INTERNATIONAL CONFERENCE

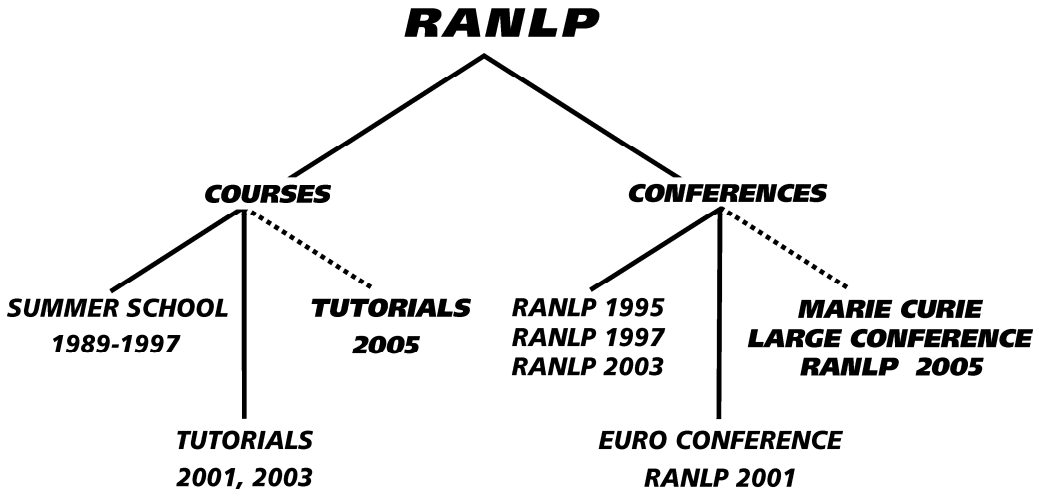
RECENT ADVANCES IN

NATURAL LANGUAGE PROCESSING

Supported by the European Commission as a Marie Curie Large Conference,
contract MLCF-CT-2004-013233

P R O C E E D I N G S

Edited by
Galia Angelova, Kalina Bontcheva, Ruslan Mitkov, Nicolas Nicolov, Nikolai Nikolov



Borovets, Bulgaria

21-23 September 2005

INTERNATIONAL CONFERENCE

RECENT ADVANCES IN

NATURAL LANGUAGE PROCESSING

Supported by the European Commission as a Marie Curie Large Conference,
contract MLCF-CT-2004-013233

P R O C E E D I N G S

Edited by
Galia Angelova, Kalina Bontcheva, Ruslan Mitkov, Nicolas Nicolov, Nikolai Nikolov

Borovets, Bulgaria

21-23 September 2005

INTERNATIONAL CONFERENCE
RECENT ADVANCES IN
NATURAL LANGUAGE PROCESSING'2005

PROCEEDINGS

Borovets, Bulgaria
21-23 September 2005

ISBN 954-91743-3-6

Designed and Printed by INCOMA Ltd.
Shoumen, BULGARIA

ORGANISERS AND SPONSORS

The International Conference RANLP–2005 is organised by:

Linguistic Modelling Department, Institute for Parallel Processing (IPP), Bulgarian Academy of Sciences (BAS)

and

Association for Computational Linguistics - Bulgaria

RANLP–2005 is a Marie Curie Large Conference supported by:

The European Commission via contract MLCF-CT-2004-013233

IPP-BAS (BIS-21++ Centre of Excellence)

Association for Computational Linguistics - Bulgaria

The team behind RANLP–2005:

Galia Angelova	Bulgarian Academy of Sciences, Bulgaria, OC Chair
Kalina Bontcheva	University of Sheffield, U.K.
Ruslan Mitkov	University of Wolverhampton, U.K., PC Chair
Nicolas Nicolov	Umbria, Inc., Boulder, U.S.A.
Nikolai Nikolov	INCOMA Ltd., Shoumen, Bulgaria

PROGRAMME CHAIR

Ruslan Mitkov (University of Wolverhampton)

PROGRAMME COMMITTEE

Eneko Agirre (University of the Basque Country, Donostia)
Elisabeth Andre (University of Augsburg)
Galia Angelova (Bulgarian Academy of Sciences)
Amit Bagga (Ask Jeeves, Piscataway)
Branimir Boguraev (IBM, Yorktown Heights)
Kalina Bontcheva (University of Sheffield)
António Branco (University of Lisbon)
Eugene Charniak (Brown University, Providence)
Dan Cristea (University of Iasi)
Hamish Cunningham (University of Sheffield)
Walter Daelemans (University of Antwerp)
Ido Dagan (Bar Ilan Univ. & FocusEngine, Tel Aviv)
Robert Dale (Macquarie University)
Thierry Declerck (DFKI GmbH, Saarbrücken)
Gaël Dias (Beira Interior University, Covilhã)
Rob Gaizauskas (University of Sheffield)
Alexander Gelbukh (Nat. Polytechnic Inst., Mexico)
Ralph Grishman (New York University)
Walther von Hahn (University of Hamburg)
Jan Hajic (Charles University, Prague)
Johann Haller (University of Saarland)
Catalina Hallett (The Open University, Milton Keynes)
Graeme Hirst (University of Toronto)
Eduard Hovy (ISI, University of Southern California)
Nikiforos Karamanis (University of Wolverhampton)
Martin Kay (Stanford University)
Dorothy Kenny (Dublin City University)
Alma Kharrat (Microsoft Natural Language Group)
Sandra Kübler (University of Tübingen)
Manfred Kudlek (University of Hamburg)
Lori Lamel (LIMSI/CNRS, Orsay)
Shalom Lappin (King's College, London)
Yves Lepage (ATR, Japan)
Anke Luedeling (Humboldt University, Berlin)
Bente Maegaard (CST, University of Copenhagen)
Bernardo Magnini (ITC-IRST, Trento)
Nuno Mamede (INESC, Lisbon)
Inderjeet Mani (Georgetown University)
Carlos Martin-Vide (Univ. Rovira i Virgili, Tarragona)
Tony McEnery (Lancaster University)
Wolfgang Menzel (University of Hamburg)
Rada Mihalcea (University of North Texas, Denton)
Andres Montoyo (University of Alicante)
Rafael Muñoz-Guillena (University of Alicante)
Masaki Murata (NICT, Japan)
Makoto Nagao (NICT, Japan)
Preslav Nakov (University of California at Berkeley)
Ani Nenkova (Columbia University)
John Nerbonne (University of Groningen)
Nicolas Nicolov (Umbria, Inc., Boulder)
Kemal Oflazer (Sabanci University, Istanbul)
Constantin Orăsan (University of Wolverhampton)
Chris Paice (Lancaster University)
Manuel Palomar (University of Alicante)
Victor Pekar (University of Wolverhampton)
Fabio Pianesi (ITC-IRST, Trento)
Stelios Piperidis (ILSP, Athens)
John Prager (IBM, Yorktown Heights)
Gábor Prószéky (MorphoLogic, Budapest)
Stephen Pulman (Oxford University)
James Pustejovsky (Brandeis University)
José Francisco Quesada Moreno (University of Seville)
Dragomir Radev (University of Michigan)
Fuji Ren (University of Tokushima)
Ellen Riloff (University of Utah)
Anne de Roeck (The Open University, Milton Keynes)
Antonio Rubio (University of Granada)
Franco Salvetti (Umbria, Inc., Boulder)
Frédérique Segond (Xerox Research Centre, Grenoble)
Kiril Simov (Bulgarian Academy of Sciences)
Richard Sproat (Univ. of Illinois at Urbana-Champaign)
Steffen Staab (University of Koblenz-Landau)
Keh-Yih Su (Behavior Design Corporation, Hsinchu)
John Tait (University of Sunderland)
Kristina Toutanova (Stanford University)
Isabel Trancoso (INEC, Lisbon)
Jun'ichi Tsujii (University of Tokyo)
Hans Uszkoreit (University of Saarland, Saarbrücken)
Karin Verspoor (Los Alamos National Laboratory)
Piek Vossen (Irion Technologies, Delft)
Yorick Wilks (Sheffield University)
Michael Zock (LIMSI/CNRS, Orsay)

REVIEWERS

In addition to the members of the Programme Committee, the following colleagues were involved in the reviewing process:

Iñaki Alegria (University of the Basque Country, Donostia)
Elsa Alves (Beira Interior University, Covilhã)
Salah Ait-Mokhtar (Xerox Research Centre Europe, Grenoble)
Le Ha An (University of Wolverhampton)
Maria Aretoulaki (Intervoice Ltd., Manchester)
Victoria Arranz (ELDA, Paris)
Patricio Martínez-Barco (University of Alicante)
Jorge Baptista (University of Algarve)
Svetla Boytcheva (Sofia University)
Borja Navarro Colorado (University of Alicante)
Bonaventura Coppola (ITC-IRST, Trento)
Courtney Corley (University of North Texas, Denton)
Andras Csomai (University of North Texas, Denton)
Víctor M. Darriba Bilbao (University of Vigo)
Díaz de Ilarraza, Arantza (U. of the Basque Country, Donostia)
Pavlin Dobrev (Prosyst Labs, Sofia)
Richard Evans (University of Wolverhampton)
Jesús Vilares Ferro (Universidad de La Coruña, La Coruña)
Manuel Vilares Ferro (University of Vigo)
Kilian Foth (University of Hamburg)
Maria Gavrilidou (ILSP, Athens)
Jorge Graña Gil (Universidad de La Coruña, La Coruña)
Laura Hasler (University of Wolverhampton)
Florentina Hristea (University of Bucarest)
Hitoshi Isahara (Communications Research Laboratory)
Stephan Kepser (University of Tübingen)
Zornitca Kozareva (University of Alicante)
Milen Kouylekov (ITC-IRST, Trento)
Hristo Krushkov (University of Plovdiv)
Olga Krasavina (Humboldt University, Berlin)
Lothar Lemnitzer (University of Tuebingen)
Yaoyong Li (University of Sheffield)
Leandro Rodríguez Liñares (University of Vigo)
Fernando Magán Muñoz (Centro "Ramón Piñeiro")
David Martínez (University of the Basque Country, Donostia)
Diana Maynard (University of Sheffield)
Miguel Angel Molinero (Centro "Ramón Piñeiro")
Vivi Nastase (University of Ottawa)
Matteo Negri (ITC-IRST, Trento)
Petya Osenova (Bulgarian Academy of Sciences)
Juan Otero Pombo (University of Vigo)
Anca Pascu (University of Western Brittany, Brest)
Katerina Pastra (ILSP, Athens)
Viktor Pekar (University of Wolverhampton)
Krasimira Petrova (Sofia University)
Emanuele Pianta (ITC-IRST, Trento)
Luis Pineda (UNAM, Mexico)
Paloma Moreda Pozo (University of Alicante)
Rudy Prabowo (University of Wolverhampton)
Georgiana Puscasu (University of Wolverhampton)
Jean-Michel Renders (Xerox Research Centre, Grenoble)
Francisco J. Ribadas (University of Vigo)
Francisco Mario Barcala Rodríguez (Centro "Ramón Piñeiro")
Bogdan Sacaleanu (DFKI, Saarbrücken)
Horacio Saggion (University of Sheffield)
Maximiliano Saiz Noeda (University of Alicante)
Estela Saquete Boró (University of Alicante)
Mark Stevenson (University of Sheffield)
Armando Suárez (University of Alicante)
Carlo Strapparava (ITC-IRST, Trento)
Valentin Tablan (University of Sheffield)
Alfonso Ureña (Universidad de Jaén)
Masao Utiyama (NICT, Japan)
Nathan Vaillette (University of Tübingen)
Maria Vargas-Vera (The Open University, Milton Keynes)
Nikolay Vazov (Sofia University)
Sonia Vázquez (University of Alicante)
Joan Miquel-Verges (University of Vigo)
Yannick Versley (University of Tübingen)
Andreas Wagner (University of Tübingen)
Ting Wang (University of Sheffield)
Holger Wunsch (University of Tübingen)
Eiko Yamamoto (NICT, Japan)
Milena Yankova (Ontotext Lab, Sofia)
Heike Zinsmeister (University of Tübingen)

PROGRAMME COMMITTEE COORDINATORS

Irina Temnikova (Bulgarian Academy of Sciences)
Albena Strupchanska (Bulgarian Academy of Sciences)

TABLE OF CONTENTS

KEYNOTE PAPERS

REGULAR PAPERS, SHORT PAPERS and POSTERS

NLP: An Information Extraction Perspective

Ralph Grishman
Department of Computer Science
New York University
715 Broadway, 7th Floor
New York, NY 10003 U.S.A.
grishman@cs.nyu.edu

This talk will look at some current issues in natural language processing from the vantage point of information extraction (IE), and so give some perspective on what is needed to make IE more successful. By IE we mean the identification of important types of relations and events in unstructured text. IE provides a nice reference point because it is compatible with a wide range of technologies: fairly simple methods can already have some degree of success at this task, while 'really good' IE will require all the tools of 'deep understanding'.

The Challenges of IE

IE is a domain-specific task; the important types of objects and events for one domain (e.g., people, companies, being hired and fired) can be quite different from those for another domain (e.g., genomics). IE for a domain can be broken down into three tasks:

1. determining what the important types of facts are for the domain
2. for each type of fact, determining the various ways in which it is expressed linguistically
3. identifying instances of these expressions in text

While there is a fuzzy boundary between these tasks, this division will provide a basis for organizing this talk, starting with the last of these tasks.

Identifying instances of a linguistic expression

To find instances of a particular IE-relevant expression, it clearly won't do to just look for word-by-word matches; to be at all successful, matching must occur at a structural level. So the crucial problem here, at the heart of many NLP applications, is the accurate identification of the structure of sentences and entire discourses.

This structure exists on many levels: the structure of names; the grammatical structure of sentences; and coreference structure across a discourse (and even across multiple discourses). Each of these is important to IE … to figuring out the participants in an event. And each of these has been studied separately and quite intensively over the past decade. Annotated corpora have been prepared for each of these levels of structure, and a wide range of models and machine learning methods have been applied to construct analyzers (particularly for name and grammatical structure). Except for coreference analysis, the result of these efforts have in general been quite satisfactory levels of performance … on the order of 90% accuracy for names and for grammatical constituents.[1]

In a typical system, these analyzers are applied sequentially to preprocess a text for extraction. Unfortunately, the analysis errors of the individual stages not only add up, they compound: an error in an early stage will often lead to further errors as analysis progresses. The net result is that overall analysis performance, and hence extraction performance, is still not very good. For the MUC evaluations in the 1990's, recall on the event task rarely broke the 60% 'ceiling' (Hirschman 1998), and it's not clear if we are doing much better today.

One limitation is the reliance on relatively local features in the early stages of analysis. Most NE (named entity) analyzers, for example, are based on simple models that look only one or two tokens ahead and behind. This fails to capture such basic tendencies as the increased likelihood of a name that was mentioned once in a document being mentioned again. To account for this, some systems employ a name cache or, more elaborately, features based on the context of other instances of the same string (Chieu and Ng 2002) – in effect, trying to do simple coreference within the name tagger. However, preferences which depend on more complex syntactic structures – for instances, that names appearing as the

[1] When tested on texts similar in genre and time period to those on which the analyzer was trained.

subjects of selected verbs are likely to be person names – remain difficult to capture because the structures are simply not available at this stage of analysis.

A more general approach harnesses the richer representations of the later stages to aid the performance of earlier ones. We generate multiple hypotheses in the first stage and then rescore them using information from subsequent stages. In general, we rely on the idea that the discourse is coherent – that in a properly-analyzed discourse, there will be many connections between entities. For example, we expect that a correct name tagging will license more coreference relations as well as more semantic relations (such as 'X is located in Y', 'X works for Y', etc.). By evaluating the result of these later stages of analysis for each hypothesized set of name tags, a system can use these later stages to improve name tagging. (Ji and Grishman 2005) generated N-best NE hypotheses and rescored them after coreference and semantic relation identification; they obtained a significant improvement in Chinese NE performance. (Roth and Yi 2004) built separate probabilistic models for name classification[2] and for semantic relation identification, and then used a linear programming model to capture the interactions between names and relations and to maximize the total probability (the product of name and relation probabilities). They obtained significant improvements in both name classification and relation detection. We can expect this 'global optimization' approach will be extended in the future to integrate a wider range of analysis levels and provide further performance improvements, possibly even incorporating cross-document information.

While such approaches should reduce analysis error, we need to consider how to deal with the error that remains. 'Deeper' representations can in principle do a better job in supporting IE (by identifying the common features of variant syntactic forms), but they will generally involve greater error. This is a dilemma which has faced IE developers for a decade. It has led many groups to rely on partial parsing which, while less informative, is more accurate than full parsing.[3] However, machine learning methods which can handle large numbers of features have allowed recent systems to integrate information from multiple levels of representation in predicting the existence of IE relations and events (Kambhatla 2004). (Zhao et al. 2004, Zhao and Grishman 2005) have shown how using kernel methods to combine information from n-grams, chunks, and grammatical relations can improve extraction performance over using a single level of representation. In some cases where there is an error in the deep analysis, a correct extraction decision will still be made based on the shallow features.

Finding linguistic expressions of an event or relation

The methods just described will give us a better chance of identifying instances of a particular linguistic expression, but we are still faced with the problem of finding the myriad linguistic expressions of an event – all (or most of) the paraphrases of a given expression of an event. A direct approach is to annotate all the examples of an event in a large corpus, and then collect and distill them either by hand or using some linguistic representation and machine learning method. However, good coverage may require a really large corpus, which can be quite expensive. Could we do better?

We need first of all to differentiate syntactic and semantic paraphrase. Syntactic paraphrases are applicable over broad (grammatical) classes of words – relations between active, passive, and relative clauses, for example, as well as complement alternations. Many of these can be addressed by using a deeper syntactic representation that captures the commonality among such different expressions. In particular, a predicate-argument representation, such as is being encoded for English in PropBank (Kingsbury and Palmer 2002) and NomBank (Meyers et al. 2004), would collapse many of these syntactic paraphrases.

What remains are the much more varied and numerous semantic paraphrases. There are dozens of ways of saying that a company hired someone, or that two people met. Lexical-semantic resources (such as WordNet) provide some assistance (Stevenson and Greenwood 2005), but they are largely limited to single-word paraphrases and so cover only a portion of the myriad expressions required for an IE task. To complement these manually-prepared resources, efforts have been underway for the past few years to learn paraphrase relations from corpora. The basic idea is to identify pairs of expressions A C B and A D B which involve the same arguments (A, B) and most likely convey the same information; then C and D stand a good chance of being paraphrases. One source of such pairs are two translations of the same text (Barzilay and McKeown 2001). If we can sentence-align the

[2] They assumed the extent of the names in the text was given.

[3] Many groups made this choice prior to the recent improvements in treebank-trained parsers, but the choice is still not clear-cut.

texts, the corresponding sentences are likely to carry the same information. Another source are comparable news articles – articles from the same day about the same news topic (Shinyama et al. 2002). The opening sentences of such articles, in particular, are likely to contain phrases which convey the same information. The likelihood is even greater if we focus on phrases which are both relevant to the same topic (see the next section). Finally, frequency can build confidence: if we have several pairs of individuals A_1, B_1; A_2, B_2; … which appear in both context C (A_1 C B_1; A_2 C B_2) and context D (A_1 D B_1; A_2 D B_2), then C and D stand a good chance of being paraphrases. This general approach has been used to find paraphrases for individual relations (Brin 1998; Agichtein and Gravano 2000; Lin and Pantel 2001) and to collect the primary paraphrase relations of a domain (Sekine 2005).

Discovering what's important

Finally, there may be situations where we don't have specific event or relation types in mind … where we simply want to identify and extract the 'important' events and relations for a particular domain or topic. (Riloff 1996) introduced the basic idea of dividing a document collection into relevant (on topic) and irrelevant (off topic) documents, and selecting constructs which occur much more frequently in the relevant documents. Her approach relied on a relevance-tagged corpus. This idea was extended by (Yangarber et al. 2000) to bootstrap the discovery process from a small 'seed' set of patterns which define a topic. Sudo generalized the form of the discovered patterns (Sudo et al. 2003) and created a system which started from a narrative description of a topic and used this description to retrieve relevant documents (Sudo et al. 2001).

These methods have been used to collect the linguistic expressions for a specific set of event types, and they are effective when these events form a coherent 'topic' … when they co-occur in documents. Because these methods are based on the distribution of constructs in documents, they may gather together related but non-synonymous forms like 'hire', 'fire', and 'resign', or 'buy' and 'sell'. However, by coupling these methods with paraphrase discovery, it should be possible to both gather relevant expressions and group those representing the same event types (Shinyama et al. 2002).

Acknowledgements

This research was supported by the Defense Advanced Research Projects Agency under Grant N66001-04-1-8920 from SPAWAR San Diego, and by the National Science Foundation under Grant 03-25657. This paper does not necessarily reflect the position or the policy of the U.S. Government.

References

(Agichtein and Gravano 2000) Eugene Agichtein and Luis Gravano. Snowball: Extracting Relations from Large Plain-Text Collections. *Proc. Fifth ACM Int'l Conf. on Digital Libraries,* 2000.

(Barzilay and McKeown 2001) Regina Barzilay and Kathleen R. McKeown. Extracting paraphrases from a parallel corpus. *Proc. ACL/EACL 2001*.

(Brin 1998) Sergei Brin. Extracting Patterns and Relations from the World Wide Web. *Proc. World Wide Web and Databases International Workshop*, pp. 172-183. Number 1590 in LNCS, Springer, March 1998.

(Chieu and Ng 2002) Hai Leong Chieu and Hwee Tou Ng. Named entity recognition: a maximum entropy approach using global information. *Proc. 19th Int'l Conf on Computational Linguistics (COLING 2002),* Taipei, August 2002, 190-196.

(Hirschman 1998) Lynette Hirschman. Language understanding evaluations: lessons learned from MUC and ATIS. *Proc. 1st Int'l Conf. on Language Resources and Evaluation (LREC 1998)*, Granada, Spain, May 1998, 117-122.

(Ji and Grishman 2005) Heng Ji and Ralph Grishman. Improving name tagging by reference resolution and relation detection. *Proc. 43rd Annl. Meeting Assn. for Computational Linguistics*, Ann Arbor, MI, June 2005, 411-418.

(Kambhatla 2004) Nanda Kambhatla. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. *Companion Volume to Proc. 42nd Annl. Meeting Assn. Computational Linguistics (ACL 2004)*, Barcelona, Spain, July 2004, 178-181.

(Kingsbury and Palmer 2002) Paul Kingsbury and Martha Palmer. From TreeBank to PropBank. *Proc. 3rd Int'l Conf. on Language Resources (LREC-2002)*, Las Palmas, Spain, 2002.

(Lin and Pantel 2001) Dekang Lin and Patrick Pantel. Discovery of inference rules for question answering. *Natural Language Engineering* **7**(4): 343-360.

(Meyers et al. 2004) Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. Annotating noun argument structure for NomBank. *Proc. 4th Int'l Conf. on Language Resources and Evaluation (LREC 2004)*, Lisbon, Portugal, 2004.

(Riloff 1996) Ellen Riloff. Automatically generating extraction patterns from untagged text. *Proc. Thirteenth National Conference on Artificial Intelligence*, 1996.

(Roth and Yih 2004) Dan Roth and Wen-tau Yih. A linear programming formulation for global inference in natural language tasks. *Proc. Conf. on Computational Natural Language Learning (CoNLL-2004)*, Boston, MA, 2004, 1-8.

(Sekine 2005) Satoshi Sekine. Automatic paraphrase discovery based on context and keywords between NE pairs. To appear

in *Proc. 3rd Int'l Workshop on Paraphrasing*, Jeju Island, South Korea, Oct. 2005.

(Shinyama et al. 2002) Yusuke Shinyama, Satoshi Sekine, Kiyoshi Sudo and Ralph Grishman. Automatic Paraphrase Acquisition from News Articles. *Proc. Human Language Technology Conference (HLT 2002)*, San Diego, CA, 2002.

(Stevenson and Greenwood 2005) Mark Stevenson and Mark Greenwood. A semantic approach to IE pattern induction. *Proc. 43rd Annl. Meeting Assn. for Computational Linguistics*, Ann Arbor, MI, June 2005, 379-386.

(Sudo et al. 2001) Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. Automatic pattern acquisition for Japanese information extraction. *Proc. Human Language Technology Conference (HLT 2001),* San Diego CA, 2001.

(Sudo et al. 2003) Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. An improved extraction pattern representation model for automatic IE pattern acquisition. *Proc. 41st Annl. Meeting Assn. for Computational Linguistics (ACL 2003)*, Sapporo, Japan, 2003.

(Yangarber et al. 2000) Roman Yangarber, Ralph Grishman, Pasi Tapanainen, and Silja Huttunen. Automatic acquisition of domain knowledge for information extraction. *Proc. 18th Int'l Conf. on Computational Linguistics (COLING 2000)*, Saarbrucken, Germany, August 2000, 940-946.

(Zhao et al. 2004) Shubin Zhao, Adam Meyers, and Ralph Grishman. Discriminative slot detection using kernel methods. *Proc. 20th Int'l Conf. on Computational Linguistics (COLING 2004)*, Geneva, August 2004.

(Zhao and Grishman 2005) Shubin Zhao and Ralph Grishman. Extracting relations with integrated information using kernel methods. *Proc. 43rd Annl. Meeting Assn. for Computational Linguistics*, Ann Arbor, MI, June 2005, 419-428.

Linguistic Challenges for Computationalists

John Nerbonne*
Humanities Computing
University of Groningen
NL 9700 AS Groningen, The Netherlands
j.nerbonne@rug.nl

Abstract

Even now techniques are in common use in computational linguistics which could lead to important advances in pure linguistics, especially language acquisition and the study of language variation, if they were applied with intelligence and persistence. Reliable techniques for assaying similarities and differences among linguistic varieties are useful not only in dialectology and sociolinguistics, but could also be valuable in studies of first and second language learning and in the study of language contact. These techniques would be even more valuable if they indicated relative degrees of similarity, but also the source of deviation (contamination). Given the current tendency in linguistics to wish to confront the data of language use more directly, techniques are needed which can handle large amounts of noisy data and extract reliable measures from them. The current focus in Computational Linguistics on useful applications is a very good thing, but some further attention to the linguistic use of computational techniques would be very rewarding.

1 Introduction

The goal of this paper is to urge computational linguists to explore issues in other branches of linguistics more broadly. Computational linguistics (CL) has developed an impressive array of analytical techniques, especially in the past decade and a half, techniques which are capable of assaying linguistic structure of various levels from fairly raw textual data. The goal will be to note how these techniques might be applied to illuminate other issues of broad interest in linguistics.

The thesis my plea is based on—that there are opportunities for computational contributions to "pure" linguistics—is not absolutely new, of course, as many computational linguists have been involved in issues of pure linguistics as well, including especially grammatical theory. And we will naturally attempt to identify such work as we become more concrete (below). We aim to spark discussion by identifying less discussed areas where computational forays appear promising, and in fact, we will not dwell on grammatical theory at all.

It is best to add some *caveats*. First, the sort of appeal we aim at can only be successful if it is sketched with some concrete detail. If we attempted to argue the usefulness of computational techniques to general linguistic theory very abstractly, virtually everyone would react, "Fine, but how can we contribute more concretely?" But we can only provide more concrete detail on a very limited number of subjects. Of course, we are limited by our knowledge of these subjects as well, but the first *caveat* is that this little essay cannot be exhaustive, only suggestive. We should be delighted to hear promptly of several further areas of application for computational techniques we omit here.

Second, the exhortation to explore issues in other branches of linguistics more broadly takes the form of an examination of selected issues in non-computational linguistics together with suggestions on how computational techniques might shed added light on them. Since the survey is to be brief, the suggestions about solutions—or perhaps, merely perspectives—of necessity will also be brief. In particular, they will be no more than *suggestions*, and will make no pretense at demonstrating anything at all.

Third, we might be misconstrued as urging you to ignore useful, money-making applications in favor of dedicating yourselves to the higher goal of collaborating in the search for scientific truth. But both the history of CL and the usual modern attitude of scientists toward applications convinces me that the application-oriented side of CL is very important and eminently worthwhile. Per-

*We are grateful to the Netherlands Organization for Scientific Research, NWO, for support (project "Determinants of Dialect Variation, 360-70-120, P.I. J. Nerbonne). It was stimulating to discuss the general issue of engineering work feeding back into pure science with Stuart Schieber, who organized a course with Michael Collins of MIT at the Linguistics Institute of the Linguistic Society of America at MIT, Summer 2005 contrasting science and engineering in CL.

haps you should indeed turn a deaf ear to the seductions of filthy mammon and consecrate yourself to a life of (pure) science, but this is a matter between you and your clergyman (or analyst). You have not gotten this advice here.

Fourth, and finally, we might be viewed as advocating different sorts of APPLICATIONS, namely the application of techniques from one linguistic subfield (CL) to another (dialectology, etc.). In this sense modern genetics APPLIES techniques from chemistry to biological molecules to determine the physical basis of inheritance, anthropology APPLIES techniques from nuclear chemistry (carbon dating) to date human artefacts, and astronomy APPLIES techniques from optics (glass) and electromagnetism (radio astronomy) to map the heavens. In all of these case is the primary motivation is scientific curiosity, not utilitarian, and this view is indeed parallel to the step advocated here.

2 Computational Linguistics

Computational Linguistics (CL) is often characterized as having a theoretical and an application-oriented, or engineering side (Joshi 99; Kay 02). The theoretical side of CL is concerned with processes involving language and their abstract computational characterization, including processes such as analyzing (parsing), and producing (generating) language, but also storing, compressing, indexing, searching, sorting, learning and accessing language. The computational characterization of these processes involves investigating algorithms for their accuracy and time and space requirements, finding appropriate data structures, and naturally testing these ideas, where possible, against concrete implementations.

The application-oriented, or engineering side of the field concerns itself with creating useful computational systems which involve language manipulation in some way, e.g. lexicography tools; speech understanding (in collaboration with speech recognition); machine translation, including translation aids such as translation memories, multilingual alignment, and specialized lexicon construction; speech synthesis, especially intonation; term extraction, information retrieval, document summarization, data (text) mining, and question answering; telephone information systems and natural language interfaces; automatic dictionary and thesaurus access, grammar checking, including spell-checking; document management, authoring (especially in multi-author systems), and conformance to specifications in so-called "controlled language systems"; foreign language aids (such as access to bilingual dictionaries), foreign language tutoring systems, and communication aids (for the handicapped). See Cole et al. (1996) for further discussion of these, and other areas of application for language technology.

We have been overly compulsive about listing the engineering activities not only to remind the reader how extensive these are, but also to emphasize that the breadth of these activities would be unthinkable if it were not for a rich "infrastructure" of language technology tools which the field is constantly creating. For the most part the techniques we urge you to apply more broadly have been developed in order to build better and more varied applications, as this has been the great motor in the recent dynamics of computational linguistics. But some of the techniques have also been useful in theoretical computational linguistics, and the distinction will play no role here. In fact, perhaps the simplest view is to acknowledge that applications and theory make use of common technology, a sort of technical infrastructure, and to emphasize the opportunities this provides.

3 Dialectology

We shall examine dialectology first because it is an area we have directly worked in, and for which we therefore need to rely less on speculation about the potential benefits of a computational approach. Given the greater amount of direct experience with this work, we may use it to distill some of the characteristics we need to seek in other areas in which computational techniques might be promising.

Dialectology studies the patterns of variation in a language and especially its geographic conditioning (Chambers & Trudgill 80). In London people say [wɒtə] for 'water', with a voiceless [t] and no trace of final [r], in New York most people say [wɑɾɪ̥], with a "tapped" [t], and in Boston [wɑɾə]. These differences are systematic, but not exceptionless, and they appear to involve potentially every level of linguistic structure, pronunciation, morphology, lexicon, syntax, and discourse. Because differences appear to involve exceptions, it is advantageous to process a great deal of mate-

rial and to apply statistical techniques to the analysis. Fortunately, dialectologists have been assiduous in collecting and archiving a great deal of data, especially involving pronunciation and lexical differences.

Once we have agreed that we need to subject a great deal of data to systematic analysis, we have *a fortiori* accepted the need for automating the analysis, and since it is linguistic material, it would be strange if this did not lead us to computational linguistics. In fact EDIT DISTANCE, well-known to computational linguists by its wide variety of applications, may be applied fairly directly to the phonetic transcripts of dialect pronunciations (Nerbonne *et al.* 99). The application of edit distance to pronunciation transcripts yields, for each pair of words, at each pair of field work sites, a numerical characterization of the difference. Because pronunciation differences are characterized numerically, we thereby initiate a numerical analysis of data that dialectologists had normally regarded as categorical—with all the advantages which normally accrue to numerical data analysis.

Nerbonne (2003) discusses at greater length the computational issues in analyzing, presenting and evaluating dialectological analyses, including those which go beyond pronunciation. These issues include the use of lemmatizers or stemmers to clean up word-form data for lexical analysis, raising the edit distance from strings to sets of strings in order treat data collections with alternative forms, and the proper treatment of frequency in detection of linguistic proximity. Opportunities for the application of standard CL techniques in computational linguistics abound. Heeringa (2004) summarizes current thinking on measuring dialectal pronunciation differences, including the thorny issue of evaluating the quality of results. Figure 1 illustrates the results of applying these techniques to Bulgarian data.

It is important to report here, as well, that specialists in dialectology—and not only computational linguists—are enthusiastic about the deployment of computational tools. A common remark by dialectologists is that that the new techniques allow a more comprehensive inclusion of all available data, effectively answering earlier complaints that analyses of dialect areas and/or dialect continua relied too extensively on the analysts' choice of material. William

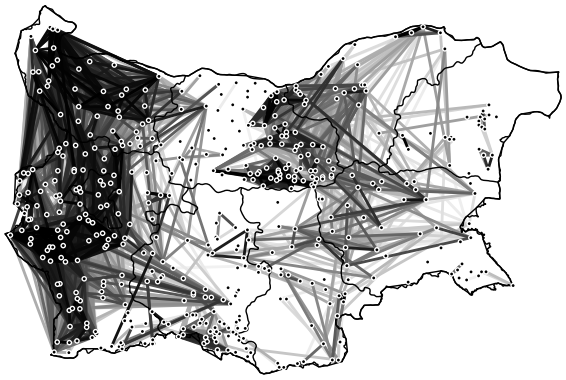


Figure 1: In this line map the average Levenshtein distances between 490 Bulgarian dialects are shown for 36 words. Darker lines join varieties with more similar pronunciations, while lighter lines indicate more dissimilar ones. From collaborative work in progress with Petya Osenova, Bulgarian Academy of Science, and Wilbert Heeringa, Groningen.

Kretzschmar leads the American *Linguistic Atlas Projects* (LAP), and has collaborated in various analyses and workshops (Nerbonne & Kretzschmar 03). He has *inter alia* included a pointer to CL work on the home page of the LAP site he maintains at `http://us.english.uga.edu/`, and he is presently collaborating on a project to publish a second volume of papers focused on computational techniques (Nerbonne & Kretzschmar 06).

Finally, let us note that the computational step may introduce such genuinely novel opportunities that we find ourselves in a position to ask questions which simply lay beyond earlier methodology. Given our numerical perspective on dialect difference, we may e.g. ask, via a regression analysis, how much of the aggregate varietal difference is explained by geography, or whether travel time is a superior characterization of the geography relevant to linguistic variation (Gooskens 04), or whether larger settlements tend to share linguistic variants more than smaller ones—something one might expect if variation diffused via social contact (Heeringa & Nerbonne 02). The introduction of CL techniques enables us to ask more abstract questions in a way we can still link to concrete linguistic analysis.

This work also suggests many related paths of exploration. For example, even if a distance measure allows the mapping of the dialectolog-

ical landscape well, it seems ill-equipped to assay one extreme result of dialect differentiation, i.e. the failure of comprehensibility. The reason for this failure is the fact the comprehensibility is not symmetrical, while linguistic distance by definition is: it may reliably be the case that speakers of one variety understand the speaker of another better than *vice versa*. For example, Dutch speakers find it easier to understand Afrikaans than vice versa (Gooskens & vanBezooijen 06). If this is due to language differences, it calls for the development of an asymmetrical measure of the relative difficulty of mapping from one language to another, or something similar.[1]

The computational work has been successful in dialectology because there were large reservoirs of linguistics data to which analyses could be applied, i.e., dialect atlases, because distinguishing properties resisted simple categorical characterization, and naturally because there were promising computational techniques for getting at the crucial phenomena.

As we turn to other areas, we shall ask ourselves whether we are likely to satisfy these desiderata. When even one is missing, the result can be disappointing. For example, sociolinguistics has largely succeeded dialectology in attracting scholarly interest. The linguistic issues are not wildly different—different social groups use different language varieties, and these may differ in all the ways in which geographical varieties do (pronunciation, lexicon, etc.). It would be straightforward and interesting to apply the techniques sketch above to linguistic varieties associated with different social groups. But there is no tradition in sociolinguistics like that of the dialect atlas, i.e. collecting speech samples from a large set of sociolects. So the opportunity does not present itself.

4 Diachronic Linguistics

Diachronic linguistics investigates how languages change, and, most spectacularly, how a single language many evolve into many related ones. It regularly attracts a good deal of scholarly attention (Gray & Atkinson 03; Eska & Ringe 04) as computational biologists have applied their techniques for tracking genetic evolution to linguistic data. Although the scholarship is at times forbidding in its expectations about philological expertise, the problem appears to allow neat enough formulations so that one may be optimistic about computational investigations.

Essentially, we are given a set of cognate words in several putatively related languages, and we construct hypotheses about the most recent common ancestor—the protolanguage—as well as a simple set of sound changes leading from the protolanguage to the individual descendants. For example, we note that the word for father has an initial /f/ in Germanic (English *father*), /p/ in Romance, Greek and Indic (French *père*, Greek *patera*, and Hindi *pitā*), and no initial consonant in some Celtic languages (Irish *athair*). This suggests that we postulate a /p/ in the protolanguage and changes from /p/ to /f/ for Germanic and /p/ to $\emptyset$ for the relevant Celtic varieties. But we gain confidence in these postulates only when the same rules are shown to operate on other forms, i.e. when the correspondences recur (as the p/f/$\emptyset$ definitely does). It is surprising that CL should turn over to the biologists such a well-structured problem in linguistic computation.[2]

Our community has contributed to this area, especially Brett Kessler, who investigated how to test when sound correspondences exceed chance levels (Kessler 01), and Grzegorz Kondrak, who modified the edit distance algorithm mentioned above, in order to identify cognates, align them, and on that basis postulate recurrent sound combinations (Kondrak 02). But these studies deserve follow-ups, tests on new data, and extensions to other problems. Among many remaining problems we note that it would be valuable to detect borrowed words, which should not figure in cognate lists, but which suggest interesting influence; to operationalize the notion of semantic relatedness relevant to cognate recognition; to quantify how regular sound change is; or to investigate the level of morphology, which is regarded as especially probative in historical reconstruction. But we emphasize that there are likely to be interesting opportunities for contributions with respect to detail as well, perhaps in the construction of instruments to examine data more insightfully, to measure hypothesized aspects, or to quantify the empirical base on which historical

[1] Nathan Vaillette, University of Massachusetts has explored this problem using relative entropy in unpublished work.

[2] See also Benedetto et al. (2002) for attempt to reconstruct linguistic history using relative entropy, but especially Goodman (2002) for criticism of Benedetto.

hypotheses are made.

5 Language Acquisition

Studies of children's acquisition of language are interesting to all sorts of inquiries because language is a defining characteristic of us as humans. They occupy an important position in linguistics due to the linguistic argument that innate, specifically linguistic mechanisms must be postulated to account for acquisition (Pinker 94, Chap. 9). The innate organizing principles of language are postulated to be part of human genetic constitution, and therefore the source of universal properties which all languages share. At the same time psychologists have shown that some acquisition is mediated by sensitivity to statistical trends in data (Saffran *et al.* 99). And children naturally need minimally to learn which of all the languages they are genetically predisposed toward is the one in use locally. Finally, CL has explored machine learning techniques extensively over the past decade (Manning & Schütze 99). Surely CL is positioned to contribute crucially to this scientific discussion with interesting implemented models of specific phenomena, and in particular with models aimed at broader coverage or so one would think.

On the other hand, machine learning techniques do not translate to computational models of acquisition very directly, at least not as normally used by CL, namely to optimize performance on technical tasks that may have no interesting parallel in a child's acquisition of language, e.g. the task of recognizing named entities, persons, places and organizations. In addition, even idealized simulations of acquisition might wish to impose restrictions on the sort of mechanisms to be used, e.g. that they may apply incrementally, and on the input data, e.g. that it reflect children's experience.

Fortunately, these differences in tasks, mechanisms and input data may be overcome, and CL has not been inactive in examining language acquisition. Brent (1997) is an early collection of articles on computational approaches to language acquisition, including especially Brent's own work applying minimal description length to the problem of segmenting the speech stream into words, and using only phonotactic and distributional information (Brent 99b; Brent 99a). There have been a number of other studies focusing on phonotactics (Nerbonne & Stoianov 04; Nerbonne & Konstantopoulos 04), the acquisition of morphophonemic rules (Gildea & Jurafsky 96; Albright & Hayes 03), morphology (Goldsmith 01), and syntax (Niyogi & Berwick 96). Albright and Hayes's work is especially worth recommending to a CL audience as it is clear and explicit about linguistic concerns in modeling acquisition computationally.

Most relevant to the sort of CL contribution I have in mind is the series of workshops organized by William Gregory Sakas of CUNY, *Psycho-Computational Models of Human Language Acquisition*. The first took place in 2004 in Geneva in coordination with COLING and the second in 2005 in Ann Arbor in coordination with the ACL's special interest group on natural language learning (`http://www.colag.cs.hunter.cuny.edu/psychocomp/`). It is clear from the proceedings of these workshops that new syntheses of linguistic, psychological and computational perspectives enjoy a good deal of interest (Yang 04).

It is also clear that there is an enormous interest in further questions about segmentation, alignment, constituency, local and long-distance relations, modification, and ill-formed input in addition to the usual questions about the generality of solutions wit respect to various language types.

Finally, it is worth emphasizing here more than elsewhere that contributions need not take the form of simulations of human learning (even if this is the case for most of the studies cited). There is great potential interest in characterizing easy vs. difficult material, in what happens when second and third languages are learned (contamination), and in how languages are lost. In addition to simulation, we should also be thinking of how to operationalize measures of language proficiency that could use speech as directly as possible. At the moment, extremely crude measures such as mean length of utterance (MLU) and type/token ratio enjoy great popularity, but one suspects that this is due more to their ease of computation than to their reflection of linguistic sophistication. Ideally we should like to automate our detection of the mastery of various linguistic structures, rules and exceptions. That is clearly a long way off in its full generality, but perhaps realizable in some instances with standard techniques.

6 Language Contact

Language contact study is an active branch of linguistics focused on recognizing and analyzing the ways in which languages borrow from one another (Thomason & Kaufmann 88; van Coetsem 88). It is growing in popularity, perhaps due to increases in mobility and the realization that multilingual speakers often, albeit unconsciously, impose the structures of one language on another. Mufwene (2001) urges us to view extreme contact effects such as koinéization, creolization and pidginization as various degrees to which language mixtures may develop (instead of as the results of very different processes, as earlier scholarship had held). Language contact study is, moreover, linked to second-language acquisition in an obvious way: if second-language speakers habitually impose elements of their native language onto another, then those element are good candidates for long-term borrowing whenever these languages are in contact.

It might seem as if we could use the same tools for the study of contact effects that we developed for dialectology. After all, if one variety of a language adopts elements of another, it should become more similar. Indeed given the sort of data in dialect atlases, one can perform these analyses and determine the convergence of some varieties toward a putative source of contamination, at least the convergence with respect to other varieties (Heeringa *et al.* 00; Gooskens & Heeringa 04). Furthermore, one could examine the role of geography in this convergence.

But language contact data collections are not usually designed as dialect atlases, with a number of distinct collection sites, and a controlled set of linguistic variables to be assayed. Recently, we obtained data of a rather different sort, and set ourselves the task of developing computational tools for its analysis.[3] Watson collected recordings of Finnish emigrants to Australia in the mid 1990's (Watson 96), and this group could be divided into adult emigrants and child emigrants, using puberty (16 years old) as the dividing line. The challenge was the development of a technique to determine whether there were significant changes in the syntax of the two groups.

[3]What follows is an informal synopsis of work in progress being conducted with Wybo Wiersma of Groningen and Lisa Lena Opas-Hänninen, Timo Lauttamus and Pekka Hirvonen of Oulu University.

Following an obvious tack from CL, we settled on using n-grams of part-of-speech tags (POS tags) assigned by the TnT tagger (Brants 00) as a probe to determine syntactic similarity. In order not to be swamped by fine distinctions we used trigrams of a small tag set (50 tags). Up to this point we were rediscovering an idea others had introduced (Aarts & Granger 98). To compare one corpus with another, we measured the difference in the two vectors of trigram frequencies using cosine (*inter alia*). To determine whether the difference is statistically significant, we applied permutation-based statistics, roughly resampling the union of the two data sets (using some complicated normalizations) and checking the degree of difference. A difference is significant at the level $p < p'$ iff it is among the most extreme p' fraction of the resampled data.

Because the technique is still under development, we cannot yet report much more. The differences are indeed statistically significant, which, in itself, is not surprising. The corpora are quite raw, however, so that the differences we are finding to-date are dominated by hesitation noises and errors in tagging. The promise is in the technique. If we have succeeded in developing an automated measure of syntactic difference, we have opportunities for application to a host of further questions about syntactic differences, e.g., about where these differences are detectable, and where not; about the time course of contamination effects (do second-language learners keep improving, or is there a ceiling effect?); and about the role of the source language in the degree of contamination. Some crucial computational questions would remain, however, concerning detecting the source of contamination.

7 Other Areas

As noted in the introduction, this brief survey has tried to develop a few ideas in order to convince you that there are promising lines of inquiry for computationalists who would seek to contribute to a broader range of linguistic subfields. We suspect that there are many other areas, as well.

We have deliberately omitted grammatical theory from the list of potential near-term adopters of computational techniques. There are two reasons for eschewing a sub-focus on grammar here, the first being the fact that the potential relevance of computational work to grammatical the-

ory has been recognized for a long time, as grammar has been cited since the earliest days of CL as a likely beneficiary of closer engagement (Kay 02). But second, even as computational grammar studies uncover new means of contributing to the study of pure grammar (van Noord 04), it seems to be a minority of grammarians who recognize the value of computational work. Many researchers have explored this avenue, but the situation has stabilized to one in which computational work is pursued vigorously by small specialized groups (Head-Driven Phrase Structure Grammar and Lexical-Functional Grammar), and largely ignored by most non-mainstream grammarians. We deplore this situation as do others (Pollard 93), but it unfortunately appears to be quite stable.

In addition to the areas discussed above, it is easily imaginable that CL techniques could play an interesting role in a number of other linguistic subareas. As databases of linguistic typology become more detailed and more comprehensive, they should become attractive targets for data-mining techniques (`http://www-uilots.let.uu.nl/td/`). Psycholinguistic studies of processing are promising because they provide a good deal of empirical data. We shall be content with a single example. Moscoso del Prado Martin (2003) reviews a large number of studies relating the difficulty of processing complex word forms, i.e., those involving inflectional and/or derivational structure to the "family size" of a word form, i.e. how many other word forms are related to it. He is able to show that a simple characterization of family size and frequency due to information theory correlates highly with processing difficulty.

8 Conclusions

We have urged computational linguists to consider how much they might contribute to curiosity-driven research into language, i.e. linguistic theory, focusing on examples in dialectology, diachronic linguistics, language acquisition and language contact. We have suggested that there are many avenues to pursue for those with a broader interest in language, and also that the tools and training one receives in developing language technology will be of direct use. We have not suggested that contributions in pure science are any easier or harder to make, and the experience has been general that the dynamics involved in pursuing non-applied goals are every bit as demanding, and every bit at provocative: a successful effort invariably suggests new questions and new avenues to explore.

We have been careful to avoid deprecating application-research and, at the risk of repetition, restate that the development of useful applications is a most valuable aspect of current CL. We encourage colleagues to think of both channels of activity rather than to force a choice of one over the other.

If we are right that most of the interesting techniques for exploring issues in non-computational linguistics have arisen through the development of techniques for engineering activities, then we may have another case where applied science furthers the progress of pure science (Burke 85). In making this remark, we are reneging on the promise in Section 2 not to concern ourselves with whether a particular technique originated in theoretical vs. applied CL, but given the preponderance of applied work in CL, it would be surprising if it were not true in many instance that techniques from engineering were being conscripted for work in theory.

The use of a stemmer to extract lexical differences from lists of word forms in dialectology (Nerbonne & Kleiweg 03) is an example of the sort of contribution where a technique developed only for application purposes could be put to a purely scientific use, that of detecting lexical overlap across a dialect continuum. The Porter stemmer which was used for this purpose is not to be confused with a genuine lemmatizer, which is interesting both linguistically and practically. But it usually reduces word forms to the same stem when they in fact are elements of the same inflectional paradigm. It was developed for use in information retrieval (Porter 80), not for the purpose of exploring linguistic structure or its processing, but its use in dialectology has no ambitions toward practical application.

This would appear to bea genuine case of an engineering technique serving a purpose in curiosity-driven research. To the extent CL is involved in other pure science (beyond CL proper), this sort of cross-fertilization must be standard. Only time will tell whether it will remain true of future computational forays into pure linguistics.

References

(Aarts & Granger 98) Jan Aarts and Sylviane Granger. Tag sequences in learner corpora: A key to interlanguage grammar and discourse. In Sylviane Granger, editor, *Learner English on Computer*, pages 132–141. Longman, London, 1998.

(Albright & Hayes 03) Adam Albright and Bruce Hayes. Rules vs. analogy in English past tenses: A computational/experimental study. *Cognition*, 90:119–161, 2003.

(Benedetto *et al.* 02) Dario Benedetto, Emanuele Caglioti, and Vittorio Loreto. Language trees and zipping. *Physical Review Letters*, 88(4):048702, 2002.

(Brants 00) Thorsten Brants. TnT — a statistical part of speech tagger. In *6th Applied Natural Language Processing Conference*, pages 224–231, Seattle, 2000. ACL.

(Brent 97) Michael Brent, editor. *Computational Approaches to Language Acquisition.* MIT Press, Cambridge, 1997.

(Brent 99a) Michael Brent. An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning Journal*, 34:71–106, 1999.

(Brent 99b) Michael Brent. Speech segmentation and word discovery: A computational perspective. *Trends in Cognitive Science*, 3:294–301, 1999.

(Burke 85) James Burke. *The Day the Universe Changed.* Little, Brown & Co., Boston, 1985.

(Chambers & Trudgill 80) J.K. Chambers and Peter Trudgill. *Dialectology.* Cambridge University Press, Cambridge, 1998, [1980].

(Cole *et al.* 96) Ronald A. Cole, Joseph Mariani, Hans Uszkoreit, Annie Zaenen, and Victor Zue. *Survey of the State of the Art in Human Language Technology.* National Science Foundation and European Commission, `www.cse.ogi.edu/CSLU/HLTsurvey/`, 1996.

(Eska & Ringe 04) Joseph F. Eska and Don Ringe. Recent work in computational linguistic phylogeny. *Language*, 80(3):569–582, 2004.

(Gildea & Jurafsky 96) Daniel Gildea and Daniel Jurafsky. Learning bias and phonological rule induction. *Computational Linguistics*, 22(4):497–530, 1996.

(Goldsmith 01) John Goldsmith. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27(2):153–198, 2001.

(Goodman 02) Joshua Goodman. Extended comment on 'Language trees and zipping'. *Condensed Matter Archive*, Feb. 21, 2002. `arXiv:cond-mat/0202383`.

(Gooskens & Heeringa 04) Charlotte Gooskens and Wilbert Heeringa. The position of Frisian in the Germanic language area. In Dicky Gilbers, Maartje Schreuder, and Nienke Knevel, editors, *On the Boundaries of Phonology and Phonetics*, pages 61–88. CLCG, Groningen, 2004.

(Gooskens & vanBezooijen 06) Charlotte Gooskens and Renée van Bezooijen. Mutual comprehensibility of written Afrikaans and Dutch: Symmetrical or asymmetrical? *Literary and Linguistic Computing*, 21, 2006. accepted, 7/2005.

(Gooskens 04) Charlotte Gooskens. Norwegian dialect distances geographically explained. In Britt-Louise Gunnarson, Lena Bergström, Gerd Eklund, Staffan Fridella, Lise H. Hansen, Angela Karstadt, Bengt Nordberg, Eva Sundgren, and Mats Thelander, editors, *Language Variation in Europe. Papers from the Second International Conference on Language Variation in Europe ICLAVE 2, June 12-14, 2003*, pages 195–206. Uppsala University, Uppsala, Sweden, 2004.

(Gray & Atkinson 03) Russell D. Gray and Quentin D. Atkinson. Language-tree divergence times support the Anatolian theory of Indo-European origin. *Nature*, 426(27 Nov.):435–439, 2003.

(Heeringa & Nerbonne 02) Wilbert Heeringa and John Nerbonne. Dialect areas and dialect continua. *Language Variation and Change*, 13:375–400, 2002.

(Heeringa 04) Wilbert Heeringa. *Measuring Dialect Pronunciation Differences using Levenshtein Distance.* Unpublished PhD thesis, Rijksuniversiteit Groningen, 2004.

(Heeringa *et al.* 00) Wilbert Heeringa, John Nerbonne, Hermann Niebaum, and Rogier Nieuweboer. Measuring Dutch-German contact in and around Bentheim. In Dicky Gilbers, John Nerbonne, and Jos Schaeken, editors, *Languages in Contact*, pages 145–156. Rodopi, Amsterdam-Atlanta, 2000.

(Joshi 99) Aravind K. Joshi. Computational linguistics. In Robert A. Wilson and Frank C. Keil, editors, *The MIT Encyclopedia of the Cognitive Sciences*, pages 162–164. MIT Press, Cambridge, MA, 1999.

(Kay 02) Martin Kay. Introduction. In Ruslan Mitkov, editor, *Handbook of Computational Linguistics*, pages xvii–xx. Oxford University Press, Oxford, 2002.

(Kessler 01) Brett Kessler. *The Significance of Word Lists.* CSLI Press, Stanford, 2001.

(Kondrak 02) Grzegorz Kondrak. *Algorithms for Language Reconstruction.* Unpublished PhD thesis, University of Toronto, 2002.

(Manning & Schütze 99) Chris Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing.* MIT Press, Cambridge, 1999.

(Moscoso del Prado Martín 03) Fermin Moscoso del Prado Martín. *Paradigmatic Structures in Morphological Processing: Computational and Cross-Linguistic Experiemental Studies.* Unpublished PhD thesis, Radboud University Nijmegen, 2003.

(Mufwene 01) Salikoko Mufwene. *The Ecology of Language Evolution.* Cambridge University Press, Cambridge, 2001.

(Nerbonne & Kleiweg 03) John Nerbonne and Peter Kleiweg. Lexical variation in LAMSAS. *Computers and the Humanities*, 37(3):339–357, 2003. Special Iss. on Computational Methods in Dialectometry ed. by John Nerbonne and William Kretzschmar, Jr.

(Nerbonne & Konstantopoulos 04) John Nerbonne and Stasinos Konstantopoulos. Phonotactics in inductive logic programming. In Mieczyslaw A. Klopotek, Slawomir T. Wierzchon, and Krzysztof Trojanowski, editors, *Intelligent Information Processing and Web Mining. Proceedings of the International IIS: IIPWM '04 Conference held in Zakopane, Poland*, Advances in Soft Computing, pages 493–502, Berlin, 2004. Springer.

(Nerbonne & Kretzschmar 03) John Nerbonne and William Kretzschmar, editors. *Computational Methods in Dialectometry*, volume 37 (3). 2003. Special Iss. of *Computers and the Humanities*.

(Nerbonne & Kretzschmar 06) John Nerbonne and William Kretzschmar, editors. *Progress in Dialectometry*, volume 21. 2006. Special Issue of *Literary and Linguistic Computing*, accepted to appear in 2006.

(Nerbonne & Stoianov 04) John Nerbonne and Ivilin Stoianov. Learning phonotactics with simple processors. In Dicky Gilbers, Maartje Schreuder, and Nienke Knevel, editors, *On the Boundaries of Phonology and Phonetics*, pages 89–121. CLCG, Groningen, 2004.

(Nerbonne 03) John Nerbonne. Linguistic variation and computation. In *Proceedings of the 10th Meeting of the European Chapter of the Association for Computational Linguistics*, volume 10, pages 3–10, 2003.

(Nerbonne *et al.* 99) John Nerbonne, Wilbert Heeringa, and Peter Kleiweg. Edit distance and dialect proximity. In David Sankoff and Joseph Kruskal, editors, *Time Warps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison, 2nd ed.*, pages v–xv. CSLI, Stanford, CA, 1999.

(Niyogi & Berwick 96) Partha Niyogi and Robert C. Berwick. A language learning model for finite parmeter spaces. *Cognition*, 61:161–193, 1996.

(Pinker 94) Steven Pinker. *The Language Instinct*. W. Morrow and Co., New York, 1994.

(Pollard 93) Carl Pollard. On formal grammars and empirical linguistics. In Andreas Kathol and M. Bernstein, editors, *ESCOL '93: Proc. of the 10th Eastern States Conference on Linguistics*, Columbus, 1993. Ohio State University.

(Porter 80) Martin F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

(Saffran *et al.* 99) Jenny R. Saffran, E.K. Johnson, Richard N. Aslin, and Elissa L. Newport. Statistical learning of tonal sequences by human infants and adults. *Cognition*, 70:27–52, 1999.

(Thomason & Kaufmann 88) Sarah Thomason and Terrence Kaufmann. *Language Contact, Creolization, and Genetic Linguistics*. University of California Press, Berkeley, 1988.

(van Coetsem 88) Frans van Coetsem. *Loan Phonology and the Two Transfer Types in Language Contact*. Publications in Language Sciences. Foris Publications, Dordrecht, 1988.

(van Noord 04) Gertjan van Noord. Error mining for wide-coverage grammar engineering. In *Proc. of 42nd Meeting of the Association for Computational Linguistics*, pages 446–453. ACL, Barcelona, 2004.

(Watson 96) Greg Watson. The Finnish-Australian English corpus. *ICAME Journal: Computers in English Linguistics*, 20:41–70, 1996.

(Yang 04) Charles Yang. Universal grammar, statistics or both? *Trends in Cognitive Science*, 8(10):451–456, 2004.

Comparison of Topic Identification methods for Arabic language

M. Abbas [1] and K. Smaili [2]
[1] CRSTDLA , Alger, Algérie
[2] INRIA-LORIA, Parole team B.P. 101 -
54602 Villers les Nancy, France
Tel.: +33 (0)3 83 59 20 22 - Fax: +33 (0)3 83 59 19 27
e-mail: smaili@loria.fr

Abstract

In this paper we present two well-known methods for topic identification. The first one is a TFIDF classifier approach, and the second one is a based machine learning approach which is called Support Vector Machines (SVM). In our knowledge, we do not know several works on Arabic topic identification. So that we decide to investigate in this article. The corpus we used is extracted from the daily Arabic newspaper *Akhbar Al Khaleej',* it includes 5120 news articles corresponding to 2.855.069 words covering four topics : sport, local news, international news and economy.
According to our experiments, the results are encouraging both for SVM and TFIDF classifier, however we have noticed the superiority of the SVM classifier and its high capability to distinguish topics.

1 Introduction

Topic identification has several applications : documents categorization, selecting documents for WEB engines, speech recognition systems, etc. State-of-the-art continuous speech recognition systems suffer from various problems. In unrestricted speech recognition process the vocabulary has to be as large as possible. Increasing the vocabulary increases the search space and results in performance degradation. A language model is one of the knowledge source which is used by an automatic speech recognition system in order to find the best hypotheses respecting linguistic criteria. One way to improve the results of a speech recognition system is to adapt the language model in accordance to the concerned utterance context. The problem of topic adaptation has already been largely addressed. In (Martin et al., 1997), (M. Mahajan and Huang, 1999),(Yang, 1999), (Bigi et al., 2000), (Bigiet al., 2001), (Brun et al., 2002), topic information is exploited in different ways, resulting each time in a significant reduction of the perplexity of the baseline language model and in sometimes in an improvement of the word error. Hence, these studies highlight the importance of topic adaptation.

Topic Identification is a supervised learning task consisting in identifying the topic of a text among a set of predefined topics. There is no formal definition of this concept. In what follows, a topic is viewed as a subset of the language associated to particular events. A document will be considered to be on a particular topic whenever its content is connected to the associated event. In this article we present the performance of two classifiers: TFIDF and SVM which are evaluated on Arabic corpus extracted from *Akhbar Al Khaleej.* To our knowledge, topic identification for Arabic is very little covered, that is why our purpose in this article is to highliht this subject. We begin by presenting the specificity of Arabic language, then we give details about the two methods used in this paper and we present the results.

1.1 An overview of Arabic morphology

Arabic is a semitic language which is written from right to left, unlike Latin languages. An Arabia word may be composed of a stem, prefix and suffix. The stem is composed of a root and pattern morphemes. The suffix can be composed of several sub-suffixes including inflectional markers for tense, gender, and/or numbers. The prefix includes zero or several sub-prefixes as some prepositions, conjunctions, determiners, possessive pronouns and pronouns. Most Arabic morphemes are defined by three consonants, to which various affixes can be attached to create a word. For example, from the tri-consonant "ktb" **كتب**

, we can inflect several different words concerning the idea of writing as presented in Table 1

There are many, many other derivations from this stem. The following example gives an idea about the different morphological segments existing in the word, and shows their equivalent in English:

And by her relations → **وبعلاقاتها**

Arabic	English
كَتَبَ	wrote
كِتَاب	book
كُتُب	books
يَكتُبُ	he writes
سَيَكتُبُ	he will write
كَاتِب	author
...	...

Table 1: An example of an Arabic word

Arabic	English
وَ	and
بِ	by
علَاقَاتِ	relations
هَا	her

Table 2: An example of an Arabic word

وَ بِعَلَاقَاتِهَا →

The example cited above shows that an Arabic word may correspond to several English words. Because of the variability of prefixes and suffixes, the morphological analysis is an important step in Arabic text processing. This makes segmentation of Arabic textual data different and more difficult than Latin languages. In the following, we developed a tool which split a word into prefixes, stem and suffixes. Some prefixes and stems have been kept, the suffixes have been removed for topic identification. This is due to the fact that we need only the sub-words which are meaningful for this task. have to be represented.

1.2 Documents representation

To process the documents, we have to build internal representations by transforming a document d to compact vector form. This operation is generally done after the tokenization of the corpus as explained in the previous section. The dimension of the vector corresponds to the number of distinct words or tokens in the training set. Each entry in the vector represents the weight of each term. For our purpose, after removing the non content words, we calculated both the frequency of each word, which is called Term Frequency, and the documents frequency of a word, that means the number of documents in which the word w occurs at least once. A general vocabulary is based on the word frequencies extracted from the Arabic newspaper corpus *Akhbar Al Khaleej* which contains 5120 news articles corresponding to more than 2.8 million of words. The first vocabulary contains 103706 distinct words, and finally the vocabulary used included all the words which appear more than 2. This leads to a vocabulary of 42877.

2 Topic Detection

Given a set of topics $T_1, T_2, \ldots, T_k$, the topic detection task consists in finding the topic(s) treated in a piece of text W (paragraph, article, ...).

Topic identification is based on topic training corpora, which represent the specificities of each topic. Given a text W, we want to identify the topic treated in this text. To do that, its specificities are compared with the ones of each topic.

3 The TFIDF classifier

The idea of this algorithm is to represent each document d as a vector $D = (d_1, d_2, \ldots, d_v)$ in a vector space. The vector elements are calculated as the combination of the term frequency $TF(w, d)$, which is the number of times the word w occurs in the document d, and the inverse document frequency $IDF(w)$ (Salton, 1991; Seymore and Rosenfeld, 1997).

$DF(w)$ is the number of documents in which the word w occurs at least once.
The value d_i is called the weight of word w_i in document d, and is given by the relation:
$d_i = TF(w, d) * IDF(w)$ with $IDF(w) = log(\frac{N}{DF(w)}$ N is the total number of documents.
To calculate the similarity between a document D_i and D_j we used the equation 1:

$$Sim(D_j, D_i) = \frac{\sum_{k=1}^{|V|} d_{jk} d_{ik}}{\sqrt{\sum_{k=1}^{|V|} (d_{jk})^2 \sum_{k=1}^{|V|} (d_{ik})^2}} \quad (1)$$

Topic	Training	Distinct words
International	755000	15078
Economy	578000	21108
Local	893000	17213
Sports	628000	13632

Table 3: Training corpora by topic

	Recall	Precision	F1
International news	97.65	99.2	98.42
Local news	85.94	79.71	82.71
Economy	85.15	85.82	85.84
Sport	94.53	100	97.19

Table 4: The performance of the TFIDF classifier

An article is assigned to the topic which gives the highest similarity.

4 The SVM method

The well known SVMs (Support Vector Machines) introduced by V. Vapnik (Vapnik, 1995) achieve biclass categorization. They have the advantage of being robust where it can handle a large number of features with good generalization performance. Another advantage of the SVM classifier is its capability to work with real and large-scale data. Basic SVM algorithm is able to recognize two different types of objects (vectors). The algorithm offers to do classification by building hyperplane in the R^N vector space and checking at which side found each vector. This operation may be described by a linear decision function: $f(x) = \sum_{i=1}^{n} w_i * x_i + b$ with w vector orthogonal to hyperplane and b distance from hyperplane to the origin. To decide to which class x belongs, one has to study the sign of the decision function $y = sgn(f(x)$. Since text categorization has been shown to be a linear problem (Joachims, 1998), and since exploratory research with other kernels did not yield performance improvements, we use only linear kernels. The SVM classification was performed with SVM^{light} (Joachims, 1998)

5 Experiments

In this section the TFIDF classifier and the SVM method are evaluated on real data extracted from an Arabic daily newspaper. We used 5120 articles, 90% of this corpus have been reserved for training and the rest for test. Table 5 summarizes the number of words for each topic and the number of words kept for a topic representation [1]

All the experiments presented in the next sections have been evaluated by the well-known measures : recall, precision and F1 given below.

[1] all the words occurred more than 3 times

$$\text{Recall} = \frac{\text{Nb texts correctly labelled}}{\text{Nb texts of topic}}$$

$$\text{Precision} = \frac{\text{Nb texts correctly labelled}}{\text{Nb texts labelled}}$$

$$F_1 = \frac{2*\text{Recall}*\text{Precision}}{\text{Recall}+\text{Precision}}$$

5.1 The TFIDF classifier

We withdrawn the non content words. In addition, we removed the words occurring less than 3 times. Consequently, each document is represented by a vector of 42877 words. The table 4 presents the recall, precision and F1 measure values for the four topics :

The best result is obtained for the international news and followed by sport news.

5.2 The SVM method

The Joachims tool SVM^{light} is used in our experiments for biclass discrimination. We used 1152 articles from each topic for training and 128 articles for test. Training consists of presenting positive and negative data. The negative data in our experiments consists of any other topic different from the one we want to learn. In all the experiments, we kept the same number of articles for positive and negative data. The table 5.2 shows respectively the values of recall, precision and F1 measure. This table shows that the SVM gives good results for Arabic topic identification. In fact, International news topic is well discriminated. It is never confused with Economy and Sport and reciprocally. In less than 1% of cases it is confused with local news topic. It is clear from this table that local news topic is the one which is slightly confused with all the other topics even with sport which could be considered as a very special. This topic has to be splited to more precise sub-topics. Table 6 shows the decrimination between a specific topic and a mixture of the the three other topics. This leads to the same conclusion, the Arabic topics are well discriminated.

To give an idea about the performances of both

	International			Local			Economy			Sport		
Topic	Rec	Prec	F_1	Rec	Prec	F_1	Rec	Prec	F_1	Rec	Prec	F_1
International	-	-	-	99.22	100	99.61	100	99.22	99.61	100	100	100
Local	99.22	100	99.61	-	-	-	89.06	92.68	90.83	97.66	99.21	98.43
Economy	100	99.22	99.61	89.06	92.68	90.83	-	-	-	97.66	100	98.81
Sport	100	100	100	97.66	99.21	98.43	97.66	100	98.81	-	-	-

Table 5: Recall, precision and F_1 for SVM biclass discrimination

	Recall	Precision	F1
International news	99.21	100	99.60
Local news	89.68	93.39	91.49
Economy	96.03	91.67	93.79
Sport	96.83	100	98.39

Table 6: SVM discrimination between a topic and topic mixtures

	Recall	Precision	F1 measure
TFIDF	90.82	91.18	90.95
SVM	97.26	98.52	97.88

Table 7: The mean values of recall, precision and F1

methods (SVM, TFIDF), we summarized the values of recall, precision and F1 measure, from the previous tables, in the table 7. We can conclude that SVM overcomes the results of TFIDF classifier for Arabic topic identification even if we showed in other works (Brun et al., 2002) that SVM is not the best method for classification. Neverthless, for Arabic language and with 4 topics the SVM performance are very interesting and important.

6 Conclusion

In this work we investigated topic identification for Arabic language, two well-known methods have been tested : TFIDF and SVM. The SVM methods achieves very high results 97.88 in terms of F_1. This method shows its capability to discriminate topics. Some of the studied topics are distinguished very easily. The SVM classifier outperforms the results obtained by TFIDF by more than 7.5% in terms of F_1 measure. As presented in (Yang, 1999), it would be interesting to study the methods performance according to the size of training data. This study is under work, we have now to increase the number of topics for Arabic and to compare the results obtained with those we achieved for French with other methods (Bigi et al., 2001). The idea is to try to understand if these methods are sensitive to the language.

References

B. Bigi, R. De Mori, M. El-Bèze, and T. Spriet. 2000. A fuzzy decision strategy for topic identification and dynamic selection of language models. *Special Issue on Fuzzy Logic in Signal Processing, Signal Processing Journal*, 80(6).

B. Bigi, A. Brun, J.P. Haton, K. Smaïli, and I. Zitouni. 2001. Dynamic topic identification: Towards combination of methods. In *Recent Advances in Natural Language Processing (RANLP)*, pages 255–257, Tzigov Chark, Bulgarie.

A. Brun, K. Smaïli, and J.P. Haton. 2002. Contribution to topic identification by using word similarity. In *International Conference on Spoken Language Processing (ICSLP2002)*.

T. Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *European Conference on Machine Learning (ECML)*, pages 137–142.

D. Beeferman M. Mahajan and X. Huang. 1999. Improved topic-dependent language modeling using information retrieval techniques. In *Proc. of the Int. Conf. on Acoustics, Speech, and Signal Processing.*

S. Martin, J. Liermann, and H. Ney. 1997. Adaptive topic-dependent language modelling using word-based varigrams. In *Proceedings 3rd European Conference on Speech Communication and Technolog.*

G. Salton. 1991. Developments in Automatic Text Retrieval. *Science*, 253:974–979.

K. Seymore and R. Rosenfeld. 1997. Using Story Topics for Language Model Adaptation. In *Proceeding of the European Conference on Speech Communication and Technology.*

V. Vapnik. 1995. *The Nature of Statistical Learning Theory.* Spinger, New York.

Yiming Yang. 1999. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1-2):69–90.

Summarizing Reports on Evolving Events; Part I: Linear Evolution

Stergos D. Afantenos and **Vangelis Karkaletsis**
National Center for Scientific Research (NCSR) "Demokritos", Greece
{stergos,vangelis}@iit.demokritos.gr

Panagiotis Stamatopoulos
Department of Informatics, University of Athens, Greece
takis@di.uoa.gr

Abstract

We present an approach for summarization from multiple documents which report on events that evolve through time, taking into account the different document sources. We distinguish the evolution of an event into *linear* and *non-linear*. According to our approach, each document is represented by a collection of messages which are then used in order to instantiate the cross-document relations that determine the summary content. The paper presents the summarization system that implements this approach through a case study on linear evolution.

1 Introduction

With the advent of the Internet, access to many sources of information has now become much more easier. One problem that arises though from this fact is that of the information overflow. Imagine, for example, that someone wants to keep track of an event that is being described on various news sources, over the Internet, as it evolves through time. The problem is that there exist a plethora of news sources that it becomes very difficult for someone to compare the different versions of the story in each source. Furthermore, the Internet has made it possible now to have a rapid report of the news, almost immediately after they become available. Thus, in many situations it is extremely difficult to follow the rate with which the news are being reported. In such cases, a text summarizing the reports from various sources on the same event, would be handy. In this paper we are concerned with the automatic creation of summaries from multiple documents which describe an event that evolves through time. Such a collection of documents usually contains news reports from various sources, each of which provides novel information on the event as it evolves through time. In many cases the sources will agree on the events that they report and in some others they will adopt a different viewpoint presenting a slightly different version of the events or possibly disagreeing with each other. Such a collection of documents can, for example, be the result of a Topic Detection and Tracking system (Allan *et al.* 98).

The identification of similarities and differences between the documents is a major aspect in Multi-document Summarization (Mani 01; Afantenos *et al.* 05a; Afantenos *et al.* 05b). (Mani & Bloedorn 99), for example, identify similarities and differences among *pairs* of isolated documents by comparing the graphs that they derive from each document, which are based heavily on various lexical criteria. Our approach, in contrast, does not take into consideration isolated pairs of documents, but instead tries to identify the similarities and differences that exist between the documents, taking into account the time that the incidents occurred and the document source. This enables us to distinguish the document relations into *synchronic* and *diachronic* ones. In the synchronic level we try to identify the similarities and differences that exist between the various sources. In the diachronic level, on the other hand, we try to identify similarities and differences across time focusing on each source separately.

Another twofold distinction that we made through our study (Afantenos *et al.* 05b) concerns the type of evolution of an event, distinguishing between *linear* and *non-linear* evolution, and the rate of emission of the various news sources, distinguishing between *synchronous* and *asynchronous* emission of reports. Figure 1 depicts the major incidents for two different events: a linearly evolving event with synchronous emission and a non-linearly evolving one with asynchronous emission of reports. Whereas in the linearly evolving events the main incidents happen in constant and possibly predictable quanta of time,[1] in the non-linear events we can make no predictions as to when the next incident will occur. As you can see in Figure 1 we can have within a small amount of time an explosion of incidents followed by a long time of sparse incidents, etc.

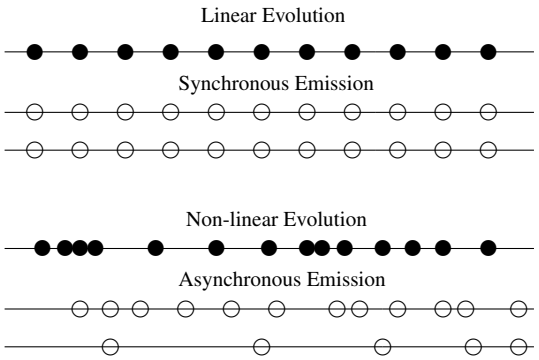


Figure 1: Linear and Non-linear evolution

In order to represent the various incidents that are described in each document, we introduce the notion of *messages*. Messages are composed from a name, which reflects the type of the incidents, and a list of arguments, which take their values from the domain ontology. Additionally, they

[1] This means that if the first news story q_0 comes at moment t_0, then we can assume that for each source the story q_n will come at time $t_n = t_0 + n * t$, where t is the constant amount of time that it takes for the news to appear.

have associated with them the *time* that the message refers to, as well as the document *source*.

The distinction between linear and non-linear evolution affects mainly the *synchronic relations*, which are used in order to identify the similarities and differences between two messages from different sources, at about the same time. In the case of linear evolution all the sources report in the same time. Thus, in most of the cases, the incidents described in each document refer to the time that the document was published. Yet, in some cases we might have temporal expressions in the text that modify the time that a message refers to. In such cases, before establishing a synchronic relation, we should associate this message with the appropriate time-tag. In the case of non-linear evolution, each source reports at irregular intervals, possibly mentioning incidents that happened long before the publication of the article, and which another source might have already mentioned in an article published earlier. In this case we shouldn't rely any more to the publication of an article, but instead rely on the *time* tag that the messages have (see section 2). Once this has been performed, we should then establish a *time window* in which we should consider the messages, and thus the relations, as synchronic.

In the following section, we make more concrete and formal the notion of the messages and relations. In section 3 we briefly present our methodology and describe its implementation through a particular case study. Section 4 presents in more detail the related work, and section 5 concludes presenting ongoing work on *non-linear* summarization and our future plans.

2 Some Definitions

In our approach (Afantenos *et al.* 05b; Afantenos *et al.* 04) the major building blocks for representing the knowledge on a specific event are: the *ontology* which encodes the basic entity types (concepts) and their instances; the *messages* for representing the various incidents inside the document; and the *relations* that connect those messages across the documents. More details are given below.

Ontology. For the purposes of our work, a domain ontology should be built. The ontology we use is a taxonomic one, incorporating *is-a* relations, which are later exploited by the messages and the relations.

Messages. In order to capture what is represented by several textual units, we introduce the notion of *messages*. A message is composed from four parts: its *type*, a list of *arguments* which take their values from the concepts of the domain *ontology*, the *time* that the message refers, and the *source* of the document that the message is contained. In other words, a message can be defined as follows:

$$\texttt{message\_type (arg}_1\texttt{, ... , arg}_n\texttt{)}$$
$$\text{where } \texttt{arg}_i \in \text{Domain Ontology}$$

Each message m is accompanied by the time ($m.\texttt{time}$) that it refers and its source ($m.\texttt{source}$). Concerning the source, this is inherited by the source of the document that contains the message. Concerning the time of the message, it is inherited by the publication time of the document, unless there exists a temporal expression in the text that modifies the time that a message refers. In this case, we should interpret the time-tag of the message, in relation to that temporal expression. A message definition may also be accompanied by a set of *constraints* on the values that the arguments can take. We would like also to note that messages are similar structures (although simpler ones) with the templates used in the MUC.[2] An example of a message definition will be given in the case study we present in section 3.

Relations. In order to define a relation in a domain we have to provide a *name* for it, and describe the conditions under which it will hold. The name of the relation is in fact *pragmatic* information, which we will be able to exploit later during the generation of the summary. The conditions that a relation holds are simply some rules which describe the *temporal distance* that two messages should have (0 for synchronic and more than 1 for diachronic) and the characteristics that the arguments of the messages should exhibit in order for the relation to hold.

Furthermore, it is crucial to note here the importance that time and source position have on the relations, apart from the values of the messages' arguments. Suppose, for example, that we have two identical messages. If they have the same temporal tag, but belong to different sources, then we have an *agreement* relation. If, on the other hand, they come from the same source but they have chronological distance one, then we speak of a *stability* relation. Finally, if they come from different sources and they have chronological distance more than two, then we have no relation at all. We also do not have a relation if the messages have different sources and different chronological distances. Thus we see that, apart from the characteristics that the arguments of a message pair should exhibit, the source and temporal distance also play a role for that pair to be characterized as a relation. In section 3 we will give concrete examples of messages and relations for a particular case study.

3 A Case Study of Linear Evolution

The methodology was originally presented in (Afantenos *et al.* 04). It involves four stages:

1. Corpus collection
2. Creation of a domain ontology
3. Specification of the messages
4. Specification of the relations

The topic we have chosen is that of the descriptions of football matches. In this domain, we have several events that evolve; for example, the performance of a player or

[2] http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_toc.html

a team as the championship progresses. According to the definitions we have given, the evolution of this domain is *linear*. The reason for this is that we have a match each week which is then being described by several sources.

As our methodology requires, in order to create multi-document summaries of evolving events, we have to provide some knowledge of the domain to the system. This knowledge is provided through the ontology and the specification of the messages and the relations, following the four steps described above.

3.1 Domain Knowledge

Corpus Collection. We manually collected descriptions of football matches, from various sources, for the period 2002-2003 of the Greek football championship. The language used in the documents was also Greek. This championship contained 30 rounds. We focused on the matches of a certain team, which were described by three sources. So, we had in total 90 documents.

Ontology Creation. An excerpt of the taxonomic ontology we have created is shown in Figure 2.

```
Degree                Card
Person                  Yellow
  Referee               Red
  Assistant Referee   Team
  Linesman            Temporal Concept
  Coach                 Minute
  Player                Duration
  Spectators              First Half
    Viewers               Second Half
    Organized Fans        Delays
Round                     Whole Match
```

Figure 2: An excerpt from the domain ontology

Messages' Specifications. We concentrated in the most important events, that is on events that evolve through time, or events that a user would be interested in knowing. At the end of this process we concluded on the following set of 23 message types:

Absent, Behavior, Block, Card, Change, Comeback, Conditions, Expectations, Final_Score, Foul, Goal_Cancelation, Hope_For, Injured, Opportunity_Lost, Penalty, Performance, Refereeship, Satisfaction, Scorer, Successive_Victories, Superior, System_Selection, Win

An example of full message specifications is shown in Figure 3. We should note that this particular message type is not accompanied by constraints. Also, associated with it we have the `time` and `source` tags.

performance (entity, in_what, time_span, value)

entity	:	player or team
in_what	:	Action Area
time_span	:	Minute or Duration
value	:	Degree

Figure 3: An example of message specifications

Specification of the Relations. We identified twelve cross-document relations, six on the synchronic and six on the diachronic axis (see Table 1).

Diachronic Relations	***Synchronic Relations***
– POSITIVE GRADUATION	– AGREEMENT
– NEGATIVE GRADUATION	– NEAR AGREEMENT
– STABILITY	– DISAGREEMENT
– REPETITION	– ELABORATION
– CONTINUATION	– GENERALIZATION
– GENERALIZATION	– PRECISENESS

Table 1: Synchronic and Diachronic Relations in the Football Domain

Since this was a pilot-study during which we examined mostly the feasibility of our methodology, we limited the study of the cross-document relations, in those ones that connect the *same* message types. Thus both the synchronic and the diachronic relations connect the same types, although further studies might reveal that different message types can be connected with some sort of relations. Furthermore, concerning the *diachronic* relations we limited our study in relations that have chronological distance only one.[3] Examples of such specifications for the message type `performance` are shown in Figure 4.

Performance
Assuming we have the following two messages:

$\text{performance}_1\ (\text{entity}_1, \text{in\_what}_1, \text{time\_span}_1, \text{value}_1)$
$\text{performance}_2\ (\text{entity}_2, \text{in\_what}_2, \text{time\_span}_2, \text{value}_2)$

Then we have a Diachronic relation if

$(\text{performance}_1\texttt{.time} < \text{performance}_2\texttt{.time})$ and
$(\text{performance}_1\texttt{.source} = \text{performance}_2\texttt{.source})$

and a Synchronic relation if

$(\text{performance}_1\texttt{.time} = \text{performance}_2\texttt{.time})$ and
$(\text{performance}_1\texttt{.source} \neq \text{performance}_2\texttt{.source})$

More specifically, we have the following Synchronic and Diachronic relations:

Diachronic Relations

- **Positive Graduation** iff
  $(\text{entity}_1 = \text{entity}_2)$ and $(\text{in\_what}_1 = \text{in\_what}_2)$ and $(\text{time\_span}_1 = \text{time\_span}_2)$ and $(\text{value}_1 < \text{value}_2)$
- **Stability** iff
  $(\text{entity}_1 = \text{entity}_2)$ and $(\text{in\_what}_1 = \text{in\_what}_2)$ and $(\text{time\_span}_1 = \text{time\_span}_2)$ and $(\text{value}_1 = \text{value}_2)$
- **Negative Graduation** iff
  $(\text{entity}_1 = \text{entity}_2)$ and $(\text{in\_what}_1 = \text{in\_what}_2)$ and $(\text{time\_span}_1 = \text{time\_span}_2)$ and $(\text{value}_1 > \text{value}_2)$

Synchronic Relations

- **Agreement** iff
  $(\text{entity}_1 = \text{entity}_2)$ and $(\text{in\_what}_1 = \text{in\_what}_2)$ and $(\text{time\_span}_1 = \text{time\_span}_2)$ and $(\text{value}_1 = \text{value}_2)$
- **Near Agreement** iff
  $(\text{entity}_1 = \text{entity}_2)$ and $(\text{in\_what}_1 = \text{in\_what}_2)$ and $(\text{time\_span}_1 = \text{time\_span}_2)$ and $(\text{value}_1 \approx \text{value}_2)$
- **Disagreement** iff
  $(\text{entity}_1 = \text{entity}_2)$ and $(\text{in\_what}_1 = \text{in\_what}_2)$ and $(\text{time\_span}_1 = \text{time\_span}_2)$ and $(\text{value}_1 \neq \text{value}_2)$

Figure 4: Specifications of Relations

[3] Chronological distance zero makes the relations synchronic.

A question that can arise is the following: *How does time affect the relations you create?* To answer that question, imagine having two identical messages, in different documents. If the documents have chronological distance zero, then we have an *agreement* relation. If the messages come from the same source but have chronological distance 1, then we have a *stability* relation. Finally, if the messages come from different sources and have chronological distance more than one, then we have no relation at all. Thus, indeed, time does affect the relations.

An Example At this point we would like to give a more concrete example. Two sources, A and B, for a particular match, describe the performance of a player as follows:

A The performance of Nalitzis, for the whole match was mediocre.

B In general, we can say that Nalitzis performed modestly, throughout the match.

The messages that represent those two sentences are the following:

A `performance (Nalitzis, general, whole_match, 50)`

B `performance (Nalitzis, general, whole_match, 50)`

The number 50 represents the mediocre performance of the player, since the `degree` is realized as an integer in the scale of 0 to 100. According to the specifications of the relations (see Figure 4) we would have an *Agreement* synchronic relations between those two messages. In the next game we have the following description:

A Nalitzis shown an excellent performance throughout the game.

The message that results from this sentence is the following:

A `performance (Nalitzis, general, whole_match, 100)`

Now, between the two messages from source A we have a *Positive Graduation* diachronic relation.

3.2 The System

Our summarization system is a query-based one, since the summary is an answer to a natural language query that a user has posed. Such queries concern the evolution of several events in the domain. In order to create the summaries we have to extract, from the documents, the messages with their arguments, and the relations that connect them, and subsequently organize them into a structure which we call a *grid* (see Figure 5). This grid reflects exactly the fact that the domain that we have used in this case study exhibits linear evolution. If we take a horizontal "slice" of the grid, then we will have descriptions of events from all the sources, for a particular time unit. If, on the other hand, we take a vertical "slice" of the grid, then we have the description of the evolution of an event from a particular source.

In order to extract the messages from the documents, our system employs an Information Extraction (IE) subcomponent. Relations between the messages are identified according to the conditions associated with each one. After the user has issued the query, the system identifies the various messages that are relevant to this query, as well as the relations that connect them. Thus, in essence the system *extracts a subgrid* from the original grid which is, in fact, the answer to the user query. This subgrid is passed to a Natural Language Generation (NLG) subcomponent which creates the final summary.

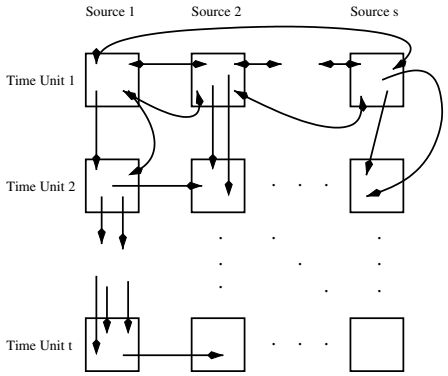


Figure 5: The Grid structure with Synchronic and Diachronic Relations

3.2.1 Messages Extraction

This subsystem was developed using the *Ellogon* text engineering platform.[4] Its architecture is depicted in Figure 6. It involves the following processing stages.

Preprocessing. This stage includes the tokenization, sentence splitting and the Named Entity Recognition and Classification (NERC) sub-stages. During NERC, we try to identify the Named Entities (NEs) in the documents and classify them into the categories that the ontology proposes.

The next two processing stages are the core of message extraction. In the first one we try to identify the *type* of each extracted message, while in the second we try to fill its *argument values*.

Message Classification. Concerning the identification of the message types, we approached it as a classification problem. From a study that we carried out, we concluded that in most of the cases the mapping from sentences to messages was one-to-one, *i.e.* in most of the cases one sentence corresponded to one message. Of course, there were cases in which one message was spanning more than one sentence, or that one sentence was containing more than one message. We managed to deal with such cases during the arguments' filling stage.

In order to perform our experiments we used a bag-of-words approach according to which we represented each sentence as a vector from which the stop-words and the words with low frequencies (four or less) were removed. The features used are divided into two categories: *lexical* and *semantic*. As lexical features we used the words of the

[4] `www.ellogon.org`

Figure 6: The message extraction subsystem

sentences both stemmed and unstemmed. As semantic features we used the *NE types* that appear in the sentence. Of course, in order to perform the training phase of the experiments, in each of the vectors we appended the *class* of the sentence, *i.e.* the type of message; in case a sentence did not corresponded to a message we labeled that vector as belonging to the class *None*. This resulted into *four* series of vectors and corresponding experiments that we performed.

In order to perform the classification experiments we used the WEKA platform (Witten & Frank 00). The Machine Learning algorithms that we used where three: *Naïve Bayes, LogitBoost* and *SMO*. For the last two algorithms, apart from the default configuration, we performed some more experiments concerning several of their arguments. Thus for the LogiBoost we experimented with the number of iterations that the algorithm performs and for the SMO we experimented with the complexity constant, with the exponent for the polynomial kernel and with the gamma for the RBF kernel. For each of the above combinations we performed a *ten-fold cross-validation* with the annotated corpora that we had. The results of the above experiments are presented in Table 2.

Taking a look at that table there are several remarks that we can make. Firstly, the LogitBoost and the SMO classifiers that we used outperformed, in all the cases, the Naïve Bayes which was our baseline classifier. Secondly, the inclusion of the NE types in the vectors gave a considerable enhancement to the performance of all the classifiers. This is rather logical, since almost all the messages contain in their arguments NEs. The third remark, concerns the stemmed and the unstemmed results. As we can see from the table, the algorithms that used vectors which contained unstemmed words outperformed the corresponding algorithms which used vectors whose words had been stemmed. This is rather counterintuitive, since in most of the cases using stemming one has better results.

Ultimately, the algorithm that gave the best results, in the experiments we performed, was the SMO with the default configuration for the unstemmed vectors which included information on the NE types. This classifier managed to correctly classify 2974 out of 3735 messages (including the *None* class) or about 80% of the messages. Thus, we integrated this trained classifier in the message extraction subsystem, which you can see in Figure 6.

Arguments' Filling In order to perform this stage we employed several domain-specific heuristics. Those heuristics take into account the constraints of the messages, if they do have. As we noted above, one of the drawbacks of our classification approach is that there are some cases in which we do not have a one-to-one mapping from sentences to messages. During this stage of message extraction we used heuristics to handle many of these cases.

In Table 3 we show the final performance of the subsystem as a whole, when compared against manually annotated messages on the corpora used. Those measures concern only the message types. As you can see from that table although the vast majority of the messages extracted are correct, these represent 68% of all the messages.

Precision	:	91.1178
Recall	:	67.7810
F-Measure	:	77.7357

Table 3: Evaluation of the messages' extraction stage

3.2.2 Extraction of Relations

As is evident from Figure 4, once we have identified the messages in each document and we have placed them in the appropriate position in the grid, then it is fairly straightforward, through their specifications, to identify the cross-document relations among the messages.

In order to achieve that, we implemented a system which was written in Java. This system takes as input the extracted messages with their arguments from the previous subsystem and it is responsible for the incorporation of the ontology, the representation of the messages and the extraction of the synchronic and diachronic cross-document relations. Ultimately, through this system we manage to represent the *grid*, which carries an essential role for our summarization approach.

The reason for this is that since our approach is a query based one, we would like to be able to pose queries and get the answers from the grid. The system that we have created implements the API through which one can pose queries to the grid, as well as the mechanism that extracts from the whole grid structure the appropriate messages and the relations that accompany them, which form an answer to the question. Those extracted messages and relations form a sub-grid which can then be passed to an NLG system for the final creation of the summary.

Concerning the statistics of the extracted relation, these are presented in Table 4. The fact that we have lower statistical measures on the relations, in comparison with the message types, can be attributed to the argument extraction subsystem, which does not perform as well as the message classification subsystem.

Precision	:	89.0521
Recall	:	39.1789
F-Measure	:	54.4168

Table 4: Recall, Precision and F-Measure on the relations

Classifier		Correctly Classified Instances	Classifier		Correctly Classified Instances
Without NE types			Including NE types		
stemmed	Naïve Bayes	60.6693 %	stemmed	Naïve Bayes	63.8286 %
	LogitBoost default	72.7443 %		LogitBoost default	78.0991 %
	LogitBoost $I = 5$	71.8876 %		LogitBoost $I = 5$	76.1981 %
	LogitBoost $I = 15$	72.2892 %		LogitBoost $I = 15$	78.2062 %
	SMO default	73.6011 %		SMO default	75.9839 %
	SMO $C = 0.5$ $E = 0.5$ $G = 0.001$	68.9692 %		SMO $C = 0.5$ $E = 0.5$ $G = 0.001$	72.5301 %
	SMO $C = 1.5$ $E = 1.5$ $G = 0.1$	74.4578 %		SMO $C = 1.5$ $E = 1.5$ $G = 0.1$	75.7965 %
unstemmed	Naïve Bayes	62.2758 %	unstemmed	Naïve Bayes	64.2035 %
	LogitBoost default	75.8768 %		LogitBoost default	78.9023 %
	LogitBoost $I = 5$	74.9398 %		LogitBoost $I = 5$	77.4565 %
	LogitBoost $I = 15$	76.6533 %		LogitBoost $I = 15$	79.4645 %
	SMO default	79.2503 %		SMO default	79.6252 %
	SMO $C = 0.5$ $E = 0.5$ $G = 0.001$	75.2343 %		SMO $C = 0.5$ $E = 0.5$ $G = 0.001$	76.8675 %
	SMO $C = 1.5$ $E = 1.5$ $G = 0.1$	77.9920 %		SMO $C = 1.5$ $E = 1.5$ $G = 0.1$	78.5007 %

Table 2: The results from the classification experiments

As of writing this paper, everything has been implemented except the mechanism that transforms the natural language queries to the API that will extract the subgrid. Additionally, we do not have a connection with an NLG system, but instead we have implemented some simple template-based mechanism.

4 Related Work

The work that we present in this paper is concerned with multi-document summarization of events that evolve through time. Of course, we are not the first to incorporate directly, or indirectly, the notion of time in our approaches to summarization. (Lehnert 81), for example, attempts to provide a theory for what she calls *narrative summarization*. Her approach is based on the notion of "plot units", which connect *mental states* with several relations, and are combined into very complex patterns. This approach is a single-document one and was not implemented. Recently, (Mani 04) attempts to revive this theory of narrative summarization, although he also does not provide any concrete computational approach for its implementation.

From a different viewpoint, (Allan *et al.* 01) attempt what they call *temporal summarization*. In order to achieve that, they take the results from a Topic Detection and Tracking system for an event, and they put all the sentences one after the other in a chronological order, regardless of the document that it belonged, creating a stream of sentences. Then they apply two statistical measures *usefulness* and *novelty* to each ordered sentence. The aim is to extract those sentences which have a score over a certain threshold. This approach does not take into account the document sources, and it is not concerned with the evolution of the events; instead they try to capture novel information.

As we have said, our work requires some domain knowledge which is expressed through the ontology, and the messages' and relations' specifications. A system which is based also on domain knowledge is SUMMONS (Radev & McKeown 98; Radev 99). The domain knowledge for this system comes from the specifications of the MUC conferences. This system takes as input several MUC templates and, applying a series of operators, it tries to create a baseline summary, which is then enhanced by various named entity descriptions collected from the Internet. One can argue that the operators that SUMMONS uses resemble our cross-document relations. This is a superficial resemblance, since our relations are divided into *synchronic* and *diachronic*, thus reporting similarities and differences in two different directions: source and time. Additionally our system is a query-based one.

Concerning now the use of relations, (Salton *et al.* 97) for example, try to extract paragraphs from a single document by representing them as vectors and assigning a relation between the vectors if their similarity exceeds a certain threshold. Then, they present various heuristics for the extraction of the best paragraphs.

Finally, (Radev 00) proposed the Cross-document Structure Theory (CST) which incorporated a set of 24 domain independent relations that exist between various textual units across documents. In a later paper (Zhang *et al.* 02) reduce that set into 17 relations and perform some experiments with human judges. Those experiments reveal several interesting results. For example, human judges annotate only sentences, ignoring the other textual units (phrases, paragraphs, documents) that the theory suggests. Additionally, we see a rather small inter-judge agreement concerning the type of relation that connects two sentences. (Zhang *et al.* 03) and (Zhang & Radev 04) continue the work with some experiments, during which they use Machine Learning techniques to identify the cross-document relations. We have to note here that although a general *pool* of cross-document relations might exist, we believe, in contrast with (Radev 00), that those relations are dependent on the domain, in the sense that one can choose from this pool the appropriate subset of relations for the domain under consideration, possibly enhancing this subset with completely domain specific relations that will suit ones needs. Another significant difference from our work, is that our main goal is to create summaries that show the evolution of an event, as well as the similarities or differences that the sources have during the evolution of an event.

5 Conclusions and Future Work

The aim of this paper was to present our approach to the problem of *multi-document summarization of evolving events*. We divide the evolution of the events into linear and non-linear. In order to tackle the problem, we

introduced cross-document relations which represent the evolution of the events in two axes: *synchronic* and *diachronic*. Those relations connect messages, which represent the main events of the domain, and are dependent on the domain ontology. We also presented, through a case study, an implementation for a linearly evolving domain, namely that of the descriptions of football matches. The system we have built automatically extracts the messages and the synchronic and diachronic relations from the text. A particular point of concern is the recall (approximately 40%) of the relations' extraction sub-system, which is due to the heuristics used for the filling the arguments of the messages. Apart from enhancing our heuristics, we also plan to study their effect on the quality of the generated summary.

Currently we are working on a more complicated domain, namely that of events with hostages, whose evolution, according to the specification that we gave in the introduction of this paper, can be characterized as non-linear. The main challenges in non-linear evolution concern the synchronic relations. A related problem, which we investigate, is that of the *temporal expressions* which may make several messages refer back in time, in relation to the publication time of the article that contains the messages. We also examine in depth the role that time has on the relations. Additionally, we examine the existence of relations between different message types. Concerning now the classification experiments and the argument extraction, we intend to enhance them by adding more semantic features incorporating also the Greek WordNet.[5]

References

(Afantenos *et al.* 04) Stergos D. Afantenos, Irene Doura, Eleni Kapellou, and Vangelis Karkaletsis. Exploiting cross-document relations for multi-document evolving summarization. In G. A. Vouros and T. Panayiotopoulos, editors, *Methods and Applications of Artificial Intelligence: Third Hellenic Conference on AI, SETN 2004*, volume 3025 of *Lecture Notes in Computer Science*, pages 410–419, Samos, Greece, May 2004. Springer-Verlag Heidelberg.

(Afantenos *et al.* 05a) Stergos D. Afantenos, Vangelis Karkaletsis, and Panagiotis Stamatopoulos. Summarization from medical documents: A survey. *Journal of Artificial Intelligence in Medicine*, 33(2):157–177, February 2005.

(Afantenos *et al.* 05b) Stergos D. Afantenos, Konstantina Liontou, Maria Salapata, and Vangelis Karkaletsis. An introduction to the summarization of evolving events: Linear and non-linear evolution. In *Natural Language Understanding and Cognitive Science NLUCS - 2005*, pages 91–99, Maiami, USA, May 2005.

(Allan *et al.* 98) James Allan, Jaime Carbonell, George Doddington, Jonathan Yamron, and Yiming Yang. Topic detection and tracking pilot study: Final report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, pages 194–218, February 1998.

(Allan *et al.* 01) James Allan, Rahuk Gupta, and Vikas Khandelwal. Temporal summaries of news stories. In *Proceedings of the ACM SIGIR 2001 Conference*, pages 10–18, 2001.

(Lehnert 81) Wendy G. Lehnert. Plot units: A narrative summarization strategy. In W. G. Lehnert and M. H. Ringle, editors, *Strategies for Natural Language Processing*, pages 223–244. Erlbaum, Hillsdale, New Jersey, 1981.

(Mani & Bloedorn 99) Inderjeet Mani and Eric Bloedorn. Summarizing similarities and differences among related documents. *Information Retrieval*, 1(1):1–23, 1999.

(Mani 01) Inderjeet Mani. *Automatic Summarization*, volume 3 of *Natural Language Processing*. John Benjamins Publishing Company, Amsterdam/Philadelphia, 2001.

[5] `www.ceid.upatras.gr/Balkanet/resources.htm`

(Mani 04) Inderjeet Mani. Narrative summarization. *Journal Traitement Automatique des Langues (TAL): Special issue on "Le résumé automatique de texte: solutions et perspectives"*, 45(1), Fall 2004.

(Radev & McKeown 98) Dragomir R. Radev and Kathleen R. McKeown. Generating natural language summaries from multiple on-line sources. *Computational Linguistics*, 24(3):469–500, September 1998.

(Radev 99) Dragomir R. Radev. *Generating Natural Language Summaries from Multiple On-Line Sources: Language Reuse and Regeneration*. Unpublished PhD thesis, Columbia University, 1999.

(Radev 00) Dragomir R. Radev. A common theory of information fusion from multiple text sources, step one: Cross-document structure. In *Proceedings of the 1st ACL SIGDIAL Workshop on Discourse and Dialogue*, Hong Kong, October 2000.

(Salton *et al.* 97) Gerald Salton, Amit Singhal, Mandar Mitra, and Chris Buckley. Automatic text structuring and summarization. *Information Processing and Management*, 33(2):193–207, 1997.

(Witten & Frank 00) Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, San Francisco, 2000.

(Zhang & Radev 04) Zhu Zhang and Dragomir Radev. Learning cross-document structural relationships using both labeled and unlabeled data. In *Proceedings of IJC-NLP 2004*, Hainan Island, China, March 2004.

(Zhang *et al.* 02) Zhu Zhang, Sasha Blair-Goldensohn, and Dragomir Radev. Towards cst-enhanced summarization. In *Proceedings of AAAI-2002*, August 2002.

(Zhang *et al.* 03) Zhu Zhang, Jahna Otterbacher, and Dragomir Radev. Learning cross-document structural relationships using boosting. In *Proccedings of the Twelfth International Conference on Information and Knowledge Management CIKM 2003*, pages 124–130, New Orleans, Louisiana, USA, November 2003.

Truecasing For The Portage System

Akakpo Agbago, Roland Kuhn, George Foster
Institute for Information Technology, National Research Council of Canada
{Akakpo.Agbago, Roland.Kuhn, George.Foster}@nrc-cnrc.gc.ca

Abstract

This paper presents a truecasing technique - that is, a technique for restoring the normal case form to an all lowercased or partially cased text. The technique uses a combination of statistical components, including an N-gram language model, a case mapping model, and a specialized language model for unknown words. The system is also capable of distinguishing between "title" and "non-title" lines, and can apply different statistical models to each type of line. The system was trained on the data taken from the English portion of the Canadian parliamentary Hansard corpus and on some English-language texts taken from a corpus of China-related stories; it was tested on a separate set of texts from the China-related corpus. The system achieved 96% case accuracy when the China-related test corpus had been completely lowercased; this represents 80% relative error rate reduction over the unigram baseline technique. Subsequently, our technique was implemented as a module called Portage-Truecasing inside a machine translation system called Portage, and its effect on the overall performance of Portage was tested. In this paper, we explore the truecasing concept, and then we explain the models used.

1. Introduction

Many natural language processing engines output text that lacks case information – by convention, usually in lowercase. For instance, Portage-Truecasing is incorporated in a machine translation system called Portage whose initial translations are generated in lowercase format. Thus, to complete the translation task, the system needs a truecasing module that will change some of the characters in the initial translation to uppercase. Systems that carry out named entity recognition, spelling correction, and grammar correction may also require truecasing modules to function properly.

To illustrate the use of truecasing, consider the following example. Let us assume that an automatic speech recognition or machine translation system outputs the sentence *"sir john a macdonald drank old covenanter whiskey"*. The sentence is much easier to read and to understand in its truecase form: *"Sir John A MacDonald drank Old Covenanter whiskey"*. In this version, "Sir John A MacDonald" and "Old Covenanter" are clearly understood to be names. (After truecasing, the typical next step is punctuation insertion).

Few people have worked on this problem. The most recent papers are by Chelba and Acero [2] and by Lita *et al.* [5]. Chelba and Acero's technique is based on maximum "a posteriori" (MAP) adaptation of Maximum Entropy Markov Models (MEMMs) to solve this problem. These authors obtained a 35-40% relative improvement for the baseline MEMM over a 1-gram baseline, and a 20-25% relative improvement for the MAP-adapted MEMM over the baseline MEMM (in tests done on Broadcast News data). Lita *et al*. [5] used a truecasing approach based on trigram language modeling. They obtained relative error rate improvement over a unigram baseline of about 50% (from 96% accuracy to 98%) on a news articles from which titles, headlines, and section headers had been excluded, and an even greater relative error rate improvement of about 66% (from about 94% accuracy to about 98%) over the baseline on a test corpus comprising titles, headlines, and section headers. Finally, Mikheev's work [1] targeted the parts of a text where capitalization is expected, such as beginning of sentences and quotations. Similarly, Kim and Woodland [3] used rule-based techniques to generate punctuation and beginning of sentence capitalization for speech recognition outputs.

We began by implementing a unigram baseline system that yielded 19.35% case error; implementation of a trigram-related model similar to that of Lita *et al*. lowered this to 5.24% (relative error rate reduction of 73%). Careful study of the problems seen on a development set showed that many of the errors came from titles, and from "unknown" words – *i.e*., those encountered during testing but not during training. Thus, we extended the basic approach by incorporating a **title**

detector which attempts to label lines as being either "title" or "non-title". This gives us the option of training separate title and non-title casing models for application at runtime. In addition, we grouped "unknown" words into four classes. For each such class, the case probabilities are determined from the cases of low-frequency words in the training data that fall into that class.

The language models described in this paper were trained using the SRI Language Modeling Toolkit (SRILM). Since one of the goals of this work was to improve the performance of a machine translation (MT) system participating in a NIST MT task, much of the training data was drawn from the 2004 NIST "Large" Chinese-English training corpus. This "C/E" corpus includes texts from a variety of China-related sources. Additional training material was drawn from the Canadian parliamentary Hansard corpus. The test data were the 2004 NIST C/E evaluation data.

The metric employed for the C/E MT NIST task is BLEU (see Papineni et al. [4]), which measures the similarity of the translation system's output with one or more reference translations. In this paper, we measure the performance of the truecasing module both by how accurately it assigns case to normal text that has had case information removed, and by its effect on BLEU. We define "case accuracy" per word - a case error in a single character of a word is counted as a case error for that word. The goal of optimizing performance according to one of these metrics may conflict with optimizing performance according to the other. Suppose that the MT system outputs "elephants in africa mostly has long nose" and the truecasing module converts this to "elephants in Africa mostly has long nose". We might be tempted to add a rule to the truecasing module that imposes uppercase for the first letter in every sentence. Though this rule might help performance according to the "case accuracy" metric, it may hurt the BLEU score. In the example, if the reference sentence were "Most elephants in Africa have long noses", BLEU will assign a higher score to "elephants in Africa mostly has long nose" than to "Elephants in Africa mostly has long nose" (because the form of "elephants" in the reference is all-lowercase).

The layout of this paper is as follows: section 2 will outline the problem, section 3 will describe the statistical models, section 4 will describe the experiments and their results, and section 5 will discuss these results.

2. The Problem of Truecasing

The truecasing problem is not obvious until one faces a real example. Consider the sentence "*indian paratroopers will command a joint alpha-tango military exercise with the special forces of the us pacific command*". In languages employing the Latin alphabet, a sentence typically begins with uppercase. Therefore, "indian" should be "Indian" with little ambiguity. The word "us" could remain lowercase but the word sequence "us pacific command" suggests that the all-uppercase form "US" is more likely. Thus, word context can provide clues to case. In a syntactic approach, some aspects of context could be exploited by means of Part-Of-Speech (POS) tagging. For instance, the tagger might tag "the us pacific command" as "the <noun phrase>" and use the information that "us" is part of a noun phrase to generate "the US Pacific Command".

At the beginning of our work on truecasing, we investigated the distribution of the casing errors of a unigram truecaser. This system, which was used as the baseline in subsequent experiments, assigns to words observed in the training data the most frequent case observed. New words seen in the test data for the first time – the so-called "unknown" words - are left in lowercase. The resulting error distribution is plotted below (with words of similar frequency in the training corpus binned together).

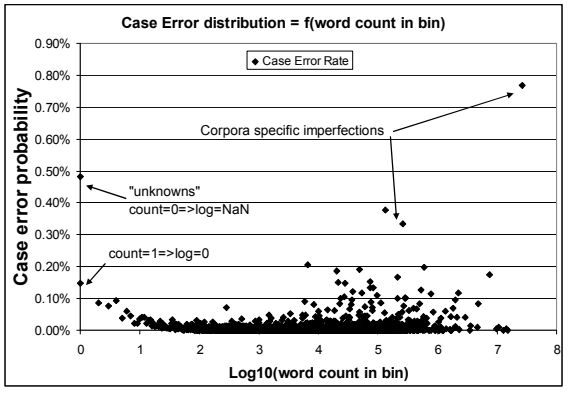


Figure 1: Case error distribution of the baseline truecaser as a function of word count in our training corpus

The point marked "unknowns" appears for convenience on the y axis (though its true x coordinate is not 0 but -∞); it represents words

appearing in the test data that were not seen in the training data. It is not surprising that these words have a higher case error rate than the words of count 1: the baseline system has not learned anything from the training data about the "unknown" words. At the high end of the x axis, we see that a few very frequent words such as "the" also have a high error rate. This is partly because of tokenization problems (*e.g.*, "the" sometimes has a hyphen glued to the end of it, or a quotation mark glued to the front of it) and partly because "the" and similar words often appear in titles, which are particularly tricky.

In the truecasing approach we used (similar to [5]), an N-gram language model (LM) is used to model the contextual information. In the example, if the trigram "US Pacific Command" has often been seen, then the system will be inclined to carry out truecasing correctly. The "case mapping" model smooths the N-gram model. If (for instance) the erroneous sequence "will Command a" occurred once in the training data, this smoothing ensures that an occurrence of "command" preceded by "will" and followed by "a" will still receive the correct all-lowercase form in the system's output.

We need a third kind of model to deal with "unknown" words – *i.e.*, those that were not observed in the training data. In the example, it is quite likely that no form of "*alpha-tango*" (a rare military code word) has been observed. Nevertheless, the "unknown word" model we provide will be capable of converting it to the correct form, "*Alpha-Tango*".

3. Scoring Function and Models

We gave in chapter 2 some motivations for three sub-models. To use the specific contribution of each sub-model, we combine them into a scoring function Ω formulated by Eq.1. The sub-models are:

- An N-gram model called θ_N to capture the contextual information surrounding a word;
- A case mapping model called Φ to capture the probabilities for different cases of a word;
- An "unknown word" model called Л to provide for unseen cases.

$$\Omega = \theta_N \otimes \Phi \otimes \Pi \qquad \text{Eq.1}$$

3.1 Terminology

Let S denote a sequence of words s_i, with case information included. Let C() denote the function that gives only the casing of a string, and L() the function that returns its lowercase form, thus leaving only information about the uncased word sequence. Let AU denote "All Uppercase", FU "First letter Uppercase", AL "All lowercase", and MC "Mixed Case"; for S = "USA is an acronym for United States of America", C(S) = AU AL AL AL AL FU FU AL FU, and L(S) = "usa is an acronym for united states of america". Truecasing is applied when we know L(S) and are trying to obtain C(S). If both the case information C(S) and the word information L(S) are known for a string S, S is completely defined.

3.2 N-Gram model θ_N

One way of estimating the probability that s_i has a particular case $C(s_i)$ would be to assume recursively that we already know the case of the words preceding a particular word s_i in the string S. This line of thought leads to the N-gram component θ_N of the truecaser. For instance, for N=3, let $P_{\theta 3}(C(s_i) \mid L(s_i), s_{i-2}s_{i-1})$ denote the probability that the $C(s_i)$ form of s_i (rather than some other form) occurs after the cased word sequence $s_{i-2}s_{i-1}$. An example: if $L(s_i)$ = "america", and that $s_{i-2}s_{i-1}$ = "States of", the trigram-related probability of "America" is $P_{\theta 3}(C(s_i)=FU \mid L(s_i)=$"america", "States of"$)$.

3.3 Case mapping model Φ

Another way of estimating the probability that s_i has a particular case $C(s_i)$ would be to ignore context and rely on the case forms observed in the training data for s_i. This leads to the case mapping model, $P_\Phi(C(s_i) \mid L(s_i))$. For example, the probability of "America" given that some form of "america" has occurred is denoted $P_\Phi(C(s_i)=FU| L(s_i)=$ "america"$)$. Using Φ alone would be equivalent to considering the most probable case pattern for a word everywhere. This sub-model is used to smooth the model θ_N.

3.4 Unknown word model Л

Finally, the sub-model called Л deals with words s_i that weren't observed in the training data. It was constructed by defining classes based on the form

of a word – for instance, the presence of non-word symbols (*e.g.*, internal hyphen). It's formulated as

$$P_{\Pi}(C(s_i) \mid L(s_i)) \approx P(C(s_i) \mid Class(L(s_i)))$$

The conditional probability on the right side above is calculated from the case statistics for words that belong to the class, and that occur exactly once in the training data. Our assumption is that low-frequency words in a given class tend to follow similar patterns of case.

How should the function *Class(L(s$_i$))* be defined? Depending on the test corpus, the nature of such "unknown" words may vary. They include rare proper names such as "agbago" and mixed alphanumeric expressions such as "$2563US" or "675km" or "220kV". Other forms are compounded name entities and character sequences resulting from words in non-alphabetic languages. This last type of "unknown" word sometimes occurs in the English portion of the C/E corpus when Chinese characters have been inserted in English text (*e.g.*, to clarify the meaning of an English word to Chinese readers).

Based on the characteristics of the C/E corpus, we decided to define the following "unknown" word classes:

1. *quantity words*: "unknown" tokens starting or/and ending with numbers. Example: "us$0.19", "10kV", "rmb0.308".
2. *acronyms*: "unknown" tokens containing a sequence of single letters followed by periods. Example: "u.s.", "u.s.-south".
3. *hyphenated words*: "unknown" tokens made up of at least two components joined by a hyphen, where each component consists of a sequence of alphabetic characters. Example: "belarus-russian", "jong-il".
4. *regular uniform words*: "unknown" tokens consisting entirely of alphabetic characters. Example: "abesie", "badeshire".

These classes are considered in the precedence order just given. Thus, an unknown token is only considered for class 2 if it has been rejected for class 1, and so on (that's why "u.s.-south" is assigned to class 2 and not class 3). "Unknown" tokens not falling into one of these four classes are left in all-lowercase form (an example is the "unknown" token "*cafâ¨ä¦*" we observed during our tests, which results from a word that combines alphabetic letters and Chinese characters).

3.5 Scoring function Ω

The θ_N , Φ and Л components of Portage's truecasing module (defined above) are true probabilities. The scoring function Ω combines them in the following way:

$$\Omega(C(s_i) \mid L(s_i)) \approx \begin{cases} P_{\Pi}(C(s_i) \mid L(s_i)), & if\ s_i\ unknown \\ P_{\theta_N}(C(s_i) \mid L(s_i), s_{i-1}s_{i-2}) & \\ \quad * P_{\Phi}(C(s_i) \mid L(s_i)), & else \end{cases}$$

Although Ω defined in this way is not a probability because of the product term, it has certain advantages (*e.g.*, ease of implementation in the SRILM framework). The way Ω is formulated indicates that at the step i, we already know the case of the words preceding s_i in the string S. To get a sense of how Ω works, consider the following training text:

"*Akakpo is the son of Agbago. So his name is said and written as Akakpo Agbago in Canada but Akakpo AGBAGO in Togo. Akakpo AGBAGO is unique in Togo. Akakpo is a last name for many. Agbago is a good guy. Agbago is smart. Agbago is kind.*"

And the following test text:

"Akakpo agbago"

Let's redefine the θ component slightly so it's based on bigrams rather than trigrams, and let's ignore smoothing and assume the component models use frequencies directly to estimate probabilities.

Then using these training and testing texts, we obtain:

Step i = 1:

$$\begin{aligned} \Omega(Akakpo \mid akakpo) &= P_{\theta_N}(Akakpo \mid Akakpo) \\ &\quad * P_{\Phi}(Akakpo \mid akakpo) \\ &= 1 * 1 \\ &= 1 \quad "akakpo"\ is\ known \end{aligned}$$

$$\hat{C}(s_1) = \text{Akakpo}$$

Step i = 2:

$$\Omega(\text{Agbago} \mid \text{agbago, Akakpo}) =$$
$$P_{\theta_N}(\text{Agbago} \mid \text{agbago, Akakpo})$$
$$* P_{\Phi}(\text{Agbago} \mid \text{agbago})$$
$$= \frac{1}{3} * \frac{5}{7} = \frac{5}{21}$$
$$\Omega(\text{AGBAGO} \mid \text{agbago, Akakpo}) =$$
$$P_{\theta_N}(\text{AGBAGO} \mid \text{agbago, Akakpo})$$
$$* P_{\Phi}(\text{AGBAGO} \mid \text{agbago})$$
$$= \frac{2}{3} * \frac{2}{7} = \frac{4}{21} \quad \text{"agbago" is known}$$
$$\hat{C}(s_2) = \text{Akakpo Agbago}$$

Thus, the scoring function Ω, if trained on this corpus, would tend to predict "Agbago" rather than "AGBAGO" after "Akakpo". This prediction is incorrect in Togo, but correct in Canada (and most of the English-speaking world) – an example of how slippery the notion of correct casing can get.

We also tried a different approach in which we find the cased form that maximizes the trigram probability, given the lowercase form and the two preceding cased forms. Let S denote the entire cased word sequence, and L the corresponding sequence of lowercased words. By Bayes's Law, we have

$$P(S \mid L) = P(L \mid S) * P(S) / P(L)$$

However, by definition we know the lowercase word sequence L. Thus, we want to maximize

$$P(S \mid L) \propto P(L \mid S) * P(S)$$

Substituting in the trigram estimate of $P(s_i)$, we see that at each step we are trying to maximize

$$P(L(s_i) \mid s_{i-2}, s_{i-1}, s_i) * P(s_i \mid s_{i-2}, s_{i-1})$$

Thus, we search over the cased forms s_i of $L(s_i)$ observed in the training data to find the one that maximizes this expression. For an observed form s_i of $L(s_i)$, $P(L(s_i)|s_{i-2},s_{i-1},s_i)$ will be 1. In initial experiments, this approach yielded inferior performance to that obtained by using the scoring function Ω above.

4. Experiments and Results

We used the SRILM package, along with some code we wrote ourselves, to handle the training (creation of the language models) and the case decoding (also called "disambiguation"). The θ models are produced in the ARPA N-gram LM format and the Φ and Л models in SRILM "V1 to V2" mapping format.

The resources used were as follows:
- Training corpus: contains 366,532,578 tokens (~words).
- Test corpus: contains 451,154 tokens (~words)

Recall that the training corpus comes from the English-language half of the 2004 NIST "Large" Chinese-English (C/E) training corpus (which includes material from Hong Kong Hansard and news sources such as Xinhua News Agency, Associated Press, Agence Française de Press, *etc.),* supplemented by material from the Canadian Hansard corpus. The test corpus is the 2004 NIST C/E test set.

To the scoring function Ω described above, we added some **heuristics**. These are:

- Junk cleaning: we removed from the training data various special tags; also, all lines in which most words are uppercased (these turned out to be extremely atypical).
- Title detection and processing: titles show unusual casing patterns. Unfortunately, in English there are no explicit rules for casing in titles; frequently, casing is left to the whims of the author. We implemented a title detection module that relied on domain specific aspects of our data, which consisted mainly of newswire data. For instance, the presence of a date, name of a news agency, and "reported by" followed by a personal name was taken to indicate a title. Used on training data, this module makes it possible to train "body only" or "title only" models; used on test, it makes it possible to apply different models or rules to title and body. The best-performing system shown in Figure 2 was trained only on the portion of the training corpus classified as "body" by the title detector. For casing of test text, this body-only system was applied both to portions of the test corpus classified as "body" and as "title". Then, words in the title that were longer than four letters were systematically uppercased. It might seem more logical to use a model trained on titles to assign case to words

in titles, but the main characteristic of titles in the training text is inconsistency in case assignment. Thus, the title detector's usefulness for training is that it enables us to **remove** titles from the training data.

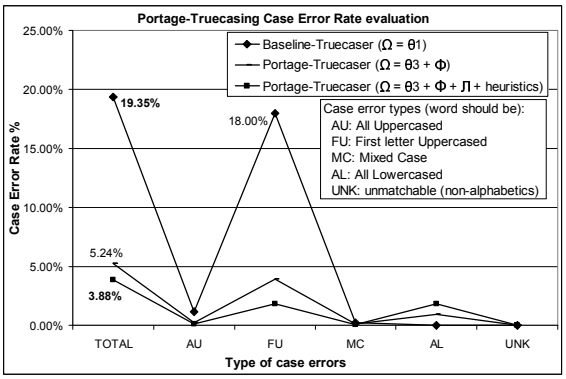


Figure 2: performance of Portage-Truecasing

The performance of Portage-Truecasing is plotted in Figure 2 and shows 80% improvement in relative error rate over the unigram baseline technique, $\Omega = \theta_1$ (from 19.35% to 3.88%). The figure also shows case error by the correct case type – *e.g.*, the points above "AU" show error rates for words that **should** be written all-uppercase. From the figure, it is clear that the effort that went into classifying different types of unknown words for the Л model and into developing heuristics (junk cleaning and title detection) was justified: it yielded an improvement of 26% relative (from 5.24% to 3.88% error). If we do **not** include the heuristics but only Л, the improvement is only 13% relative (to 4.55% case error – this point is not shown in the figure 2). Figure 3 provides an analysis of Portage-Truecasing errors by word count, as was done in Figure 1 for the baseline truecaser.

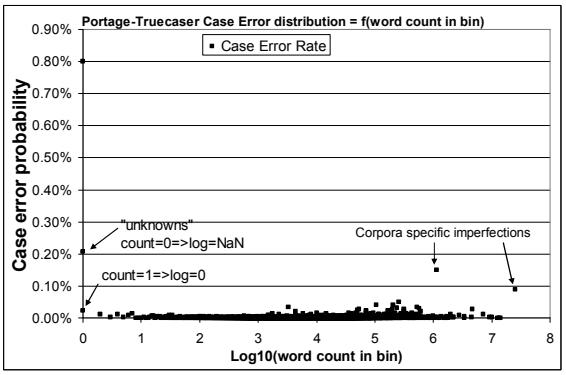


Figure 3: Case error distribution of Portage-Truecaser as a function of word count in our training corpus

Since the module is used for machine translation (MT), we ran it on the output of the MT system and obtained the BLEU results in Table 1. The last two columns show that the unknown word model Л helps performance, while the use of heuristics causes slight deterioration. As noted earlier, it is not surprising that the best-performing system according to case error rate (this system includes the heuristics) is not the same as the best-performing system according to BLEU (this system does not include the heuristics). For most applications, the case error rate is more informative.

Baseline-Truecaser ($\Omega = \theta_1$)	Portage-Truecaser ($\Omega=\theta_3+\Phi$)	Portage-Truecaser ($\Omega=\theta_3+\Phi$+Л +heuristics)	Portage-Truecaser ($\Omega=\theta_3+\Phi$+Л)
17.98%	23.77%	23.74%	23.83%

Table 1: BLEU score

5. Discussion

In this paper, we presented a module designed as part of a machine translation system that uses three statistical language models to assign case to a text. It reduces the case error rate of a unigram baseline truecaser by 80% relative, achieving 96% global accuracy on the test corpus. We designed the system in a manner that allowed us to quickly test different variants; this was fortunate, because the best variant according to case error rate and the best variant according to BLEU turned out to be different.

Some specific problems we encountered were:

- *Inconsistency*: there was a non-negligible proportion of case inconsistencies in training and test corpora. This happens because the corpora are agglomerations of texts written by different people with different formatting styles, competences, and working tools. Furthermore, not much attention is paid to enforcing casing standards, even where these exist. Named entities (e.g., "United States Government" *vs.* "United States government") and titles tend to be subject to casing inconsistency.
- *Portage specific side effects*: errors from other components of the system, particularly the tokenizer, had a strong negative impact on performance.

For future work, we could consider turning scoring function Ω into a true probability by interpolating the θ_N and Φ terms instead of multiplying them – *i.e.*, defining it as:

$$\Omega(C(s_i) \mid L(s_i)) \approx \begin{cases} P_{\Pi}(C(s_i) \mid L(s_i)), & \text{if } s_i \text{ unknown} \\ \lambda * P_{\theta_N}(C(s_i) \mid L(s_i), s_{i-1}s_{i-2}) + \\ (1-\lambda) * P_{\Phi}(C(s_i) \mid L(s_i)), & \text{else} \end{cases}$$

Alternatively, we could approximately keep Ω in its current form, but incorporate power terms α and β that would depend on the frequency of a word (where K is a normalization factor):

$$\Omega(C(s_i) \mid L(s_i)) \approx \begin{cases} P_{\Pi}(C(s_i) \mid L(s_i)), & \text{if } s_i \text{ unknown} \\ P_{\theta_N}(C(s_i) \mid L(s_i), s_{i-1}s_{i-2})^{\alpha} \\ \quad * P_{\Phi}(C(s_i) \mid L(s_i))^{\beta} / K, & \text{else} \end{cases}$$

It is interesting to think about how one would build a truecaser optimized for MT (*i.e.*, to maximize the BLEU score). MT output is not exactly the same as regular text. One might consider training the truecaser on output from the MT system whose words have been assigned case in some other way (*e.g.*, by "pasting" onto them case patterns from corresponding words in reference data).

References:

[1] A. Mikheev, "*A knowledge-free method for capitalized word disambiguation*", 37th conference on Association for Computational Linguistics, pp. 159 – 166, 1999, College Park, Maryland, ISBN.

[2] C. Chelba and A. Acero, "*Adaptation of Maximum Entropy Capitalizer: Little Data Can Help a Lot*", Conf. on Empirical Methods in Natural Language Processing (EMNLP) July 2004, Barcelona, Spain.

[3] J.H. Kim and P. C. Woodland, "*Automatic Capitalization Generation for Speech Input*", Computer Speech and Language, 18(1):67–90, January 2004.

[4] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "*BLEU: A method for automatic evaluation of machine translation*", IBM Technical Report RC22176, Sept. 2001.

[5] L. Lita, A. Ittycheriah, S. Roukos, and N. Kambhatla, "*tRuEcasIng*", Proceedings of ACL 2003, pp. 152–159, Sapporo, Japan.

Exploring feature spaces with svd and unlabeled data for Word Sense Disambiguation

Eneko Agirre
IXA NLP Group
Univ. of the Basque Country
Donostia, 20018
e.agirre@ehu.es

Oier Lopez de Lacalle
IXA NLP Group
Univ. of the Basque Country
Donostia, 20018
jibloleo@si.ehu.es

David Martínez
IXA NLP Group
Univ. of the Basque Country
Donostia, 20018
davidm@si.ehu.es

Abstract

Current Word Sense Disambiguation systems suffer from the lack of hand-tagged data, as well as performance degradation when moving to other domains. In this paper we explore three different improvements to state-of-the-art systems: 1) using Singular Value Decomposition in order to find correlations among features, trying to deal with sparsity, 2) using unlabeled data from a corpus related to the evaluation corpus, and 3) splitting the feature space into smaller, more coherent, sets. Each of the proposals improves the results, and properly combined they achieve the best results to date for the Senseval 3 lexical sample dataset. The analysis of the results provides further insights and possibilities for the future.

1 Introduction

Many current Natural Language Processing (NLP) systems rely on linguistic knowledge acquired from tagged text via Machine Learning (ML) methods. Statistical or alternative models are learned, and then applied to running text. The main problem faced by such systems is the sparse data problem, due to the small amount of training examples. Focusing on Word Sense Disambiguation (WSD), only a handful of occurrences with sense tags are available per word. For example, if we take the word *channel*, we see that it occurs 5 times in SemCor (Miller *et al.* 93), the only all-words sense-tagged corpus publicly available: the first sense has four occurrences, the second a single occurrence, and the other 5 senses are not represented. For a few words, more extensive training data exists: The Lexical Sample task of Senseval-2 (Edmonds & Cotton 01) provides 145 occurrences for *channel*, but still some of the senses are represented by only 3 or 5 occurrences.

In addition to the sparse data problem, supervised WSD systems are usually trained and tested in texts coming from the same corpus. When training and testing instances come from distinct sources with domain or genre differences, the performance typically drops accordingly (Martínez & Agirre 00).

The impact of the above problems (sparsity and domain shifts) is exemplified by the frustrating handful of systems which are able to beat the simple Most Frequent Sense baseline in the realistic all-words task in both Senseval-2 and Senseval-3 (Snyder & Palmer 04). In these exercises the best systems were trained over SemCor, and the test texts came from The Wall Street Journal and the Brown corpus.

One possible solution to the above problems is to use unlabeled data and appropriate learning techniques that can take advantage of them. Unlabeled data might alleviate the need of hand-labeled data, and, in addition help to adapt the system to new domains. Recently, there have been several attempts in the WSD literature which use co-training (Mihalcea 04) and Principal Component Analysis (Su *et al.* 04). The results have been mixed, with some improvements over baseline supervised systems, but still below the best purely supervised system in the Senseval lexical sample tasks. An exception is (Gliozzo *et al.* 05), which improves the best Senseval-3 results using a combination of kernels and domain information modeled with Singular Value Decomposition (SVD). This last system is closely related to ours, and we will highlight the differences in the related work section.

Alternatively, there is also the preoccupation about the best way to apply ML techniques to supervised settings. The first issue is to represent the context with appropriate features. The last Senseval exercises show that the more feature types one throws into the algorithm, the better are the results (Agirre & Martínez 04). Still, it is not clear which is the best way to profit from the very rich feature space. Apart from the sparsity problem already mentioned, large feature spaces tend to have highly redundant and heterogeneous features (see Section 2.2). As a potential solution,

we interpret that SVD (cf. Section 3.1) collapses similar features (i.e. having similar distributions), and will thus be helpful against sparsity and redundancy. Regarding heterogeneity, splitting the feature space might allow the learning algorithm to better capture the patterns in the data.

In this paper we explore three different ways to improve feature modeling:

- Using SVD in order to find correlations among features, trying to deal with sparsity.
- Using unlabeled data from a corpus related to the evaluation corpus coupled with SVD as above.
- Splitting the feature space into smaller, more coherent, sets, trying to better model the feature space.

These improvements need to be combined with state-of-the-art ML algorithms. The methods based on the spatial representation of features (such as Support Vector Machines, Vector Space Models and k-Nearest Neighbors) seem to be the best performing, and we have focused on them (cf. Section 2.3)

We will show that each of the modifications in the feature space improves the results, and properly combined they achieve the best results to date for the Senseval 3 lexical sample dataset. The analysis of the results will provide further insights and possibilities for the future.

The paper is structured as follows. Section 2 reviews the experimental setting and state-of-the-art WSD systems that we used as baselines, including the feature set and ML methods used. Section 3 introduces the improvements proposed in this paper. Section 4 presents the results of these improvements. Section 5 introduces the combination method and its results. Section 6 presents the discussion and related work. Finally, Section 7 draws the conclusions and the future work.

2 Experimental setting and baseline systems

In order to organize the experiments we started building state-of-the-art WSD systems based on our previous experience (Agirre & Martínez 04). In the next sections we will present briefly the main components of the WSD system, that is, the features used to represent the context and the ML algorithms applied. But we first describe the target WSD task and the evaluation methodology.

2.1 Corpus and evaluation

The experiments have been performed using the Senseval-3 English Lexical-Sample data (Mihalcea *et al.* 04). The source corpora was the BNC (Leech 92). WordNet 1.7.1. (Fellbaum 98) was chosen as the sense inventory for nouns and adjectives, while the verb senses came from the *Wordsmyth* dictionary[1]. 57 words (nouns, verbs, and adjectives) were tagged, with 7,860 instances for training and 3,944 for testing.

For the development and fine-tuning of our systems, we have used 3-fold cross validation over the training set, where the three folds were built following stratified sampling. The final evaluation and the comparison with other systems was made over the testing set. The usual precision and recall figures were computed for each system. In all the cases reported here coverage was 100% and precision equalled recall, so we use recall in all tables.

2.2 Features

The feature types can be grouped in three main sets:

Local collocations: bigrams and trigrams formed with the words around the target. These features are constituted by lemmas, word-forms, or PoS tags[2]. Other local features are those formed with the previous/posterior lemma/word-form in the context.

Syntactic dependencies: syntactic dependencies were extracted using heuristic patterns, and regular expressions defined with the PoS tags around the target[3]. The following relations were used: object, subject, noun-modifier, preposition, and sibling.

Bag-of-words features: we extract the lemmas of the content words in the whole context, and in a $\pm$4-word window around the target. We also obtain salient bigrams in the context, with the methods and the software described in (Pedersen 01).

2.3 ML methods

Given an occurrence of a word, the ML methods below return a weight for each sense ($weight(s_k)$). The sense with maximum weight will be selected.

[1] http://www.wordsmyth.net/

[2] The PoS tagging was performed with the fnTBL toolkit (Ngai & Florian 01).

[3] This software was kindly provided by David Yarowsky's group, from the Johns Hopkins University.

Each occurrence or instance is represented by the features found in the context (f_i).

For the **Vector Space Model** (VSM) method, we represent each occurrence context as a vector, where each feature will have a 1 or 0 value to indicate the occurrence/absence of the feature. For each sense in training, one centroid vector is obtained ($\vec{C_{s_k}}$). These centroids are compared with the vectors that represent testing examples ($\vec{f}$), by means of the cosine similarity function (eq. (1)). The closest centroid assigns its sense to the testing example.

$$weight(s_k) = cos(\vec{C}_{s_k}, \vec{f}) = \frac{\vec{C}_{s_k} \cdot \vec{f}}{|\vec{C}_{s_k}||\vec{f}|} \qquad (1)$$

Regarding **Support Vector Machines** (SVM) we utilized SVM-Light, a public distribution of SVM by (Joachims 99). The weight for each sense is given by the distance to the hyperplane that supports the classes, that is, the sense s_k versus the rest of senses.

The k **Nearest Neighbor** (k-NN) is a memory based learning method (eq. (2)), where the neighbors are the k most similar contexts, represented by feature vectors ($\vec{c_i}$), of the test vector ($\vec{f}$). The similarity among instances is measured by the cosine of their vectors (as in eq. (1)). The test instance is labeled with the sense obtaining the maximum the sum of the weighted vote of the k most similar contexts. The vote is weighted depending on its (neighbor) position in the ordered rank, with the closest being first. Eq. (2) formalizes k-NN, where C_i corresponds to the sense label of the i-th closest neighbor.

$$\arg\max_{S_j} = \sum_{i=1}^{k} \begin{cases} \frac{1}{i} & \text{if } C_i = S_j \\ 0 & \text{otherwise} \end{cases} \qquad (2)$$

3 Improvements for feature modeling

This section presents the three improvements that we propose here as solutions to the data sparsity, redundancy and heterogeneity problems. First, we present the use of SVD on the training and test sets. Next, we introduce unlabeled data into the SVD procedure. Finally, we split the feature space into two smaller sets.

3.1 Singular Value Decomposition (SVD)

SVD is a technique to reduce the dimensions of any problem represented by vectors. It has been widely used in Text Categorization, being the basis of Latent Semantic Analysis. SVD reduces the dimensionality of the feature vectors, finding correlations between features, and helping to deal with data sparseness. We will review briefly SVD as we applied it to WSD.

Let $C = \{t_1, t_2, ..., t_n\}$ be a corpus (set of occurrences of target word), where t_i is an instance from the training set. Let $F = \{f_1, f_2, ..., f_m\}$ be the features appeared in C, let $M \ni \mathbf{R}^{m \times n}$ be a feature-by-instance matrix representing C, where $t_{ij} \in M$ is the frequency of feature f_i in instance t_j. Each word in the Lexical Sample has its own M feature-by-instance matrix. Instead of the frequency, one can try more sophisticated weighting schemes, as we will see in Section 4.2.

SVD decomposes the feature-by-instance matrix (M) into the product of three matrices (eq. (3)):

$$M = U\Sigma V^T = \sum_{i=1}^{k=min\{m,n\}} \sigma_i u_i v i^T \qquad (3)$$

U and V, row and column matrix, respectively, have orthonormal columns and Σ is a diagonal matrix which contains k eigenvalues in descending order. Note that in WSD problems the number of instances is much lower than the number of features ($n << m$), so k is always equal to the number of instances. By selecting the first p eigenvalues, we reduce the current space to p dimensions, and can thus project the instances (both training and test) to a reduced space. The equation (4) shows how to make this projection, where $\vec{t}^T$ is the transpose of the vector of features corresponding to one occurrence of the target word.

$$\vec{t_p} = \vec{t}^T U_p \Sigma_p^{-1} \qquad (4)$$

Once we project all training and testing instances into the reduced space, we can apply any ML algorithm as usual. SVD has been performed with SVDPACK[4] and GTP[5]. VSM and SVM were fed with the results from SVDPACK and k-NN with the results of GTP.

3.2 Singular Value Decomposition with unlabeled data

The sense (label) of an instance is not used in the process of doing SVD. Taking advantage of this, we can use unlabeled data to have a larger

[4] http://www.netlib.org/svdpack

[5] http://wwww.cs.utk.edu/~lsi

matrix for each word, and hopefully obtain better correlations in the reduced space. We have used the BNC corpus to get large amounts of unlabeled instances, and thus augment the feature-by-instance matrix M from the previous section into M'. In our experiments we have tested different amounts of unlabeled data, trying with 25% or 50% of the occurrences of the word. We call this process **background learning**.

Once we have done the SVD decomposition of M' we obtain the new U' and $\Sigma_p'^{-1}$, we project training and testing instances as in eq. (4) and proceed applying any ML method.

3.3 Splitting feature space

As seen in Section 2.2, WSD uses a high number of heterogeneous features. The methods mentioned in 2.3 are all based on geometrical properties of the feature space. If we split the problem (the whole space of features) into more coherent feature sets, the classification algorithms should find easier its way in such a simple space. We can thus build separate classifiers for each set of features, and hopefully obtain better results.

In order to test this hypothesis we split the features (cf. Section 2.2) in two subsets:

- **Topical features**: Comprising the bag-of-word features.
- **Local features**: Comprising the local collocations and the syntactic dependencies.

4 Preliminary results

In this section we describe the results of the systems presented in the previous sections: we first comment the baseline methods, then some parameter tuning over SVD, and finally the improved algorithms.

4.1 Results of baseline methods

Initially we tried with k-NN, SVM and VSM (section 2.3). VSM has no parameters, but k-NN needs to find an optimal k (number of neighbors) and SVM allows to optimize the "soft margin". We used 3-fold cross-validation on the Senseval-3 Lexical Sample training set. For k-NN we only tried two values: $k = 5$ and $k = 4$. For SVM we used the "soft margin" value obtained in previous experiments.

Table 1 shows the results from cross-validation. We can see that the results of VSM and k-NN are very similar, with VSM outperforming k-NN for 0.3 points, and SVMperforming lower. For the rest of the paper, we set $k = 5$ for all uses of k-NN. The results on the test set are shown in Table 2, with VSM increasing its advantage over k-NN and SVM in the middle of both.

Classifiers	Recall
k-NN k=5	67.7
k-NN k=4	67.4
SVM	62.3
VSM	68.0

Table 1: Results for baseline classifiers in 3-fold cross-validation (Senseval-3 training set).

Classifiers	Recall
k-NN k=5	70.5
SVM	71.2
VSM	71.5

Table 2: Results for baseline classifiers in the Senseval-3 Lexical Sample test set.

4.2 Parameter setting for SVD

SVD needs to set several parameters which can affect the performance. In order to set those parameters we run several preliminary experiments using SVD coupled with k-NN using 3-fold cross-validation as before. In the rest of the paper, SVD was performed using the following parameters:

- Number of desired **dimensions**: We tried with 100, 200, 300, 500 and 1000 dimensions, and the best performance was obtained with 200 dimensions.
- **Weighting scheme** for the frequencies in the feature-by-instance matrix: We tried different classic schemes, including local weighting formulas such as term frequency (tf), *log* and *binary*, and global measures like *idf* and *entropy*. For this work we have used *log* and *entropy* weighting scheme, replacing $t_{ij} \in M$ (cf. Section 3.1) by $log(t_{ij}) \cdot entropy(i)$.
- **Threshold** for global frequency (g): After building the matrix we can remove features that are very common (the less informative). We tried with different thresholds, and finally we chose to accept all features ($g = 0$).

Classifiers	Recall
k-NN k=5	67.7
SVM	62.3
VSM	68.0
k-NN-SVD k=5	69.8
SVM-SVD	61.2
VSM-SVD	63.9

Table 3: Results for k-NN and VSM with SVD in 3-fold cross-validation (Senseval-3 training set).

k-NN ($k = 5$)	Recall	diff.
plain	67.7	—
local+topical	69.4	+1.7
SVD	69.6	+1.9
SVD (25% BNC)	69.2	+1.5
SVD (50% BNC)	69.6	+1.9

Table 4: Improved k-NN classifier in 3-fold cross-validation (Senseval-3 training set). Plain stands for baseline k-NN.

Classifiers	Recall	diff.
plain	70.5	—
local+topical	70.8	+0.3
SVD	70.7	+0.2
SVD (25% BNC)	70.8	+0.2
SVD (50% BNC)	71.2	+0.7
VSM	71.5	+1.0
SVM	71.2	+0.7

Table 5: Improved k-NN classifier in the Senseval-3 Lexical Sample test set. Plain stands for baseline k-NN. VSM and SVM results are also provided for comparison.

4.3 Results of improved systems

In this section, we show how the proposed improvements affect the performance. Table 3 presents the results of doing SVD, and then applying VSM, SVM and k-NN over the reduced space. We can observe that only k-NN improves performance, with VSM and SVM getting lower results. These and other prior experiments motivated us to only use k-NN on the improved systems.

Table 4 shows the results on the training set for the baseline k-NN systems, as well as all improvements explored. The difference over the baseline system shows that all improvements were positive, raising from 1.5 to 1.9 the performance of the baseline. Still, there is no improvement observed when introducing unlabeled data into SVD (25% BNC and 50% BNC in Table 4) compared to using labeled data only (SVD in Table 4).

Table 5 shows the same data for all baseline systems (including VSM and SVM) on the test set. The improvement here is lower but consistent with Table 4. The only difference is that using 25% or 50% of the BNC as unlabeled data for SVD is better than not using labeled data. Table 5 also presents the results of the other two baseline systems, showing that all k-NN systems are below VSM and SVM. This motivated us to try to combine the k-NN classifiers.

5 Combining several k-NN systems

The results from the previous section show that the improved systems (Section 3) are able to increase the results of k-NN, but are still below our SVM and VSM baseline systems. The key observation here is that under each of the improved classifiers there is a slightly different feature space. All of them provide improvements, and are therefore able to generalize interesting properties of the problem space. If we are able to combine them properly, we might be able to further improve the results.

The combination of classifiers is an active area of research. Here we exploited the fact that a k-NN classifier can be seen as k points casting each one vote, making easy a combination of several k-NN classifiers. For instance, if we have two k-NN classifiers of $k = 5$, c_1 and c_2, then we can combine them into a single classifier equivalent to $k = 10$. In order to carry through the properties of each feature space, we decided to weight each vote by the cosine similarity of that point instead of the rank. We need to note that this combination method was also used in the previous section to combine the local and topical classifiers.

Table 6 shows the results over the training set. Plain stands for the baseline k-NN system. The following rows show the improved systems from the previous Section. Then the results of combining the algorithms two by two are shown, where each of the improved systems has been combined with the baseline k-NN system. The results show that all combinations attain better results than any of their components. We can also see that, in this setting, using unlabeled data (plain+SVD with 50%) improves slightly over not using it (plain+SVD). Finally, the full combina-

k-NN($k = 5$)	**Recall**	**diff.**
plain	67.7	—
local+topical	69.4	+1.7
SVD	69.6	+1.9
SVD (25% BNC)	69.2	+1.5
SVD (50% BNC)	69.6	+1.9
plain + local+topical	69.9	+2.2
plain + SVD	70.7	+3.0
plain + SVD (25% BNC)	70.7	+3.0
plain + SVD (50% BNC)	70.8	+3.1
full combination	**71.9**	+4.2

Table 6: Results for different combinations of k-NN classifiers in 3-fold cross-validation (Senseval-3 training set)

Classifiers	**Recall**	**diff.**
plain	70.5	—
local+topical	70.8	+0.3
SVD	70.7	+0.2
SVD(%25 BNC)	70.8	+0.2
SVD(%50 BNC)	71.2	+0.7
plain + local+topical	71.5	+1.0
plain + SVD	71.2	+0.7
plain + SVD (25% BNC)	72.3	+1.8
plain + SVD (50% BNC)	72.7	+2.2
full combination	**73.4**	+2.9
SVM	71.2	—
VSM	71.5	—
Best S3	72.9	—
(Gliozzo *et al.* 05)	73.3	—

Table 7: Results for different combinations of k-NN classifiers in the Senseval-3 Lexical Sample test set. Plain stands for baseline k-NN. VSM and SVM results are also provided, as well as the best Senseval-3 system and the best result published to date.

tion of all 5 systems provides the best results. Note that for the full combination, we applied SVD (with only labeled data, plus 25% of BNC and 50% of BNC) also to the local and topical classifiers.

The results on the test set, Table 7, confirm the cross-validation results. Note that unlabeled data makes a more significant improvement over plain+SVD. Below the combined system, Table 7 also shows our baseline systems, as well as the best system in the Senseval 3 competition and the best reported result to date. The full combination of our k-NN systems attains the best results of them all.

6 Discussion and related work

The results show that we have been able to better model the feature space. SVD helps to find correlations among the features, and thus alleviate the sparse data and redundancy problems. Including unlabeled data provides very narrow performance increases, but combined with the other classifiers it makes a difference. Splitting the feature space in two and combining the two spaces also improves the results. These improvements in isolation are not very large. In fact, the resulting k-NN systems are below our SVM and SVM baseline systems for the original feature set. But when we combine the k-NN algorithms over each of the feature spaces, we attain the best results to date in the Senseval-3 dataset.

We think that the reason explaining the extraordinary performance of the combination is that each of the changes in the feature space helps finding regularities in the data that k-NN could not find before. When we combine each of the simpler k-NN systems, we are looking for the word sense that is closest to the target instance in as many of the changed feature spaces as possible.

Some of the findings in this paper are confirmed in related work, but this paper integrates them in a single task (WSD) and shows that they provide the best performance. For instance, (Kohomban & Lee 05) show in a different WSD task that building separate k-NN classifiers from different subset of features and combining them works better than constructing a single classifier with the entire feature set. In (Gliozzo *et al.* 05), instead of splitting the feature space and then combining the classifiers, they use specialized kernels to model the similarity for each kind of features. They also use SVD but only for bag-of-words features, while we apply SVD to all features. The good performance of coupling k-NN and SVD are well known in the ML literature, e.g. (Thomasian *et al.* 05) on a image retrieval task. (Dietterich 98) says that spliting features only works when the feature space is higly redundant. We already mentioned in the Introduction other works which make use of unlabeled data on a WSD setting.

7 Conclusions and Future Work

In this paper we have explored feature modeling, trying to tackle sparse data, redundancy and heterogeneity in the feature set. We have proposed and evaluated three improvements: 1) using SVD in order to find correlations among features and deal with sparsity and redundancy, 2) using unlabeled data from a corpus related to the evaluation corpus in order to provide background knowledge, and 3) splitting the feature space into smaller, more coherent, sets. Each of the proposals improves the results for a k-NN classifier, and properly combined they provide the best results to date for the Senseval-3 lexical sample dataset.

In the discussion we have argued that this improvements help to model better the feature space, which, coupled with a ML algorithm well suited for combination such as k-NN, explain the good results. This opens new feature modeling possibilities. In particular we are thinking of finer splits of the feature space, using kernels to better model similarity for certain features. On the other hand we have shown that unlabeled data helps, and we would like to better explore which is the situation when the training and test data come from distinct corpora or domains.

Acknowledgements

We wish to thank Basilio Seirra and Ana Zelaia, from the University of Basque Country, for helping us with the SVD and k-NN methods. This research has been partially founded by the European Commision (MEANING IST-2001-34460), Ministry of Education and Science (CESS-ECE HUM2004-21127-E) and the Basque Goverment (EUSEMCOR S-PE04UN10).

References

(Agirre & Martínez 04) E. Agirre and D. Martínez. Smoothing and Word Sense Disambiguation. In *Proceedings of EsTAL - España for Natural Language Processing*, Alicante, Spain, 2004.

(Dietterich 98) Thomas G. Dietterich. Machine-learning research: Four current directions. *The AI Magazine*, 18(4):97–136, 1998.

(Edmonds & Cotton 01) P. Edmonds and S. Cotton. SENSEVAL-2: Overview. In *Proceedings of the Second International Workshop on evaluating Word Sense Disambiguation Systems.*, Toulouse, France, 2001.

(Fellbaum 98) C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

(Gliozzo *et al.* 05) Alfio Massimiliano Gliozzo, Claudio Giuliano, and Carlo Strapparava. Domain Kernels for Word Sense Disambiguation. *43nd Annual Meeting of the Association for Computational Linguistics. (ACL-05)*, 2005.

(Joachims 99) T. Joachims. Making Large–Scale SVM Learning Practical. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 169–184, Cambridge, MA, 1999. MIT Press.

(Kohomban & Lee 05) Upali S. Kohomban and Wee S. Lee. Learning Semantic Classes for Word Sense Disambiguation. In *43nd Annual Meeting of the Association for Computational Linguistics. (ACL-05)*, University of Michigan, Ann Arbor, 2005.

(Leech 92) G. Leech. 100 million words of English: the British National Corpus. *Languaje Research*, 28(1):1–13, 1992.

(Martínez & Agirre 00) David Martínez and Eneko Agirre. One Sense per Collocation and Genre/Topic Variations. *Conference on Empirical Method in Natural Language*, 2000.

(Mihalcea 04) Rada Mihalcea. Co-training and Self-training for Word Sense Disambiguation. In *In Proceedings of the Conference on Natural Language Learning (CoNLL 2004)*, Boston, USA, 2004.

(Mihalcea *et al.* 04) R. Mihalcea, T. Chklovski, and Adam Killgariff. The Senseval-3 English lexical sample task. In *Proceedings of the 3rd ACL workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL)*, Barcelona, Spain, 2004.

(Miller *et al.* 93) G.A. Miller, C. Leacock, R. Tengi, and R.Bunker. A Semantic Concordance. In *Proceedings of the ARPA Human Language Technology Workshop. Distributed as* Human Language Technology *by San Mateo, CA: Morgan Kaufmann Publishers.*, pages 303–308, Princeton, NJ, 1993.

(Ngai & Florian 01) G. Ngai and R. Florian. Transformation-Based Learning in the Fast Lane. *Proceedings of the Second Conference of the North American Chapter of the Association for Computational Linguistics, pages 40-47, Pittsburgh, PA, USA*, 2001.

(Pedersen 01) T. Pedersen. A Decision Tree of Bigrams is an Accurate Predictor of Word Sense. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-01)*, Pittsburgh, PA, 2001.

(Snyder & Palmer 04) B. Snyder and M. Palmer. The English all-words task. In *Proceedings of the 3rd ACL workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL)*, Barcelona, Spain, 2004.

(Su *et al.* 04) Weifeng Su, Dekai Wu, and Marine Carpuat. Semi-Supervised Training of a Kernel PCA-Based Model for Word Sense Disambiguation. *20th International Conference on Computational Linguistics (COLING-2004)*, 2004.

(Thomasian *et al.* 05) A. Thomasian, Y. Li, and L. Zhang. Exact k-NN queries on clustered SVD datasets. *Information Processing Letters (ILP)*, 94:247–252, 2005.

The SenSem Project: syntactico-semantic annotation of sentences in Spanish

Alonso, Laura* **Capilla, Joan Antoni†** **Castellón, Irene*** **Fernández-Montraveta, Ana**** **Vázquez, Gloria†**

*Department of Linguistics, Universitat de Barcelona, Spain
{lalonso,icastellon}@ub.edu
**Department of English and German Philology, Universitat Autònoma de Barcelona, Spain
ana.fernandez@uab.es
†Department of English and Linguistics, Universitat de Lleida, Spain
{jcapilla,gvazquez}@dal.udl.es

Abstract

This paper presents SenSem, a project[1] that aims to systematize the behavior of verbs in Spanish at the lexical, syntactic and semantic level. As part of the project, two resources are being built: a corpus where sentences are associated to their syntactico-semantic interpretation and a lexicon where each verb sense is linked to the corresponding annotated examples in the corpus. Some tendencies that can be observed in the current state of development are also discussed.

1 Introduction

The SenSem project[2] aims to build a databank of Spanish verbs based on a lexicon that links each verb sense to a significant number of manually analyzed corpus examples. This databank will reflect the syntactic and semantic behavior of Spanish verbs in naturally occurring text.

We analyze the 250 verbs that occur most frequently in Spanish. Annotation is carried out at three different levels: the verb as a lexical item, the constituents of the sentence and the sentence as a whole. The annotation process includes verb sense disambiguation, syntactic structure analysis (syntagmatic categories, including the annotation of the phrasal heads, and syntactic functions), interpretation of semantic roles and analysis of various kinds of sentential semantics. It is precisely this last area of investigation which sets our project apart from others currently being carried out with Spanish (Subirats and Petruck, 2003 and García De Miguel and Comesaña, 2004).

Abstracting from the analysis of a significant number of examples, the prototypical behavior of verb senses will be systematized and encoded in a lexicon. The description of verb senses will focus on their properties at the syntactico-semantic interface, and will include information like the list of syntactico-semantic frames in which a verb can possibly occur. In addition, selectional restrictions will be automatically inferred from the words marked as heads of the constituents. Finally, the usage of prepositions will be studied.

The conjunction of all this information will provide a very fine-grained description of the syntactico-semantic interface at sentence level, useful for applications that require an understanding of sentences beyond shallow parsing. In the fields of automatic understanding, semantic representation and automatic learning systems, a resource of this type will be especially valuable.

In the rest of the paper we will describe the corpus annotation process in more detail and examples will be provided. Section 2 offers a general overview of other projects similar to SenSem. In section 3, the levels of annotation are discussed, and the process of annotation is described in section 4. We then proceed to present

[1] Databank Sentential Semantics: "Creación de una Base de Datos de Semántica Oracional". MCyT (BFF2003-06456).
[2] http://grial.uab.es/projectes/sensem.php

the results obtained to date and the current state of annotation, and we put forward some tentative conclusions obtained from the results of the annotation thus far.

2 Related Work

As shown by Levin (1993) and others (Jones et al., 1994; Jones, 1995; Kipper et al., 2000; Saint-Dizier, 1999; Vázquez et al., 2000), syntax and semantics are highly interrelated. By describing the way linguistic layers inter-relate, we can provide better verb descriptions since generalizations from the lexicon that previously belonged to the grammar level of linguistic description can be established (lexicalist approach).

Within the area of Computational Linguistics, it is common to deal with both fields independently (Grishman et al., 1994; Corley et al., 2001). In other cases, the relationship established between syntactic and semantic components is not fully exploited and only basic correlations are established (Dorr et al., 1998; McCarthy, 2000). We believe this approach is interesting even though it does not take full advantage of the existing link between syntax and semantics.

Furthermore, we think that in order to coherently characterize the syntactico-semantic interface, it is necessary to start by describing linguistic data from real language. Thus, a corpus annotated at syntactic and semantic levels plays a crucial role in acquiring this information appropriately.

In recent years, a number of projects related to the syntactico-semantic annotation of corpora have been carried out. The length of the present paper does not allow us to consider them all here, but we will mention a few of the most significant ones.

FrameNet (Johnson and Fillmore, 2000) is a lexicographic resource that describes approximately 2.000 items, including verbs, nouns and adjectives that belong to diverse semantic domains (communication, cognition, perception, movement, space, time, transaction, etc.). Each lexical entry has examples extracted from the British National Corpus that have been manually annotated. The annotation reflects argument structure and, in some cases, also adjuncts.

PropBank (Kingsbury and Palmer, 2002; Kingsbury et al., 2002) is a project based on the manual semantic annotation of a subset of the Penn Treebank II (a corpus which is syntactically annotated). This project aims to identify predicate-argument relations. In contrast with FrameNet, the sentences to be annotated have not been pre-selected so examples are more varied.

Both FrameNet and Propbank work with the use of corpora, although their objectives are a bit different. In FrameNet, a corpus is used to find evidence about linguistic behavior and to associate examples to lexical entries, whereas in Propbank, the objective is to enrich a corpus that has been already annotated at a syntactic level so that it can be exploited in more ambitious NLP applications.

For Spanish, only a few initiatives address the syntactico-semantic analysis of corpus. The DataBase "Base de Datos Sintácticos del Español Actual" (Muñiz et al., 2003) provides the syntactic analysis of 160.000 sentences extracted from part of the ARTHUS corpus of contemporary texts. Syntactic positions are currently being labeled with semantic roles (García de Miguel and Comesaña, 2004).

FrameNet-Spanish (Subirats and Petruck, 2003) is the application of the FrameNet methodology for Spanish. Its target is to develop semantic frames and lexical entries for this language. Each verb sense is associated to its possible combinations of participants, grammatical functions and phrase types, as attested in the corpus.

The SenSem project provides a different approach to the description of verb behavior. In contrast with FrameNet, its aim is not to provide examples for a pre-existing lexicon, but to shape the lexicon with the corpus examples annotated. Another difference from the FrameNet approach is that the semantic roles we use are far more general, they are related to syntactic functions, and are less class-dependent.

Finally, to the best of our knowledge, no large-scale corpus annotation initiative associates semantics to sentence such as their aspectual interpretations or types of causativity.

3 Levels of annotation

As mentioned previously, we are describing verb behavior so only constituents directly related to the verb will be analyzed. Elements beyond the scope of the verb (i.e. extra-sentential elements such as logical linkers, some adverbs, etc.) are disregarded. The following is an example of scope of annotation:

> ...El presidente, que ayer **inició** una visita oficial a la capital francesa, hizo estas declaraciones…
>
> *...The president, who **began** an official visit to the French capital yesterday, stated…*

Were we annotating the verb *iniciar* –begin– we would ignore the participants of the main sentence and only take into account the elements within the clause.

If we were annotating the verb *hacer* –make– we would annotate the subject to include the entire relative clause, with the word *president*" as the head of the whole structure. The relative clause will not be further analyzed.

Sentences are annotated at three levels: sentence semantics, lexical and constituent level.

3.1 Sentential semantics level

At this level, different aspects of sentential semantics are accounted for. With regard to aspectual information, a distinction is made among three types of meaning, *eventive*, *procedural* or *stative*, as in the following examples:

> *event:* ...El diálogo **acabará** hoy…
> *...The conversations **will finish** today...*
> *process:* …cuando le preguntaron de qué **había vivido** hasta aquel momento ...
> *...when he was asked what he **had been living on** until then...*
> *state:* ...El gasto de personal **se acerca** a los 2.990 millones de euros...
> *...Personnel expenses **come close** to 2,990 million euros...*

Apart from aspectual information, we also annotate sentential level meanings using labels like *anticausative, antiagentive, impersonal, reflexive, reciprocal* or *habitual.* This feature is useful to account for the variation in the syntactic realizations of the argument structures of each verb sense. For example, the next sentence with the verb "*abrir*" (*open*) presents an antiagentive interpretation:

> *agentive:* **El alcalde de Calafell** […] abrirá un expediente…
> *... **The mayor of Calafell** […] will open administrative proceedings ...*
> *antiagentive:* … **el vertedero de Tivissa** no se abrirá sin consenso.
> *... **Tivissa's rubbish dump** will not be opened without a consensus.*

3.2 Lexical level

At the lexical level, each example of a verb is assigned a sense. We have developed a verb lexicon in which the possible senses for a verb are defined, together with its prototypical event structure and thematic grid, and a list of synonyms and antonyms and its related synsets in WordNet (Fellbaum, 1998).

Various lexicographic sources have been taken as references to build the inventory of senses for each verb, mainly the *Diccionario de la Real Academia de la Lengua Española* and the *Diccionario Salamanca de la Lengua Española*. Less frequent meanings are discarded, together with archaic and restricted uses.

This inventory of senses for each verb is only preliminary, and can be modified whenever the examples found in the corpus indicate the existence of a distinct sense which has not been considered. Different senses are distinguished by different thematic grids, different event structures, different selectional restrictions or different subcategorizations.

3.3 Constituent level

Finally, at the constituent level, each participant in the clause is tagged with its constituent type (e.g.: *noun phrase, completive, prepositional phrase*) and syntactic function (e.g.: *subject, direct object, prepositional object*).

Arguments and adjuncts are also distinguished. Arguments are defined as those participants that are part of the verb's lexical semantics. Arguments are assigned a semantic role describing their relation with the verb (e.g.: agent, theme, initiator…). In SenSem, each sense is associated with a prototypical thematic grid describing the possible arguments a verb may take, but, as in the case of senses, this thematic grid is only preliminary and is modified when corpus examples provide enough evidence.

The head of the phrase is also signaled in order to acquire selectional restrictions for that verb sense.

Sometimes, information that has been considered relevant in that it may alter some other information declared at a different level has also been included; for example, negative polarity or negative adverbs are also indicated.

4 Annotation process

The SenSem corpus will describe the 250 most frequently occurring verbs in Spanish. Frequency has been calculated in a journalistic corpus. For each of these verbs, 100 examples are extracted randomly from 13 million words of corpora obtained from the electronic version of the Spanish newspapers, *La Vanguardia* and *El Periódico de Catalunya*. The corpus has been automatically tagged and a shallow parsing analysis has been carried out to detect the personal forms of the verbs under consideration. We

do not take into account uses of the verb as an auxiliary. We also disregard any collocations or idioms in which the verb might participate.

The manual annotation of examples is carried out via a graphical interface, seen in Figure 1, where the three levels are clearly distinguished.

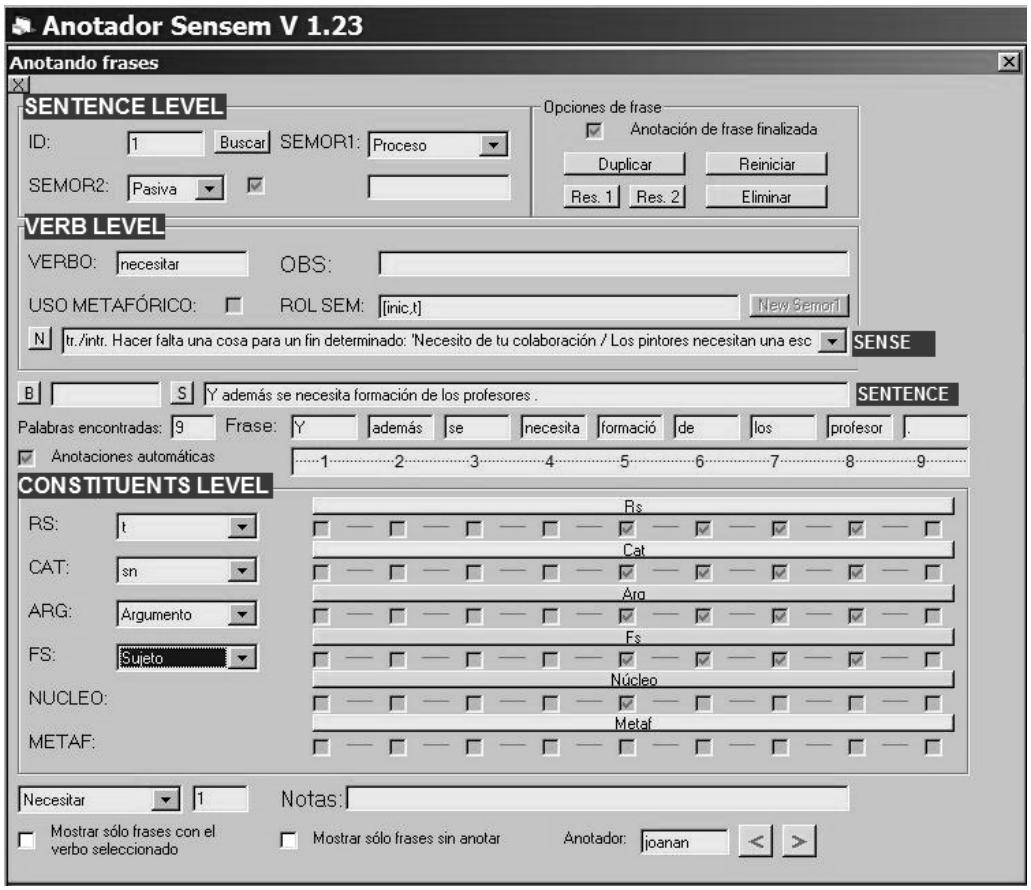


Figure 1. Screenshot of the annotation interface.

The interface displays one sentence at a time. First, when a verb sense is selected from the list of possible senses, its prototypical event structure and semantic roles are displayed for the annotator to take into account. Then, the clause is assigned its aspectual semantics, and constituents are identified and analyzed by selecting the words that belong to it. The head of the arguments and its possible metaphorical usage are also signaled in order to facilitate a future automatic extraction of selectional restrictions. Finally, annotators specify any applicable semantics at clause level (e.g.: anticausative, reflexive, stative, etc.), and state any particular fact that they consider might be of use in future revision and correction processes.

The distribution of the corpus among annotators has evolved since the earlier stages of the project. In an initial stage, when the annotation guidelines were not yet consolidated, each of the 4 annotators was given 24 different sentences of the same verb, plus 4 common sentences that were separately annotated by all of them. Later on, these sentences were compared in order to identify those aspects of the annotation that were unclear or prone to subjectivity, as explained in the following section. In the current stage, the annotation guidelines have been well established. Annotators work with sets of 100 sentences corresponding to a single verb. All annotations are revised and any possible errors are corrected.

The final corpus will be available to the linguistic community by means of a soon to be created web-based interface.

5 Preliminary Results of Annotation

At this stage of the project, 77 verbs have already been annotated, which implies that the corpus at this moment is made up of 7,700 sentences (199,190 words). A total of 900 sentences out of these 7,700

have already been validated, which means that a corpus of approximately 25,000 words has already undergone the complete annotation process.

5.1 Data analysis

In this section we describe the information about verb behavior that can be extracted from the corpus in its present state. We have found that, out of the 199,190 words that have already been annotated, 182,303 are part of phrases which are an argument of the verb and 16,887 are adjuncts.

With regard to aspectuality, there is a clear predominance of events (74.26% of the sentences) over processes (20.67%) and states (8.96%). This skewed distribution of clause types, with a clear predominance of events, may be exclusive to the journalistic genre. We have yet to investigate its distribution in other genres.

As concerns syntactic functions, seen in Table 1, the most frequent category is direct object, with a significant difference in subjects. This is not surprising if we take into account that Spanish is a pro-drop language. However, prepositional objects are less frequent than subjects, and indirect objects are also scarce. Thus, the clausal core appears to be predominantly populated by the least marked constituents.

Function	ratio
direct object	39.83 %
subject	22.57 %
circumstantial	23.16 %
prepositional object	12.65 %
indirect object	1.97 %

Table 1. Distribution of syntactical functions in the annotated examples.

The distribution of semantic roles can be seen in Table 2. Themes are predominant, as would be expected given that the most common syntactic function is that of direct object, and that there is a high presence of antiagentive, anticausative and passive constructions. Within the different types of the semantic role theme, unaffected themes (moved objects) appear most frequently.

At the constituent level, the semantic role chosen for each phrase is often predictive of the other labels of that phrase, following what was expected from linguistic introspection: agents tend to be noun phrases with subject function, themes tend to be noun phrases with subject or object function (if they occur in a passive, antiagentive, anticausative or stative sentence), etc. +

Role	Ratio
not- affected theme	53.47 %
affected theme	14.36%
agent and cause	14.02%
initiator	2.97%

Table 2. Distribution of semantic roles in the annotated examples.

Thus, the associations made between labels in different levels have been used as a first step to semi-automate the annotation process: once a role is selected, the category and function most frequently associated with it and its role as a verb argument are pre-selected so that the annotator only has to validate the information.

5.2 Inter-annotator agreement

In order to measure inter-annotator agreement, four sentences of 59 verbs have been annotated by 4 different judges so that divergences in criteria could be found. These common sentences were used in the preliminary phase with the aim of both training the annotators and detecting points of disagreement among them. This comparison has helped us refine and settle the annotation guidelines and facilitate the subsequent revision of the corpus.

In order to detect these problematic issues, we calculated inter-annotator agreement for all levels of annotation. An overview of the most representative values for annotator agreement can be seen in Table 3.

We determined pair wise proportions of overall agreement, that is, the ratio of cases in which two annotators agreed with respect to all cases.

In addition, we also obtained the kappa coefficient (Cohen, 1960), which gives an indication of stability and reproducibility of human judgments in corpus annotation. The main advantage of this measure is that it factors out the possibility that judges agree by chance. Kappa measures range from k=-1 to k=1, with k=0 when there is no agreement other than what would be expected by chance, k=1 when agreement is perfect, and k=-1 when there is systematic disagreement. Following the interpretation proposed by Krippendorf (1980) and Carletta et al. (1996), for corpus annotation, kappa>0.8 indicate good stability and reproducibility of the results, while k<0.68 indicates unreliable annotation.

category	agreement	kappa
eventual semantics		
event	*66%*	.11
state	90%	.33
process	76%	.06
argumentality		
argument	82%	.54
adjunct	64%	.46
semantic role		
initiator	70%	.37
agent	84%	.81
cause	91%	.89
experiencer	97%	.92
theme	68%	.43
affected theme	74%	.55
non-affected theme	70%	.34
goal	79%	.70
syntactic function		
agentive complement	100%	1.00
subject	87%	.83
direct object	80%	.63
indirect object	77%	.79
prepositional object 1	67%	.65
prepositional object 2	66%	.28
prepositional object 3	78%	.24
Circumstantial	62%	.42
Predicative	76%	.16
syntactic category		
noun phrase	78%	.67
prepositional phrase	72%	.53
adjectival phrase	88%	.69
negative adverbial	100%	1.00
adverbial phrase	77%	.54
adverbial clause	68%	.66
gerund clause	72%	.65
relative clause	82%	.16
completive clause	95%	.93
direct speech	96%	.95
infinitive clause	94%	.98
prep. completive clause	96%	.44
prep. infinitive clause	81%	.57
personal pronoun	97%	.81
relative pronoun	98%	.96
other pronouns	94%	.82

Table 3. Inter-annotator agreement for a selection of annotated categories

As a general remark, agreement is comparable to what is reported in similar projects. For example, Kingsbury et al. (2002) report agreement between 60% and 100% for predicate-argument tagging within Propbank, noting that agreement tends to increase as annotators are more trained. In SenSem, the level of annotation that is comparable to predicate-argument relations, semantic role annotation, is clearly within this 60%-100% range.

It is noteworthy that the values obtained for the kappa coefficient are rather low. After a close inspection, we found that these low values of kappa are mainly due to the fact that the annotation guidelines were still not well-established at this point of annotation, and that annotators were still under training. This led us to further describe and exemplify cases detected as having a low agreement value once the preliminary exploration of the corpus had concluded. As a result, we expect values for kappa to increase in evaluations that will be carried out in a later stage of the project.

Agreement within aspectual interpretations of sentences is very close to chance agreement. The stative interpretation seems to be more clearly perceived than the rest. Events and processes at times seem to be confused. In order to reduce this source of disagreement, each verbal sense was associated to its prototypical aspectual semantics, as determined by its lexical meaning. For example, the verb *aceptar* (*accept*) is associated to the semantics "event" for its sense "*to receive (something offered), especially with gladness or approval*" and to the semantics "state" for its sense "*to be able to endure, hold or admit (in an ordered system)*". We expect that this change in the annotation procedure will dramatically increase inter-annotator agreement for this feature.

Agreement is also low in the categorization of constituents as arguments or adjuncts. To improve consistency in the annotation of arguments, the prototypical subcategorization frame for each verb sense has been provided, making it easier for annotators to identify arguments associated with a verb and to label the rest of constituents as adjuncts. For example, in the case of the verb *accept*, the eventive sense is associated with a subcategorization frame of the kind *[agent,theme]*, while the second is associated with *[theme,theme]*. The criteria to distinguish constituents dominated by a verb (arguments or adjuncts) and those beyond clausal scope have also been clarified.

In contrast, the tagging of semantic roles appears to rely on linguistic intuition much more than the above features. There seems to be perfect agreement for very infrequent roles (indirect cause, instrument, location). More frequent roles show a higher level of disagreement: initiators are significantly less clearly

perceived than agents or causes (note differences in k agreement). It is also clear that fine-grained distinctions are more difficult to perceive than coarse-grained ones, as exemplified by low agreement within the superclass of theme.

Among syntactic functions, the agentive complement of passives presents perfect agreement. Agreement is also high for subjects and indirect objects, but the distinction between different kinds of prepositional objects and circumstantial complements is not clearly perceived. Therefore, a clearer decision-making procedure was established in the annotation guidelines to distinguish among these. We expect that these changes will improve consistency significantly.

Finally, agreement is rather high for some syntactic categories: pronouns, adverbs of negation, adjectival complements, completive clauses, infinitive clauses and direct speech present k > .7 and ratios of agreement over 90%. However, major categories present a rather high ratio of disagreement, as well as those categories that are mostly considered adjuncts. This seems to be a direct effect of the variability in the assignment of argumentality, semantic role and syntactic function features. We expect that a thorough inspection of the relations between variability in roles and functions with relation to variability in categories will provide a clearer view of this aspect.

After the study of inter-annotator agreement, the guidelines for annotation have been settled (Vázquez et al. 2005). These guidelines serve as a reference for annotators, and we believe they will increase the overall consistency of the resulting corpus. Later on in the process of annotation, another study of inter-annotator agreement will be carried out to determine the consistency achieved in corpus annotation.

6 Conclusions and Future Work

The linguistic resource we have presented constitutes an important source of linguistic information useful in several natural language processing areas as well as in linguistic research. The fact that the corpus has been annotated at several levels increases its value and its versatility.

The project is in its second year of development, with still a year and a half to go. During this time we intend to continue with the annotation process and to develop a lexical database that will reflect the information found in the corpus. We are aware that the guidelines established in the annotation process are going to bias, to a certain extent, the resulting resources, but nevertheless we believe that both tools are of interest for the NLP community.

All tools developed in the project and the corpus and lexicon themselves will be available to all researchers who might have interest in exploiting them.

References

(Carletta et al. 1996) J. Carletta, A. Isard, S. Isard, J. C. Kowtko, G. Doherty-Sneddon and A. H. Anderson, HCRC Dialogue Structure Coding Manual, HCRC Technical report HCRC/TR-82, 1996.

(Cohen 1960) J. Cohen, A coefficient of agreement for nominal scales. in Educational & Psychological Measure, 20, 1960, pp. 37-46.

(Corley et al. 2001) S. Corley, M. Corley, F. Keller, M. W. Crocker, S. Trewin, Finding Syntactic Structure in Unparsed Corpora in Computer and the Humanities, 35, 2001, pp. 81-94.

(Dorr et al. 1998) B. Dorr, M. A. Martí, I. Castellón, Spanish EuroWordNet and LCS- Based Interlingual MT. Proceeding of the ELRA Congress. Granada, 1998.

(Fellbaum 1998) C. Fellbaum, A Semantic Network of English Verbs, in Christiane Fellbaum (ed.). WordNet: An Electronic Lexical Database, MIT Press, 1998.

(Garcia de Miguel & Comesaña 2004) J.M. Garcia de Miguel and S. Comesaña, Verbs of Cognition in Spanish: Constructional Schemas and Reference Points, in A. Silva, A. Torres, M. Gonçalves (eds) Linguagem, Cultura e Cogniçao: Estudos de Linguística Cognitiva, Almedina, 2004, pp. 399-420.

(Grishman et al. 1994) R. Grishman, C. Macleod and A. Meyers. 1994. Comlex Syntax: Building a computational lexicon. Proceedings of COLING.

(Johnson &Fillmore 2000) C. Johnson and C. J. Fillmore, The FrameNet tagset for frame-semantic and syntactic coding of predicate-argument structure. Proceedings NAACL 2000, Seattle WA, USA, 2000, pp. 56-62.

(Jones 1994) D. Jones (ed), Verb classes and alternations in Bangla, German, English and Korean. Memo nº 1517. MIT, Artificial Intelligence Laboratory, 1994.

(Jones 1995) D. Jones, Predicting Semantics from Syntactic Cues -- Evaluating Levin's English Verb Classes and Alternations. UMIACS TR-95-121, University of Maryland, 1995.

(Kingsbury & Palmer. 2002) P. Kingsbury and M. Palmer, From Treebank to Propbank. Third International Conference on Language Resources and Evaluation, LREC-02, Las Palmas, Spain, 2002.

(Kingsbury et al. 2002) P. Kingsbury, M. Palmer and M. Marcus., Adding Semantic Annotation to the Penn TreeBank. Proceedings of the Human Language Technology Conference. San Diego, California, 2002.

(Kipper ET AL. 2000) K. Kipper, H. T. Dang and M. Palmer, Class-Based Construction of a Verb Lexicon. AAAI-2000 Seventeenth National Conference on Artificial Intelligence, Austin, TX, USA, 2000.

(Krippendorf 1980) K. Krippendorf, Content analysis: an introduction, Sage, 1980.

(Levin 1993) B. Levin, English Verb Classes and Alternations: A Preliminary Investigation. The University of Chicago Press, 1993.

(McCarthy 2000) D. McCarthy, Using semantic preferences to identify verbal participation in role switching alternations. Proceedings of NAACL 2000, Seattle, WA, USA, 2000.

(Muñiz ET AL. 2003) E. Muñiz, M. Rebolledo, G. Rojo, M.P. Santalla and S. Sotelo, Description and Exploitation of BDS: a Syntactic Database about Verb Government in Spanish, in Galia Angelova, Kalina Bontcheva, Ruslan Mitkov, Nicolas Nicolov, Nikolai Nikolov (eds.). Proceedings of RANLP 2003. Borovets, Bulgaria, 2003, pp. 297-303.

(Saint-Dizier 1999) P. Saint-Dizier, Alternations and verb semantic classes for French: analysis and class formation . Saint-Dizier, P. (ed.). Predicative Forms in Natural Languages and Lexical Knowledge Bases. Holanda. Kluwer: 139-170.

(Subirats-Rüggeberg & Petruck 2003) Subirats-Rüggeberg, C. y M. R. L. Petruck, Surprise: Spanish FrameNet! Presentation at Workshop on Frame Semantics, Proceedings of the International Congress of Linguists, Praga., 2003.

(Vázquez et al. 2000) G. Vázquez, A. Fernández and M. A. Martí, Clasificación verbal. Alternancias de diátesis, Universitat de Lleida, 2000.

(Vázquez et al. 2005) G. Vázquez, A. Fernández and L. Alonso, Guidelines for the syntactico-semantic annotation of a corpus in Spanish, RANLP, Bulgaria, 2005.

Finite State Morphology of Amharic

Saba Amsalu and **Dafydd Gibbon**
Fakultät für Linguistik und Literaturwissenschaft
Universität Bielefeld
Universität strasse 25
D-33501, Germany
{saba,gibbon}@uni-bielefeld.de

Abstract

For several computational linguistic tasks we require a morphological decomposition strategy. This paper describes non–linear morphology, modelled with finite–state (FS) techniques and implemented in a well–known FS toolset. We present a complete analysis of Amharic words of all categories. Analyses display the root, pattern and feature tags indicating part of speech, person, number, gender, mood, tense, etc.

1 Introduction

Amharic is a Semitic language, the official language of Ethiopia. Document production in Amharic is increasing rapidly, with conventional printing and word–processing, but little has been done to exploit these documents as a valuable resource for use in automatic language processing. Experimental computational work on specific aspects of Amharic is in progress at Addis Ababa University and elsewhere; e.g. (Alemayehu & Willett, 2002), (Fissaha & Haller, 2003a), (Fissaha & Haller, 2003b) and (Alemu, Asker & Getachew, 2003). We report here on the first complete account of finite–state Amharic morphology for all parts of speech, which was designed as a front–end for parallel corpus alignment, and implemented using the Xerox Finite State Tools.

2 Objectives

The goal of this work is to construct a generic morphological analyser for applications such as machine translation, sense disambiguation, lexicography, and terminology extraction. We aim to construct a tool that will analyse Amharic words from a natural language text transliterated into phonemic ASCII respresentation (SERA)[1]. The system has to produce accurate component roots/stems and feature tags that indicate part of speech, person, number, gender, mood, tense, etc. ; and it also has to give correct surface forms when run in the reverse direction.

[1]SERA (System for Ethiopic Representation in ASCII) is widely used for transliteration between Ethiopic syllables and ASCII

3 Amharic Morphology

Amharic verbs exhibit the typical Semitic non–linear word formation with intercalation (interdigitation) of consonantal roots with vocalic patterns. This also applies to deverbal nouns and adjectives. We use the term 'root' for lexical morphemes consisting of consonants, 'radical' for consonant constituents of roots; and 'stem' for intercalated forms.

3.1 Verbs

Verbs are morphologically the most complex POS in Amharic, with many inflectional forms; numerous words with other POS are derived primarily from verbs. Roots mainly consist of three radicals. It is controversial whether non–triradical roots are derived from triradicals; see (Dawkins, 1960); cf. (Bender & Fulas, 1978); (Yimam, 1999). Dawkins' classification is shown in Table 1. Simple verbs have five verbal stems that are formed by intercalation of vowels with skeleton patterns of the types CVCVC, CVCC etc.; see (Dawkins, 1960) (Bender & Fulas, 1978). These stems are: Perfective, Contingent, Jussive, Gerundive and Infinitive.

Aspect	Pattern	Stem	Description
Perfect	CVC̲VC	säb̲är	broke
Contingent	CVCC	säbr	break, will break
Jussive	CCVC	sbär	break! let sb. break!
Gerund	CVCC	säbr	breaking
Infinitive	CCVC	sbär	to break

Table 2:
Conjugation of a typical triradical Type A verb root *sbr*.

In Amharic verbs, the only vowel which is genuinely intercalated is *ä*. (cf. Table 2) shows the conjugation of the root *sbr*–typical triradical, type A (penultimate gemination in perfective stem only). When vowels other than the usual *ä*

Group	Examp.	Vowelled Form	Base Form	Gloss
Uncontracted tri-radical	ሰብር	sbr	sbr	break
Contracted tri-radical with a vowel instead of last radical	ሰምአ	sma	smh	hear
Contracted tri-radical with a vowel instead of penultimate radical	ልአከ ጥኡሰ ሾዐም	lak Tes Som	lhk $T^{y}s$ $S^{w}m$	send smoke appoint
Uncontracted four-radical	ምንዝር	mnzr	mnzr	change
Contracted four-radical	ዝንግአ ግብኝኤ	znga gbNe	zngh gbN^{y}	forget visit

Table 1: Dawkins' classification of roots.

occur in stems, it is the result of historical consonantal reduction, or to conditioning by sharp or flat consonants. The vowel *a* occurs due to the reduction of the glide *h* in the root. The vowel *o* alternatively occurs in dialects in cases where flat consonants such as $k^{w}ä$, $q^{w}ä$, $g^{w}ä$ etc. occur to create the forms *ko, qo, go* etc. When the vowel is short it is converted to *u* instead of *o*. The vowel *e* also refers to an underlining sharp consonant such as $C^{y}ä$, $T^{y}ä$, making *Ce, Te*.

The stems have the patterns of gemination, commonly referred to as Types A, B and C (the Fidel script does not distinguish between geminate consonants; they are read but not written):

- *Type A*: penultimate consonant geminates in Perfect only
- *Type B*: penultimate consonant geminates throughout the conjugation
- *Type C*: penultimate consonant geminates in Perfect and Contingent.

Several linguists have categorised Amharic verbs formally on the basis of root and stem structure; cf. (Bender, 1968), (Bender & Fulas, 1978), (Dawkins, 1960), (Markos, 1994). A detailed study of verb morphology is given by (Bender, 1968) and (Bender & Fulas, 1978): 42 verb classes based on three main morphotactic criteria which provide input to phonological rules:

1. consonantal skeleton (one or more radicals);
2. gemination pattern (Types A, B, C);
3. occurrence of vowels other than *ä* (i.e. *e, o, a*).

Amharic verbs are not derived from other POS but from other verbs, mainly by affixation, penultimate consonant reduplication and vowel insertion; cf. (Amare, 1989), (Yimam, 1995). Except for the second person masculine jussive, the stem is always minimally inflected with a subject marker. The verb may be inflected for Person, Gender, Number, Mood and Tense. The verb is also inflected for beneficative, malfactive, causative, transitive, passive, dative, negative (Berhane, 1992).

3.2 Nouns

Amharic nouns are either simplex (e.g. *bEt* 'house', *merEt* 'earth' and *Isat* 'fire') or derived. The latter are derived from verb roots, adjectives or other nouns (e.g. *TyaqE* 'question' from *Tyq* 'to ask' , *degnet* 'generosity' from *deg* 'generous',*xumet* 'post, title' from xum 'an appointed person').

Deverbal nouns are derived from verb roots by intercalating different vowels between the radicals, by adding suffixes to the root without vowel intercalation, or by consonant reduction; cf. (Dawkins, 1960), (Amare, 1989), (Yimam, 1995). Affixation is the major process when deriving them from adjectives and other nouns. Nouns

Singular	Plural	(Alternative)	Gloss
mezgeb	mezagbt	mezgeboc	archive(s)
anbessa	anabst	anbessoc	lion(s)

Geez pl.noun	Amharic pl.
Mekuannt	mekuanntoc
Liqawnt	liqawntoc

Table 3: Treatment of Geez singular and plural borrowings.

are inflected for Number, Gender, Case and Definiteness. Most plural nouns are formed by adding a plural marker affix (*–oc* or *–woc* — their distribution is determined phonologically) to the singular form, although when referring to groups belonging to a certain tribe or country *–yan* is affixed. Nouns from the liturgical Geez language do not necessarily have these plural suffixes. Often, another operation in addition to plural marker affixation occurs. Table 3 lists noun borrowings from Geez: some Geez plural nouns are incorporated into Amharic as singulars and get an additional plural marker. Some collective nouns are, however, formed by full reduplication of the singular noun with insertion of a linking vowel *a*.

There are two genders in amharic, masculine and feminine. For things that are not naturally male or female, the gender female tends to be used when the entity is small or adorable; the gender male is used otherwise. The feminine gender suffix (*–it* or *–yt*, phonologically conditioned) is used to mark femininess in cases which otherwise would be masculine.

Definiteness markers are suffixes that vary depending on the gender of the noun (*–u* or *–wa* for feminine and *–u* or *–w* for masculine).

3.3 Pronouns, Adjectives, Adverbs, Prepositions, Conjunctions

Amharic pronouns can be free or bound to other POS. In the accusative and genitive, free personal pronouns take the affixes for nouns.

Adjectives are generally derived from verbs. The number of simplex adjectives is relatively small. Some simple adjectives are *qey* 'red', *deg* 'generous'. Adjectives are also derived from nouns or from verbal morphemes (Amare, 1989): cf. *brtu* 'strong', from *brth* 'be strong', *hayleNa* 'forceful', from *hayl* 'force, energy'. Like nouns, adjectives are inflected for Number, Case, Gender and Definiteness.

Adverbs in Amharic are very few, about seven common items, some derived from adjectives by suffixing *Na*; cf. (Amare, 1989) and (Yimam, 1995). Adverbial functions are often accomplished with noun phrases, prepositional phrases and subordinate clauses.

Conjunctions and prepositions have similar behaviours, and are often placed in the same class (*mestewadid*): no affixation, not used as base for derivations, syncategorematic and only occurring with other words.

3.4 Compounding

Amharic has compound verbs, nouns and adjectives. Compound verbs are created by combining the words *ale* 'said' or *aderege* 'did', with meaningless morphemes such as *qeT*: *qeT ale* 'he stood straight up', *qeT aderege* 'he made sth. straight'.

Compound nouns are formed by concatenating two nouns or a noun and an adjective with the linking vowel *–e–*: *bEtekrstiyan* 'church' = *bEt+e+krstiyan* = 'house+e+Christian'.

Compound adjectives are also formed by concatenating a noun and an adjective: *IgreqeCn* 'wanderer' = *Igr+e+qeCn* = 'leg+e+thin'.

Graphemic changes occur in word formation due to occurrence of vowels in sequence, and palatisation: $aa \rightarrow a$, $ia \rightarrow iya$ and when a dental consonant is followed by the vowel e or i it changes to palatal $de \rightarrow je$, $di \rightarrow ji$ or sometimes $di \rightarrow j$.

4 The morphological analyser

The morphological analyser takes a string of morphemes as an input and gives an output of lexical forms, i.e. underlying morphemes and morphosyntactic categories.

Many basic procedures in natural language processing standardly employ FS techniques for implementation, including tokenisation, phonological and morphological analysis, shallow parsing, spelling correction and others; cf. (Karttunen, 2003). Morphological constructions can be described particularly efficiently with regular expressions; cf. (Beesley & Karttunen, 2003), (Kay, 1987), (Koskenniemi, 1984), and (Kiraz, 2000). Morphological analysis using finite state transducers (FSTs) is based on the assumption that the mapping of words to their analysis constitutes a regular relation, i.e. the underlying forms constitute a regular set, the surface forms constitute a regular set, and there is a (possibly many–to–many) regular relation between these sets. In languages whose morphotactics is morph concatenation only, FSTs are straightforward to apply. Handling non–concatenative (or partially concatenative) languages is more challenging; cf. especially (Kay, 1987), (Beesley & Karttunen, 2003), (Trost, 2003).

4.1 Formal properties of word forms

The basic morphological modelling convention for Amharic is that there is a small finite upper

bound to root length (e.g. *sbr*) and to intercalated stems:

$root + vocalism + template = stem$

e.g. $sbr + \ddot{a} + CVCC = s\ddot{a}br$

Words are constructed from stems by concatenation of prefixes and suffixes. The reversibility property of FSTs is useful: the 'generate' mode is used for generation, the 'accept' mode for analysis (cf. Figure 1).

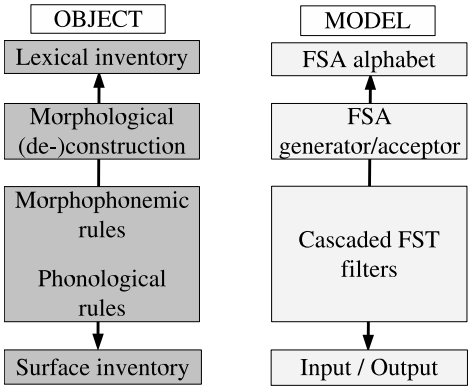


Figure 1: Modelling conventions for FSTs.

The absence of a lexicon of Amharic words in their base form is a major problem. About 1277 Amharic verb roots were compiled from (Bender & Fulas, 1978); other irregular verbs were gathered from (Dawkins, 1960). Deverbal nouns and adjectives were also obtained from these sources. Non–derived adjectives, adverbs, prepositions and conjunctions are few, and were manually collected. Simplex nouns are also hard to find. Lists of names were collected from the Bible, as well as place–names, kinship terms, body parts, local environmental terms and numbers (cardinal and ordinal), and implemented with the Xerox lexicon compiler (LEXC).

Semitic stem interdigitation has been treated several times; cf. (Kay, 1987), (Kataja & Koskenniemi, 1988), (Beesley & Karttunen, 2003). Kay designed a multitape FS technique for the interdigitation of roots, CV–templates and vocalisations in Arabic, and (Kataja & Koskenniemi, 1988) demonstrated interdigitation of Semitic roots (taking Ancient Akkadian as an example) using intersection over regular languages.

In (Beesley & Karttunen, 2003) a 'merge' operator for Arabic stems is described, a pattern filling algorithm which combines two regular languages, a CV template and fillers (root & vocalisation). The output of the merge operator is a regular expression that can be computed by the compile-replace algorithm of XFST. This algorithm works well for Amharic too. A more straightforward approach, however, would be to simply insert vocalisation between radicals. This requires accessing positions between consonant sequences. We used a novel bracketing 'diacritic' convention to locate vowel positions and right and left contexts to descriminate between different positions.

4.2 Internal changes

Derived verbs with internal changes involving penultimate consonant reduplication and vowel insertion are handled mostly by single replace rules. For example to generate *säbabär* from *säbär*, the rule used is:

$\{b\}(-->)\{bab\}jj\ddot{a}\_\ddot{a}$ which results in säbabär, while retaining the original underived *säbär*.

4.3 Affix concatenation

The regular operation concatenation is used to concatenate affixes to the stem. When concatenating, illegal sequences of vowels are avoided by using replace rules and also impermissible affix combinations are controlled by introducing constraints:

$[P1][P2][P3][P4][P5][stem1jstem2j...]$

[[S1|S2|S3] [S4] [S5][S6|S7]] [S8]

where P1-P5 stand for prefix categories and S1-S8 are suffix categories that a verb stem can take. Prefixes, stems and suffixes have specific positions. In case of prefixes, all categories may occur together, but no more than one from each category. There are constraints on the suffixes: [S1|S2|S3] are alternatives and cannot exist together in one word. The same is true for [S6|S7]. Similar procedures of concatenation are applied for other POS as well.

4.4 Full stem reduplication

Reduplication of collective nouns is handled by using the self concatenation operation *word^2* which concatenates a word to itself with the compile-replace algorithm of (Beesley & Karttunen, 2003), and using a bracketing rule to find the mid position to insert the vowel.

A second method that also gives the same results is without using the compile-replace algorithm just with the self concatenation operator and a temporary file to deal with singleton elements in the lexicon at a time to avoid over production of unwanted results. This operation de-

mands the use of a shell script outside the Finite State Tool we used (Xerox Finite State Tool-XFST).

4.5 Phonological processes

During affix concatenation, it is possible for vowels to occur in sequence that would result in a change of grapheme. To handle this problem simple replace rules are used. For example,

$\{aa\} -> \{a\}$, replaces the sequence *aa* by *a*.

$\{ae\} -> \{aye\}$ replaces the sequence *ae* by *aye*.

Finally, palatisation was handled by a replace rule that replaces dentals with palatals:

$\{di\}(->)\{pi\}$, maps *di* to *pi* and retains *di*

$\{di\} -> \{p\}$, maps the retained *di* to *p*

(the order of operation matters)

$\{de\} -> \{pe\}$ maps each *de* to *pe*

The transducers created for each class of verbs are finally merged by the union operation. This single transducer is then used whenever analysis of surfaces forms need to be made. The transducers for the different POS are not put together for evaluation purposes cf. Section 5.

5 Evaluation and conclusion

A preliminary evaluation of the system was made by analysing words from Amharic corpus (The Book of Matthew in the bible, Chapters 1–5). The evaluation hypothesis was that for each word class the words in it should be analysed correctly. A total number of 1620 words which contain words of all parts of speech were input into the transducers of each class. The results showed that among 468 verbs in the corpus 94% were analysed in total but taking the first 100 of analysed verbs 32% consisted also wrong analysis together with the correct ones. Among 650 nouns that exist in the corpus 85% were correctly analysed,with only a few about 7 that contain wrong analysis. For adjectives of 76 a recall of 88% with less than 1% wrong plus correct analysis was obtained. Other parts of speech were all correctly recognised. Since the input consisted of all classes of words, there were false positives. The precision levels in cases of nouns, adjectives and adverbs were 94%, 81%, and 91% respectively; while that of verbs was down to 54%. An attempt to improve the precision for verbs increased it to 65% but with an adverse effect on the recall. The low results in the precision of verb analysis are primarily a result of rules that are not inclusive for all members in a class. In addition, there is no standard spelling, creating flexibility in spelling the same words one way or another.

The results show that even without more contextual information for purposes of disambiguation, the basic recall result is already very useful. The next stage of development is to incorporate the output of the analyser into a syntax–aware tagging utility; we predict that this will increase the precision result drastically.

References

Nega Alemayu and Peter Willett. 2002. Stemming of Amharic Words for Information Retrieval. Literary and Linguistic computing 17 (1), p. 1–17.

Atelach Alemu, Lars Asker, and Mesfin Getachew. 2003. Natural Language Processing for Amharic: Overview and Suggestions for a Way Forward. In Proceedings of the 2nd Workshop on Treebanks and Linguistic Theories, Växjö University, Sweden, November.

Getahun Amare. 1989. Amarenja Souasou Bek'elal Ak'erareb (Amharic Grammar Presented in an Easy Way). Addis Abbaba: Business Printing Press.

Kenneth R. Beesley and Lauri Karttunen. 2003. Finite State Morphology. Stanford: CSLI.

M. Lionel Bender. 1968. Amharic Verb Morphology: A Generative Approach. PhD. Dissertaion, Graduate School of Texas.

M. Lionel Bender and Hailu Fulas. 1978. Amharic Verb Morphology. East Lansing: Michigan State University, African Studies Center.

Girmay Berhane. 1992 Word Formation in Amharic. Journal of Ethiopian Languages and Literature. No. 2. p. 50–75

C. H. Dawkins. 1960. The Fundamentals of Amharic. Sudan Interior Mission, Addis Ababa, Ethiopia.

Sisay Fissaha and Johann Haller. 2003. Amharic Verb Lexicon in the Context of Machine Translation. TALN, p. 183–192.

Sisay Fissaha and Johann Haller. 2003. Application of Corpus-based Techniques to Amharic Texts. In Proceedings of the 10th Conference on Traitement Automatique des Langues Naturelles, volume 2, p. 173-182, Batz-sur-Mer, France, June.

Lauri Karttunen. 2003. Finite–State Technology. In: The Oxford Handbook of Computational linguistics, p. 339–357. Oxford University Press.

Laura Kataja and Kimmo Koskenniemi. 1988. Finite State Description of Semitic Morphology: a case study of Ancient Akkadian. Proceedings of 12th Conference on Computational Linguistics, I, p. 313–315.

Martin Kay. 1987. Nonconcatenative Finite–State Morphology. EACL 1987, p. 2–10.

George Anton Kiraz. 2000. Multitiered Nonlinear Morphology Using Multitape Finite Automation: A Case Study on Syriac and Arabic. Computational Linguistics, Volume 26 Issue 1, p. 178–181.

Kimmo Koskenniemi. 1984. A General Computational Model for Word–Form Recognition and Production. Proceedings of the 22nd Conference of the ACL, p. 178–181, California.

Habte Mariam Markos. 1991–1994. Towards the Identification of the Morphemic Components of the Conjugational Forms of Amharic. Proceedings of the Eleventh International Conference of Ethiopian Studies. Addis Ababa, vol. 1, p. 465–479.

Harald Trost. 2003. Morphology. In: The Oxford Handbook of Computational linguistics, p. 25–47. Oxford University Press.

Baye Yimam. 1999. Root Reductions and Extensions in Amharic. Ethiopian Journal of Languages and Literature, No 9, p. 56–88.

Baye Yimam. 1995. yamargnasewasew (Amharic Grammar). Addis Ababa: EMPDA.

SpeechLab 2.0 – A High-Quality Text-to-Speech System for Bulgarian

Maria Andreeva and **Ivaylo Marinov** and **Stoyan Mihov**
Institute for Bulgarian Language,
Bulgarian Association for Computational Linguistics,
Institute for Parallel Processing
Bulgarian Academy of Sciences
stoyan@lml.bas.bg

Abstract

SpeechLab 2.0 is a high quality and very efficient text-to-speech engine for Bulgarian, which applies a sophisticated rule-based approach for tokenization, part-of-speech tagging, phonetization and prosody annotation. All stages of the text processing, including grammatical and accentual dictionary application, contextual rules for grammatical and prosodic annotations and phonetization, are implemented as a pipeline of finite state bimachines and subsequential transducers. The main advantage of the new method for text processing for speech synthesis is its very high efficiency for complex linguistic analysis. Some specific aspects of the Bulgarian phonology and prosody are presented as well. Using the FD-PSOLA algorithm for signal generation our system delivers Bulgarian speech with very high intelligibility and naturalness by a processing speed of 960 words/sec.[1]

Keywords: text to speech, text processing, finite state devices, natural language processing

1 Introduction

A modern text-to-speech (TTS) system has to provide a number of features including sophisticated text analysis and robustness, to deliver intelligibility and naturalness of the generated speech and to achieve high performance and robustness. Currently a number of high quality TTS systems for English, French and German have been developed (Santen *et al.* 97; Dutoit 94). The development of a high quality TTS system for a new language is still a challenge.

To the best of our knowledge, no comprehensive achievements have been made so far in the development of a high-quality speech synthesizer for Bulgarian. The first Bulgarian TTS system called "Betsy" has been developed by Borislav Zahariev in the framework of his Ph.D Thesis (Zahariev 93). While being quite an achievement for the time it was created, Betsy's voice sounds unnatural and applies only a few linguistic rules, which makes it hard to understand and does not suffice the need for a fast and reliable text-to-speech system. In (Totkov *et al.* 03) the authors report for another Bulgarian TTS system. This system also suffers from the lack of more advanced text analysis techniques for accent and prosody annotation.

In this paper we present a new approach for text processing, which we use in the SpeechLab 2.0 TTS system. We apply a rule-based approach implemented by a pipe finite state devices (Kaplan & Kay 94; Roche & Schabes 97). Although finite state devices have been used extensively for phonetic translation (Sproat 97; Laporte 97), our method differs significantly in respect to the following three features: First we use only subsequential transducers and bimachines, which have been constructed by determinization of regular relations. Second, all the finite-state devices (including the dictionaries) are text rewriting devices – the whole text is deterministically transduced by each of the transducers. Third, starting from the input text all finite-state devices are simply applied in a pipe, which provides a simple implementation, application of complex linguistic rules and dictionaries and results in very high efficiency.

The first phase of text processing proceeds with the GrammLab system (Doychinova & Mihov 04) for Part-of-Speech annotation. At the second phase we apply an Bulgarian accentual dictionary with 1 million wordforms, an English pronunciation dictionary with 60000 wordforms and a user defined custom pronunciation dictionary. The third phase proceeds with 98 contextual rules for phonetization, accent determinization, unknown words processing. At last 91 rules for prosody annotation are applied. For the signal synthesis we use a diphone concatenation system based on the well-known FD-PSOLA method (Moulines & Charpentier 90), which delivers a high quality Bulgarian speech.

In the next section we present the general text analysis provided by the system. Section 3 describes the details of the phonetization procedure

[1]This work was partly funded by a grant from Microsoft.

for Bulgarian. We present the Bulgarian prosody annotation in Section 4. The technology used for text processing in the SpeechLab 2.0 system is presented in Section 5 and the implementation details are given in Section 6. Finally the conclusion presents some general comments and directions for further work.

2 General text analysis

The general text analysis component in SpeechLab 2.0 consists of four stages – tokenization, grammar dictionary application, unknown words guessing and contextual part-of-speech (POS) disambiguation. At the end the input text is annotated with token tags and POS tags. We have used the tagger described in (Doychinova & Mihov 04) for the implementation of this subsystem.

The overall precision of the tagger is over 98.4% for full disambiguation and the processing speed is over 34K words/sec on a personal computer.

Below we give a short overview of the four stages.

2.1 Tokenizer

The system uses a sophisticated tokenizer, which marks and categorizes tokens as numeral expressions, dates, hours, abbreviations, URLs, items, punctuation, words, abbreviations etc. Words in Latin and Cyrillic are differently marked when capitalized or upper case. Punctuation is classified according to the type and the function of the sign. Sentence boundaries are recognized as well. The tokenizer is implemented as a composition of 53 contextual rewriting rules composed into 4 bimachines.

2.2 Grammatical dictionary

The dictionary assigns to each dictionary word in the text its ambiguity class and its initial tag. The initial tag is usually the most probable tag for the ambiguity class. For example, the class which consists of adverbs and neutral adjectives gets adverb as a most probable tag, because in an representative corpus these words are adverbs 2 times more often than adjectives. The dictionary contains about 1 million wordforms. Since each wordform can occur in lower case, upper case or capitalized, the dictionary contains 3 millions strings. It is implemented as one (big) rewriting rule represented by a subsequential transducer using the construction method presented in (Mihov & Schulz 04).

2.3 Unknown word guesser

The guesser handles the words that are not in the dictionary. The constructed rules are analyzing the suffix for guessing the word's morphology and for assigning the initial part-of-speech tag and the ambiguity class. The guesser is implemented as a composition of 73 rewrite rules all compiled into a single bimachine.

2.4 Contextual Disambiguation Rules

The part-of-speech ambiguity ratio for a representative Bulgarian corpus is 1.51 tags/word, which means that in average every second word is ambiguous. For solving the ambiguities we apply 148 contextual rules, which can utilize part-of-speech, lexical or dictionary information on the context. All 148 contextual rules are composed into 4 bimachines, which we apply in the pipeline.

3 Phonetization

We present a brief description of the phonetic inventory of Bulgarian, with a discussion of the approach used to select and segment phonetic units for the system. One of the main specifics of Bulgarian language is the combination of phonetic and morphological orthographic principle. This is the reason for us to choose the description for text-to-speech processing by means of phonetic and grammatical rules that rewrite the text by annotating it with phonetic description.

For the development of the formal rules a morphologically tagged corpus of 50000 words was transcribed. A set of rewrite rules were developed for the preprocessing of the orthographic format of written text into a phonetic alphabet, serving as input to the speech synthesizer. Our work was oriented towards capturing the specifics of the Literary Bulgarian speech and attaining intelligibility and expressiveness of the generated speech.

3.1 Phonetic Description of Bulgarian

The first stage of formal description included an large size Bulgarian accentual dictionary and the elaboration of letter-to-sound rules that capture the close correspondence existing between letters (symbolic representation) and pronunciation.

One of the specifics of the Bulgarian language is the free word stress. The change of stress position in the different forms of the word, homographs and the accent features and specifics of some clitics in different word order constructions gives rise

to many problems in text-to-speech preprocessing.

Stress movement is accounted for by means of a dictionary consisting of over 1 million units with marking of primary and supplementary stresses. Most cases of homography are resolved by the tagger. For the description of the phonetic peculiarities of some specific word order models and grammatical constructions a set of rules were created as well as special dictionaries of some phrases. Those rules have been used to resolve cases like the one in "Добър вечер" (Good evening) where in the adjective "добър" (good) the vowel "о" is stressed, while usually "ъ" is the stressed vowel. Various combinations of prepositions, conjunctions and particles are analyzed for stress movements as well.

For the purposes of our text-to-speech system we adapted the traditional phonological system of Bulgarian to a phonetic system consisting of 45 phonemes, including 39 consonants and 6 vowels with additional accounting for certain specifics of speech such as stressed and unstressed vowels. etc. The precise number of Bulgarian phonemes was determined with regard to the necessary-and-sufficient condition in the representation of the groups of vowels and consonants needed for the diphone concatenation and its later implementation in the generation of natural sounding Bulgarian speech. Speech was modeled as a linear sequence of these phones. Table 3.1 shows the correspondence between the phonemes, their pronunciation being represented by the corresponding symbol established in the standard Phonetic Alphabet - SAMPA, and the characters of the Bulgarian alphabet, and an example of the occurrence of each phoneme in a Bulgarian word.

Consonants			
Non-palatalized	Palatalized	Vowels	Semivowel
б[b] баба	[p'] пял	ъ[@] пън	[j] ял
в[v] вада	[b'] бял	о[o] кон	
г[g] гарга	[t'] тях	у[u] тур	
д[d] домат	[d'] дял	е[e] фес	
ж[Z] жаби	[k'] кяр	и[i] пир	
дж[dZ] джам	[g'] гюл	а[a] чар	
з[z] змия	[ts'] цял		
дз[dz] дзвън	[dz'] дзян		
п[p] папка	[f'] фют		
Ф[f] фонтан	[v'] вял		
к[k] котка	[s'] сял		
т[t] телефон	[z'] зян		
Ш[S] шапка	[x'] хюм		
Ч[tS] чаша	[m'] мях		
с[s] сам	[n'] ням		
ц[ts] цаца	[l'] лях		
х[x] хора	[r'] ряз		
р[r] роза			
л[l] лекар			
м[m] мама			
н[n] нос			

Table 1: Bulgarian Phonemic System.

3.2 Structural rule-based representation of Bulgarian speech

Rules are used to perform accurate phonological description of Bulgarian speech. The representation of Bulgarian speech is based on a set of phonological distinctive features – atomic units by means of which phonemes are described. The groups of consonants and vowels are further divided into subgroups on the basis of the correlative phonetic features regularly occurring in literary speech. The determination of the set of vowels differs from the traditional approach in that it includes all stressed vowels and their unstressed allophones, the reduction feature (in unstressed position) conforming to reduction rules established in literary tradition. The group of consonants (obstruents) is divided into subgroups based on the correlative characteristics +voice and -voice; +palatalness and -palatalness. The glide [j] groups together the sounds represented in spelling by the letters "й", "ъ" and the glide formant of the sounds represented graphically by "я", "ю".

Other special rules describe the letters standing for two sounds like "щ", "ю", "я" or single phonemes represented by digraphs such as "дз", "дж". In these cases the letters may at the same time represent two distinct sounds occurring on a morpheme boundary. The problem of ambiguity in these cases is sufficiently resolved with special rules and a dictionary of words that include these digraphs, most of which are in fact rare or dialect words.

The organization of the phonemes into groups is the basis for the elaboration of formal rules for resolving problems such as stressed or unstressed vowels, voice assimilation and palatalization of consonants.

The possible sound alternations are described by means of context rules, based on the contradistinction of correlative features. The assimilation in Bulgarian may be described as an anticipatory (regressive) adoption ("copying") of a certain feature by the target sound from the sound in the immediate environment following it. The source of assimilation is the second sound in the sequence. One of the most important features of Bulgarian consonants is the alternation of voiced and voiceless consonants in certain positions in words.

The rules for the combination of two or more

consonants replace voiced consonants with voiceless and vice versa under certain conditions (the influence of the first neighboring consonant regarded backwards). For example the word "отдавна" (a long time ago), in the combination of "тд", "д" voices "т" and the word is pronounced [odd‘avna]. In the word “безсилен” (feeble), in the combination "зс", "с" influences "з" and the word is pronounced [bess‘ilen].

A special place in the description is determined for the phonemes "в" and "в’", after which both voiced and voiceless consonants can be pronounced - for example “звяр” [zv’‘ar] (beast), “свят” [sv’‘at] (world), “двор” [dv‘or] (yard), “творчество” [tv‘ortSestvo] (creative work).

In connected speech words form a continuous speech chain. Another important phonetic characteristic of the Bulgarian language, which should be described with formal rules, is the word boundaries assimilation. This is the reason for the characteristic changes that occur word initially and word finally in neighboring words. Each word or a set of words is regarded as a string of phonemes, whose boundaries are determined by special rules. For example, some consonants are not influenced by the assimilation under certain conditions.

A special case is presented by the negation particle "не" [ne] (not), which is unstressed in the construction "Той не ме удари" [t‘oj ne m‘e ud‘ari] (He has not hit me.), since it is followed by the short personal pronoun "ме" (me). In the fragment "не зелен, но не и червен" [n‘e zel‘en no n‘e i tServ‘en] (not green but not red as well) the particle is stressed, because it is followed by an adjective. The formal rules for the particle "не" (not) also subsume grammatical constructions with the auxiliary verb "съм" (to be), otherwise unstressed in all its forms, but assuming stress in all forms to the exception of the 3d person, singular, Present form “е” [e] (is), if it is preceded by the negative particle "не" (not), e. g. "Той е човек. Ти не си човек" [t‘oj e tSov‘ek t‘i ne s‘i tSov‘ek] (He is a man. You are not a man.). According to the their stress features function words were described and classified in three classes: always stressed, always unstressed and stressed under specific conditions.

Special rules for the determination of stress positions are needed for the definite forms of masculine nouns and definite masculine word forms of adjectives, numerals in some of which the definite article receives the word stress - "‘а/‘я" "‘ят" (pronounced [‘@/j‘@] and [j‘@t]) for example: "мъж" [m‘@Z] (man) - "мъжа" [m@Z‘@] (the man) - "мъжът" [m@Z‘@t] (the man); "ден" [d‘en] (day) - "деня" [den’‘@] (the day) - "денят" [den’‘@t] (the day). The same problem occurs also with first person, singular and third person plural present tense forms of Bulgarian verbs belonging to I and II conjugation type ending in "‘а/‘я" and "‘ат/‘ят" (pronounced [‘@/j‘@] and [‘@t/j‘@t]), for example "чета" [tSet‘@] (I read) - "чет‘ат" [tSet‘@t] (they read); "плат‘я" [plat’‘@] (I pay) - "плат‘ят" [plat’‘@t] (they pay).

Many problems are caused by so called homographs – words that are represented identically but differ in pronunciation. When the corpus was phonetized some interesting cases were described and classified in groups:

- Nouns in which the change in stress leads to a change in the category gender for example: “техника” (technician / technics), “физика” (phisician / phisics)
- Words in which the change in stress changes their part of speech, for example: “трупа” (the corpse / group), “душа” (the shower / soul)
- Verbs in which change in stress leads to change of the verb tense, for example “заделя”, “заделят” (put aside), “споделя”, “споделят” (share)
- Words that coincide in all their grammatical forms, but have different semantics for example “вълна” (wool / wave), “пара” (steam / penny), “блажен” (fat / blessed)

These cases cannot be resolved by the tagger and context rules, but can be partially reduced by placing secondary (weaker) stress to the vowels in question, or rather vowel length markers, to the wordform already recognized by the system as a homograph. In such a way an auditory impression of a correct pronunciation of these forms is created, while at the same time the possibility for a completely wrong pronunciation of the synthesized word in the concrete instance is eliminated.

4 Prosody annotation

We consider the intonation as a set of prosodic elements characterizing human speech and its re-

alization in the act of the verbal communication. The grouping of the prosodic elements according to their significance and their classification in different groups shows that some of them are obligatory for the naturalness and intelligibility of speech, while others (not obligatory in this respect) are characteristic for individual speech and express the attitude of the speaker. The first group includes the different speech melody types characterizing the main communicative types of sentences in Bulgarian, the correct putting of stress of words and definition of pauses with respect to the punctuation as well as the logical accentuation of words. Non-obligatory prosodic elements include emotional speech indications such as sudden changes of voice, changes in voice power, lengthened vowels, slower or faster articulation of some words in the sentence.

Along with the description of the regularities of sound patterns of Bulgarian, phonology is concerned with the more abstract description of the intonation patterns viewed as a suprasegmental property of speech indispensable for the intelligibility of automatically generated speech.

We used a testing tool for interactive modeling of prosody changes to the synthesized speech. Using it we conducted a number of intonation modeling tests and experiments for the elaboration of formal rules describing the Bulgarian prosody.

4.1 Pauses

The rules determining the pauses in a string play a very important role in speech synthesis. The generated pauses should not interrupt speech or change the meaning, but should add to the melody contour of the segments. In different types of sentences different pauses operate. Their major role is in the process of the definition of the intonation segments' boundaries. All these factors were taken in consideration in the formulation of the formal pause definition rules to account for the different cases:

- the phoneme string is temporarily interrupted by a new line, new page or new paragraph mark;
- different length of pauses is associated with the different punctuation marks;
- in cases where two punctuation marks are combined the longest pause is considered;
- in sentences, which exceed a given number of wordforms determined by physiological factor;
- alternation on word boundaries where assimilation is forbidden a short pause interrupts the phoneme string. For example the cases when a word ends with "щ", "жд" и "ст";

4.2 Intonation

The rules describe the intonation contour of the four main communicative types of sentences in Bulgarian and their further subdivision into more detailed subtypes:

- Intonation of a text segment ending with a period "." - for example: "Той ходеше по улицата, по която беше вървял преди година." (He was walking on the street, on which he walked a year before.)
- Intonation of a text segment ending with a question mark "?" which does not contain question word or particle - for example: "Вие мразите ония горе? Не знаете нищо? Имате домашно по математика?" (You hate those people above? You don't know anything? You have a homework on mathemathics?)
- Intonation of a text segment ending with a question mark "?", containing at least one question word - for example: "Колко струва това палто? Къде зимуват раците? Кой отговаря за тази работа?" (How much costs this coat? Where do the crabs spend the winter? Who is responsible for this job?)
- Intonation of a text segment ending with a question mark "?", containing a question particle - for example: "Сега ли трябва да ходя на работа? Той нали ще дойде на време?" (Should I go to work now? He will come on time, won't he?)
- Intonation of a text segment ending with a question mark "?", containing both question word and particle - for example: "Кой знае колко е страдала, колко е мислила за него и колко дълго го е очаквала?" (Who knows how much she suffered, how much she was thinking about him and how long she was wating him?)

The methodology adopted for the representation of these groups is based on the proper determination of a minimal intonation segment - the intonema. By intonema we mean any stressed content word, which attracts proclitics or enclitics and determines their tonal value.

For example in a sentence containing one content word and ending with a punctuation mark ".", as "Във вазата." (In the vase.), the system will track the main word stress and will add the unstressed preposition "във" (in) to construct an intoneme and will define its intonation contour as rise-fall marked by the punctuation mark ".".

If the sentence comprises two intonation segments, the first one ending with ",", and the second with ".", as in "Трябва да изберем човек, в който да сме сигурни." (You have to choose a man, of whom we can be sure.), the program will determine the intoneme of the first segment, finishing with "," as rise-fall and will apply it to the intoneme preceding the comma. The second segment in this sentence will be synthesized applying the rules for melody contour determined by a punctuation mark ".".

The intoneme generation involves the application of additional rules that take into account the characteristics of speech resulting from factors such as the number of intonemes in an intonation segment and the length of individual intonemes. On the operation of these rules, the system generates rules for segment initial, segment final and segment internal intonemes, as well as for special intonemes such as question words, question particles, etc.

5 Text processing technology

As already mentioned the whole text analysis module is implemented as a pipeline of bimachines and subsequential transducers. The text is transduced by each of the finite state devices deterministically. After each step the text is enriched with additional information, which is used by the following steps. At the end the text is annotated with token tags, POS tags, annotations for phonemes and prosody.

For the construction of the rewriting rules we used the methods presented in (Kaplan & Kay 94; Gerdemann & vanNoord 99; Ganchev *et al.* 03). All the rules are specified in the form $\alpha \rightarrow \beta / L\_R$, where α, β, L and R are regular expressions. This means that each occurrence of a substring presented as α is replaced by β if it occurs in the context of substrings of L and R ((Kaplan & Kay 94)). For solving the conflicts we used the "first left longest match" strategy with the implementation given in (Gerdemann & vanNoord 99). The actual realization of 2-tape automata and the operations on them are implemented using the "one-letter automata" methodology given in (Ganchev *et al.* 03). The rules are then composed and represented by bimachines by the techniques presented in (Roche & Schabes 97). For the construction of the sequential transducer for the rewriting dictionary rules we applied the new method we developed and presented in (Mihov & Schulz 04).

We have developed a new rule compiler called "Siera" for practical implementation of the above functionality. This system allows the description and construction of all needed language resources including rule compilation from regular expressions to one-letter automaton; composition, union, concatenation of one-letter automata; conversion of functional one-letter automata to subsequential transducer or bimachine; compilation of rewrite dictionaries to susequential transducer and many others. In practice the entire text analysis is described as set of Siera scripts.

This methodology has provided a powerful, flexible and comfortable linguistic development environment while resulting to an exceptionally high performance.

6 Implementation details

The SpeechLab 2.0 system is implemented in ANSI C with a platform independent core. It was successfully tested under Linux and Windows.

The base pitch can be set in the range of one octave and the speed range is between two times slower and two times faster than normal. The usual options for reading the punctuation marks or for spelling given expressions are provided. For the fulfillment of specific requirements the system is supplied with a module for dynamic setting of specific configuration options. For example, the user can switch off the English dictionary, intonation, stresses and pauses between words, numbers can be read by digits, dates and abbreviation expansion can be controlled.

The size of text processing module is 120 MB and the size of the voice module is about 10 MB. Using memory map files the memory occupied in

the computer is about 15 MB.

The speed of the synthesis including the saving of 16KHz 16 bit audio stream is 963 words per second on a 3 GHz Pentium 4 computer running Linux. Even on a 100 MHz Pentium II computer with only 32 MB SpeechLab 2.0 was able to provide real-time synthesis.

7 Conclusion

The presented SpeechLab 2.0 system provides a high quality Bulgarian speech. People with visual disabilities have extensively tested the system. Currently it is available free of charge for all Bulgarian visually disabled people from "Horizonti" foundation and the Union of the Blinds in Bulgaria. In result of the tests SpeechLab 2.0 was acknowledged to fulfill the requirements of the visually impaired people.

Although the tests revealed a high quality of the synthesized speech, there is still room for improvement in a couple of directions. In the near future we plan to create a more sophisticated syntax analysis, better homograph resolution rules and more complex intonation contour annotation.

References

(Doychinova & Mihov 04) Veselka Doychinova and Stoyan Mihov. High performance part-of-speech tagging of bulgarian. In *Proceedings of AIMSA 2004, LNAI #3192*, 2004.

(Dutoit 94) T. Dutoit. High quality text-to-speech synthesis: A comparison of four candidate algorithms. In *Proceedings of ICASSP 94 (1)*, 1994.

(Ganchev *et al.* 03) Hristo Ganchev, Stoyan Mihov, and Klaus U. Schulz. One-letter automata: How to reduce k tapes to one. Technical Report CIS-Bericht, Centrum für Informations- und Sprachverarbeitung, Universität Munchen, 2003.

(Gerdemann & vanNoord 99) Dale Gerdemann and Gertjan van Noord. Transducers from rewrite rules with backreferences. In *Proceedings of EACL 99*, 1999.

(Kaplan & Kay 94) Ronald Kaplan and Martin Kay. Regular models of phonological rule systems. *Computational Linguistics*, 20(3):331–378, 1994.

(Laporte 97) Eric Laporte. Rational transductions for phonetic conversion and phonology. In *E. Roche and Y.Schabes eds., Finite-State Language Processing*. MIT Press, 1997.

(Mihov & Schulz 04) Stoyan Mihov and Klaus U. Schulz. Efficient dictionary-based text rewriting using sequential transducers. *Natural Language Engineering*, Submitted, 2004.

(Moulines & Charpentier 90) E. Moulines and F. Charpentier. Pitch synchronous waveform processing techniques for text-to-speech synthesis using diphones. *Speech Communication*, 9(5-6), 1990.

(Roche & Schabes 97) Emmanuel Roche and Yves Schabes. *Finite-State language processing (Introduction)*. MIT Press, 1997.

(Santen *et al.* 97) Santen, Sproat, Olive, and Hirschberg. *Progress in Speech Synthesis*. Springer-Verlag, 1997.

(Sproat 97) Richard Sproat. *Multilingual Text-to-Speech Synthesis, the Bell Labs Approach*. Kluwer, 1997.

(Totkov *et al.* 03) G. Totkov, D. Blagoev, and V. Angelova. Towards bulgarian text-to-speech system. In *Proc. of the Int. Conference "10 years Computer Systems Dept."*, 2003.

(Zahariev 93) Borislav Zahariev. *Microcomputer Systems for Text-to-Speech Synthesis*. Unpublished PhD thesis, Bulgarian Academy of Sciences, 1993.

Multi-Perspective Evaluation of the FAME Speech-to-Speech Translation System for Catalan, English and Spanish

Victoria Arranz*, Elisabet Comelles[†] and David Farwell[†°]

* ELDA – Evaluation and Language Resources Distribution Agency
55-57, rue Brillat Savarin, 75013 Paris, France
[†]TALP Research Centre, Universitat Politècnica de Catalunya
C/ Jordi Girona 1-3, 08034 Barcelona, Spain
[°] Institució Catalana de Recerca i Estudis Avançats
arranz@elda.org,{comelles,farwell}@lsi.upc.edu

Abstract

This paper describes the final evaluation of the FAME interlingua-based speech-to-speech translation system for Catalan, English and Spanish. It is an extension of the already existing NESPOLE! System that translates between English, French, German and Italian. However, the FAME modules have now been integrated in an Open Agent Architecture platform, thus offering a number of technical advantages for a multi-modal environment. The article describes the system architecture and the components of the translation module including the speech recognition component, the analysis chain, the generation chain and the speech synthesizer. We describe three types of evaluation (task-oriented, performance-based and of user satisfaction) and present the results of a multi-perspective evaluation of our system. We also compare the results of the system with those obtained by a stochastic translator developed independently within the FAME project.

1 Introduction

The FAME interlingual speech-to-speech translation system (SST) for Catalan, English and Spanish has been developed at the Universitat Politècnica de Catalunya (UPC), Spain, as part of the recently completed European Union-funded FAME project (Facilitating Agent for Multicultural Exchange) that focused on the development of multi-modal technologies to support multilingual interactions (see http://isl.ira.uka.de/fame/ for details). The FAME system is an extension of the existing NESPOLE! translation system (Metze, *et al*. 02; Taddei, *et al*. 03) to Catalan and Spanish in the domain of hotel reservations. At its core is a robust, scalable, interlingual speech-to-speech machine translation system having cross-domain portability that allows for effective translingual communication in a multi-modal setting. However, despite being originally developed within the NESPOLE! framework, it was later ported to an Open Agent Architecture that resulted in a number of benefits (see Section 2), including a speed up of the end-to-end translation process.

The main advantage of following an interlingual approach lies in the ease with which new languages may be added to the translation system. This was important considering that a) efforts reported here required a systematic and efficient addition of two new languages (and where the coverage needed to be compliant with the four previously developed languages in NESPOLE!), b) there were strict time restrictions that required a final system running at a large public event 6 months before project completion[1] and c) there was a very limited amount of development data, given the difficulty of obtaining and preparing it. With interlingua-based approaches only analysis and generation grammars need to be developed for the new languages, thus avoiding the development of transfer modules for every language pair involved. Furthermore, the developers do not need to be bilingual: only a monolingual source-language analysis developer or monolingual target-language generation developer is required for each language.

The complexity of dealing with spontaneous speech was also something that had to be taken into account. Some of the main problems to be overcome when translating spontaneous speech were disfluencies, incomplete sentences, non-grammatical sentences, etc. And this kind of phenomena was also a major reason for using an interlingual approach based on semantics, since it allows the translation of sentence fragments, non-

[1] A public demonstration of the end-to-end translation system took place at the Forum of Cultures in Barcelona, during last July 2004.

grammatical sentences etc., without being totally dependent on well-formed syntax.

Portability was also an issue taken into account as it is one of the main drawbacks generally attributed to interlingual systems. However, in the case of our system this was not the case. The structure of the grammars presents a clear division between the rules and lexical items that are portable to other domains and those that are task-specific. This allowed us to develop the translation modules quickly and efficiently and to port newly acquired vocabulary and forms of expression to the other languages. In addition, other partners in the project were able to port the Spanish and Catalan components to a very different domain: the medical domain (Schultz, *et al.* 04).

The interlingua used for the FAME SST system is Interchange Format (IF), the interlingua used by the C-STAR Consortium (see http://www.c-star.org for details) and which has been adapted for this effort. Its central advantage for representing dialogue interactions such as those typical of speech-to-speech translation systems is that it focuses on identifying the speech acts and the various types of requests and responses typical of a given domain. Thus, rather than capturing the detailed semantic and stylistic distinctions, it characterizes the intended conversational goal of the interlocutor (for a full description of the IF, please refer to Levin, *et al.* 02). Even so, in mapping to IF it is necessary to take into account a wide range of structural and lexical properties related to Spanish and Catalan. For further details, please refer to (Comelles 04) and (Arranz, *et al.* 05a: Arranz, *et al.* 05b).

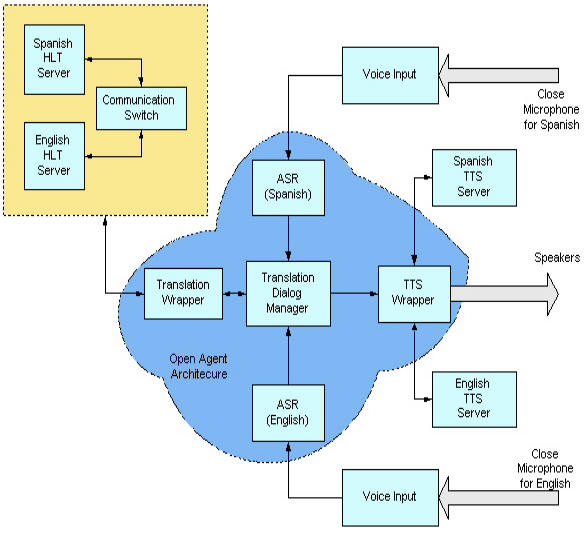


Figure 1: Open Agent Architecture of the Translation System

2 System Architecture

Although the system architecture was initially based on NESPOLE!, all of the modules have now been integrated in an Open Agent Architecture platform, as shown in Figure 1 (Holzapfel, *et al.* 03, for details see http://www.ai.sri.com/~oaa). This type of multi-agent framework offers a number of technical features for a multi-modal environment that are highly advantageous for both system developers and users, specially when considering the complex number and nature of the modules that needed to be integrated within the full FAME project (e.g., modules for image and video processing, speech recognition and synthesis, information retrieval, topic detection, etc.).

Broadly speaking, the FAME system consists of an analysis component and generation component. The analysis component automatically transcribes spoken source language utterances and then maps that transcription into an interlingual representation. The generation component then maps from interlingua into target language text and then produces a synthesized spoken version of that text.

For both Catalan and Spanish speech recognition, we used the JANUS Recognition toolkit (JRTk) developed at UKA and CMU (Woszczyna, *et al.* 93). For the text-to-text component, the analysis side utilizes the top-down, chart-based SOUP parser (Gavaldà 00) with full domain action level rules to parse input utterances. Natural language generation is done with GenKit, a pseudo-unification based generation tool (Tomita *et al.* 88). For both Spanish and Catalan, we use a Text-to-Speech (TTS) system fully developed at the UPC, which uses a unit-selection based, concatenative approach to speech synthesis.

For the initial development of the Spanish analysis grammar, the already existing NESPOLE! English and German analysis grammars were used as a reference point. Despite using these grammars, great efforts were taken to overcome important differences between English and German and the Romance languages dealt with. The Catalan analysis grammar, in turn, was adapted from the Spanish analysis grammar and, in this case, the process was rather straightforward. The generation grammar for Spanish was mostly developed from scratch, although some of the underlying structure was taken from the NESPOLE! English generation grammar. Language-dependent properties such as word order, gender and number agreement, etc. needed to be dealt with representationally but, on the whole, starting with existing structural descriptions was useful. On the other hand, the

generation lexica play a very major role in the generation process and these had to be developed from scratch. Again, however, the Catalan generation lexicon was adapted from the Spanish directly with almost no significant complication.

3 Evaluation

The evaluation performed was done on real users of the SST system, in order to:

- examine the performance of the system in as real a situation as possible, as if it were to be used by a real tourist trying to book accommodation in Barcelona,
- study the influence of using automatic speech recognition (ASR) in translation,
- compare the performance of a statistical approach and an interlingual approach in a restricted semantic domain and for task of this kind[2],
- investigate the relevance of certain standard evaluation methods used in statistical translation when applied to evaluate interlingual translation.

3.1 Evaluation: Data Recording and Treatment

Prior to the evaluation *per se*, a number of tasks had to be done to obtain the necessary data. These included dialogue and data recording during real[3] system usage; adapting the translation system to register every utterance from the different translation approaches and from ASR; recruiting people to play the roles of the users; designing the scenarios for the users; designing the sequence of events for the recording sessions; transcribing all speech data; etc.

Conversations took place between an English-speaking client and a Catalan- or Spanish-speaking travel agent. Twenty dialogues were carried out by a total of 12 people. Of these, 10 people were completely inexperienced with respect to the task and unfamiliar with the system, while 2 were familiar with both the task and the system. The former (the 10 speakers) participated in 2 dialogues each and the latter (the other 2) participated in 10 dialogues each. That way, each dialogue would resemble a real-situation dialogue where one of the speakers would always be familiar with the task and the system while the other one would not. It should also be added that all English speakers recruited for the evaluation were non-native speakers of the language. We consider this realistic as most of the potential clients who might use such a system would actually be from non-English speaking countries. Although some of them had a very high proficiency in English, this was not the case with all of them, and it should be noted that the results from automatic speech recognition and translation have suffered from this.

Five different scenarios were designed per speaker (agent or client) and they were available in all relevant languages (agent scenarios in Catalan and Spanish and client scenarios in English). Before starting the recording of the data, speakers were provided with very basic knowledge about the system, such as where to click to start or stop recording, where to find the necessary information regarding the scenarios, and so on. Computer screens only showed the user their own scenario related information and system interface. The system interface provided them with the ASR output of their own contribution and the translation output (from both the interlingual and statistical systems) of the other user's utterances. The former allowed the speakers to check if the ASR had recognised their utterances properly and thus allow for translation to go on or intervene before communication failure took place, say by repeating their utterance. The latter allowed them to have the two translation outputs from the other speaker's utterances on the screen given that the synthesiser only provided one of the translations (choosing between the two resulting translations based on a very simple algorithm).

Dialogue recording took place in a room set up for that purpose. Speakers were situated separately with their respective computers in such a way that they could only view their own computer screen. Once recording was finished and all conversations were registered, the following steps were followed to prepare the data for evaluation:

- All speech files were transcribed and concatenated into dialogue units. That is, all utterances were grouped according to the dialogue they belonged to. During transcription, speech disfluencies were also marked and all utterances were tagged.

[2] A statistical system was built in parallel within the FAME project so as to compare approaches, both in terms of results and efforts. Refer to (Arranz et al., 2005) for full details.

[3] In order to perform a quantitative evaluation of the system, a number of scenarios as real as possible were set up with external users and in reality-resembling situations.

- Reference translations were created for each speaker+dialogue file, so as to evaluate translation using BLEU and mWER metrics.

3.2 Task-Oriented Evaluation Metrics

A task-oriented methodology was developed to evaluate both the end-to-end system (with ASR and TTS) and the source language transcription to target language text subcomponent. An initial version of this evaluation methodology had already proven useful during system development since it allowed us to analyse content and form independently and, thus, contributed to practical system improvements.

The evaluation criteria used were broken down into three main categories (*Perfect, Ok* and *Unacceptable*), while the second was further subdivided into *Ok+, Ok* and *Ok-*. During the evaluation these criteria were independently applied to *form* and to *content*. In order to evaluate *form*, only the generated output was considered by the evaluators. To evaluate *content*, evaluators took into account both the input utterance or text and the output text or spoken utterance. Thus, the meaning of the metrics varies according to whether they are being used to judge *form* or to judge *content*:

- **Perfect**: well-formed output (*form*) or full communication of speakers' information (*content*).
- **Ok+/Ok/Ok-**: acceptable output, grading from some minor error of *form* (e.g., missing determiner) or missing information (*Ok+*) to some more serious problem of *form* or *content* (*Ok-*) resulting in awkwardness or important missing information.
- **Unacceptable:** unacceptable output, either essentially unintelligible or simply totally unrelated to the input in terms of information content.

3.2.1 Task-Oriented Evaluation Results

The results obtained from the evaluation of the end-to-end translation system for the different language pairs are shown in Tables 1, 2, 3 and 4, respectively. After studying the results we can conclude that many of the errors obtained are caused by the ASR component. However, it should be pointed out that results remain rather good since, for the worst of our language pairs (English-Spanish), a total of 62.4% of the utterances were judged acceptable in regard to content. This is comparable to evaluations of other state-of-the-art systems such as NESPOLE! (Lavie *et al*. 02), which obtained slightly lower results and were performed on Semantic Dialog Units (SDUs)[4] instead of utterances (UTTs), thus simplifying the translation task. The Catalan-English and English-Catalan pairs were both quite good with 73.1% and 73.5% of the utterances being judged acceptable, respectively, and the Spanish-English pair performed very well with 96.4% of the utterances being acceptable.

SCORES	FORM	CONTENT
PERFECT	70.59%	31.93%
OK+	5.04%	15.12%
OK	6.72%	9.25%
OK-	9.25%	16.80%
UNACCEPTABLE	8.40%	26.90%

Table 1: Evaluation of End-to-End Translation (with ASR) for the Catalan-English Pair. Evaluation based on 119 UTTs.

SCORES	FORM	CONTENT
PERFECT	92.85%	71.42%
OK+	4.77%	11.90%
OK	1.19%	7.14%
OK-	0%	5.96%
UNACCEPTABLE	1.19%	3.58%

Table 2: Evaluation of End-to-End Translation (with ASR) for the Spanish-English Pair. Evaluation based on 84 UTTs.

SCORES	FORM	CONTENT
PERFECT	64.96%	34.19%
OK+	15.39%	11.97%
OK	8.54%	14.52%
OK-	5.12%	12.82%
UNACCEPTABLE	5.99%	26.50%

Table 3: Evaluation of End-to-End Translation (with ASR) for the English-Catalan Pair. Evaluation based on 117 UTTs.

SCORES	FORM	CONTENT
PERFECT	64.80%	17.60%
OK+	4.80%	10.40%
OK	12.00%	18.40%
OK-	8.80%	16.00%
UNACCEPTABLE	9.60%	37.60%

Table 4: Evaluation of End-to-End Translation (with ASR) for the English-Spanish Pair. Evaluation based on 125 UTTs.

[4] SDUs are smaller meaning-porting units, where usually several of them are contained within a dialogue utterance.

As seen in these tables, better results have been obtained for the Spanish-English and Catalan-English directions. We should point out that Catalan and Spanish Language Models used for speech recognition were developed specifically for this task while the English Language Models used were those provided by the project's partners. In addition, we should also consider that a great effort was devoted to the development of both Catalan and Spanish analysis and generation grammars. However, English analysis and generation grammars were not so developed. Because generation from a well-formed IF is more robust than from a fragmented IF, better analysis components tend to result in better overall throughput. These two factors are why better results are achieved when Spanish and Catalan are source languages.

3.3 Statistical Evaluation Metrics

Evaluation of our end-to-end speech-to-speech translation system has also been carried out by means of statistical metrics such as BLEU and mWER. We anticipated that results would drop drastically when compared to the manual evaluation presented in Section 3.2. and this turned out to be the case. This considerable drop is due to a number of factors:

- Resulting translations are compared to a single reference translation, thus failing to account for language variety and flexibility and negatively impacting results,
- BLEU and mWER penalise all diversions from the reference translation, even if these result from minor errors that do not affect intelligibility,
- The English-speaking volunteers for the evaluation were not native speakers, which considerably complicated the task of speech recognition at some points.

3.3.1 Statistical Evaluation Results

Results obtained both from the statistical approach and the interlingua-based one are shown below in Tables 5 and 6, respectively:

Language Pairs	# sentences	mWER	BLEU
CAT2ENG	119	74.66	0.1218
ENG2CAT	117	77.84	0.1573
SPA2ENG	84	61.10	0.1934
ENG2SPA	125	80.95	0.1052

Table 5: Results of the Statistical MT System

Language Pairs	#sentences	mWER	BLEU
CAT2ENG	119	78.98	0.1456
ENG2CAT	117	81.19	0.2036
SPA2ENG	84	60.93	0.3462
ENG2SPA	125	86.71	0.1214

Table 6: Results of the Interlingua-based MT System

The results are consistent with respect to the relative performance of the different systems in terms of language pairs. The Spanish-to-English systems, both statistical and rule-based, performed best. The English-to-Spanish systems, both statistical and rule-based, performed the worst. The Catalan-to-English and English-to-Catalan systems performed somewhere in between with the latter slightly outperforming the former.

As for the relative performance of the statistical systems as opposed to the rule-based systems, the results are entirely contradictory. The mWER scores of the statistical system are consistently better than those of the rule-based systems (apart from the Spanish-to-English case where the two systems essentially performed equally). On the other hand, the BLEU scores of the rule-based system are consistently better than those of the statistical systems. It is unclear how this happened although it is likely that since the BLEU metric rewards overlapping strings of words (as opposed to simply matching words) that the rule-based systems produced a greater number of correct multiword sub-strings than the statistical systems did. In any case, were it not for the low performance of all the systems and the very limited size of the test corpus, this would be a very telling result with regard to the validity of the evaluation metrics.

3.4 User Satisfaction Evaluation

A final evaluation of user satisfaction was carried out both from a quantitative and a qualitative point of view. Both of them are detailed below. Evaluation is carried out on a system which provided the user with both the interlingual-based and statistical-based translations.

3.4.1 Quantitative Study

The quantitative study of the user satisfaction consists in measuring the results obtained from the end-to-end translation system according to a number of metrics established for that purpose. Both metrics 1 and 3 are used as reference points for determining the values for metrics 2 and 4, respectively. The metrics used are detailed below:

1. *Number of turns per dialogue:* This establishes the number of turns per dialogue.
2. *Success in communicating the speaker's intention/Successful turns per dialogue:* This measures the success of each turn.
3. *Number of items of target information per dialogue*: Each turn may comprise 1 or more items of target information and, thus, this number is always higher than the number of turns. *Items of target information* refers to the different blocks of semantic information contained in each sentence or turn.
4. *Successful items of target information obtained*: This measures the number of successful blocks of semantic information passed from one user to the other (agent and client).
5. *Number of disfluencies per dialogue*: This refers to the number of disfluencies uttered by the users, covering mostly erroneous clicks (mistaken clicks of the mouse), pauses, doubts and mistakes while speaking.
6. *Number of repetitions per dialogue*: This reflects the number of repeated turns per dialogue so as to show how many repetitions the users have had to go through to achieve their goal.
7. *Number of abandoned turns per dialogue:* This presents those turns that have been abandoned by the user, mostly after several repetitions.

Before showing the table, results obtained from metrics 2 and 4 should be further explained given that they seem to provide much lower results than they actually do. The success obtained both at the level of a turn and of an item of target information is shown in a global way, that is, taking into account the full number of repetitions (which are considered in reference metrics 1 and 3). Thus, a dialogue may be successful by means of some repetitions while the numbers of success in metrics 2 and 4 are rather low. In order to establish this success, one should also look at metric 7, which reflects the number of abandoned turns and, thus, failures in transmitting target information (speaker's intention).

Last but not least, and as already explained in Section 3.1, users playing the role of the English-speaking client were not native speakers of English, which certainly makes speech recognition an even more complex task. This is particularly so in some dialogues where the speakers have little mastery of the language.

Table 7 shows the results obtained with the above metrics:

Dialogues	M-1	M-2	M-3	M-4	M-5	M-6	M-7
Eng/Spa-1	7	7	11	10	0	0	0
Eng/Spa-2	24	13,5	33	21,5	0	4	2
Eng/Spa-3	24	19	36	27	2	1	1
Eng/Spa-4	12	7,5	22	15	1	2	1
Eng/Spa-5	22	16,5	36	28	2	4	1
Eng/Spa-6	18	7	18	8	5	9	0
Eng/Spa-7	9	6	14	10	3	1	0
Eng/Spa-8	32	14,5	42	21	0	10	3
Eng/Cat-9	23	9,5	33	15,5	1	12	1
Eng/Cat-10	40	20,5	52	26,5	3	17	0
Eng/Cat-11	20	9	34	16	1	8	1
Eng/Cat-12	37	16	44	18,5	1	15	1
Eng/Cat-13	6	5,5	12	11	1	1	0
Eng/Cat-14	37	18,5	48	27	0	14	4
Eng/Spa-15	25	8	35	15	0	13	2
Eng/Spa-16	37	24	52	35,5	2	9	2
Eng/Cat-17	11	5,5	23	12	0	5	0
Eng/Cat-18	31	15	43	24	1	14	1
Eng/Cat-19	7	4,5	13	9	1	2	0
Eng/Cat-20	23	14,5	32	20,5	1	9	1

Table 7: User Satisfaction Results

As observed in M-7, 7 dialogues have successfully communicated all information; 8 dialogues have only given up on one turn, and 3 dialogues on 2. The remaining 2 dialogues have abandoned 3 and 4 turns, respectively. This is not an important loss, bearing in mind that after analysing the results, it was observed that a large number of problems come from very simple turns like greetings and thanking.

3.4.2 Qualitative Study

The quantitative study presented above has been supplemented by a qualitative evaluation based on users' responses to a brief questionnaire. Below is a detailed analysis of users' opinions of the system, looking at the results obtained both per question and per questionnaire. The average response is 3.4 points out of 5.

Results per question

Question 1: I understood the information the system passed on to me.

3 points out of 5 → 30%
4 points out of 5 → 40%
5 points out of 5 → 30%
} Ave. 4.0 pts

Question 2: The system understood what I told it to pass on.

2 points out of 5 → 10%
3 points out of 5 → 60%
4 points out of 5 → 30%
} Ave. 3.0 pts

Question 3: At each point during the interchange I understood what I could say.

2 points out of 5 →	20%	Ave. 3.6 pts
4 points out of 5 →	30%	
5 points out of 5 →	50%	

Question 4: The dialogue was normal and natural.

2 points out of 5 →	30%	Ave. 3.5 pts
3 points out of 5 →	20%	
4 points out of 5 →	10%	
5 points out of 5 →	40%	

Question 5: I succeeded in getting what I wanted done.

3 points out of 5 →	50%	Ave. 4.0 pts
4 points out of 5 →	20%	
5 points out of 5 →	30%	

Question 6: I would use this system again to help reserve a hotel room.

1 point out of 5 →	20%	Ave. 3.0 pts
2 points out of 5 →	10%	
3 points out of 5 →	40%	
4 points out of 5 →	20%	
5 points out of 5 →	10%	

Question 7: The system behaved as expected.

2 points out of 5 →	20%	Ave. 3.0 pts
3 points out of 5 →	40%	
4 points out of 5 →	40%	

Question 8: The system allowed me to easily correct any errors that arose.

4 points out of 5 →	70%	Ave. 4.5 pts
5 points out of 5 →	30%	

Question 9: The dialogue was very long.

1 point out of 5 →	40%	Ave. 2.0 pts
2 points out of 5 →	40%	
3 points out of 5 →	20%	

Question 10: I had trouble with turns about:

Hotel names →	12.50%
Hotel categories →	25.00%
Room Types →	18.75%
Dates →	25.00%
Prices →	6.25%
Other – Greetings →	12.50%

Average of questions 1-9: **3.4 points out of 5.0**

Results per questionnaire:

- Questionnaire 1 → 3.5 pts of 5
- Questionnaire 2 → 3.1 pts of 5
- Questionnaire 3 → 2.5 pts of 5
- Questionnaire 4 → 3.6 pts of 5
- Questionnaire 5 → 4.3 pts of 5
- Questionnaire 6 → 3.4 pts of 5
- Questionnaire 7 → 3.2 pts of 5
- Questionnaire 8 → 3.4 pts of 5
- Questionnaire 9 → 3.7 pts of 5
- Questionnaire 10 → 2.8 pts of 5

An informal inspection of the results per questionnaire indicates that the reaction of the users as a whole was consistent and weakly positive (taking 3.0 as a median).

4 Conclusions

This article has described the FAME interlingua-based speech-to-speech translation system for Catalan, English and Spanish and the different evaluations performed on real users and in lifelike situations. Three different types of evaluation were carried out so as to check a) the performance of the system, b) the influence of ASR in translation, c) the comparison in performance of the interlingua and the stochastic systems developed within FAME for the domain and task set, and d) the relevance of certain standard evaluation metrics when applied to spoken language interlingual translation.

The different evaluations prove that the system is already at an interesting and promising stage of development. In addition to these evaluations, a public demonstration of the system took place with untrained users participating and testing the system. Results from this open event were also very satisfactory.

Having reached this level of development, our next step will be to solve some remaining technical problems and to expand the system both within this domain and to others. Among the technical problems, we need to focus on improving the ASR component, which seems to be an important source of errors. For this purpose, further domain-specific data needs be collected so as to develop better language models. Another problem to be confronted is dealing with degraded translations. An option here may be to incorporate within the dialogue model strategies for the speakers to be able to request repetitions or reformulations.

Last but not least, a detailed study has been carried out of the pros and cons of the interlingua representation used by our system (Arranz *et al*. 05b). A number of problems have come to light in

the process of applying IF to representing the Romance languages described. As a result, a number of changes and improvements have been proposed for implementation during the next stage of system development.

5 Acknowledgements

This research was partially financed by the FAME (IST-2001-28323) and ALIADO (TIC2002-04447-C02) projects. We would also like to thank, very specially, Climent Nadeu and Jaume Padrell, for all their help and support in numerous aspects of the project. We are also grateful to other UPC colleagues, José B. Mariño and Adrià de Gispert, for fruitful exchanges during the development of both SST systems. Last but not least, we are strongly indebted to our colleagues at CMU, Dorcas Alexander, Donna Gates, Lori Levin, Kay Peterson and Alex Waibel, for all their feedback and help.

References

(Arranz, *et al*. 05a) V. Arranz, E. Comelles, A. de Gispert, D. Farwell, C. Nadeu, J. B. Mariño, J. Padrell, H. Rodríguez and A. Febrer, Speech-to-Speech Translation: Systems and Evaluations, FAME D9.2 deliverable, UPC, January 2005.

(Arranz, *et al*. 05b) V. Arranz, E. Comelles, D. Farwell, *Proposals to Handle Language-Dependent Limitations in an Interlingual Representation*, To be presented at the Conference on Lesser Used Language and Computational Linguistics, Bolzano, IT. October 26-27, 2005.

(Comelles 04) E. Comelles, *Sistema de Traducció Interlingua: Gramàtiques d'anàlisi del català i castellà*. Universitat de Barcelona, 2004.

(Gavaldà 00) M. Gavaldà, *SOUP: A Parser for Real-world Spontaneous Speech*. In *Proceedings of the 6th International Workshop on Parsing Technologies (IWPT-2000)*, Trento, Italy, 2000.

(Holzapfel, *et al*. 03) H. Holzapfel, I. Rogina, M. Wölfel, and T. Kluge, *FAME Deliverable D3.1: Testbed Software, Middleware and Communication Architecture*, 2003.

(Lavie, *et al*. 02) A. Lavie, F. Metze, R. Cattoni, E. Constantini, *A Multi-Perspective Evaluation of the NESPOLE! Speech-to-Speech Translation System*. In *Proceedings of ACL-2002 Workshop on Speech-to-Speech Translation: Algorithms and Systems*. Philadelphia, PA, 2002.

(Levin, *et al*. 02) L. Levin, D. Gates, D. Wallace, K. Peterson, A. Lavie, F. Pianesi, E. Pianta, R. Cattoni, N. Mana, *Balancing Expressiveness and Simplicity in an Interlingua for Task based Dialogue*. In *Proceedings of ACL-2002 workshop on Speech-to-speech Translation: Algorithms and Systems*. Philadelphia, PA, 2002.

(Metze, *et al*. 02) F. Metze, J. McDonough, J. Soltau, C. Langley, A. Lavie, L. Levin, T. Schultz, A. Waibel, L. Cattoni, G. Lazzari, N. Mana, F. Pianesi, E. Pianta, *The NESPOLE! Speech-to-Speech Translation System*. In *Proceedings of HLT-2002*. San Diego, California, 2002.

(Searle 69) J. Searle, *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press: Cambridge, UK, 1969.

(Schultz, *et al*. 04) T. Schultz, D. Alexander, A.W. Black, K. Peterson, S. Suebvisai, A, Waibel, A Thai Speech Translation System for Medical Dialogs. HLT/NAACL-2004, Boston, U.S., 2004.

(Taddei, *et al*. 03) L. Taddei, L. Besacier, R. Cattoni, E. Costantini, A. Lavie, N. Mana, E. Pianta, *NESPOLE! Deliverable D17: Second Showcase Documentation*, 2003. See the NESPOLE! Project web site: http://nespole.itc.it.

(Tomita & Nyberg 88) M. Tomita, E.H. Nyberg, *Generation Kit and Transformation Kit, Version 3.2, User's Manual. Technical Report CMU-CMT-88-MEMO*. Center for Machine Translation, Carnegie Mellon University, Pittsburgh, PA, 1988.

(Woszczyna, *et al*. 93) M. Woszczyna, N. Coccaro, A. Eisele, A. Lavie, A. McNair, T. Polzin, I. Rogina, C. Rose, T. Sloboda, M. Tomita, J. Tsutsumi, N. Aoki-Waibel, A. Waibel, W. Ward, *Recent Advances in JANUS: A Speech Translation System*. In *Proceedings of Eurospeech-1993*. Berlin, 1993.

A framework for representing and managing linguistic annotations based on typed feature structures

X. Artola, A. Díaz de Ilarraza, N. Ezeiza, K. Gojenola* G. Labaka A. Sologaistoa A. Soroa
Faculty of Computer Science, Donostia / *School of Engineering, Bilbo
University of the Basque Country (UPV/EHU)
The Basque Country
jipdisaa@si.ehu.es

Abstract

In this paper we present a framework for dealing with linguistic annotations. Our aim is to establish a flexible and extensible infrastructure which follows a coherent and general representation scheme. This proposal provides us with a well-formalized basis for the exchange of linguistic information. We use TEI-P4 conformant feature structures as a representation schema for linguistic analyses. We have identified the consistent underlying data model which captures the structure and relations contained in the information to be manipulated. This data model has been represented by classes following the object-oriented paradigm. The huge amount of information generated is stored in an XML database that provides fast answers to common queries. With the aim of helping users to manipulate linguistic annotations generated by the different tools, we have designed and implemented a component-based software, EULIA, that facilitates operations on the linguistic annotations.

Keywords: linguistic annotations, NLP software engineering, stand-off annotation

1 Introduction

In this paper we present a framework for creating, browsing and editing linguistic annotations generated by a set of different linguistic processing tools[1](Artola *et al.* 00).

The objective is to establish a flexible and extensible infrastructure for consulting, visualizing, and modifying annotations generated by existing linguistic tools, following a coherent and general representation scheme (Artola *et al.* 02).

The main goal of this proposal is to set up a well-formalized basis for the exchange of linguistic information among tools. We use TEI-P4 conformant (http://www.tei-c.org/P4X/DTD/) typed feature structures as a representation schema for linguistic analyses.

We have identified the consistent underlying data model which captures the structure and relations contained in the information to be manipulated. This data model is represented by classes that are encapsulated in several library modules, following the object-oriented paradigm.

[1]URL: http://ixa.si.ehu.es

Besides, we have also implemented EULIA, an extensible, component-based software architecture to integrate language engineering applications. EULIA is a user-oriented linguistic data manager, with an intuitive and easy-to-use GUI that offers help in data browsing, manual disambiguation and annotation tasks.

The rest of the paper is organized as follows. In section 2 we present some related work. Section 3 will be dedicated to explain the annotation framework proposed; that is, the representation scheme used for the linguistic information obtained from the different tools. In section 4 we explain the information flow among the different linguistic processors integrated so far. Section 5 presents LibiXaML, the program library which deals with the different types of linguistic information, i.e., with what we call the "annotation web". In section 6, the library-oriented approach we use to store information is presented. Section 7 describes EULIA, an application implemented for facilitating the work with the annotation web. Finally, section 8 presents conclusions and future work.

2 Related work

There is a general trend for establishing standards for effective language resource management (ISO/TC 37/TC 4 (Ide & Romary 04)), the main objective of which is to provide a framework for language resource development and use. A key issue in software development in NLP processes is the definition of a framework for linguistic knowledge representation. Such a framework has to satisfy needs entailed by the different tools and has to be general enough (Basili *et al.* 98). It is not trivial to adopt a formalism to represent this information and different approaches have been considered for this task. For example, ALEP (Advanced Language Engineering Platform) (Simkins 94) can be considered the first integrating environment for NLP design, where all the components (linguistic information, processing modules and resources) are homogeneously described using the ALEP User Language (AUL) based on a DAG formalism. Perhaps the most influential system in the area is GATE (Cunningham *et al.* 96; Bontcheva *et al.* 04; Neff *et al.* 04) which provides a software infrastructure on which NLP applications may be combined into larger application systems. Following this tendency, ATLAS and

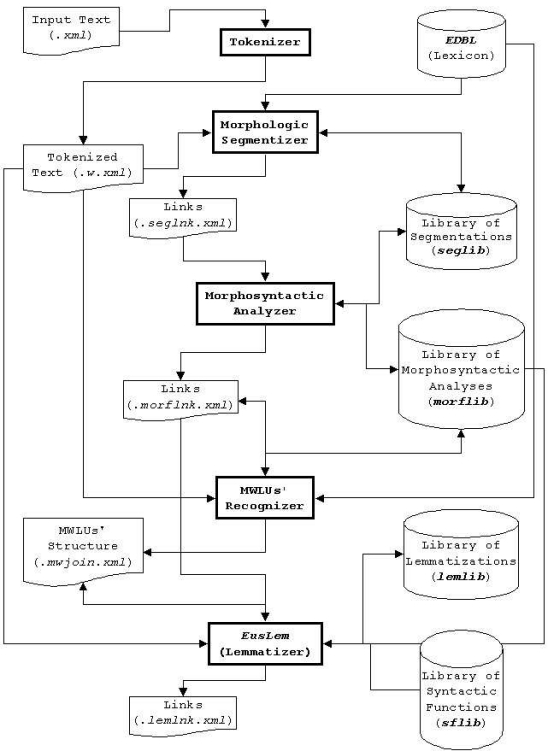


Figure 1: The multi-document annotation web (1)

MAIA (Bird *et al.* 00; Laprun *et al.* 02) provide an architecture targeted at facilitating the development of lingistic annotation applications. In Talent (Neff *et al.* 04), the authors present a pipeline architecture allowing for rapid prototyping and application development. The UIMA model (Ferrucci & Lally 04) permits the implementation of middleware frameworks that allow component-based infrastructure for enabling the rapid combination of linguistic technologies.

The annotation framework presented in this paper follows the stand-off markup approach and it has been inspired on TEI-P4 guidelines (Sperberg-McQueen & Burnard 02) to represent linguistic information obtained by a wide range of linguistic tools. The reason for taking this approach is that our representation requirements are not completely fulfilled by the annotation schemes proposed in the systems mentioned before. For instance, the TIPSTER architecture [Grishman 97] used in GATE version 1 exhibits problems when encoding some linguistic structures, as those referred to non-continuous multiword lexical units. The ATLAS system, based on the so-called Directed Annotation Graphs (DAG) for annotation purposes, exhibits the same restrictions. GATE version 2 (Cunningham *et al.* 02) tried to solve this problem combining TIPSTER and DAG, but the solution they propose makes the annotation of some simple, non-continuous features complex and non-intuitive.

Basque is an agglutinative language and the morphological information we want to attach to every word-form obliges us to use a rich model to represent it. The models used by these well-known systems don't fullfil this requirement. Our stand-off markup annotation system can represent any kind of linguistic information or structure. Following the TEI guidelines, we can deal with any kind of linguistic annotation by means of few elements such us anchors, joins, links and feature structures.

3 The annotation framework

Two main features characterize our annotation framework:

1. The variety of anchors to which the linguistic information can be attached ranges from single tokens, continous and discontinous multi-token lexical units, and different kinds of spans up to even particular word interpretations.

2. The richness and complexity of the linguistic information we need to represent. For example, in morphological analysis, we want to describe phenomena such as intra-word ellipsis or the inner structure of derivatives and compounds

In our case, within this framework of stand-off linguistic annotation, the output of each analysis tool may be seen as composed of several XML documents: *the annotation web*. Figure 1 and Figure 2 show the currently implemented document model including the representation schemes used in tokenization, segmentation, morphosyntactic analysis, multiword recognition, lemmatization/disambiguation, shallow syntax

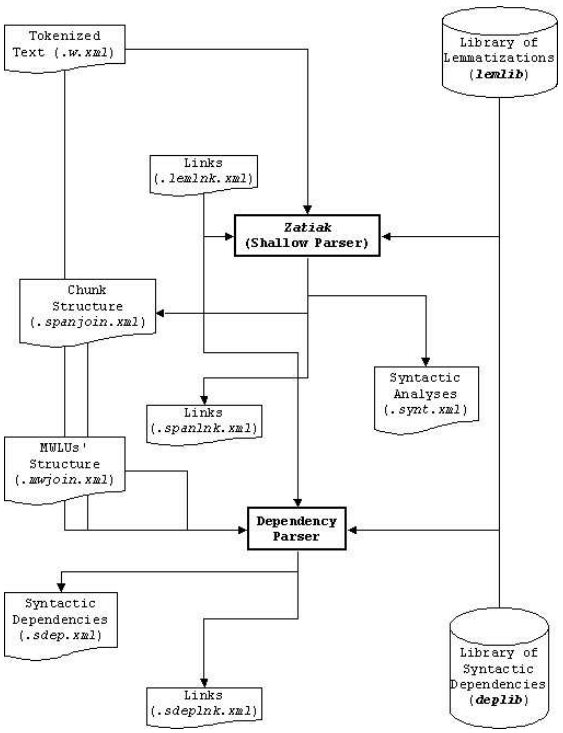


Figure 2: The multi-document annotation web (2)

and dependency-based analysis. This model fulfils the general requirements proposed in the standards (Ide & Romary 04), as in (Bird *et al.* 00; Schäfer 03):

- It provides a way to represent different types of linguistic information, ranging from the general to the fine-grained one where partial results and ambiguities can be easily represented.
- It uses feature structures as a general data model, thus providing a formal semantics and a well known logical operation set over the linguistic information represented by them.
- A general abstract model has been identified over the particular linguistic processors. Therefore, NLP applications are able to import/export the information they need in a unified way.
- The representation model doesn't dependend on any linguistic theory nor any particular processing software.

3.1 The annotation web

As said above, linguistic information is attached to the analyzed text and represented as a set of XML documents that constitute the annotation web. Looking at the characteristics of the documents to be generated, we have identified different groups and types of documents. Next, we will present all of them indicating the elements defined and the corresponding class used for their representation in our model.

- Text anchors: text elements found in the input text.
  - Single-word tokens recognized by the tokenizer. They are tagged with the XML <w> element, and represented in our model by the W class.
  - Multiword lexical units: the collection of "multiword tokens" identified in the input. The MWSTRUCT class represents the constituents of a multiword unit, which is encoded by means of a <join> element that gathers the individual constituents of the unit.
  - The structure of syntactic chunks recognized in the text: the collection of "spans" identified in the input. The SPANSTRUCT class represents the constituents of a chunk that are also tagged by means of <join> elements.
- Analysis collections: collections of linguistic analyses obtained by the different tools. Due to the complexity of the information to be represented we decided to use feature structures (FS) as a general data structure. The use of feature structures quickly spread to other domains within linguistics since Jacobson (Jacobson 49) first used them for the representation of phonemes. Feature structures serve as a general-purpose linguistic metalanguage; this reason led us to use them as the basis of our encoding. The feature structures we use fulfill the TEI's guidelines for typed FSs, and the schema of all the inputs/outputs in the tool pipeline has been thoroughly described by means

of Relax NG Schemas.

- Links between anchors and their corresponding analyses, tagged by means of <link> elements. They are represented by the LINK class

The multi-document annotation web gives, as pointed out in (Ide & Véronis 95; Ide & Romary 04), more independence and flexibility to the different processes, and greater facilities for their integration. In figure 3 we will show an example which illustrates how the multi-document annotation web looks like once the lemmatization process is carried out.

4 The I/O stream between programs

There are many linguistic tools integrated so far. Figure 1 and 2 illustrate the integration of the lexical database (Aldezabal *et al.* 01) and the rest of the tools, emphasizing that the communication among the different processes is made by means of XML documents. Let us describe these processes in sequence:

1. Having an XML-tagged input text file, the tokenizer takes this file and creates, as output, a *.w.xml* file, which contains the list of the tokens and sentences recognized in the input text. The tokenized text is of great importance in the process, in the sense that it intervenes as input for different processes.

2. After the tokenization process, the segmentizer takes as input the tokenized text and the general lexicon issued from the lexical database, and updates the library of segmentation analyses (FSs describing the different morphemic segments found in each word token; one FS per different distinct word-form) producing as well a document (*.seglnk.xml*) containing the links between the tokens in the *.w.xml* file and their corresponding analyses (one or more) in the library. The standoff framework we follow in annotating the documents allows us to attach easily different analyses to one token.

3. After that, the morphosyntactic treatment module takes as input the output of the segmentation process and updates the library of morphosyntactic analyses *morflib*. It produces a *.morflnk.xml* document containing the links between the tokens in the *.w.xml* file and their corresponding analyses (one or more) in *morflib*.

   The library *morflib* will be later enriched by the MWLUs' treatment module (Alegria *et al.* 04). This module performs the processing of multiword lexical units producing a document that describes, by means of a collection of <join> elements *.mwjoin.xml*, the structure of the MWLUs identified in the text. This module has obviously access to the morphosyntactic analyses and to the *.morflnk.xml* document, into which it will add the links between the *.mwjoin.xml* document and the library.

4. The morphosyntactic analyses and the output of the tokenizer constitute the input of the *Euslem* lemmatizer (Ezeiza *et al.* 98). The lemmatizer updates the library of lemmatizations and produces the *.lemlnk.xml* document that contains the links between the tokens and MWLUs, and their corresponding lemmatization analyses. Besides, it updates the *.mwjoin.xml* document removing the incorrect joins previously included in it.

5. In figure 2, the syntactic process is depicted. The *Zatiak* surface syntax parser (Aduriz *et al.* 04) identifies the chunks in the text (phrases, verb chains and so on) based on the syntactic functions that, following the Constraint Grammar formalism (Karlsson *et al.* 95), the lemmatizer has associated to each word of the text. In this process a named-entity recognizer is also included. This process produces three documents: a *.spanlnk.xml* document that describes which tokens and MWLUs belong to each chunk in the text; a *.synt.xml* document that contains syntactic features associated to each chunk; and a *.spanjoin.xml* document containing the links between the chunks and the *.synt.xml* document. Note that the syntactic analyses contained in the *synt.xml* document correspond to a single input text, since, obviously, there is no general library containing syntactic analysis.

6. Finally, a dependency grammar parser establishes the dependencies between the components of the sentence in order to obtain a syntactic tree. It takes as input the library of the different syntactic dependencies *deplib* and obtains an *.sdep.xml* document describing the syntactic dependencies found in the sentences (Aranzabe *et al.* 04) and a *.sdeplnk.xml* document containing the links between the dependencies and the library.

Figure 3 shows a sample of the annotation web, result of the lemmatization. The input-text is at the upper-left part of the drawing. A multiword expression *Hala ere* (Basque for *however*) and a single-word token *ere* (Basque for *also*) have been emphasized to illustrate the relationships established between these items in the text and their corresponding lemmatizations represented by feature structures contained in the document at the upper-right part of the drawing. The document called *tokenized text* contains the results of the tokenization process: the sequence of tokens identified in the text with the indication of the character offsets corresponding to each token in the source. Similarly, a document called *MWLU's structure* contains the results of the MW expressions processing: the sequence of multi-word elements identified

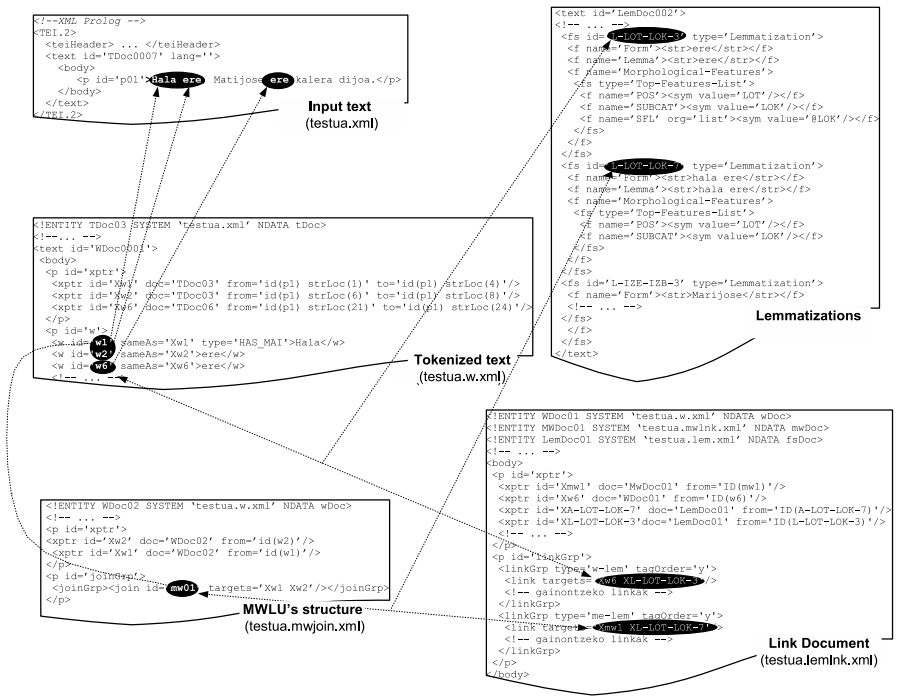


Figure 3: Output of the lemmatizer: a sample of the multi-document annotation web

in the text with the indication of the single tokens belonging to each one of them. Finally, the actual annotations (and the ambiguities, if any) are represented by the link document, that attaches the different items in the source text (single- or multi-word tokens) to their corresponding lemmatizations. TEI external pointers are used to refer to elements not present in the same document.

5 LibiXaML: A program library for dealing with the annotation web

We identified the consistent underlying data model which captures the structure and relations contained in the information to be manipulated. This data model is represented by classes which are encapsulated in several library modules, following the object oriented paradigm. These modules offer the necessary types and operations the different tools need to perform their task when recognizing the input and producing their output. LibiXaML manipulates: features, feature structures, values, XML documents containing linguistic information of different types, document headers, and so on.

The class methods in LibiXaML allow:

- Getting the necessary information from an XML document containing tokens, links, multiword structure joins, FSs, etc.
- Producing with ease the corresponding output according to a well-defined XML description.

The class library has been implemented in C++ and it contains about 100 classes. For the implementation of the different classes and methods we make use of the LT XML system (Thompson *et al.* 97), a tool architecture for XML-based processing of text corpora. The current release of LibiXaML works on Unix and can be soon found at `http://ixa.si.ehu.es/ixa/resources/libixaml`.

6 Storing linguistic information in general FS-Libraries

Considering the huge amount of information obtained in these linguistic processes, it is crucial to get an optimal storage of data in order to provide a fast answer when retrieving and searching this information. We are experimenting two ways of doing things:

- Document-oriented approach: the segmentations, morphosyntactic analyses and lemmatizations (FSs) obtained by the different analysis tools applied on a given document constitute FS collections which are stored in files specifically attached to that document.
- Library-oriented approach: the segmentations, morphosyntactic analyses and lemmatizations (FSs) obtained by the different analysis tools applied on a given document are added to general FS collections stored in big FS libraries.

The second approach saves lots of disk space and speeds up the analysis procedures because the analysis of most word forms must not be repeated since their results will be already stored in the library. So, performing the analysis is just a matter of retrieving the

corresponding analysis identifiers in the library and establishing a link to them.

Using the library-oriented technique to store information requires a more powerful indexing scheme, which will avoid, most of the time, the access to the actual XML FS library.

In order to test our annotation framework on a running environment, we have tokenized, segmentized and morphologically analyzed a text corpus containing 426,205 tokens (71,893 of them are the punctuation marks). The annotation web issued from morphological analysis has been stored, according to the library-based approach, in the following manner:

- feature structures that represent the morphological analyses corresponding to the word forms in the corpus (one for each different word form) have been loaded on an XML-native database (Berkeley DBXML);
- links between text elements and their corresponding analyses have been stored in a relational database (Berkeley DB) for faster retrieval;
- tokenization results and the original text are left in the file system.

A query prototype has been developed on this architecture and some experiments have been carried out on it. For now, this prototype provides us with a quite basic functionality, allowing to pose complex XPath expressions as queries. The XPath expressions are evaluated against the morphological analyses in the XML database that has been adequately indexed, returning as result the identifiers of the feature structures that meet the constraints expressed by the query; next, these identifiers are searched in the relational database containing the links in order to get the identifiers of the corresponding tokens, which are then retrieved on the original text to get their contexts. The final result is that, for example, to get a concordance (KWIC) that contains the words whose morphological analyses meet the constraints in the query along with their contexts takes around one second in a SUN workstation.

7 EULIA: An application to create, browse and disambiguate linguistic annotation based on the annotation web

In order to work on the annotation framework here explained, we have developed EULIA an environment that implements an extensible, component-based software architecture to integrate natural language engineering applications and to exploit the data created by these applications. The main functions of EULIA are the following ones:

- search, queries and analysis of results.
- submit a text to be analyzed.
- consultation and browsing of the linguistic annotation attached to texts.
- manual disambiguation of analysis results.
- manual annotation facilities and suitable encoding for new linguistic information.
- personalization of users.

Regarding the interface, the main window is divided into two parts: a Multi-Document Interface (MDI) panel where linguistic information is shown in an understandable way. The interface provides hypertextual facilities, showing the linguistic information associated to items selected on the left part. The environment is designed as a tool for general users and linguists.

8 Conclusion and future work

In this paper we present a framework for dealing with language annotations.

Our proposal provides a flexible and extensible infrastructure for consulting, visualizing, and modifying annotations generated by existing linguistic tools. In this framework, the fact that different analysis sets (segmentations, complete morphosyntactic analyses, lemmatization results, and so on) linked to text anchors are stored in analysis libraries in a stand-off fashion implies a reduction in time and space resources. Regarding the physical storage of the annotation information, we have already implemented the document-based storage approach and are now refining the library-based approach previously explained because we think that the use of XML native databases should be a good solution for fast retrieval and searching on these huge analysis libraries. So, we are planning to move progressively to this library-based approach. The work done so far confirms the scalability of our approach.

Very few studies have used the stand-off markup based on TEI-P4 guidelines. From our point of view, the TEI-P4 approach gives us the expresiveness required by the complexity of the linguistic information we want to represent both when establishing diverse kinds of anchors to which attach information, and when defining specialized FS types for this information.

We have designed and implemented LibXaML, a component-based library that represents the different types of information to be manipulated.

Morever, EULIA, an extensible, component-based software architecture to integrate natural language engineering applications facilitates the work on these annotations offering help in data browsing, manual disambiguation and annotation tasks.

9 Acknowledgements

This research was partially funded by the Basque Government and University (HIZKING21 project and 9/UPV00141.226-14601/2002)

References

(Aduriz *et al.* 04) Itziar Aduriz, Maxux Aranzabe, Jose Mari Arriola, Arantza Díaz de Ilarraza, Koldo Gojenola, Maite Oronoz, and Larraitz Uria. *Computational Linguistics and Intelligent Text Processing*, chapter A Cascaded Syntactic Analyser for Basque, pages 124–135. 2945 LNCS Series. Springer Verlag, 2004.

(Aldezabal *et al.* 01) Izaskun Aldezabal, Olatz Ansa, Bertol Arrieta, Xabier Artola, Aitzol Ezeiza, Gregorio Hernández, and Mikel Lersundi. EDBL: a general lexical basis for the automatic processing of Basque. In *IRCS Workshop on linguistic databases.*, Philadelphia. USA, 2001.

(Alegria *et al.* 04) Iñaki Alegria, Olatz Ansa, Xabier Artola, Nerea Ezeiza, Koldo Gojenola, and Ruben Urizar. Representation and Treatment of Multiword Expressions in Basque. In *ACL workshop on Multiword Expressions*, Barcelona, 2004.

(Aranzabe *et al.* 04) Maxux Aranzabe, Jose Mari Arriola, and Arantza Díaz de Ilarraza. Towards a dependency parser for Basque. In *Proc. of International Conference on Computational Linguistics. COLING'2004*, Geneva, 2004.

(Artola *et al.* 00) Xabier Artola, Arantza Díaz de Ilarraza, Nerea Ezeiza, Koldo Gojenola, Aitor Maritxalar, and Aitor Soroa. A proposal for the integration of NLP tools using SGML-Tagged documents. In *Proc. of the Second Int. Conf. on Language Resources and Evaluation*, Athens (Greece), 2000.

(Artola *et al.* 02) Xabier Artola, Arantza Díaz de Ilarraza, Nerea Ezeiza, Koldo Gojenola, Gregorio Hernández, and Aitor Soroa. A class library for the integration of NLP tools: Definition and implementation of an abstract data type collection for the manipulation of SGML documents in a context of stand-off linguistic annotation. In *Proc. of the Third Int. Conf. on Language Resources and Evaluation*, Las Palmas (Spain), 2002.

(Basili *et al.* 98) Roberto Basili, Massimo Di Nanni, and Maria Teresa Pazienza. Engineering of IE Systems: An Object-oriented approach. In Maria Terese Pazienza, editor, *Information Extraction: Towards scalable, Adaptable Systems*, volume 1714 of *Lecture Notes in Artificial Intelligence*, pages 134–164. Springer-Verlag, 1998.

(Bird *et al.* 00) Steven Bird, David Day, John Garofolo, Henderson Henderson, Christophe Laprun, and Mark Liberman. ATLAS: A flexible and extensible architecture for linguistic annotation. In *Proc. of the Second International Conference on Language Resources and Evaluation*, pages 1699–1706, Paris (France), 2000.

(Bontcheva *et al.* 04) Kalina Bontcheva, Valentin Tablan, Diana Maynard, and Hamish Cunningham. Evolving GATE to meet new challenges in language engineering. *Natural Language Engineering*, 10(3-4):349–373, 2004.

(Cunningham *et al.* 96) Hamish Cunningham, Yorick Wilks, and Robert J. Gaizauskas. GATE: a General Architecture for Text Engineering. In *Proceedings of the 16th conference on Computational linguistics*, pages 1057–1060. Association for Computational Linguistics, 1996.

(Cunningham *et al.* 02) H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, and C. Ursu. *The GATE User Guide.* 2002. http://gate.ac.uk/.

(Ezeiza *et al.* 98) Nerea Ezeiza, Itziar Aduriz, Iñaki Alegria, Jose Mari Arriola, and Ruben Urizar. Combining Stochastic and Rule-based Methods for Disambiguation in Agglutinative Languages. In *Proc. of COLING-ACL'98*, pages 10–14, Montreal (Canada), 1998.

(Ferrucci & Lally 04) David Ferrucci and Adam Lally. UIMA: an architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering*, 10(3/4):327–348, 2004.

(Ide & Romary 04) Nancy Ide and Laurent Romary. International standard for a linguistic annotation framework. *Natural Language Engineering*, 10(3-4):211–225, 2004.

(Ide & Véronis 95) Nancy Ide and Jean Véronis, editors. *Text Encoding Initiative. Background and Context.* Kluwer Academic Pub, 1995.

(Jacobson 49) Roman Jacobson. The identification of phonemic entities. *Travaux du Cercle Linguistique de Copenhague*, 5:205–213, 1949.

(Karlsson *et al.* 95) Fred Karlsson, Atro Voutilainen, Juha Heikkilä, and Arto Anttila, editors. *Constraint Grammar: A Language-independent System for Parsing Unrestricted Text*, volume 4 of *Natural Language Processing.* Natural Language Processing, Mouton de Gruyter, Berlin and New York, 1995.

(Laprun *et al.* 02) Cristophe Laprun, Jonathan. Fiscus, John. Garofolo, and Silvai. Pajot. A practical introduction to ATLAS. In *Proceedings of the Third International Conference on Language Resources and Evaluation*, 2002.

(Neff *et al.* 04) Mary S. Neff, Roy J. Byrd, and Branmir K. Bougaraev. The Talent system: TEXTRACT architecture and data model. *Natural Language Engineering*, 10(3-4):307–326, 2004.

(Schäfer 03) Ulrich Schäfer. WHAT: An XSLT-based infrastructure for the integration of natural language processing components. In *Proceedings of the Workshop on the Software Engineering and Architecture of Language Technology Systems (SEALTS), HLT-NAACL03*, Edmonton (Canada), 2003.

(Simkins 94) N. K. Simkins. An Open Architecture for Language Engineering. In *First CEC Language Engineering Convention*, Paris (France), 1994.

(Sperberg-McQueen & Burnard 02) C. M. Sperberg-McQueen and L. Burnard, editors. *TEI P4: Guidelines for Electronic Text Encoding and Interchange.* Oxford, 4 edition, 2002.

(Thompson *et al.* 97) H.S. Thompson, R.Tobin, D. Mckelvie, and C. Brew. *LT XML Software API and toolkit for XML processing.* 1997. http://www.ltg.ed.ac.uk/software/xml/index.html.

Indexing and Querying Linguistic Metadata and Document Content

Niraj Aswani and **Valentin Tablan** and **Kalina Bontcheva** and **Hamish Cunningham***
Department of Computer Science
University of Sheffield
Regent Court, 211 Portobello Street
Sheffield, S1 4DP, UK
{niraj,valyt,kalina,hamish}@dcs.shef.ac.uk

Abstract

The need for efficient corpus indexing and querying arises frequently both in machine learning-based and human-engineered natural language processing systems. This paper presents the ANNIC system, which can index documents not only by content, but also by their linguististic annotations and features. It also enables users to formulate versatile queries mixing keywords and linguistic information. The result consists of the matching texts in the corpus, displayed within the context of linguistic annotations (not just text, as is customary for KWIC systems). The data is displayed in a graphical user interface, which facilitates its exploration and the discovery of new patterns, which can in turn be tested by launching new ANNIC queries.

1 Introduction

The need for efficient corpus indexing and querying arises frequently both in machine learning-based and human-engineered natural language processing systems. A number of query systems have been proposed already and (Christ 94), (Mason 98), (Bird *et al.* 00a) and (Gaizauskas *et al.* 03) are amongst the most recent ones. In this paper, we present a full-featured annotation indexing and retrieval search engine, called ANNIC (ANNotations-In-Context), which has been developed as part of GATE (General Architecture for Text Engineering) (Cunningham *et al.* 02).

Whilst systems such as (McKelvie & Mikheev 98), (Gaizauskas *et al.* 03) and (Cassidy 02) are targeted towards specific types of documents, (Christ 94), (Bird *et al.* 00a) and (Mason 98) are general purpose systems. ANNIC falls in between these two types, because it can index documents in any format supported by the GATE system (i.e., XML, HTML, RTF, e-mail, text, etc). These existing systems were taken as a starting point, but ANNIC goes beyond their capabilities in a number of important ways. New features address issues such as extensive indexing of linguistic information associated with document content, independent of document format. It also allows indexing and extraction of information from overlapping annotations and features. Its advanced graphical user interface provides a graphical view of annotation mark-ups over the text along with an ability to build new queries interactively. In addition, ANNIC can be used as a first step in rule development for NLP systems as it enables the discovery and testing of patterns in corpora.

Section 2 introduces the GATE text processing platform which is the basis of this work. Following this, we briefly describe how Lucene is used to index documents (Section 3). This section also provides details of the ANNIC implementation and the changes made in Lucene.

2 GATE

GATE is a large-scale infrastructure for natural language processing applications (Cunningham *et al.* 02). Lingustic data associated with language resources such as documents and corpora is encoded in the form of annotations. GATE supports a variety of formats including XML, RTF, HTML, SGML, email and plain text. In all cases, when a document is created/opened in GATE, the format is analysed and converted into a single unified model of annotation. The annotation format is a modified form of the TIPSTER format (Grishman 97) which has been made largely compatible with the Atlas format (Bird *et al.* 00b), and uses 'stand-off markup' (Thompson & McKelvie 97). The annotations associated with each document are a structure central to GATE, because they encode the language data read and produced by each processing module. Each annotation has a start and an end offset and a set of features associated with it. Each feature has a name and a relative value, which holds the descriptive or analytical information such as Part-of-speech and sense tags, syntactic analysis, named entities identifica-

*This work was partially supported by an AHRB grant ETCSL and an EU grant SEKT.

tion and co-reference information etc.

JAPE, Java Annotation Patterns Engine, is part of the GATE system. It is an engine based on regular expression pattern/action rules over annotations. JAPE is a version of CPSL (Common Pattern Specification Language). This engine executes the JAPE grammar phases - each phase consists of a set of pattern/action rules. The left-hand-side (LHS) of the rule represents an annotation pattern and the right-hand-side (RHS) describes the action to be taken when pattern found in the document. JAPE executes these rules in a sequential manner and applies the RHS action to generate new annotations over the matched regular expression pattern. Rule prioritisation (if activated) prevents multiple assignments of annotations to the same text string.

This paper demonstrates how ANNIC indexes GATE processed documents with their annotations and features and enables users to formulate versatile queries using JAPE patterns. The result consists of the matching texts in the corpus, displayed within the context of linguistic annotations (not just text, as is customary for KWIC systems). The data is displayed in a graphical user interface, which facilitates its exploration and the discovery of new patterns, which can in turn be tested by launching new ANNIC queries.

3 Apache Lucene

ANNIC is built on top of the Apache Lucene[1] a high performance full-featured search engine implemented in Java, which supports indexing and search of large document collections. Our choice of IR engine is due to the customisability of Lucene.

Lucene document is a basic unit of indexing and search operations. All information associated with Lucene documents is stored in units called fields, where each field has its name and a textual value. (e.g. contents, url, modified date etc.). *Analyzer* knows what to parse and how to convert text into the format that *Index Writer* understands. Index Writer builds a *Token Stream (a sequence of words)*, which describes information about the token text. *Token* contains linguistic properties, and other information such as the start and end offsets and the type of the string (i.e. the lexical or syntactic class that the token belongs to). We will use the term *Lucene token* to refer to the tokens created by Lucene. *Filters* take the *stream of tokens* as input and add or delete Lucene tokens in the token stream. For example, a stemmer would add a new Lucene token with base word for each word that is not in its base form and a stop word filter would remove all stop words from the token stream so that they do not get indexed. Not only Lucene provides the ability to create user defined queries through its API, it also supports a wide range of predefined queries. This includes wild character queries, boolean queries, phrase queries etc.

[1] http://lucene.apache.org

Table 1: Lucene Token Generation

Lucene Token	Position Increment
John	1
wants	1

Table 2: Lucene Token Generation

Lucene Token	Position Increment
John	1
wants	1
want	0

Every Lucene token has its own *position* in the *token stream*. This position remains relative to its previous Lucene token and is stored as a *position increment factor* in the token stream. Consider the example in Table 1 and Table 2 which show the strings, the Lucene tokens derived from them, and their respective position increments in the token stream. Executing a stemmer over the above sentence would generate two extra words, which are stored with 0 increments immediately after the word they refer to in the token stream.

Along with its position increment attribute, each Lucene token in the token stream comprises of four attributes: text (e.g. wants), start offset, end offset and type (e.g. word). Lucene stores only the first attribute (i.e. text) in its indices.

When a Lucene query is submitted to the Lucene query parser, an array that contains hits is returned as a result. Each hit is an object that contains a pointer to the document, in which one or more patterns have been found, and the score of that hit. Documents in this hit array are organized in a descending order of their scores, i.e. the most relevant document appears first. This arrangement allows users only to refer to the n number of top most documents in the results.

4 ANNIC

The aim of the ANNIC system is to index the linguistic information and other metadata and retrieve the annotation patterns in the form of KWIC concordances (see 5). After few changes in the behaviour of the key components of Lucene, we were able to make Lucene adaptable to our requirements.

4.1 Lucene Token generation

As mentioned before, Lucene only indexes the text attribute of a Lucene token. To meet our requirements, i.e. to index the linguistic information and metadata, Lucene was modified to index also the type attribute. Type attribute holds a string assigned by lexical analyzer that defines the lexical or syntactic class of the Lucene token. GATE documents need to be separated into tokens by a tokeniser (*GATE Token* from now on) before they get indexed with ANNIC. This is required as tokens are the basic segments of any document and therefore they should be indexed in order to perform full-text search. Every annotation in GATE has a corresponding features associated with it. We create a separate Lucene token for every feature in the document. In the case where multiple annotations and their features refer to the same text in the document, we use the "Position increment" attribute to indicate their positions. Consider the following example:

E.g. the word Bill is annotated as:

GATE Token
POS: NNP
Kind: word
String: Bill
Person

Table 3 explains the token stream that contains tokens for the above annotations. The annotation type itself is stored as a separate Lucene token with its attribute type * and text as the value of annotation type. This allows users to search for a particular annotation type. In order not to confuse features of one annotation with others, feature names are qualified with their respective annotation type names. Where there exist multiple annotations over the same piece of text, only the position of the very first feature of the very first annotation is set to 1 and it is set to 0 for the rest of the annotations and their features. This enables users to query over overlapping annotations and features.

It is possible for two annotations to share the same offsets. They can share either start, end or both offsets. The built-in GATE annotation comparator is used for this purpose. First, the start offsets are compared and then the end offsets. If comparator returns two annotations as sharing both offsets, such annotations are kept on the same position in the token stream, and otherwise one after another. This may lead to a problem. What if annotations overlap each other (i.e. they share only one of the start and end offsets)? In this case, though annotations do not appear one after another, they are stored one after another. This may lead to incorrect results being returned and therefore the results are verified in order to filter out invalid overlapping patterns.

Before indexing GATE documents with Lucene, we convert them into the Lucene format and refer to them as GATE Lucene documents. In order to fetch patterns for their left and right contexts, it was necessary for some old concordances programs to have all documents available at the search time (Mason 98). This may lead to serious performance penalties. To overcome this problem, the token stream is stored in a separate file as a Java serializable object in the index directory. Later, it is retrieved in order to fetch left and right contexts of the found pattern.

4.2 Gate Query Parser

JAPE patterns support various query formats. Below we give few examples of JAPE patterns. Actual patterns can also be a combination of one or more of the following pattern clauses:

1. *String*
2. {*AnnotationType*}
3. {*AnnotationType == String*}
4. {*AnnotationType.feature == feature value*}
5. {*AnnotationType1, AnnotationType2.feature == featureValue*}
6. {*AnnotationType1.feature == featureValue, AnnotationType2.feature == featureValue*}

Order of the annotations specified in ANNIC query is very important. In Lucene, document must contain the specified keywords, no matter in which order they exist. Order is important only for the phrase queries. Since the default implementation of Lucene indexer indexes only the

Table 3: Token stream entries for the word Bill annotated as Token and Person

Sr. No.	Lucene Token Text	Lucene Token Type	Pos. Incr.	Description
1	Token	*	1	Annotation Type Token
2	NNP	Token.pos	0	pos feature with value NNP
3	word	Token.kind	0	kind feature with value word
4	Bill	Token.string	0	string feature with value Bill
5	Person	*	0	Annotation Type Person

text attribute of Lucene Token, it does not allow searching over the type attribute. Certain characters used in JAPE patterns have different meanings in Lucene. E.g. Lucene uses { } (opening and closing brackets) to recognize the range queries and these characters are used to enclose the annotation type in JAPE. Lucene query parser does not support position increments in queries. For example if one wants to search for annotations of type *Location* and *Person* referring to the same piece of text, Lucene does not support this. On the other hand, the respective JAPE pattern would be {*Location, Person*}.

JAPE patterns also support the | (OR) operator. For instance, {*A*} *(*{*B*} | {*C*}*)* is a pattern of two annotations where the first is an annotation of type *A* followed by the annotation of type either *B* or *C*.

Due to the various reasons explained above, we introduce our own query parser (ANNIC Query Parser) which accepts JAPE queries. Instead of comparing only the text attribute of Lucene Token, we also compare the type attribute. Lucene query parser, before accessing index, converts each keyword into an instance of *Term* class and compares them with the terms in index. Table 4 demonstrates how JAPE pattern tokens are converted into query terms. In order to use predefined Lucene queries (i.e. Boolean and Phrase queries), JAPE patterns with *OR* operator are normalized into the *AND normalized form* and all such patterns are *OR*ed together to form a Boolean query.

Lucene Phrase query considers its each token as a separate *term* and sets its position to the previous terms position + 1. This behaviour leads to a problem in the context of JAPE queries. For example, user issues the following query:

{*Location, Person.gender = male*}

This should search for the text that is annotated as *Location* and *Person*, where the *Person* annotation must contain a feature called *gender* with value *male*. In this case, the ANNIC query parser creates two separate terms (*Location* and *Person.gender = male*). In order to make both terms referring to the same location, positions of these terms must remain same. If the position of first term is *n*, Lucenes phrase query implementation makes the position of second term to *n+1*. This results into a pattern where the first annotation is *Location* and is followed by the annotation *Person.gender = male*. To overcome this problem, one solution is to pass customized term positions along with terms to the phrase query. Given a term and its position respective to its previous term, Lucene searches within its index to find the term only at the given position. Thus, instead of searching the second term at the *n+1* position, Lucene seeks a term that occurs at *n* position. This disables automatic increment in term's position and also allows searching for the overlapping annotation.

But even after this arrangment, there exists one major overlapping problem. For example for the text "Mr. Tim-Berners Lee told ...", where the text "Mr." is annotated as "Title", "Tim-Berners" as "FirstName", "Lee" as "Surname", "Mr. Tim-Berners Lee" as "Person" and finally "told" as "Token" with the part-of-speech tag "verb". For these annotations, the tokens "Title" and "Person" will be placed at the same position in the token stream, while "FirstName", "Surname" and "Verb" will be placed one after another after the "Title" and the "Person" annotations. This results into incorrect results when the query is : {Person} {Token.string == "told"}. When searching this pattern in the token stream, "Person" is not followed by the Token string "told", instead "Person" is followed by the annotation "FirstName", which is followed the annotation "Surname" and which is followed by the "told". To solve this problem, after converting the JAPE query into the Lucene query terms, we issue the query that contains only the initial terms which refer to the same location. For example, instead of querying with {Person}{Token.string == "told"}, we query index with {Person}. As a result this query returns all positions from the

Table 4: JAPE pattern tokens and their respective Query terms

	Query Term	
JAPE Pattern Token	**Term Text**	**Term Type**
String	String	Token.string
{annotationType}	annotationType	*
{annotationType.featureType == value}	value	annotationType.featureType

Table 5: Klene Characters

Query	Interpretation
({A})+3	({A}) | ({A}{A}) | ({A}{A}{A})
{B}({A})*3	({B}) | ({B}{A}) | ({B}{A}{A}) | ({B}{A}{A}{A})
{B}({A} | {C})+2	({B}{A}) | ({B}{C})| ({B}{A}{A}) | ({B}{A}{C}) | ({B}{C}{A}) | ({B}{C}{C})

token stream where the annotation is "Person". We compare the rest terms (i.e. "Token.string == "told") by fetching terms after the "Person" annotation and by comparing query terms with them.

Annotations in left and right contexts: As described earlier, each token stream referring to a separate document in the corpus is stored in a separate file as a Java serializable object and is retrieved once the Lucene tokens matching the query results in the token stream are known. Along with a list of documents, positions (i.e. where these annotations in the token stream appear) are also retrieved. This helps in skipping to a specific location in a token stream and reduces the lookup time. Numbers of tokens, specified in a context window field at run-time, are also fetched from the token stream before and after the pattern so as to show them as the left and right contexts in the GUI.

Klene operators: ANNIC supports two operators, + and *, to specify the number of times a particular annotation or a sub pattern should appear in the main query pattern. Here, *({A})+n means one and up to n occurrences of annotation {A} and ({A})*n* means *zero or up to n occurrences of annotation {A}*. Table 5 lists few example queries to illustrate the use of klene characters.

5 ANNIC user interface

ANNIC provides an advanced user interface at the presentation layer that allows users to index a large collection of documents (i.e. corpus), search indices and analyze the found patterns along with their left and right contexts concordances. At indexing time, the user can specify the corpus to be indexed, the annotation type that acts as document tokens, annotation set which contains the annotations to index, features and annotation types not to include in index and finally the location of index on the local or network file system. At search time, the user specifies the maximum number of documents to retrieve as results, number of tokens to show in the left and right contexts and finally the JAPE pattern query.

5.1 ANNIC Viewer

Figure 1 gives a snapshot of an ANNIC search window. The bottom section in the window contains the patterns along with their left and right context concordances and the section at top shows graphical visualization of annotations. ANNIC shows each pattern in a separate row and provides tool tip that shows the query that the selected pattern refers to. Along with its left and right context texts, it also lists the name of documents that the patterns come from. When the focus changes from one pattern to another, graphical visualization of annotations (GVA, above the pattern table) changes its current focus to the selected pattern. Here, users have an option of visualising annotations and their features for the selected pattern. The figure shows the highlighted spans of annotations for the selected pattern. Annotation types and features can also be selected from the drop-down combo box and their spans can also be highlighted into the GVA. When users choose to highlight the features of annotations (e.g. Token.category), GVA shows the highlighted spans containing values of those features. Whereas when users choose to highlight the annotation with feature all, ANNIC adds a blank span in GVA and shows all its features in a popup window when mouse enters the span region. A new query can also be generated and executed from the ANNIC GUI. When clicked on any of the highlighted spans of the annotations, the respective query clause is placed in the *New Query* text box. Clicking on *Execute* issues a new query and refreshes the GUI output. ANNIC also provides

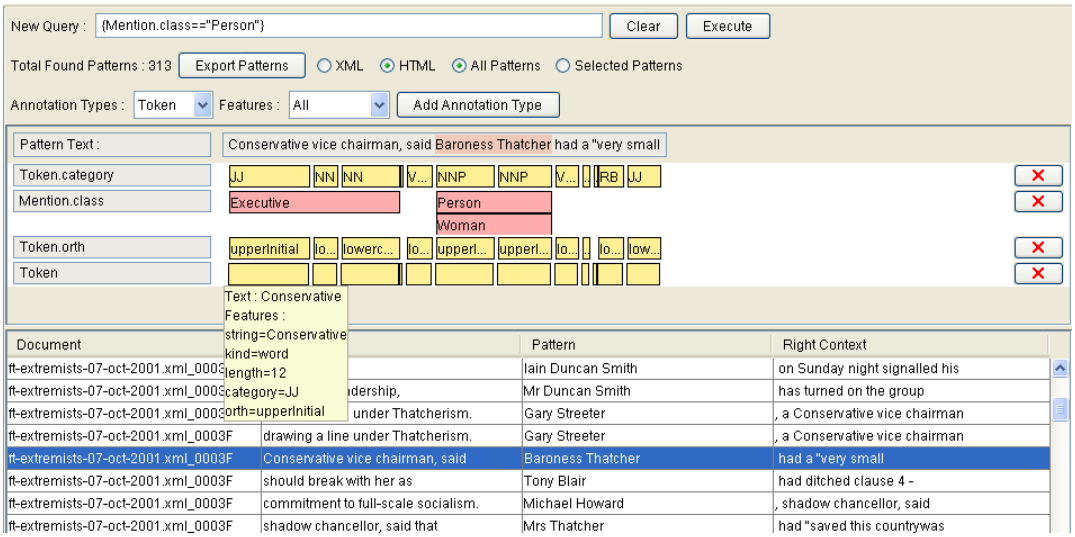


Figure 1: ANNIC Viewer

an option to export results in XML or HTML files with options of all patterns and selected patterns.

6 Applications of ANNIC

ANNIC is used as a tool aiding the development of JAPE rules. Language engineers use their intuition when writing JAPE rules trying to strike the ideal balance between specificity and coverage. This requires them to make a series of informed guesses which are then validated by testing the resulting ruleset over a corpus. ANNIC can replace the guesswork in this process with actual live analysys of the corpus. Each pattern intended as part of a JAPE rule can be easily tested directly on the corpus and have its specificity and coverage assesed almost instantaneously.

ANNIC can be used also for corpus analysys. It allows querying the information contained in a corpus in more flexible ways than simple full-text search. Consider a corpus containing news stories that has been processed with a standard named entity recognition system like ANNIE[2]. A query like {Organization} ({Token})*3 ({Token.string==’up’}|{Token.string==’down’}) ({Money} | {Percent}) would return mentions of share movements like “BT shared ended up 36p” or “Marconi was down 15%”. Locating this type of useful text snippets would be very difficult and time consuming if the only tool available were text search. ANNIC can also be useful in helping scholars to analyse linguistic

Table 6: ANNIC queries

QP	Patterns
1	{Token.string==Microsoft} | ”Microsoft Corp”
2	{Person} {Person}
3	{Person} {Token.category==IN} {Token.category==DT})*1 {Organization}
4	({Token.orth==allCaps} | {Token.orth==upperInitial}) ({Token.kind==number,Token.length==1})+2 {Token.kind==number,Token.length==1} ({Token.orth==allCaps} | {Token.orth==upperInitial})
5	({Token.kind==number})+4 ({Token.string =”/”} | {Token.string==”-”}) ({Token.kind==number})+2 ({Token.string==”/”} | {Token.string==”-”}) ({Token.kind==number})+2
6	{Title} ({Token.orth==upperInitial} | {Token.orth==allCaps}) ({FirstPerson})*1
7	{Token.category==”DT”} ({Token.category==”NNP”} | {Token.category==”NNPS”}) ({Token.category==”NNP”} | {Token.category==”NNPS”})
8	({Token.category==”DT”})*1 {Location} {Token.category==”CC”} ({Token.category==”DT”})*1 {Location}
9	{Token.category==”IN”} ({Token.category==”DT”})*1 {Location} {Token.category==”CC”} ({Token.category==”DT”})*1 {Location}
10	{Organization}{Token.category==”IN”} ({Token.category==”DT”})*1 {Location}

QP=Query Pattern

[2]GATE is distributed with an IE system called ANNIE, A Nearly-New IE system.

Table 7: ANNIC query results

	BNC 10%		HSE		NEWS	
QP	**ST**	**P**	**ST**	**P**	**ST**	**P**
1	11.276	112	0.5	0	1.252	3
2	24.798	17	2.0	0	0.933	12
3	5.23	6	7.0	6	0.432	2
4	24.33	24	26.458	14	0.803	0
5	50.139	264	110.738	39	6.652	36
6	39.029	238	120.054	180	12.37	1038
7	99.813	480	192.013	321	16.854	1261
8	62.971	81	126.823	124	5.508	281
9	52.08	43	96.735	67	3.672	134
10	6.191	10	11.875	5	0.692	11

QP=Query Pattern,ST=Search Time,P=Patterns

corpora. Sumerologists, for instance, could use it to find all places in the ETCSL corpus [3] where a particular pair of lemmas occur in sequence.

7 Performance Results

In order to evaluate the performance of ANNIC, we experimented on three different corpora (large, medium, and small), processed with GATE: 10% of the BNC (British National Corpus)(374 documents,1443.84MB), HSE (Health and Security Experiments)(192 documents,896MB), and finally the NEWS corpus (446 documents, 39.4MB).

We tested the performance with several types of queries: string only queries, combinations of strings and linguistic data, and patterns with quantified Klene operators. Table 6 lists some of the different types of queries which were issued over the indexed corpuses. Table 7 gives the statistics of output of these queries. It provides different statistics including the time taken by ANNIC to retrieve the results and the number of patterns retrieved.

8 Related Work

(McKelvie & Mikheev 98) describe a suite of programs, LT INDEX, that supports indexing of large SGML documents. It indexes elements by their position in the document structure and by their textual content. ANNIC is more generic, because it can cope with a wider range of formats, while covering the same functionality.

CUE (Corpus Universal Examiner) system (Mason 98) splits the corpus data into different data streams (e.g. actual words, POS information), which are stored along with their positioning information in the index. Unlike CUE, ANNIC maintains a fixed structured data format (Term string, Term type, position) within indices and converts all annotations and their features into this consistent format. (Christ 94) describes separate layers for their corpus query system, where index access is described at the physical layer; interpreting user queries, searching within indices and processing of results at the logical layer; and the graphical user interface at the presentation layer. Their system is aimed at indexing all text documents that their modules at the physical layer can convert into a predefined format. Similarly ANNIC also indexes any document format that is supported by the GATE. Lucene and GATE both play a vital role in carrying out the tasks at physical layer. GATE reads different kinds of documents (SGM, EML, MAIL, XHTM, RTF, XML, SGML, HTML, TXT etc.) from a file system or from the web and transforms them into GATE documents, which are then processed by the ANNIC via the GATE API. ANNIC then converts them in a format that Lucene can index and store. GATE, ANNIC and Lucene work altogether at the logical layer. GATE processes the documents and provides an API that helps ANNIC to deal with document text, annotations and their features. ANNIC takes queries from users, interprets them using the query parser and submits them to Lucene. Once the results are out, ANNIC uses respective token streams stored under the index directory to fetch the patterns and left and right contexts along with their annotations to prepare the GUI.

(Gaizauskas *et al.* 03) describe a system, XARA that indexes any well-formed XML document. It combines an indexer, a server and a windows client. Indexer requires information like how element content is to be tokenized and how tokens are to be mapped to index terms etc. ANNIC supports not only XML but many other types of documents supported by the GATE. Similar to XARA, in ANNIC as well, the decision of how documents be tokenized is left on a user (e.g., GATE supplies tokenisers for several languages).

In order to investigate new models for semi structured data that are appropriate to XML, (Buneman *et al.* 98) describes a query language that is beyond any XML query languages. They describe extraction rules that consist of expressions along the tree and are expressed using the HTML Extraction Languages (HEL). Their query

[3] http://www-etcsl.orient.ox.ac.uk/

language comes with navigation operators, regular expressions and conditions to retrieve information even from the nested structures. ANNIC query parser works on top of the GATE annotations and features and supports search over overlapping annotations and features. Its advanced user interface allows users to visualize the nested structure of the annotations with their features highlighted.

(Kazai *et al.* 04) discuss the overlapping problem in content-oriented XML retrieval. They discuss the INitiative for the Evaluation of XML Retrieval (INEX) system, which discusses the matrices to evaluate the XML retrieval results. Their argument is that if in an XML document, a sub element satisfies a content-oriented query, parent element would also satisfies the same query. Thus, instead of including only a subcomponent in the result, INEX also includes the parent component. In ANNIC, the overlapping problem, as discussed in (Kazai *et al.* 04), does not exist due to two reasons. 1) Annotations in GATE documents are stored as an annotation graph. Thus comparing the structure of XML documents where elements contain texts, in GATE documents annotations are created over the text. 2) ANNIC queries are very specific about the annotation types, i.e. query itself describes the annotation type in which the string should be searched. If user does not specify annotation type, ANNIC does it automatically to search strings with the GATE token annotation type.

References

(Bird *et al.* 00a) S. Bird, P. Buneman, and W. Tan. Towards a query language for annotation graphs. In *Proceedings of the Second International Conference on Language Resources and Evaluation*, Athens, 2000.

(Bird *et al.* 00b) S. Bird, D. Day, J. Garofolo, J. Henderson, C. Laprun, and M. Liberman. ATLAS: A flexible and extensible architecture for linguistic annotation. In *Proceedings of the Second International Conference on Language Resources and Evaluation*, Athens, 2000.

(Buneman *et al.* 98) P. Buneman, A. Deutsch, W. Fan, H. Liefke, A. Sahuguet, and W.C. Tan. Beyond XML Query Languages. In *In Proceedings of the Query Language Workshop (QL'98)*, 1998.

(Cassidy 02) S. Cassidy. Xquery as an annotation query language: a use case analysis. In *Proceedings of 3rd Language Resources and Evaluation Conference (LREC'2002)*, Gran Canaria, Spain, 2002.

(Christ 94) O. Christ. A Modular and Flexible Architecture for an Integrated Corpus Query System. In *Proceedings of the 3rd Conference on Computational Lexicography and Text Research (COMPLEX '94)*, Budapest, 1994. `http://xxx.lanl.gov/-abs/cs.CL/9408005`.

(Cunningham *et al.* 02) H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*, 2002.

(Gaizauskas *et al.* 03) R. Gaizauskas, L. Burnard, P. Clough, and S. Piao. Using the XARA XML-Aware corpus query tool to investigate the METER Corpus. In *In Proceedings of the Corpus Linguistics 2003 Conference*, pages 227–236, Lancaster, UK, 2003.

(Grishman 97) R. Grishman. TIPSTER Architecture Design Document Version 2.3. Technical report, DARPA, 1997. `http://www.itl.nist.gov/div894/-894.02/related_projects/tipster/`.

(Kazai *et al.* 04) G. Kazai, M. Lalmas, and A. Vries. The Overlapping problem in Content-Oriented XML Retrieval Evaluation. In *Proceedings of the 27th International conference on Research and development in information retrieval*, pages 72–79, Sheffield, UK, 2004.

(Mason 98) O. Mason. The CUE Corpus Access Tool. In *Workshop on Distributing and Accessing Linguistic Resources*, pages 20–27, Granada, Spain, 1998. `http://www.dcs.shef.ac.uk/~hamish/dalr/`.

(McKelvie & Mikheev 98) D. McKelvie and A. Mikheev. Indexing SGML files using LT NSL. LT Index documentation, from `http://www.ltg.ed.ac.uk/`, 1998.

(Thompson & McKelvie 97) H. Thompson and D. McKelvie. Hyperlink semantics for standoff markup of read-only documents. In *Proceedings of SGML Europe'97*, Barcelona, 1997.

An integrated approach to Word Sense Disambiguation

Jordi Atserias and **Lluís Padró** and **German Rigau**
TALP Research Center
Jordi Girona Salgado, 1-3.
08036 Barcelona
{batalla,padro}@lsi.upc.edu

IXA Group
Euskal Herriko Unibertsitatea
Donostia
{rigau}@si.ehu.es

Abstract

This paper presents an extension to perform Word Sense Disambiguation of an integrated architecture designed for Semantic Parsing. In the proposed collaborative framework, both tasks are addressed simultaneously. The feasibility and robustness of the proposed architecture for Semantic Parsing have been tested against a well-defined task on Word Sense Disambiguation (the *SENSEVAL-II English Lexical Sample*) using automatically acquired models.

1 Introduction

This paper explores the use of new robust and flexible architectures towards Natural Language Understanding (NLU). The work here presented focuses in one of the main steps in NLU, Semantic Interpretation. As a first step, our main goal is to integrate two of the tasks involved in Semantic Interpretation: Word Sense Disambiguation and Semantic Parsing.

Word Sense Disambiguation (WSD hereafter), can be defined as the process of deciding the meaning of a word in its context. Our approach uses the possible senses of a word previously defined in a sense repository. In particular, we use WordNet (Fellbaum 98), a lexical taxonomy built at Princeton University that has become *de facto* the standard sense repository in the NLP community.

The goal of *Semantic Parsing* is to identify semantic relations between words in text, resulting in structures denoting various levels of semantic interpretation. For instance, trying to identify the semantic roles of the entities, (such as *Agent* or *Patient*) (Brill & Mooney 97). In this case, the process, named *Semantic Role Labeling* (SRL), has been the goal of the *shared tasks* of the last editions of SENSEVAL[1] and CONLL[2].

In this paper we will integrate WSD in an architecture already used for Semantic Parsing (Atserias *et al.* 01), allowing both tasks to be done simultaneously. Although, this architecture allows this integration, the lack of wide coverage resources for SRL which can be related to WordNet synsets has forced us to acquire automatically the lexical models needed to carry out these tasks. Although, the models acquired are based on syntactic dependencies not roles, they allow to test the flexibility and robustness of our approach against a well established WSD task.

2 Semantic Parsing & WSD

Despite the fact that *WSD* and *Semantic Parsing* are strongly correlated, traditionally, most of the systems treat both separately. Paradoxically, *WSD* can improve *Semantic Parsing*, as the different senses of a word could present different syntactic structures (specially verbs) and the other way round, Semantic Parsing can help WSD (e.g. selectional preferences could determine the right sense of the verb (Carroll & McCarthy 00)). In this paper we present a robust and flexible architecture that aims to integrate both in a collaborative way.

Our approach to *WSD* follows the same formalization for *Semantic Parsing* that of (Atserias *et al.* 01). This formalization was based on the application of lexicalized verbal models. Those models combine syntactic information (preposition, agreement, etc) and semantic information (roles, selectional preferences, etc.) as the model shown in Table 1.

In this system *Semantic Parsing* was carried out by means of finding the model/s which are the most similar/s to the input sentence. Following this approach and connecting those models to WordNet senses, at the same time that we identify the most similar model, the correct sense of the word will be also determined. In that way, we formalize a framework where *Semantic Parsing* and *WSD* are performed simultaneously.

[1] http://www.senseval.org/
[2] http://www.cnts.ua.ac.be/conll/

model *impersonal* for "hablar" (to talk)					
Synt.	Prep.	Rol	Semantics	Agree.	Optional.
SE	x	se	Top	no	no
PP	de, sobre	entity	Top	no	yes
PP	con	destination	Top	no	yes

Table 1: Example of LEXPIR Syntactic-Semantic model for *Semantic Parsing*

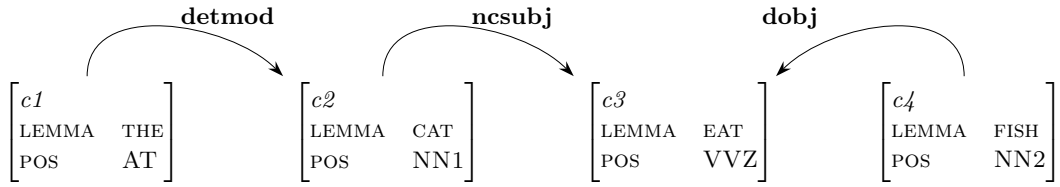


Figure 1: Syntactic Dependencies for "*The cat eats fish*"

During a pre-processing step, the input sentence containing the word to disambiguate is syntactically parsed and obtaining the syntactic dependencies between their elements using RASP (Carroll *et al.* 98). Figure 1 shows the dependency analysis obtained for the sentence *The cat eats fish*. Then each word is tagged with all its possible senses in WordNet. We use an specific tool for recognizing multi–word expressions (MWEs) according to WordNet (Arranz *et al.* 05) instead of the lemmatization/tokenization provided by RASP.

Once all possible senses in WordNet are added for each word, the input is also enriched with all the information associated to each sense using the *Multilingual Central Repository* (MCR)(Atserias *et al.* 04b): the expanded (Atserias *et al.* 04a) EuroWordNet's Top Concept Ontology (Vossen 98), Suggested Upper Merged Ontology (SUMO) (Niles & Pease 01) and MultiWordNet Domains (Magnini & Cavaglia 00).

The resulting information (syntactic dependencies and semantic information) for each word is converted to a feature structure which is the input to our system. The Figure 2 shows the feature structures obtained for the two different senses of *fish*: the food sense (*fish#n#1*) and the animal sense (*fish#n#2*). Henceforth, we will use the term *object* to refer to those feature structures.

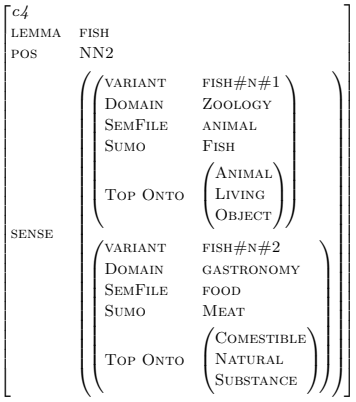


Figure 2: Object *Fish*

3 NLP and Constraint Satisfaction

Once the set of objects corresponding to the input sentence is obtained, it can be compared with the models. However, due to the richness of the language, robust methods to carry out this comparison are needed. Those methods should be capable to deal with semantic preferences or even to relax the syntactic structure. Thus, we formalize the problem of finding the most similar model for the input sentence as a Constraint Satisfaction Problem (CSP). CSPs have been already used in other NLP task: Part of Speech tagging (Padró 98), syntactic analysis (*Weighted Constraint Dependency Grammars* (Foth *et al.* 03)) or Machine Translation (Mikrokosmos's Hunter-Gatherer (Beale 96)).

In most NLP tasks, and specially in *WSD*, we need to express fuzziness, possibilities, preferences, costs, that is, soft constraints, and then the problem to be solved became over–constrained. Despite the advances in the area of solving efficiently these kind of CSP with soft constraints (or preferences) (Rudova 01), to find the best solution still remains an open issue.

A natural way to model Constraint Satisfaction Problem (CSP) is by means of *Consistent Labeling Problems* (CLP)(Messeguer & Larossa 95). Consistent Labeling Problems (CLP) can be solved efficiently via Relaxation Labeling. Relaxation labeling is a generic name for a family of iterative algorithms which perform function optimization, based on local information.

4 Consistent Labeling Problems

A *Consistent Labeling Problem* (CLP) basically stands for the problem of finding the most consistent assignments of a set of variables, given a set of constraints. Formally, a Labeling Problem is defined by a set of variables V_i, a set of labels (domain) for each variable D_i, a compatibility relation over tuples. Compatibilities are real-valued functions $r_{ij} : DxD \longrightarrow \Re$ where $r_{i,j}(a, b)$ refers to the compatibility of the simultaneous assignment of a to V_i and b to V_j. In a similar way than CSP aims to find total assignments where constraints are not violated, CLP looks for labeling where variables are highly compatible with respect to compatibility functions.

The feature structures (objects) that are the input of the system are represented in the CLP by means of a set of assignments. That is, each feature of the object is represented by a variable whose domain is the set of values of that feature. The variable $c1.att$ stands for the feature *att* of the object $c1$.

However, as can be seen in figure 2 the input objects contain complex features related to the different senses. In the CLP, these objects are amalgamated. That is, the representation associated to the different senses is combined. Figure 3 shows a simplified CLP representation for the sentence *"The cat eats fish"*. The variables amalgamate all the values of the same features for different senses. For instance, the *domain* features related to the *fish* object (c3) in figure 2 are mapped into the C3.DOMAIN variable, and the possible labels of the variable C3.DOMAIN corresponds to the union of the values for the domain feature.

The main idea of this amalgamation, is that, in a similar way than *Polaroid Words* (Hirst 87), when a model is chosen the representation of the object is selected and viceversa. The consistence between the sense selected and the selection of the corresponding labels in the variable are assured by a set of constrains.

Variable	Values
C1.POS*	{ NN1 }
C1.LEMMA*	{ cat }
C1.SENSE	{ cat#n#1, cat#n#2 ...}
C1.DOMAIN	{ Zoology, Factotum, Person, Transport}
C1.MODEL	{ NONE }
C1.ROLE	{ subj.m1.c2, subj.m2.c2}
C2.POS*	{ VVZ }
C2.LEMMA*	{ eat }
C2.SENSE	{ eat#v#1, eat#v#2 ...}
C3.DOMAIN	{Gastronomy, Chemistry, Factotum, Psychology, Zoology}
C2.MODEL	{ transitive }
C2.ROLE	{ TOP }
C3.POS*	{ NN1 }
C3.LEMMA*	{ fish }
C3.SENSE	{ fish#n#1, fish#n#2 }
C3.DOMAIN	{ Animal, Food}
C3.MODEL	{ NONE }
C3.ROLE	{ dobj.m1.c3}

Figure 3: CLP for *The cat eats fish*

However, not only the object features have to be represented in the CLP but also the relations between those objects. Most of the problems which are naturally modeled as a CLP do not have and implicit structure. Thus, to represent a structure between objects we need to use a kind-of dependency representation.

The combination of objects by means of a model is represented using two variables, a variable named *model* which represents the model which is applied and another variable named *role* which represents the dependency between the two objects. There is one special model, named NONE, to represent the null-model (that is the no application of any model) and one special role, named TOP, to represent the null-role (that is that the object do not take part in any model).

In order to identify a role from a model label we need a triplet *(role, object, model)*. For instance, the role *dobj* of the *m1* model for the object *eat* is represented as *(dobj, c3, m1)*.

Since a CLP always assigns a label to all the variables, we will use the two null-labels defined previously: NONE for the model variables (objects which do not have/use a model, usually leaf semantic objects with no sub-constituents) and the label TOP for the role variables (objects not playing a role in the model of a higher constituent, e.g. the sentence head).

4.1 Matching Roles and Objects

In order to see whether a model can be applied or not, we should determine which combination of objects could be used to fulfill the roles of the model. First we will establish which roles an object can play in isolation, that is regardless which objects fulfill the other roles of the model by means of a similarity measure between an *obj*ect and a *role*: $sim(obj, role)$. Once determined which pairs of role-object can be instantiated, it must be established which objects can be used together to best fulfill a model.

Some of the assignments/features which determine how much an object suits a role do not depend on the sense/model chosen and do not change in our amalgamated representation (static). For instance, in our representation the attributes *pos* or *lemma* are shared by all the senses.

Thus, the function *sim* could be split in two: a dynamic part sim_{dyn} and a static part sim_{static} which can be calculated only once (e.g. when building the CLP) and can be used to determine which objects can play a role initially in the CLP. On the other hand, the dynamic part, which depends on the sim_{dyn} could be represented as a set of constraint which takes into account the current state of the CLP (that is the weight associated to the assignment at each iteration). In the experiments carried out, the dynamic attributes are the *sense*, *em domain*, SUMO and *Top Onto*. For simplicity we have chosen a similarity function which combines independently the similarity of each attribute:

$$sim(obj, role) = \frac{\sum_{a \in Atts} sim_{att}(role.a, obj.a)}{\#Atts}$$

Next section describes the set of constraints which ensures a) that the model are well-formed (structural) and b) the good application of both, models and roles (matching). These constraints have a weight associated standing the compatibility ($\sim$) or the incompatibility ($\nsim$).

4.1.1 Structural Constraints

- **Object Uniqueness**: This first axiom ensures that an object can only fulfill a role:
  $[c_x.role = a] \nsim [c_x.role = b]$
  $\forall x \in Obj\ \forall a, b \in Roles(c_x) \mid a \neq b$

- **Role Uniqueness**: A role can only be fulfilled by one object:
  $[c_x.role = a] \nsim [c_y.role = a]$
  $\forall x, y \in Obj\ \forall a \in Roles \mid x \neq y$
  This constraint will avoid for instance that the object *cat* and *fish* fulfill the same role simultaneously.

- **Model Uniqueness**: The models are incompatible among them:
  $[c_x.model = a] \nsim [c_x.model = b]$
  $\forall x \in Obj\ \forall a, b \in Models, x \neq y$

- **Model Inconsistence**: A role can not be fulfilled by an object if the model to which the role belongs is not being instantiated:
  $[c_x.model = m_b] \nsim [c_y.role = (r, m_a, x)]$
  $\forall x, y \in Obj\ (r, x, m_a) \in Roles(y)$
  $m_b, m_a \in Models(x) \mid m_a \neq m_b$

- **TOP Uniqueness** Only one TOP:
  $[c_x.model = TOP] \nsim [c_y.model = TOP]$
  $\forall x, y \in Obj, x \neq y$

- **TOP Existence** At least a TOP:
  $[c_x.model = TOP] \sim \nexists\ [c_y.model = TOP]$
  $\forall x, y \in Obj \mid x \neq y$

- **NONE Support** The model NONE is compatible with the inexistence of the role assignments:
  $[c_y.model = NONE] \sim \nexists\ [c_y.role = a]$
  $\forall y \in Obj$

4.1.2 Matching Constraints

- **Model Support** In order to not penalise smaller models, the support a model receives is normalized by the number of its roles.
  $[c_x.model = m] \sim [c_y.role = (r, m, x)]$
  $\forall (r, m, x) \in Roles$
  For instance, if the model *eat-V4* has three possible roles (subj, dobj, dobj2), the constraint which supports this model depending on assignment of the role *dobj2* will be $[c3_{model} = \text{eat-V4}] \sim^{\frac{1}{3}} [c3_{role} = (\text{dobj2, eat-V4, c2})]$. The model will have also two similar constraints for the other two roles.

- **Role Support** The role support must take into account the sense which are associated to the object. Thus we need to compare each sense and the role:
  $[c_{role} = (r, m, x)] \sim^{w} [c_{sense} = s]$
  $\forall c, x \in Obj\ \forall s \in c.sense$ where w is sim_{dyn} between the senses of the object and the role. For instance, the constraint $[c3_{role} = (dob\#2,\ eat\text{-}V4,\ c2)] \sim^{2.45} [c3_{sense} = fish\#n\#2]$ will give support to the assignment *(dob#2, eat-V, c2)* taking into account the current weight of the assignment representing the sense *fish#v#2* and their similarity in WordNet[3] with the sense/s of the role *(dob#2, eat-V4, c2)*.

4.2 Sense Constraints

The following set of constraints ensures that at the same time a model is applied, the sense associated with this model is also selected, for both the head of the model and the rest of roles. As the current formalization does not include any constraint that modifies the *domain*, SUMO or *Top Onto*, these features do not need to be represented in the CLP and can be considered as *static* in the sense that we will not have to keep their consistence.

- **Head Sense Disambiguation** This set of constraints associate the application of a model with the selection of its sense for the *head* of the model:
  $[c_{sense} = s] \sim^{100} Or_{i=1}^{n} [c_{model} = m_i]$
  $\forall s \in c.sense$ and where $m_1 ... m_n$ is the set of models of c whose sense is s

  For instance, the constraint $[c2_{sense} = eat\#v\#3] \sim^{100} [c2_{model} = eat\text{-}V17]$ *or* $[c2_{model} = eat\text{-}V52]$ *or* $[c2_{model} = eat\text{-}V50]$ would give support to the assign of the third sense of *eat* if any of the models associated to the third sense (eat-V17, eat-V52, eat-V50) is selected.

[3]For the experiments we use the level of the first common ancestor

- **Role Sense Disambiguation** This set of constraints associates the sense of the role with the sense of the object which fulfills the role:
  $[c_{sense} = r.sense] \sim^{w} [c_{role} = (r, m, x)]$
  $\forall c \in Obj$ where w is $sim_{static}(obj_{r.sense}, role)$
  Where $obj_{r.sense}$ is the representation of the object corresponding to sense *r.sense*, for instance, $[c3_{sense} = fish\#n\#2] \sim^{2} [c3_{role} = (dob2,\ eat\text{-}V4,\ c2)]$ will select the second sense of *fish* if the object *c3* fulfills the role *dobj2* of model *eat-V4*. The sim_{static} will be calculated comparing the attributes associated to the object representing the second sense of *fish* and the role.

4.3 Initial Labeling

As *relaxation labeling* is an algorithm with local convergence, one of the main issues when using this algorithm is to establish the initial labeling from where the iterative process starts. Heuristically we initialize the role and model assignments according to the static similarity function, while for the sense assignments the SemCor frequency is been used.

5 Experiments

To prove the flexibility and robustness of our approach against *WSD* we applied our system to *English Lexical Sample* of SENSEVAL-II. This tasks consists on disambiguating the occurrences of 73 different words (noun, verbs and adjectives) in a corpus of 4,328 paragraphs. We choose this specific task because we plan to acquire the models from the examples of the training corpora and also because in SENSEVAL-III do not used WordNet senses for verbs directly.

In order to apply our system to this task, we need syntactic models which also contain semantic information about WordNet senses. Although there has been remarkable efforts to relate FrameNet and VerbNet with WordNet (Shi & Mihalcea 05), the coverage is still very low to face even a small Lexical Sample task (only 50 senses of the test are directly associated to a frame and only 640 sentences of the 4,328 could be solved correctly).

Thus, although its inherent complexity, we decide to build automatically these models from corpus. The acquisition this kind of models has many difficulties. First, the lack of disambiguated

corpus, or when existing their small size which makes impossible: a) to have a wide coverage of the senses in WordNet and b) to have models of all the syntactic subcategorization patterns for a sense. Moreover, state-of-the-art WSD systems and parsers still have a significant error rate that machine learning algorithms could not cope with.

5.1 Model Acquisition

In order to obtain models from semantically tagged corpus we used the same pre–processing than for the input (see section 2), obtaining for each sentence a set of syntactic dependencies enriched with semantic information from MCR. For each sentence, we extract for each word, the feature structures associated to its direct syntactic dependences (e.g. subj / obj / dobj). We take these set of relations as the set roles of a model for this word. For instance, taking the dependency analysis of sentence *The cat eats fish* in figure 1, two models could be acquired. One associated to *cat* (head) obtained from the dependency *The* —detmod→ *cat* and another associated to *eat*, using the dependencies *cat* —subj→ eat ←dobj— fish. Due to the big amount of models, our first approach for the experiments is to constraint the models to those having their *head* disambiguated.

The models have been obtained from two corpus with different characteristics. On one hand SemCor (Miller *et al.* 93), which is mostly disambiguated but due to his relatively small size (about 250.000 words) has a low sense coverage. On the other hand, SENSEVAL-II training corpus for the *English Lexical Sample* task (*Senseval*) whose 8,611 examples has only one word disambiguated. Table 2 shows the figures of the models obtained from each corpus for the words to be disambiguated in the test.

Notice that even we have obtained more models from Semcor, their sense distribution and coverage is different than for the *Training*. While *Training* models are distributed among all the senses in the test corpus, the models obtained from Semcor are associated to the most frequents.

	Number of Models
Semcor	7,344
Senseval	4,438

Table 2: Models acquired

6 Results

Table 3 shows the results (**P**recision and **R**ecall) obtained for the SENSEVAL-II *English Lexical Sample* test using the models obtained from Semcor and the *Senseval* corpus respectively.

Using the models obtained for each corpus, three different experiments have been performed, varying the level of semantic information used to determine the similarity between object and role: without any semantic information (**Syntax**), using only the information from WordNet (**Synset**) and using the information associated to each sense in the MCR.

For the syntactic attributes, we constraint the object that could instantiate a role, to those whose syntactic relation and preposition is the same. This restriction is probably too strong and drastically reduces the impact of increasing the semantic information.

	Models			
	Senseval		**Semcor**	
	P	**R**	**P**	**R**
MCR	48.3	26.9	28.3	15.9
Synset	48.2	26.9	27.5	15.5
Syntax	47.9	26.8	27.0	15.2

Table 3: Results in **P**recision and **R**ecall

Although at a synset level, the results of the system seem to be modest, when using the (*coarse*) grained evaluation of SENSEVAL-II our system reach the 59% of precision (41% using Semcor). We believe that this big diference in the figures is due to the lack of applicable models of the right sense, specially when using Semcor (a close-world-assumption is implicit in our formalization and the system chosses the most similar model among all the applicable).

Checking if each test sentence has at least a role with the same syntactic relation and preposition for a model associated to the correct sense to be disambiguated, we establish an upper bound of 70% for our system using the actual models.

We consider that the results obtained prove the feasibility of our approach, although they are slightly below the state-of-the-art of *WSD*, but highly above on the current figures for *Semantic Parsing*. Moreover, we should take into account than we have made no tuning (neither on the attributes nor on the similarity functions) and that the models used where obtained automatically.

7 Discussion

The automatically obtained models suffer several limitations and do not always allow to build an adequate semantic representation. For instance for a piece of sentence like *... clean dental surface ...* with a the dependency analysis (*dental* —mod→ *surface* —dobj→ *clean*), the system will build a representation for *dental* —mod→ *surface* which is basically associated to the semantic of his head, *surface*. As a consequence the verb *clean* is wrongly disambiguated, as the models associated to *clean#v#3* (to clean a house) are the ones more related to clean a *surface*. The fundamental piece of information that a *dental surface* is also a *body_part* is not captured by our automatically obtained models, while more simple WSD systems, such as using a bag of words, are able to capture and use that relation.

On the other hand, the current prototype makes a shallow integration of the syntactic and semantic level, so the system is sensitive to errors in the syntactic analysis being not able to disambiguate a word if a dependency analysis is not obtained.

Regarding the models acquired for Semcor, although fully disambiguated, they do not provide enough coverage. This sparseness makes more difficult to cope with inconsistencies or errors from the corpus.

The disambiguation capability of the system also depends greatly on the information available to discriminate the senses. Thus, it could be difficult be able to distinguish between senses whose MCR representation is almost the same (e.g. the five senses of *child*)).

8 Conclusions & Future Work

We have shown that it is possible to develop a more robust and flexible architecture for SEMANTIC PARSING using CSP techniques and that it can be solved efficiently using well-known optimization algorithms (such as relaxation labeling algorithms). Moreover, this formalization can be extended to other models that combine syntactic and semantic information (e.g. FrameNet).

In this paper we have presented an architecture able to integrate *Semantic Parsing* and *WSD*, where both tasks could collaborate. The system has been tested in a *WSD* task (SENSEVAL-II English Lexical Sample) using automatically acquired models.

Future lines of research include, first to extend the level of integration of *Semantic Parsing* and *WSD* using richer semantic models, and second to improve the system itself (e.g. tuning the similarity functions, propagating semantic information, etc.).

References

(Arranz *et al.* 05) V. Arranz, J. Atserias, and M. Castillo. Multiword expressions and word sense disambiguation. In Alexander Gelbukh, editor, *CICLING'05*, volume LNCS 3406, 2005.

(Atserias *et al.* 01) J. Atserias, L. Padró, and G. Rigau. Integrating multiple knowledge sources for robust semantic parsing. In *Proceedings of the International Conference, Recent Advances on Natural Language Processing RANLP'01*, Bulgaria, 2001.

(Atserias *et al.* 04a) J. Atserias, S. Climent, and G. Rigau. Towards the meaning top ontology: Sources of ontological meaning. In *4rd International Conference on Language Resources and Evaluations (LREC)*, 2004.

(Atserias *et al.* 04b) Jordi Atserias, Luis Villarejo, German Rigau, Eneko Agirre, John Carroll, Bernardo Magnini, and Piek Vossen. The MEANING multilingual central repository. In *Proceedings of the Second International Global WordNet Conference (GWC'04)*, Brno, Czech Republic, January 2004.

(Beale 96) Stephen Beale. *Hunther-Gatheter: Applying Constraint Satisfactioon, Branch-and-Bound and Solution Synthesis to computational Semantics*. Unpublished PhD thesis, Computer Research Laboratory, New Mexico State University, 1996.

(Brill & Mooney 97) Eric Brill and Raymond J. Mooney. An Overview of Empirical Natural Language Processing. *Artificial Intelligence Magazine*, 18(14):13–24, 1997. Special Issue on Empirical Natural Language Processing.

(Carroll & McCarthy 00) J. Carroll and D. McCarthy. Word sense disambiguation using automatically acquired verbal preferences. *Computers and the Humanities. Senseval*, 34(1-2), 2000.

(Carroll *et al.* 98) J. Carroll, G. Minnen, and E. Briscoe. Can subcategorisation probabilities help a statistical parser? In *Proceedings of the Sixth ACL/SIGDAT Workshop on Very Large Corpora*, pages 118–126, 1998.

(Fellbaum 98) C. Fellbaum, editor. *WordNet. An Electronic Lexical Database*. The MIT Press, 1998.

(Foth *et al.* 03) Kilian Foth, Wolfgang Menzel, and Ingo Schröder. Robust parsing with weighted constraints. to appear in Natural Language Engineering, 2003.

(Hirst 87) Graeme Hirst. *Semantic Interpretation and the Resolution of the ambiguity*. Studies in Natural Language Processing. Cambridge University Press, 1987.

(Magnini & Cavaglia 00) B. Magnini and G. Cavaglia. Integrating subject field codes into wordnet. In *Proceedings of the Second Internatgional Conference on Language Resources and Evaluation LREC'2000*, Athens. Greece, 2000.

(Messeguer & Larossa 95) Pedro Messeguer and Javier Larossa. Constraint satisfaction as global optimization. In *IJCAI*, 1995.

(Miller *et al.* 93) G. Miller, C. Leacock, R. Tengi, and R. Bunker. A Semantic Concordance. In *Proceedings of the ARPA Workshop on Human Language Technology*, 1993.

(Niles & Pease 01) I. Niles and A. Pease. Towards a standard upper ontology. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems*, pages 17–19. Chris Welty and Barry Smith, eds, 2001.

(Padró 98) Lluís Padró. *A Hybrid Environment for Syntax-Semantic Tagging*. Unpublished PhD thesis, Departament de Llenguatges i Sistemes Informàtics. Universitat Politècnica de Catalunya. Barcelona, 1998.

(Rudova 01) Hana Rudova. *Constraint Satisfaction with Preferences*. Unpublished PhD thesis, Masaryk University, 2001.

(Shi & Mihalcea 05) Lei Shi and Rada Mihalcea. Putting pieces together: Combining framenet, verbnet and wordnet for robust semantic parsing. In *CICLING'05*, Mexico, 2005.

(Vossen 98) P. Vossen, editor. *EuroWordNet: A Multilingual Database with Lexical Semantic Networks* . Kluwer Academic Publishers , 1998.

Adapting a general parser to a sublanguage

Sophie Aubin*, Adeline Nazarenko* and **Claire Nédellec****
(*) LIPN, University of Paris 13 & CNRS UMR 7030
99, av. J.B. Clément, F-93430 Villetaneuse, France
{sophie.aubin,nazarenko} at lipn.univ-paris13.fr
(**) Unité Mathématique Informatique et Génome (MIG, INRA)
Domaine de Vilvert, F-78350 Jouy en Josas Cedex, France
claire.nedellec at jouy.inra.fr

Abstract

In this paper, we propose a method to adapt a general parser (Link Parser) to sublanguages, focusing on the parsing of texts in biology. Our main proposal is the use of terminology (identification and analysis of terms) in order to reduce the complexity of the text to be parsed. Several other strategies are explored and finally combined among which text normalization, lexicon and morpho-guessing module extensions and grammar rules adaptation. We compare the parsing results before and after these adaptations.

1 Introduction

Most available NLP tools are developed for general language while processing technical texts, *i.e.* sublanguages, becomes a necessity for various applications like extracting information from biological texts (see (Grishman 01),(Pyysalo *et al.* 04), (Grover *et al.* 04) and (Akane *et al.* 05)). In order to assist the biologists in their daily bibliographical work, the ExtraPloDocs project[1] develops the natural language processing and machine learning tools that enable to build focused information extraction systems in genomics (gene-protein interaction, gene fonctionalities, gene homologies, etc.) at a reasonable cost. Beyond keyword and statistics based approaches, extracting such relational information must be based on syntax to achieve good precision and coverage (see for instance (Ding *et al.* 03)). We therefore need a reliable syntactic parsing of the texts dealing with genomics.

Instead of redeveloping new parsers for each sublanguage, we try to define a method for adapting a general parser to a specific sublanguage. This paper presents a strategy to adapt the Link Parser (LP) (Sleator & Temperley 91) to parse Medline abstracts dealing with genomics.

[1]ExtraPloDocs website : http://www-lipn.univ-paris13.fr/RCLN/Extra/ExtraPloDocs/
These results are also exploited for the development of specialized search engines in the ALVIS project (STREP) : http://cosco.hiit.fi/search/alvis.html

In this paper, we first discuss the question of sublanguages and the different strategies that can be adopted to parse technical texts. Section 3 presents the context of the adaptation of the LP to the biological domain. In section 4, we analyse several cases of parsing failure along with the solutions we propose to adapt the parser. We finally present the evaluation of the modifications we made on the LP grammar and lexicon.

2 Previous works

Sublanguages have been studied for a long time even though it remains a rather confidential part of linguistic and NLP studies. It is noticeable that in specific domains of knowledge, among certain communities and in particular types of texts, people have their own way of writing. These specific languages are called either sublanguages (Harris *et al.* 89; Grishman & Kittredge 86), restricted or specialized languages depending on the fact that one focuses on the continuity or the gap between these languages and the "usual language". In fact, a sublanguage is a restricted (fewer lexicon items and semantic classes) as well as a deviant language (original lexicon items and phrasings). This is also noticeable from a distributional point of view. As Harris noticed it, a sublanguage can be characterized by its selectional restrictions and more generally by the distribution of lexicon items and syntactic patterns.

(Sekine 97) has argued that parsing should be domain dependent. Three alternative approaches can be considered. Several NLP teams have decided to develop a specialized parser for a given sublanguage (see for instance the String project (Sager *et al.* 87) or (Pustejovsky *et al.* 02)) but this approach is considered too expensive for many applications. A second track consists in training a grammar from a specialized corpus, which requires annotated corpora that are rare in specialized domains. An intermediate approach aims at manually adapting a parser as proposed

in (Pyysalo *et al.* 04). This is our approach. This work can be considered as a preliminary work to evaluate the potentialities of automating this adaptation.

Two different approaches have been explored for the parsing evaluation. The first is linguistically oriented and based on test suites, a set of sentences that illustrates the various syntactic structures that a parser is supposed to analyse like in TSNLP (Lehman 96). The second approach, more pragmatic and more common, consists in evaluating the performances of a parser on a given corpus supposed to be representative of the textual data to parse. We will show in the following that we adopted a mixed approach.

As we will see below, one of the main problems in parsing sublanguages is the ambiguity of prepositional attachment.

3 Context

3.1 The corpora

Three different corpora were built from Medline[2] abstracts (in English) dealing with transcription in *Bacillus subtilis*. As recommended by (Prasad & Sarkar 00) and (Srinivas *et al.* 98), we mixed the two evaluation standards by randomly selecting 212 sentences that we organized according to their linguistic specificities. Despite its relatively small size, the `MED-TEST` corpus is a good sample of the sublanguage of genomics. We also used a larger corpus of full abstracts (`TRANSCRIPT`, 16,981 sentences, 434,886 words) and the `GIEC` corpus made of 160 sentences expressing gene/protein interactions. The `GIEC` corpus was built and used as a benchmark corpus in the context of the Genic Interaction Extraction Challenge[3] joint to the ICML 2005.

3.2 The initial parser choice

In the context of our IE task, and particularly for the ontology acquisition, we need reliable and precise syntactic relations between the words of the whole sentence (except empty words). For those reasons, a symbolic dependency-based parser seemed to be the most adequate.

LP presents several advantages among which the robustness, the good quality of the parsing, the adequation of the dependency technique and representation with our IE task and the declarative format of its lexicon. From the results of the evaluation that we did on different parsers with the `MED-TEST` corpus, it turned out that dependency-based parsers have better results on long and complex sentences, particularly with coordinations. This conclusion is shared by (Ding *et al.* 03) who also worked on Medline abstracts. Other experiments, in the context of the ExtrAns project (Mollá *et al.* 00), showed that 76% of 2,781 sentences from a Unix manpage corpus were completely parsed by LP with no regard to the parsing quality, while we reach only 54% on the biological corpus. When looking at the quality of the parses, we noticed different kinds of errors depending either on the biological domain or on more general linguistic difficulties like ambiguous constructions. We propose three solutions to address these issues, the text normalization, the use of terminology and the adaptation of the lexicon/grammar of LP.

4 Diagnosis and adaptation

Our analysis of the performance of the Link grammar on the biological corpus confirms previous works. The main problems can be classified along the following axes.

4.1 "Textual noise"

Scientific texts present particularities that we chose to handle in a normalization step prior to the parsing. First, the segmentation in sentences and words was taken off from the parser and enriched with named entities recognition and rules specific to the biological domain. We also delete some extratextual information that alter the parsing quality. Finally, we use dictionaries and transducers to replace genes and species names by two codes, which prevents from extending the LP dictionary too much.

4.2 Unknown words

In the `TRANSCRIPT` corpus, we identified 6,005 out-of-lexicon forms (45,804 occurences) among 12,584 distinct words, *i.e.* 47.72%. They are mostly latin words, numbers, DNA sequences, gene names, misspellings and technical lexicon.

However, LP includes a module that can assign a syntactic category to an unknown word. It is based on the word suffix. Modifying the morpho-guessing (MG) module seemed a better strategy than extending the dictionary since biological objects differ from an organism to another. We then

[2] http://www.ncbi.nlm.nih.gov/entrez/query.fcgi
[3] http://genome.jouy.inra.fr/texte/LLLchallenge

created 19 new MG classes for nouns (*-ase*, *-ity*, etc.) and adjectives (*-al*, *-ous*, etc.) along with their rule.

In the same time, we added about 500 words of the biological domain to the LP lexicon in different classes, mainly nouns, adjectives and verbs.

4.3 Specific constructions

Some words already defined in the LP lexicon present a specific usage in biological texts, which implied some modifications including moving words from one class to another and adaptating or creating rules.

The main motivation for moving words from one class to another is that the abstracts are written by non-native English speakers. This point was also raised by (Pyysalo *et al.* 04). One way to allow the parsing of such ungrammatical sentences is to relax constraints by moving some words from the countable to the mass-countable class for instance.

Some very frequent words present idiosyncratic uses (particular valency of verbs for instance), which induced the modification or creation of rules. Numbers and measure units are omnipresent in the corpus and were not necessarily well described or even present in the lexicon/grammar. Other minor changes were made that are not mentioned in this paper.

4.4 Structural ambiguity

We identified two cases of ambiguity that can be partially resolved by using terminology.

Prepositional attachment is a tricky point that is often fixed using statistical information from the text itself (Hindle & Rooth 93; Fabre & Bourigault 01), a larger corpus (Bourigault & Frérot 04), the web (Volk 02; Gala Pavia 03) or an external resources such as WordNet (Stetina & Nagao 97). The second major ambiguity factor is the attachment of series of more than two nouns. As shown in Figure 1, neither a parallel attachment (lp) nor a serial one (lp-bio) seem to be satisfying. We noticed that such cases often appear inside larger nominal phrases often corresponding to domain specific terms. For this reason, we decided to identify terms in a pre-processing step and to reduce them to their syntactic head. If needed, the internal analysis of terms is added to the parsing result for the simplified sentence (see lp-bio-t). The strategy proposed by (Sutcliffe *et al.* 95) that consists in the linkage of the words

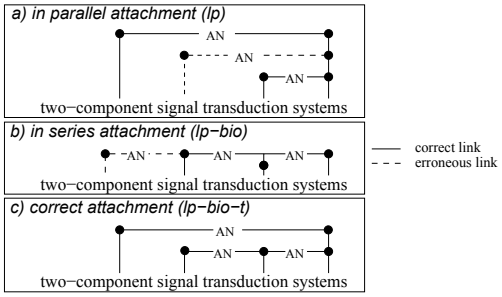


Figure 1: Series of nouns dependencies

contained in a compound (for instance *"sporulation_process"*) was excluded. It makes the lexicon size augment and does not reduce complexity for reasons due to the implementation of LP.

Figure 2 shows the influence of the adaptation on the parsing with the fixing of a segmentation error and the disambiguation of prepositional and nominal attachements.

Before practically integrating the use of terminology in our processing suite, we made a simulation of this simplification of terms.

5 Evaluation

We performed a two-stage evaluation of the modifications in order to measure the respective contribution of the LP adaptation on the one hand and of the term simplification on the other hand.

5.1 Corpus and criteria

We used a subset (10 files[4]) of the `MED-TEST` corpus but, contrary to the first evaluation (choice of a parser), we wanted to look at the quality of the whole parse and not only to specific relations.

Table 1 (for the `MED-TEST` subset) shows the way that out-of-lexicon words (OoL), i.e. unknown (UW) and guessed (GW) words, are handled by giving the percentage of incorrect morpho-syntactic category assignations with the original resources (lp), those adapted to biology (lp-bio) and finally the latter associated with the simplification of terms (lp-bio-t).

In Table 2, five criteria inform on the parsing time and quality for each sentence : the number of linkages (NbL), the parsing time (PT) in seconds, the fact that a complete linkage is found or not (CLF), the number of erroneous links (EL) and the quality of the constituency parse (CQ). (NbW) is the average number of words in a sen-

[4] 141 sentences, 2630 words

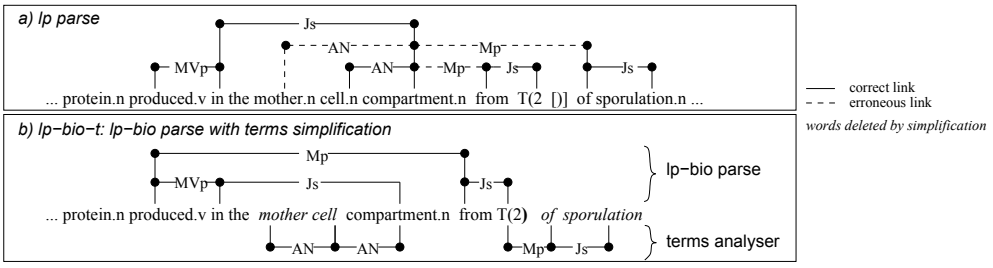


Figure 2: Example of parsing

	lp		lp-bio		lp-bio-t	
	a	b	a	b	a	b
UW	244	41.4%	53	52.8%	26	19.2%
GW	24	4.2%	72	0%	31	0%
OoL	268	38%	125	22.4%	57	8.8%

a : total MS assignations, b : % of incorrect assignations

Table 1: Incorrect MS category assignations

	lp	lp-bio		lp-bio-t	
crit.	avg	avg	%/lp	avg	%/lp
NbW	24.05	24.05	100%	18.9	78.6%
NbL	190,306	232,622	122.2%	1,431	0.75%
PT	37.83	29.4	77.7%	0.53	1.4%
CLF	0.54	0.72	133%	0.77	142.6%
EL	2.87	1.91	66.5%	1.15	40.1%
CQ	0.54	0.7	129.6%	0.8	148.1%

Table 2: Parsing time and quality

tence which varies with the term simplification. The results are given for each one of the three versions of the parser.

UW, GW, NbL, PT and CLF are objective data while EL and CQ necessitate a linguistic expertise. The CQ evaluation consisted in the assignation of a general quality score to the sentence.

5.2 Results and comments

The **extension of the MG module** reduced the number of erroneous morpho-syntactic category assignations (see Table 1) from 38% to 22.4%. 61% of the sentences where one or more assignation error was corrected by the MG module actually have better parsing results (15% have been degraded). More generally, the increase of guessed forms makes the category assignation more reliable.

The **extension of the lexicon** and the **normalization of genes and species names** discharged the two modules from 143 assignations out of 268, 50 of which were wrong. 64% of the sentences where one or more assignation error was corrected by the extension of lexicon have better parsing results (18% of the sentences were degraded).

The effect of the **rules modification and creation** is difficult to evaluate precisely though it is certain to play a part in the parsing improvement, especially the relaxing of constraints on determiners and inserts.

The most obvious contribution to the better parsing quality is the one of the **term simplification**. The drastic reduction in parsing time and number of linkages gives an idea of the reduction of complexity. It is not only due to the smaller number of words since the number of erroneous links is reduced of 60% while the number of words is reduced of only 21.4%. This confirms previous similar studies that showed a reduction of 40% of the error rate on the main syntactic relations with a French corpus.

Remaining errors are mainly due to four different phenomena. First, the normalization step, prior to the parsing, needs to be enhanced. Concerning LP, there are still lexicon gaps, wrong class assignations and a still unsatisfactory handling of numerical expressions. In addition, and like (Sutcliffe *et al.* 95), we identified a weakness of LP regarding coordination. A specific study of the coordination system in LP and in the biological texts may be necessary. Finally, some ambiguous nominal and prepositional attachments still remain in spite of the term simplification. These may be resolved in a post-processing step like in ExtrAns that uses a corpus based approach to retrieve the correct attachment from the different linkages given by LP for a sentence.

Other questions like the feeding of LP with a morpho-syntactically tagged text or the ameliora-

tion of the parse ranking in LP were not discussed in this paper but are interesting issues that we intend to study.

6 Conclusion

Since parsing is domain and language dependent, a general parser must be adapted to each given sublanguage. In the context of an IE project in biology, we have adapted the Link Parser to analyse the specific language of Medline abstracts in genomics. Our initial diagnosis mainly raised two different problems which are traditional in sublanguage analysis: the lack of lexical coverage and the structural ambiguity, especially in the cases of prepositional phrase attachments.

We showed that the lexical problem can be manually handled by introducing new words in the lexicon and by extending the morpho-guessing module. We also proposed to distinguish and combine terminological and syntactic analysis. In the same way as the morpho-syntactic tagging should be considered independently from the parsing, we argue that the terminology analysis must be handled separately. This represents the main automated part of the adaptation task. The use of terminology to alleviate the parsing task is relevant and applicable in the context of domain specific texts processing since terminology tools and lists of terms are generally available. It also reduces the part of effective modification of the lexicon/grammar of the parser. This first evaluation has shown promising results.

This work has been developed as part of the ExtraPloDocs (extraction of gene-protein interactions in Medline abstracts) and ALVIS projects. We have shown that combining the terminological and syntactic analysis has an important impact on the resulting parses because the terminological analysis simplifies the parser input.

7 Bibliography

References

(Akane *et al.* 05) Y. Akane, Y. Miyao, Y. Tateisi, and J. Tsujii. Biomedical Information Extraction with Predicate-Argument Structure Patterns. In *Proceedings of the First International Symposium on Semantic Mining in Biomedicine*, pages 60–69, 2005.

(Bourigault & Frérot 04) D. Bourigault and C. Frérot. Ambiguïté de rattachement prépositionnel : introduction de ressources exogènes de sous-catégorisation dans un analyseur syntaxique de corpus endogène. In *Actes des 11mes journées sur le Traitement Automatique des Langues Naturelles, Fès, Maroc*, 2004.

(Ding *et al.* 03) Jing Ding, Daniel Berleant, Jun Xu, and Andy W. Fulmer. Extracting Biochemical Interactions from MEDLINE Using a Link Grammar Parser. In *15th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'03)*, 2003.

(Fabre & Bourigault 01) C. Fabre and D. Bourigault. Linguistic clues for corpus-based acquisition of lexical dependencies. In *Proceedings of the Corpus Linguistics 2001 Conference, UCREL Technical Papers*, volume 13, pages 176–184. Lancaster University, 2001.

(Gala Pavia 03) N. Gala Pavia. *Un modèle d'analyseur syntaxique robuste basé sur la modularité et la lexicalisation de ses grammaires, Thèse de Doctorat.* Unpublished PhD thesis, Université Paris XI, Orsay, 2003.

(Grishman & Kittredge 86) Ralph Grishman and Richard Kittredge. *Analyzing Language in Restricted Domains. Sublanguage Description and Processing.* Lawrence Erlbaum Ass., Hillsdale, NJ, USA, 1986.

(Grishman 01) Ralph Grishman. Adaptive Information Extraction and Sublanguage Analysis. In *Proceedings of the Workshop on Adaptive Text Extraction and Mining at the 17th International Joint Conference on Artificial Intelligence (IJCAI'01)*, 2001.

(Grover *et al.* 04) Claire Grover, Maria Lapata, and Alex Lascarides. A Comparison of Parsing Technologies for the Biomedical Domain. *Journal of Natural Language Engineering*, 2004.

(Harris *et al.* 89) Zellig Harris, Michael Gottfried, Thomas Ryckman, Paul Mattick, Jr., Anne Daladier, T.N. Harris, and S. Harris. *The Form of Information in Science: Analysis of an Immunology Sublanguage.* Reidel, Dordrecht, 1989.

(Hindle & Rooth 93) Donald Hindle and Mats Rooth. Structural Ambiguity and Lexical Relations. In *Meeting of the Association for Computational Linguistics*, pages 229–236, 1993.

(Lehman 96) Sabine Lehman. TSNLP-test Suites for Natural Language Processing. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING'96)*, Budapest, 1996.

(Mollá *et al.* 00) Diego Mollá, Gerold Schneider, Rolf Schwitter, and Michael Hess. Answer Extraction Using a Dependency Grammar in ExtrAns. *Traitement Automatique de Langues (T.A.L.), Special Issue on Dependency Grammars*, 2000.

(Prasad & Sarkar 00) R. Prasad and A. Sarkar. Comparing test-suite based evaluation and corpus-based evaluation of a wide-coverage grammar for English. In *Using Evaluation within Human Language Technology Programs: Results and Trends. LREC'2000 Satellite Workshop*, pages 7–12, 2000.

(Pustejovsky *et al.* 02) J. Pustejovsky, J. Castano, and J. Zhang. Robust Relational Parsing over Biomedical Literature: Extracting Inhibit Relations. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 362–373, 2002.

(Pyysalo *et al.* 04) S. Pyysalo, P. Ginter, T. Pahikkala, Boberg J., Järvinen J., T. Salakoski, and J. Koivula. Analysis of link grammar on biomedical dependency corpus targeted at protein-protein interactions. In *Proceedings of the international Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA)*, pages 15–21, 2004.

(Sager *et al.* 87) Naomi Sager, Carol Friedman, and Margaret S. Lyman. *Medical Language Processing: Computer Management of Narrative Data.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1987.

(Sekine 97) Satoshi Sekine. The Domain Dependence of Parsing. In *Proceedings of the Applied Natural Language Processing (ANLP'97)*, pages 96–102, Washington D.C., USA, 1997.

(Sleator & Temperley 91) D. Sleator and D. Temperley. Parsing English with a Link Grammar. Technical report, Carnegie Mellon University, 1991.

(Srinivas *et al.* 98) B. Srinivas, A. Sarkar, C. Doran, and B.A. Hockey. Grammar and Parser Evaluation in the XTAG Project. In *Workshop on the Evaluation of Parsing Systems*, 1998.

(Stetina & Nagao 97) J. Stetina and M. Nagao. Corpus Based PP Attachment Ambiguity Resolution with a Semantic Dictionary. In J. Zhou and K. W. Church, editors, *Proceedings of the Fifth Workshop on Very large Corpora*, pages 66–80, Beijing, China, 1997.

(Sutcliffe *et al.* 95) R. F. E. Sutcliffe, T. Brehony, and A. McElligott. The Grammatical Analysis of Technical Texts using a Link Parser. In *Second Conference of the Pacific Association for Computational Linguistics, PACLING'95*, 19-22 April 1995.

(Volk 02) Martin Volk. Using the Web as Corpus for Linguistic Research. In Renate Pajusalu and Tiit Hennoste, editors, *Tähendusepüüdja. Catcher of the Meaning. A Festschrift for Professor Haldur Õim.* Publications of the Department of General Linguistics 3. University of Tartu, Estonia, 2002.

A syntactic strategy for filtering sentences in a question answering system

Vincent Barbier and **Anne-Laure Ligozat**
LIMSI-CNRS
University of Paris Sud
BP 133, 91403 Orsay, France
barbier,annlor@limsi.fr

Abstract

In a question answering system, the first steps consist in retrieving documents relevant to the question, from which sentences are extracted. In these steps, the possible variations between the formulations of the question and the candidate sentences should be taken into account. The selection of documents has to be large enough to ensure a high recall, but the noise generated by the reformulations has to be contained. In this article, we will present a method for filtering and reranking the candidate sentences of the documents according to syntactic criteria.

1 Introduction

In a question answering system, the first steps consist in retrieving documents relevant to the question. This selection should ideally take into account the possible reformulations of the question, in order to ensure a high recall. But accepting important semantic variations leads to very noisy results, thus the documents retrieved have to be filtered. In order to filter them, a linear distance between the terms of the question found in the documents can be calculated. But this kind of distance is not very reliable. We chose to use instead a syntactic distance between these words in order to improve the precision of our selection.

In this paper, we will make a brief presentation of our question answering system QALC. Then we will detail the difficulties of passage selection, and the different strategies that can be used to face these difficulties. We will afterwards describe our solution, based on a syntactic filtering, and present an evaluation of this solution on a corpus of questions and answers. Finally, we will give some perspectives to our work.

2 Selection of relevant documents

2.1 QALC architecture

Our question answering system QALC is composed of four main modules: question analysis, document retrieval, document processing and answer extraction (Ferret *et al.* 02). The architecture of the system is described figure 1.

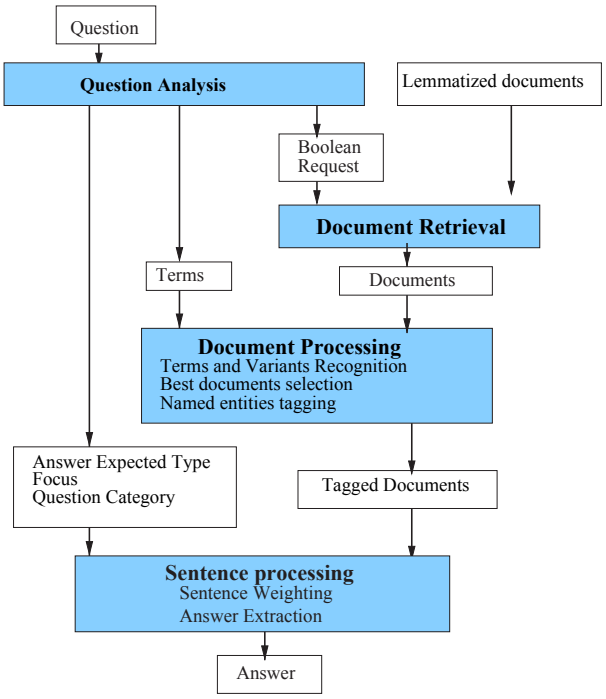


Figure 1: Architecture of the QALC question answering system

The question analysis module determines some information about the question: expected type of the answer, category of the question, keywords... This information is first used to retrieve documents thanks to the search engines, Lucene [1] for a French corpus, and MG [2] for an English one. These documents are then re-indexed by Fastr (Jacquemin 99) which recognizes morphological, syntactic and semantic variants of simple or composed terms of the question, and a subset of the highest ranked ones is kept. The named enti-

[1] http ://jakarta.apache.org/lucene/docs/index.html
[2] Managing Gigabytes, http://www.mds.rmit.edu.au/mg/intro/about_mg.html

ties tagging module is then applied to these documents. The final module is in charge of extracting the answers from weighted sentences: first, the sentences of the documents are weighted according to the presence of the terms of the question and of named entities and their linear distance, then, answers are extracted from the sentences, the process depending on the expected type of the answer.

2.2 Passage selection strategy

The sentences in which the answers are searched for are then the result of several successive selections:

- A first selection based on the non-empty words of the question retrieves the documents.
- Fastr proceeds to a second selection according to the recognition of mono and multi-term variants.
- Sentences are selected according to weights depending on the presence of the question terms and their linear distance, and on the presence of named entities of the expected type.

We chose to focus particularly on the third selection. The ranking of the sentences influences this of the answers, and thus it is crucial to be able to detect the sentences which are most likely to contain the answer. In order to assess the quality of our ranking of sentences, we calculated the Mean Reciprocal Rank (MRR[3]) over the questions of the CLEF04 multilingual question answering evaluation[4] in which we participated. The MRR is of 0.306 for these questions. As an element of comparison, one can refer to (Tellex *et al.* 03) which made a comparison of several passage retrieval algorithms, and found MRRs ranging approximately from 0.26 and 0.43.

2.3 Selecting relevant passages for question answering

The output of the retrieval engine is usually reprocessed before extracting the answer, since this output is not best suited for question answering: the documents may not be ranked if the engine is boolean, and their selection is driven by the keywords of the question rather than by the question itself. For example, the documents may not contain an entity of the expected type. This process can be partly mixed with the question extraction: most systems (re)rank the documents and restrict them to passages, and then process to the answer extraction using various strategies, ranging from expected type recognition to logic proof of the answer. The length of these passages can be more or less long.

For the passage selection, classic information retrieval models can be used, by relying on statistical information to evaluate the relevance of a document to a given query, for example with *tf*idf*. But these methods are not completely adapted to question answering since short answer strings can be found in documents concerning completely different topics. Thus more question-driven strategies are required.

For instance, in the (Moldovan *et al.* 02) system, the passage selection module associates question terms with the set of their morphological alternations, and ranks the passages by estimating the degree of lexical matching between the question and the passages.

In (Hartrumpf 04)'s system, the selected passages are sentences; the sentences of the corpus of documents are transformed into semantic networks, and a semantic network matching the question is searched for.

In our system, we chose to process the answer extraction on sentence-long passages. As semantic reformulations were used by Fastr in the document selection, the sentence selection and ranking has to counterweight the loss of precision stemming from these reformulations. A deep semantic strategy was not chosen: first, it requires knowledge bases such as Extended WordNet constructed by LCC, which can hardly be used in a multilingual context, and robustness is difficultly achieved in a deep semantic system. Our hypothesis is that a syntactic filtering could also improve the passage selection.

3 Syntactic filtering

3.1 Sentence tree reduction

The ranking of the sentences according to a linear distance between the terms of the question presents drawbacks, since this distance does not

[3]The Mean Reciprocal Rank is calculated by inverting the rank of the rank correct answer, and averaging over all questions: with x being the number of questions, $MRR = \frac{1}{x} * \sum_{questions}(\frac{1}{rank(first\_answer)})$

[4]Question Answering Evaluation Exercise, http://clef-qa.itc.it/2004/

consider syntactic aspects. Sentence scores will be deteriorated by the presence of epithets or relatives, although these elements do not alter the meaning of the sentence. An example of this kind of problem is given Figure 2 :

Question: Who was married to Whoopi Goldberg? (Qui était marié avec Whoopi Goldberg ?)
Answer: Actress Whoopi Goldberg married film industry union representative Lyle Trachtenberg during a weekend ceremony at her Pacific Palisades home.

Figure 2: *An example question from the CLEF04 campaign*

Another point justifying the use of a syntactic measure, is that it can favour sentences where the relevant words are closely linked to each other. Sentences where the question's relevant words are not directly linked to each other should be less likely to answer the question.

We aim at creating a measure that takes the two above points into account.

Our solution consists in representing the sentences as syntactic graphs and pruning any phrase that is not useful to link the relevant terms together.

In order to be tested, this measure is inserted in the sentence weighting algorithm of the QALC system.

3.2 Algorithm

The algorithm prunes the syntactic tree [5] of the retrieved sentences so as to build the best subgraph containing the elements of the question, where "best" is defined through a measure combining syntactic and semantic proximity. In this approach, the syntactic structure of the question is not taken into account. The question is considered as a set of criteria, denoted Q.

3.2.1 Mapping between information of the question and the words of the answer.

The paradigmatic criteria for matching a sentence with the question are the following :

- The expected type of the answer.

  For the moment, we only considered questions whose answer's expected type are named or numerical entities. It seems possible to extend the algorithm to WordNet types without modifying the framework of the algorithm.

- The terms of the question

  Terms of the questions can be lemmas, mono-terms or multi-terms, verbs, nouns or noun phrases. They are linked to either mono-terms or multi-terms in the answer. There can be semantic or morphological variations between the terms of the question an those of the answer. Words that appear in both answer and question without variation are referred to as lemmas.

The paradigmatic links are weighted: named entities, Fastr terms and lemmas of the question term are given different weights. For example, named entities have a weight of 2. Fastr term's weight vary according to the reliability of the term: For example, a bi-term should be scored higher than a mono-term.

3.2.2 Selecting the best combination of nodes

For each element of the question, either a term or the expected answer type, we obtain the list of the nodes of the answer's graph that are likely to be paradigmatically linked to this element.

If the element is a term of the question, it may point to a composed term of the answer. In that case it corresponds to more than one node, but will be treated as a simple term that will correspond to the head of the composed term.

In these lists of nodes there is bound to be non-relevant elements. We try to filter the non-relevant nodes by selecting a combination of nodes, keeping only one node for each criterion, so we can denote $weight(c)$ the weight of the paradigmatic link selected for criterion c. We chose the combination that maximizes a weight combining a syntagmatic and a paradigmatic constraint. The weight of a combination is computed as follows :

Syntagmatic part of the weight: We build the minimal subtree containing all the nodes of the combination, as shown in figure 3.

Let *nb_nodes* the total number of nodes of this graph and *nb_criteria* the number of nodes

[5] Syntactic analysis is performed by Charniak's parser : http://www.cs.brown.edu/people/ec/

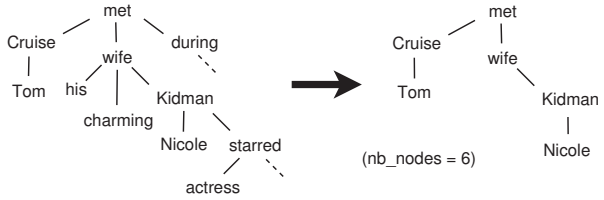


Figure 3: Syntactic tree pruning

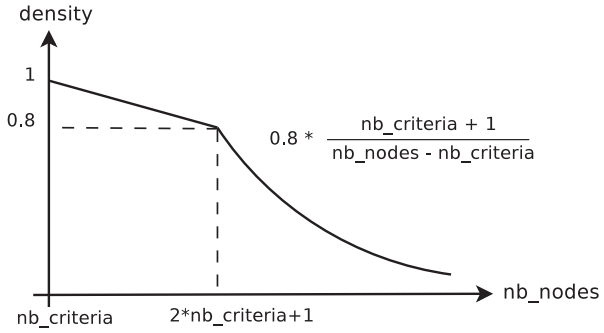


Figure 4: Threshold function for measuring syntactic density

linked to a criterion of the question. Note that $nb\_criteria < nb\_nodes$. We calculate a density which is given by the function shown in figure 4.

The function represents a fuzzy threshold function. The threshold limit is set to $nb\_nodes = 2 * nb\_criteria + 1$, which corresponds to the case of an answer graph where there is always exactly one non-relevant node between two relevant nodes.

Paradigmatic part of the weight: In order to put a disadvantage on the nodes which are not strongly related to the question's element, we take the weights of the paradigmatic relations into account. The final measure is :

$$weight(combination) = density * \sum_{c \in Q} weight(c)$$

3.2.3 Connecting the metric into the QALC architecture

The QALC system performs a sentence ranking which integrates several measures such as answer terms linear distance, Fastr terms weights, named entity weights into a global sentence weight. We connect our syntactic measure by multiplying this global weight by the density we computed. The sum of weights used for determining the best combination is no longer used in this step, for it is redundant with the QALC systems weights.

3.3 Study of our approach on the Clef 04 corpus

We evaluated our strategy over the CLEF04 corpus of questions. It has to be noted that CLEF being a multilingual evaluation, our evaluation on this corpus suffered sometimes from translation term difficulties. Table 1 shows the results of this evaluation.

	NE questions	All questions
Initial MRR	0.310	0.306
New MRR	0.360	0.338

Table 1: MRR with and without syntactic reranking

The most significant figures are those concerning NE questions, since our reranking of sentences is presently restricted to those, but the overall improvement is nevertheless interesting since the MRR also increases significantly, due to the relatively high percentage of NE questions. On NE questions, we improved our MRR by 17%.

To the question "En quelle année le Pape Jean Paul II est -il devenu pontife ?" ("In which year did Pope John Paul II become pontiff?") the following sentence gained 20 positions for example: "(...)but never again will the **election of** a non-Italian **Pope be as** startling as when Cardinal Karol Wojtyla of Krakow was **elected in 1978**." The minimal subgraph containing "Pope" and a DATE named entity can indeed be represented as shown in Figure 5, bold font words are those contained by the pruned structure.

Another example of reranking is given by question "Qui a gagné le Prix Nobel de Littérature en 1994 ?" ("Who won the Nobel Prize for literature in 1994?"), for which the answer "**Derek Walcott**, who **won** the **Nobel Prize** for **litterature**, called (...)" is ranked 7 instead of 18, thanks to the subgraph also shown in Figure 5. Note that the "1994" criterion is absent from the sentence but is present in the retrieved document, thanks to the search engine selection.

3.4 Perspectives

As noticed earlier, the syntactic structure could be taken into account, in order to privilege in the minimal subtree construction, the links between terms which were highly related in the question. Moreover, the strategy could be extended to non-NE

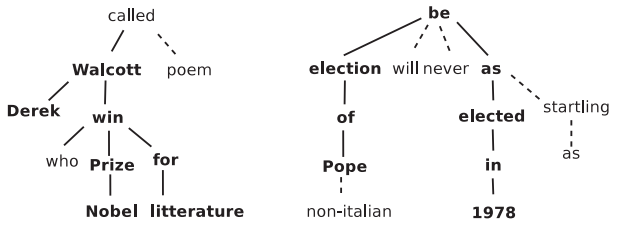


Figure 5: Some retrieved sentences.

questions, in case the semantic type of the answer can be found and verified.

Another improvement would be the use of tree edit distances to approximate syntactic similarity. (Kouleykov & Magnini 05)

This strategy could also benefit from other types of reformulations; WordNet variants could also be considered as question term reformulations. Finally, it would be interesting to test this sentence filtering one step before in our system, and to compare fully and combine the selections based on a linear distance and on a syntactic one.

4 Conclusion

By using a syntactic distance instead of a linear one to select sentences in our question answering system, we improved the ranking of these sentences, and thus our probability to find the correct answers. The type of questions for which this strategy is most relevant could be studied, in order to try and detect to which extent this strategy can replace our previous one.

References

(Ferret *et al.* 02) O. Ferret, B. Grau, M. Hurault-Plantet, G. Illouz, C. Jacquemin, L. Monceaux, I. Robba, and A. Vilnat. How nlp can improve question answering. In *Knowledge Organization Vol. 29, N3-4*, pages 135–155, 2002.

(Hartrumpf 04) S. Hartrumpf. Question answering using sentence parsing and semantic network matching. In *Results of the {CLEF} 2004 Cross-Language System Evaluation Campaign, Working Notes for the {CLEF} 2004 Workshop.*, 2004.

(Jacquemin 99) C. Jacquemin. Syntagmatic and paradigmatic representation of term variation. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL'99)*, pages 341–348, 1999.

(Kouleykov & Magnini 05) M. Kouleykov and B. Magnini. Recognizing textual entailment with tree edit distance algorithms. 2005.

(Moldovan *et al.* 02) D. Moldovan, S. Harabagiu, R. Girju, P. Morarescu, F. Lacatusu, A. Novischi, A. Badulescu, and O. Bolohan. Lcc tools for question answering. In *Proceedings of the 11th Text REtrieval Conference (TREC 2002)*, 2002.

(Tellex *et al.* 03) S. Tellex, B. Katz, J. Lin, A. Fernandes, and G. Marton. Quantitative evaluation of passage retrieval algorithms for question answering. pages 41–47, 2003.

Automatic Building of Wordnets

Eduard Barbu* and **Verginica Barbu Mititelu†**
* Graphitech Italy
2, Salita Dei Molini
38050 Villazzano, Trento, Italy
eduard.barbu@graphitech.it
†Romanian Academy Research Institute for Artificial Intelligence
13 Calea 13 Septembrie,
Bucharest 050711, Romania
vergi@racai.ro

Abstract

In what follows we will present a two-phase methodology for automatically building a wordnet (that we call target wordnet) strictly aligned with an already available wordnet (source wordnet). In the first phase the synsets for the target language are automatically generated and mapped onto the source language synsets using a series of heuristics. In the second phase the salient relations that can be automatically imported are identified and the procedure for their import is explained. The assumptions behind such methodology will be stated, the heuristics employed will be presented and their success evaluated against a case study (automatically building a Romanian wordnet using PWN).

1. Introduction

The importance of a wordnet for NLP applications can hardly be overestimated. The Princeton WordNet (PWN) (Fellbaum 1998) is now a mature lexical ontology which has demonstrated its efficiency in a variety of tasks (word sense disambiguation, machine translation, information retrieval, etc.). Inspired by the success of PWN many languages started to develop their own wordnets taking PWN as a model (cf. http://www.globalwordnet.org/gwa/wordnet_table.htm) . Furthermore, in both EuroWordNet (Vossen 1998) and BalkaNet (Tufiş 2004) projects the synsets from different versions of PWN (1.5 and 2.0) were used as ILI repositories. The created wordnets were linked by means of interlingual relations through this ILI repository[1].

The rapid progress in building a new wordnet and linking it with an already tested wordnet (usually PWN) is hindered by the amount of time and effort needed for developing such a resource. To take a recent example, the development of core wordnets (of about 20000 synsets, as is the case with the Romanian wordnet) for Balkan languages took three years (2001-2004).

[1] In both projects there was a number of synsets expressing language specific concepts added to the ILI repository.

In what follows we present a methodology that can be used for automatically building wordnets strictly aligned (that is, using only EQ_SYNONYM relation) with an already available wordnet. We have started our experiment with the study of nouns, so the data presented here are valid only for this grammatical category.

We call the wordnet already available Source wordnet (as mentioned before, this is usually a version of PWN) and the wordnet to be built and linked with the Source wordnet will be named Target wordnet.

The methodology we present has two phases. In the first one the synsets for the target language are automatically generated and mapped onto the source language synsets using a series of heuristics. In the second phase the salient relations that can be automatically imported are identified and the procedure for their import is explained.

The paper has the following organization. Firstly we state the implicit assumptions in building a wordnet strictly aligned with other wordnets. Then we shortly describe the resources that one needs in order to apply the heuristics, and also the criteria we used in selecting the source language test synsets to be implemented. Finally, we state the problem to be solved in a more formal way, the heuristics employed will be presented and their success evaluated against a case study (automatically building a Romanian wordnet using PWN 2.0).

2. Assumptions

The assumptions that we considered necessary for automatically building a target wordnet using a Source wordnet are the following:

1. There are word senses that can be clearly identified. This assumption is implicit when one builds a wordnet aligned or not with other wordnets. This premise was extensively questioned among others by (Kilgarriff 1997) who thinks that word senses have not a real ontological status, but they exist only relative to a task. We will not discuss this issue here.
2. A rejection of the strong reading of Sapir-Whorf (Caroll 1964) hypothesis (the principle of linguistic relativity). Simply stated, the principle of linguistic relativity says that

language shapes our thought. There are two variants of this principle: strong determinism and weak determinism. According to the strong determinism language and thought are identical. This hypothesis has today few followers if any and the evidence against it comes from various sources among which the possibility of translation in other language. However, the weak version of the hypothesis is largely accepted. One can view the reality and our organization of reality by analogy with the spectrum of colors which is a continuum in which we place arbitrary boundaries (white, green, black, etc.). Different languages will "cut" differently this continuous spectrum. For example, Russian and Spanish have no words for the concept blue. This weak version of the principle of linguistic relativity warns us, however, that a specific source wordnet could not be used for automatically building any target wordnet. We further discuss this bellow.

3. The acceptance of the conceptualization made by the source wordnet By conceptualization we understand the way in which the source Wordnet "sees" the reality by identifying the main concepts to be expressed and their relationships. For specifying how different languages can differ with respect with to conceptual space they reflect we will follow (Sowa 1992) who considers three distinct dimensions:

- *accidental*. The two languages have different notations for the same concepts. For example the Romanian word *măr* and the English word *apple* lexicalize the same concept.
- *systematic*. The systematic dimension defines the relation between the grammar of a language and its conceptual structures. It deals with the fact that some languages are SVO or VSO, etc., some are analytic and other agglutinative. Even if it is an important difference between languages, the systematic dimension has little import for our problem
- *cultural*. The conceptual space expressed by a language is determined by environmental, cultural factors, etc. It could be the case for example, that concepts that define the legal systems of different countries are not mutually compatible. So when someone builds a wordnet starting from a source wordnet he/she should ask himself/herself what the parts (if any) that could be safely transferred in the target language are. More precise what the parts that share the same conceptual space are.

The assumption that we make use of is that the differences between the two languages (source and target) are merely accidental: they have different lexicalizations for the same concepts. As the conceptual space is already expressed by the Source wordnet structure using a language notation, our task is to find the concepts notations in the target language.

When the Source wordnet is not perfect (the real situation), then a drawback of the automatic mapping approach is that all the mistakes existent in the source wordnet are transferred in the target wordnet: consider the following senses of the noun *hindrance* in PWN:

1. hindrance, deterrent, impediment, balk, baulk, check, handicap -- (something immaterial that interferes with or delays action or progress)

=> cognitive factor -- (something immaterial (as a circumstance or influence) that contributes to producing a result)

2. hindrance, hitch, preventive, preventative, encumbrance, incumbrance, interference -- (any obstruction that impedes or is burdensome)

=> artifact, artefact -- (a man-made object taken as a whole).

We listed the senses 1 and 2 of the word *hindrance* together with one of their hyperonyms. As one can see, in PWN a distinction is made between *hindrance* as a cognitive factor and *hindrance* as an artefact, so that these are two different senses of the word *hindrance*. According to this definition a speed bump can be classified as a hindrance because it is an artifact, but a stone that stays in the path of someone cannot be one, because it is not a man made object.

Another possible problem appears because the previously made assumption about the sameness of the conceptual space is not always true as the following example shows:

mister, Mr -- (a form of address for a man)

sir -- (term of address for a man)

In Romanian both *mister* and *sir* in the listed senses are translated by the word *domn*. But in Romanian it would be artificial to create two distinct synsets for the word *domn*, as they are not different, not even in what their connotations are concerned.

3. Selection of concepts and resources used

When we selected the set of synsets to be implemented in Romanian we followed two criteria.

The first criterion states that the selected set should be structured in the source wordnet (i.e. every selected synset should be linked by at least one semantic relation with other selected synsets). This is dictated by the methodology we have adopted (automatic mapping and automatic relation import). If we want to obtain a wordnet in the target language and not just some isolated synsets, this criterion is self-imposing.

The second criterion is related to the evaluation stage. To properly evaluate the built wordnet, it should be compared with a "golden standard". The golden standard that we use will be the Romanian Wordnet (RoWN) developed in the BalkaNet project[2].

[2] One can argue that this Romanian wordnet is not perfect and definitely incomplete. However, PWN is neither perfect. Moreover, it is indisputable that at least in the case of ontologies (lexical or

For fulfilling both criteria we chose a subset of noun concepts from the RoWN that has the property that its projection on PWN 2.0 is closed under the hyperonym and the meronym relations. Moreover, this subset includes the upper level part of the PWN lexical ontology. The projection of this subset on PWN 2.0 comprises 9716 synsets that contain 19624 literals.

For the purpose of automatic mapping of this subset we used an in-house dictionary built from many sources. The dictionary has two main components:

- The first component consists of the above-mentioned 19624 literals and their Romanian translations. We must make sure that this part of the dictionary is as complete as possible. Ideally, all senses of the English words should be translated. For that we used the (Levițchi & Bantaş 1992) dictionary and other dictionaries available on web.
- The second component of the dictionary is (Levițchi & Bantaş 1992) dictionary.

Some dictionaries (in our case the dictionary extracted from the already available Romanian wordnet) also have sense numbers specified, but, from our experience, this information is highly subjective, does not match the sense as defined by PWN and it is not consistent over different dictionaries, so we chose to disregard it.

The second resource used is the Romanian Explanatory Dictionary (EXPD 1996) whose entries are numbered to reflect the dependencies between different senses of the same word.

4. Notation introduction

In this section we introduce the notations used in the paper and we outline the guiding idea of all heuristics we used:

1. By $\mathbf{T_L}$ we denote the target lexicon. In our experiment T_L will contain Romanian words (nouns). $T_L = \{ rw_1, rw_2, \ldots rw_m \}$ where rw_i, with i=1..m, denotes a target word.
2. By $\mathbf{S_L}$ we denote the source lexicon. In our case S_L will contain English words (nouns). $S_L = \{ ew_1, ew_2, \ldots ew_n \}$ where ew_j, with j=1..n, denotes a source word.
3. $\mathbf{W_T}$ and $\mathbf{W_S}$ are the wordnets for the target language and the source language, respectively.
4. w_j^k denotes the k^{th} sense of the word w_j.
5. $\mathbf{B_D}$ is a bilingual dictionary which acts as a bridge between $\mathbf{S_L}$ and $\mathbf{T_L}$. $B_D = (\mathbf{S_L}, \mathbf{T_L}, M)$ is a 3-tuple, where M is a function that associates to each word in $\mathbf{S_L}$ a set of words in $\mathbf{T_L}$. For an arbitrary word $ew_j \in S_L$, $M(ew_j) = \{ rw_1, rw_2, \ldots rw_k \}$.

Formally the bilingual dictionary maps words and not word senses. If word senses had been mapped, then building W_T from a W_S would have been trivial.

If we ignore the information given by the definitions associated with word senses, then, formally a sense of a word in the PWN is distinguished from other word senses only by the set of relation it contacts in the semantic network. This set of relations defines what it is called the position of a word in the semantic network. Ideally, every sense of a word should be unambiguously identified by a set of connections; it should have a unique position in the semantic net. Unfortunately this is not the case in PWN. There are many cases when different senses of a word have the same position in the semantic network (i.e they have precisely the same connections with other word senses).

The idea of our heuristics could be summed up in three points:

1. Increase the number of relations in the Source wordnet to obtain a unique position for each word sense. For this an external resource can be used to which the wordnet is linked, such as Wordnet Domains.
2. Try to derive useful relations between the words in the target language. For this one can use corpuses, monolingual dictionaries, already classified set of documents etc.
3. In the mapping stage of the procedure take profit of the structures built at points 1 and 2.

We have developed so far a set of four heuristics and we plan to supplement them in the future.

5. The first heuristic rule

The first heuristic exploits the fact that synonymy enforces equivalence classes on word senses.

Let EnSyn=$\{ ew_{j_{11}}^{i_{11}}, ew_{j_{12}}^{i_{12}} \ldots ew_{j_{1n}}^{i_{1n}} \}$ where $ew_{j_{11}}$, $ew_{j_{12}}$, $ew_{j_{1n}}$ are the words in synset and the superscripts denote their sense numbers) be a S_L synset and length(EnSyn)>1. We impose the length of a synset to be greater than one when at least one component word is not a variant of the other words. So we disregard synsets such as {artefact, artifact}. For achieving this we computed the well known Levenshtein distance between the words in the synset.

The B_D translations of the words in the synset will be:

$M(ew_{j_{11}}) = \{ rw_{i_{11}}, \ldots rw_{i_{1m}} \}$

$M(ew_{j_{12}}) = \{ rw_{i_{21}}, \ldots rw_{i_{2k}} \}$

….

$M(ew_{j_{1n}}) = \{ rw_{i_{n1}}, \ldots rw_{i_{nt}} \}$

We build the corresponding $\mathbf{T_L}$ synset as

1. $M(ew_{j_{ik}})$ if $\exists\ ew_{j_{ik}} \in$ EnSyn such that the number of senses NoSenses $(ew_{j_{ik}})=1$

formal), a manually or semi-automatically built ontology is much better than an automatically built one.

2. $M(ew_{j_{11}}) \cap M(ew_{j_{12}}) \ldots \cap M(ew_{j_{1n}})$ otherwise

Words belonging to the same synset in S_L should have a common translation in $\mathbf{T_L}$. Above we distinguished two cases:

1. At least one of the words in a synset is monosemous. In this case we build the $\mathbf{T_L}$ synset as the set of translations of the monosemous word.

2. All words in the synset are polysemous. The corresponding $\mathbf{T_L}$ synset will be constructed by the intersection of all $\mathbf{T_L}$ translations of the S_L words in the synset.

Taking the actual RoWNas a gold standard we can evaluate the results of our heuristics by comparing the obtained synsets with those in the RoWN. We distinguish five possible cases:

1. The synsets are equal (this case will be labeled as Identical).
2. The generated synset has all literals of the correct synset and some more. (Over-generation).
3. The generated synset and the golden one have some literals in common and some different (Overlap)
4. The generated synset literals form a proper subset of the golden synset (Under-generation)
5. The generated synset have no literals in common with the correct one (Disjoint).

The cases Over-generation, Overlap and Disjoint will be counted as errors. The other two cases, namely Identical and Under-generation, will be counted as successes[3].

The evaluation of the first heuristics is given in Table 1, at the end of section 9.

The percents mapped column contains the percents of the synsets mapped by the heuristics from the total number of the synsets (9716). The percent errors column represents the percent of synsets from the number of mapped synsets wrongly assigned by the heuristics. The high number of mapped synsets proves the quality of the first part of the dictionary we used. The only type of error we encountered is Over-generation.

6. The second heuristic rule

The second heuristic draws from the fact that, in the case of nouns, the hyperonymy relation can be interpreted as an IS-A relation[4]. It is also based on two related observations:

1. A hyperonym and his hyponyms carry some common information.
2. The information common to the hyperonym and the hyponym will increase as you go down in the hierarchy.

Let $EnSyn_1 = \{ ew_{j_{11}}^{i_{11}}, ew_{j_{12}}^{i_{12}} \ldots ew_{j_{1t}}^{i_{1t}} \}$ and $EnSyn_2 = \{ ew_{j_{21}}^{i_{21}}, ew_{j_{22}}^{i_{22}} \ldots ew_{j_{2s}}^{i_{2s}} \}$ be two S_L synsets such that $EnSyn_1$ HYP $EnSyn_2$, meaning that $EnSyn_1$ is a hyperonym of $EnSyn_2$. Then we generate the translation lists of the words in the synsets. The intersection is computed as:

$T_L\ EnSyn_1 = M(ew_{j_{11}}) \cap M(ew_{j_{12}}) \ldots \cap M(ew_{j_{1t}})$

$T_L\ EnSyn_2 = M(ew_{j_{21}}) \cap M(ew_{j_{22}}) \ldots \cap M(ew_{j_{2s}})$

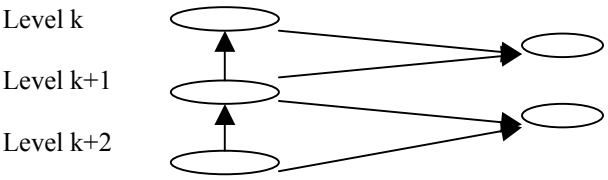
The generated synset in the target language will be computed as

$T_L\ Synset = T_L\ EnSyn_1 \cap T_L\ EnSyn_2$

Given the above consideration, it is possible that a hyponym and its hyperonym have the same translation in the other language and this is more probable as you descend in the hierarchy. The procedure formally described above is applied for each synset in the source list. It generates the lists of common translations for all words in the hyperonym and hyponym synsets and then constructs the $\mathbf{T_L}$ synsets by intersecting these lists. In case the intersection is not empty the created synset will be assigned to both S_L language synsets.

Because the procedure generates autohyponym synsets this could be an indication that created synsets could be clustered in $\mathbf{T_L}$.

It is possible that a $\mathbf{T_L}$synset be assigned to two different source pair synsets as in the figure below. So we need to perform a clean-up procedure and choose the assignment that maximizes the sum of depth level of the two synsets.



In the figure common information is found between the middle synset and the upper synset (its hyperonym) and also between the middle synset and the lower synset (its hyponym). Our procedure will prefer the second assignment.

The results of the second heuristic are presented in Table 2, at the end of section 9. The low number of mapped synsets (10%) is due to the fact that we did not find many common translations between hyperonyms and their hyponyms.

7. The third heuristic

The third heuristics takes profit of an external relation imposed over the wordnet. At IRST PWN 1.6 was augmented with a set of Domain Labels, the resulting resource being called **Wordnet Domains** (Magnini & Cavaglia 2000). PWN 1.6 synsets have been semi-automatically linked with a set of 200 domain labels taken from Dewey Decimal classification, the world most widely used library classification system. The domain labels are hierarchically organized and each synset received one or more domain labels. For the synsets that cannot be labeled unambiguously the default label "factotum" has been used.

Because in the BalkaNet project the RoWN has been aligned with PWN 2.0, we performed a mapping

[3] The Under-generation case means that the resulted synset is not reach enough; it does not mean that it is incorrect.

[4] This not entirely true because in PWN the hyperonym relation can also be interpreted as an INSTANCE-OF relation, as in PWN there are also some instances included (e.g. *New York*, *Adam*, etc.).

between PWN 1.6 and PWN 2.0. By comparison with PWN1.6, PWN 2.0 has new additional synsets and also the wordnet structure is slightly modified. As a consequence not all PWN 2.0 synsets can be reached from PWN 1.6 either because they are completely new or because they could not be unambiguously mapped. This results in some PWN 2.0 synsets that have not domains. For their labelling we used the three rules below:

1. If one of the direct hyperonym of the unlabeled synsets has been assigned a domain, then the synset will automatically receive the father's domain, and conversely, if one of the hyponyms is labelled with a domain and father lacks domain, then the father synset will receive the son's domain.
2. If a holonym of the synset is assigned a specific domain, then the meronym will receive the holonym domain and conversely.
3. If a domain label cannot be assigned, then the synset will receive the default "factotum" label.

The idea of using domains is helpful for distinguishing word senses (different word senses of a word are assigned to different domains). The best case is when each sense of a word has been assigned to a distinct domain. But even if the same domain labels are assigned to two or more senses of a word, in most cases we can assume that this is a strong indication of a fine-grained distinction. It is very probable that the distinction is preserved in the target language by the same word.

We labelled every word in the B_D dictionary with its domain label. For English words the domain is automatically generated from the English synset labels. For labelling Romanian words we used two methods:

1. We downloaded a collection of documents from web directories such that the categories of the downloaded documents match the categories used in the Wordnet Domain. The downloaded document set underwent a pre-processing procedure with the following steps:

a. Feature extraction. The first phase consists in finding a set of terms that represents the documents adequately. The documents were POS tagged and lemmatized and the nouns were selected as features.

b. Features selection. In this phase the features that provide less information were eliminated. For this we used the well known χ^2 statistic. χ^2 statistic checks if there is a relationship between being in a certain group and a characteristic that we want to study. In our case we want to measure the dependency between a term t and a category c. The formula for χ^2 is:

$$\chi^2(t,c) = \frac{N \times (AD - CB)^2}{(A+C) \times (B+D) \times (A+B) \times (C+D)}$$

Where:

- A is the number of times t and c co-occur
- B is the number of times t occur without c
- C is the number of times c occurs without t
- D is the number of times neither c nor t occurs
- N is the total number of documents

For each category we computed the score between that category and the noun terms of our documents. Then, for choosing the terms that discriminate well for a certain category we used the formula below (where m denotes the number of categories):

$$\chi^2_{\max}(t) = \max_{i=1}^{m}(\chi^2(t,c_i))$$

2. We took advantage of the fact that some words have already been assigned subject codes in various dictionaries. We performed a manual mapping of these codes onto the Domain Labels used at IRST. The Romanian words that could not be associated domain information were associated with the default factotum domain.

The following entry is a B_D dictionary entry augmented with domain information:

M(ew_1 [$D_{1,...}$]) = rw_1 [D_1, D_2 …], rw_2 [D_1, D_3 …], rw_i [D_2, D_4 …]

In the square brackets the domains that pertain to each word are listed.

Let again $EnSyn_1$={ $ew_{j_{11}}^{i_{11}}$, $ew_{j_{12}}^{i_{12}}$ … $ew_{j_{1n}}^{i_{1n}}$ } be an S_L language synset and D_i the associated domain. Then the $\mathbf{T_L}$ synset will be constructed as follows:

$\mathbf{T_L}$ Synset = $\bigcup_{m=j_{11}..j_{1n}}$ M(ew_m), where each $rw_i \in$ M(ew_m) has the property that its domain "matches" the domain of $EnSyn_1$ that is: either is the same as the domain of $EnSyn_1$, subsumes the domain of $EnSyn_1$ in the IRST domain labels hierarchy or is subsumed by the domain of $EnSyn_1$ in the IRST domain labels hierarchy .

For each synset in the S_L we generated all the translations of its literals in the $\mathbf{T_L}$. Then the $\mathbf{T_L}$ synset is built using only those $\mathbf{T_L}$ literals whose domain "matches" the S_L synset domain.

The results of this heuristic are given in Table 3 at the end of section 9.

8. The fourth heuristic rule

The fourth heuristics takes advantage of the fact that the source synsets have a gloss associated and also that target words that are translations of source words have associated glosses in EXPD. As with the third heuristic the procedure comprises a preprocessing phase. We preprocessed both resources (PWN and EXPD):

1. We automatically lemmatized and tagged all the glosses of the synsets in the S_L.
2. We automatically lemmatized and tagged all the definitions of the words that are translations of S_L words.

3. We chose as features for representing the glosses the set of nouns.

The target definitions were automatically translated using the bilingual dictionary. All possible source definitions were generated by translating each lemmatized noun word in the $\mathbf{T_L}$ definition. Thus, if a $\mathbf{T_L}$ definition of one $\mathbf{T_L}$ word is represented by the following vector $[rw_1, rw_2, \ldots rw_p]$, then the number of S_L vectors generated will be: $N= n_d * t_{w_1} * t_{w_2} * \ldots * t_{w_p}$, where n_d is the number of definitions the target word has in the monolingual dictionary (EXPD), and t_{w_k}, with k=1..p, is the number of translations that the noun w_k has in the bilingual dictionary.

By R_{Gloss} we denote the set of S_L representation vectors of $\mathbf{T_L}$ glosses of a $\mathbf{T_L}$ word: $R_{Gloss} = \{T_1, T_2 \ldots T_n\}$. By S_v we denote the vector of S_L synset gloss.

The procedure for generating the $\mathbf{T_L}$ synset is: for each S_L synset we generate the $\mathbf{T_L}$ list of the translation of all words in the synset. Then for each word in the $\mathbf{T_L}$ list of translation we compute the similarity between S_v and its R_{Gloss}. The computation is done in two steps:

1. We give the vectors in S_v and R_{Gloss} a binary representation. The number (m) of positions a vector has will be equal to the number of distinct words existent in the S_v and in all vectors of R_{Gloss}. The presence of 1 in the vector means that a word is present and the existence of 0 means that a word is absent from the vector.
2. For each T_i vector in R_{Gloss} we compute the product: $S_v \bullet T_i = \sum_{j=1..m} s_j * t_j$. If there exists at least one T_i such that $S_v \bullet T_i >= 2$ we compute $\max(S_v \bullet T_i)$ and we add the word to the $\mathbf{T_L}$ synset.

Notice that by using this heuristic rule we can automatically add a gloss to the $\mathbf{T_L}$ synset.

As one can see in Table 4 at the end of section 9 the number of incomplete synsets is high. The percent of mapped synsets is due to the low agreement between the glosses in Romanian and English.

9. Combining results

For choosing the final synsets we devised a set of meta-rules by evaluating the pro and con of each heuristic rule. For example, given the high quality dictionary the probability that the first heuristic will fail is very low. So the synsets obtained using it will automatically be selected. A synset obtained using the other heuristics will be selected and moreover will replace a synset obtained using the first heuristic, only if it is obtained independently using the heuristics 3 and 2, or by using the heuristics 3 and 4. If a synset is not selected by the above meta-rules will be selected only if it is obtained by the heuristics number 3 and the ambiguity of its members is at most equal to 2. Table 5 at the end of this section shows the combined results of our heuristics.

As one can observe there, for 106 synsets in PWN 2.0 the Romanian equivalent synsets could not be found. There also resulted 635 synsets smaller than the synsets in the RoWN.

Number of mapped synsets	Percents mapped	Error types			Correct		Percent errors
		Over-generation	Overlap	Disjoint	Under-generation	Identical	
8493	87	210	0	0	300	7983	2

Table 1: The results of the first heuristic

Number of mapped synsets	Percents mapped	Error types			Correct		Percent errors
		Over-generation	Overlap	Disjoint	Under-generation	Identical	
1028	10	213	0	150	230	435	35

Table 2: The results of the second heuristic

Number of mapped synsets	Percents mapped	Error types			Correct		Percent errors
		Over-generation	Overlap	Disjoint	Under-generation	Identical	
7520	77	689	0	0	0	6831	9

Table 3: The results of the third heuristic

Number of mapped synsets	Percents mapped	Error types			Correct		Percent errors
		Over-generation	Overlap	Disjoint	Under-generation	Identical	
3527	36	25	0	78	547	2877	3

Table 4: The results of the fourth heuristic

Number of mapped synsets	Percents mapped	Error types			Correct		Percent errors
		Over-generation	Overlap	Disjoint	Under-generation	Identical	
9610	98	615	0	250	635	8110	9

Table 5: The combined results of the heuristics

10. Import of relations

After building the target synsets an investigation of the nature of the relations that structure the source wordnet should be made for establishing which of them can be safely transferred in target wordnet. As one expects the conceptual relations can be safely transferred because these relations hold between concepts. The only lexical relation that holds between nouns and that was subject to scrutiny was the antonym relation. We concluded that this relation can also be safely imported. The importing algorithm works as described bellow.

If two source synsets S_1 and S_2 are linked by a semantic relation R in W_S and if T_1 and T_2 are the corresponding aligned synsets in the W_T, then they will be linked by the relation R. If in W_S there are intervening synsets between S_1 and S_2, then we will set the relation R between the corresponding $\mathbf{T_L}$ synsets only if R is declared as transitive (R+, unlimited number of compositions, e.g. hypernym) or partially transitive relation (R*k* with k a user-specialized maximum number of compositions, larger than the number of intervening synsets between S_1 and S_2). For instance, we defined all the holonymy relations as partially transitive (k=3).

11. Conclusion and future work

Other experiments of automatically building wordnets that we are aware of are (Atserias et al., 1997) and (Lee et al., 2000). They combine several methods, using monolingual and bilingual dictionaries for obtaining a Spanish Wordnet and, respectively, a Korean one starting from PWN 1.5.

However, our approach is characterized by the fact that it gives an accurate evaluation of the results by automatically comparing them with a manually built wordnet. We also explicitly state the assumptions of this automatic approach. Our approach is the first to use an external resource (Wordnet Domains) in the process of automatically building a wordnet.

We obtained a version of RoWN that contains 9610 synsets and 11969 relations with 91% accuracy.

The results obtained encourage us to develop other heuristics. The success of our procedure was facilitated by the quality of the bilingual dictionary we used.

Some heuristics developed here may be applied for the automatic construction of synsets of other parts of speech. That is why we also plan to extend our experiment to adjectives and verbs. Their evaluation would be of great interest in our opinion.

Finally we would like to thank the three anonymous reviewers for helping us in improving the final version of the paper.

References:

(Atserias et al. 1997) J. Atserias, S. Clement, X. Farreres, German Rigau, H. Rodríguez, Combining Multiple Methods for the Automatic Construction of Multilingual

WordNets. In *Proceedings of the International Conference on Recent Advances in Natural Language*, 1997.

(Carroll 1964) J. B. Carroll (Ed.). 1964. Language, Thought and Reality Selected writings of Benjamin Lee Whorf, The MIT Press, Cambridge, MA.

(EXPD 1996) *Dicţionarul explicativ al limbii române*, 2nd edition, Bucureşti, Univers Enciclopedic, 1996.

(Fellbaum 1998) Ch. Fellbaum (Ed.) *WordNet: An Electronical Lexical Database*, MIT Press, 1998.

(Kilgarriff 1997) A. Kilgarriff, *I don't believe in word senses*. In Computers and the Humanities, 31 (2), 91-113, 1997.

(Lee et al. 2000) C. Lee, G. Lee, J. Seo, Automatic WordNet mapping using Word Sense Disambiguation. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC 2000)*, Hong Kong, 2000.

(Levițchi & Bantaş 1992) L. Levițchi and A. Bantaş, *Dicţionar englez-român*, Bucureşti, Teora, 1992.

(Magnini & Cavaglia 2000) B. Magnini and G. Cavaglia, Integrating subject field codes into WordNet. In *Proceedings of LREC-2000*, Athens, Greece.

(Sowa 1992) J. F. Sowa, Logical Structure in the Lexicon. In J. Pustejovsky and S. Bergler (Eds.) *Lexical Semantics and Commonsense Reasoning*, LNAI 627, Springer-verlag, Berlin, 39-60, 1992.

(Tufiş 2004) D. Tufiş (Ed.), Special Issue on the BalkaNet Project of *Romanian Journal of Information Science and Technology*, vol. 7, no. 1-2, 2004.

(Vossen 1998) P. Vossen, *A Multilingual Database with Lexical Semantic Networks,* Dordrecht, Kluwer, 1998.

Composite Topics in Discourse

Frédérik Bilhaut
GREYC, UMR 6072, Université de Caen
Bd du Maréchal Juin, BP 5186, Caen Cedex
fbilhaut@info.unicaen.fr

Abstract

This paper focuses on the issue of automatic analysis of *discourse aboutness* with a view to information retrieval tasks. We introduce a functional model of textual themes funding on the notion of *composite topic*. We apply it to various discursive configurations and illustrate the fact that domain-specific knowledge may play a significant role in the structure of the related discourse, relying on the concept of *semantic axis*. We finally describe an automatic analysis method based on this model.

1 Introduction

The notion of *aboutness* can be seen as an interesting convergence area for linguistics and information retrieval (IR). In the latter case, this term usually refers to the relation that holds between a *document considered as a whole*, and a set of *terms*, also called *descriptors* or *keywords*. Although this approach lead to indisputably useful applications, it also suffers from important weaknesses: a set of terms is in itself a unsatisfactory way of representing the informational content of a document, and the issue of describing the distribution of this content with respect to finer-grained textual units is not addressed. In order to circumvent these limits, a current tendency in natural language processing (NLP) is to substitute the notions of *theme* or *topic* for the notion of descriptor, relying on *linguistic models* instead of statistical ones, and studying the *discourse structure* rather than the mere distribution of words.

On the other hand, the linguistic literature most usually comprehends the notion of aboutness at the *sentence level* with respect to its *informational structure*. In this case, the term denotes a pragmatic relation that holds between *a clause* and *the referent of a topic expression*: "the topic of a sentence is the thing which the proposition expressed by the sentence is about" (Lambrecht 1994). When considered above the sentence level, the linguistic definition of aboutness remains an open question: although the notion of *discourse topic* has been widely discussed, many contributions limit their attention to one particular level, above or below sentences, and few authors explicitly claim that sentence and discourse topics could be treated in a unified way, one notable exception being (Dik 1989). Numerous factors can explain the complexity of shifting from the sentence level to the discourse level. First, it is obviously admitted that a text is more than the sum of its individual sentences, which implies that a discourse topic can not be immediately inferred from the topics of its sentences. This lead some authors to claim that the global aboutness of a discourse should rather be made explicit in terms of *semantic macro-structures* (van Dijk 1977), which raises in turn the problem of objectifying the complex interpretative operations from which these structures result. Another issue arises from *discourse-level framing*: the scope of potentially topical referents introduced in *headings* or *discourse frame introducers* as defined in (Charolles 1997) can span over several sentences without being explicitly propagated through referential chains. As such, these entities may not fit in sentence-centred frameworks that heavily rely on referential mechanisms. One more degree of complexity is related to the fact that a *general-purpose*, *unified* notion of discourse topic may not be accessible or even desirable. On that matter we would follow (Asher 2004) who argues that "different [discourse] coherence relations make different demands on what topics should do".

Our approach is indeed targeted at one particular form of discourse coherence, based on the positioning of topical discourse referents relatively to other referents that play a specific role in the organisation of the knowledge of a domain. We formalise topics as simple semantic structures that reflect the relations between these referents, and that prove to be appropriate to represent the aboutness of various textual configurations. After introducing the notion of "composite topic" and considering how it relates to some classical, sentence-centred concepts,

we will apply it to various discursive configurations. Finally, we describe an automatic discourse analysis system that has been developed funding on the composite topics model.

2 Composite Topics

The approach of topic presented here aims at describing the aboutness of a variety of textual configurations where topical referents are explicitly *situated* in a given *setting* within a particular knowledge field. In other words, we question the discourse-level applicability of the concept of *framework* defined in (Chafe 1976) as "limit[ting] the applicability of the main predication to a certain restricted domain". For instance, although an acceptable characterisation of the aboutness of the excerpt reproduced below could be "la secondarisation des effectifs scolaires", the specificities of the geographical information lead us to consider as well the adverbials that "situate" this phenomenon in space and time. Thus, a more detailed characterisation of the informational content of the segment marked **{**...**}**$_S$ could be made up of a triad like ("la secondarisation des effectifs scolaires" ; "dans la France du Nord" ; "des années 60 à la fin des années 80"), while S_0 could be represented by ("la secondarisation..." ; "des années 60 à la fin des années 80").

① **{** Entre les années 1960 et la fin des années 1980, le nombre de collégiens et lycéens pour 100 élèves du primaire est passé de 45 à plus de 80, par le double effet de [...]. **{**Mais cette secondarisation a été fort inégale. Elle est forte dans la France du Nord [...] : les effectifs du secondaire y ont fréquemment augmenté de plus des trois quarts en vingt ans, et le rapport secondaire/primaire y a souvent plus que doublé**}**$_S$ Dans la France du Centre et du Sud-Ouest, [...] **}**$_{S0}$

More generally, our concern is to characterise the information carried by such segments using tuples named *composite topics* (CT), that are made up of:

- a *topical core* which is a referent in relation of aboutness with the segment taken as a whole;
- a set of *topical satellites* that "locate" the core in a particular conceptual field[1].

The resulting structure will be written $\tau \multimap (s_1, \ldots, s_n)$, where τ stands for the core and each s_i for a satellite. Following (Lambrecht 1994), we clearly distinguish between *topic referents* and *topic expressions* that refer to them, and we consider the constituents of a composite topic to be *referents*[2]. However, we do not make further assumptions about their actual nature, our model being independent of any particular semantic representation. Moreover, a composite topic has to be conceived as peculiar to a given segment with respect to another segment in which it is embodied. This point is of importance since, as we will see further on, the composite topic of a segment considered in isolation (that is, with respect to itself) can differ sensibly from the topic of the same segment with respect to a higher level segment. The composite topic of a segment A with respect to a segment B will be written $\mathfrak{I}(A,B)$. In the case of the previous excerpt, we would write:

$$\mathfrak{I}(S,S_0) = \text{La secondarisation des effectifs} \multimap (\text{dans la France du nord}, [1969;1990])$$

At the discourse level, composite topics will often be realised by hierarchical structures. Although the above notation is able to represent the leafs of the resulting trees, we have to introduce a notation applicable to higher-level segments. Let us take the following excerpt as an example:

L'explosion des effectifs scolaires
§ **{ {**Dans l'enseignement public, elle s'accélère en Île-de-France, en Picardie, dans le Centre, ainsi qu'en Provence ; elle reste modérée dans l'Ouest et le Nord. [...] **}**$_{S1}$ **{**L'enseignement privé enregistre des baisses d'effectifs en Bretagne, où il est fortement implanté, ainsi que dans les académies de la diagonale Pyrénées-Lorraine, où son audience est par contre traditionnellement réduite [...] **}**$_{S2}$ **}**$_{S0}$

In this case, since the composite topics of S_1 and S_2 can be represented by effectifs scolaires $\multimap$ (public) and effectifs scolaires $\multimap$ (privé), we will represent the topic of the encompassing segment S_0 using the following notation:

$$\mathfrak{I}(S_0,S_0) = \text{Effectifs scolaires} \multimap (\langle\langle \text{Statut} \rangle\rangle)$$

This notation allows us to specify that the topical core of S_0 is described in relation to several entities of a given semantic class named "statut" (in this case, "privé" and "public"). In the same manner, the composite topic of S_1 could be specified as follows:

$$\mathfrak{I}(S_1,S_0) = \text{Effectifs scolaires} \multimap (\text{Public}, \langle\langle \text{Spatial} \rangle\rangle)$$

As argued in (Bilhaut & Enjalbert 2005) where they are called *semantic axes*, such semantic classes seem to play a significant role in the cohesion of some

1 Although we make use of similar terms, we do not claim any connection with the rhetorical structure theory (RST).

2 These referents will be written in small capitals, excepted temporal ones which will be represented by time intervals.

particular discursive configurations, as will be observed further on.

3 Composite Topics and Informational Structure

It is a noteworthy fact that similar concepts have been broadly described at the sentence level. Even if the *multiple themes* of (Halliday & Hasan 1976) have to be mentioned here, we will take a particular interest in concepts formulated in (Chafe 1976), (Dik 1989) and (Lambrecht 1994). Chafe describes the *subject* as the "hitching post for the new knowledge", as opposed to *"Chinese-style" topics* (now CST) that set a "spatial, temporal or individual framework within which the main predication holds". A similar distinction is drawn by Dik between *topic* and *theme* (respectively), depending on whether the constituent takes part in the main predication, when Lambrecht uses the terms of *topic* and *scene-setting topic* (now SST). For example, in the following sentence, the topic (or subject for Chafe) would be P_2, while P_1 would be a SST (or theme for Dik and CST for Chafe):

> Dans l'Ouest$_{(P1)}$, le taux de retard scolaire$_{(P2)}$ est en régression depuis une dizaine d'années$_{(P3)}$.

When considering sentences in isolation, we will generally consider topical cores and satellites as equivalent to topics and SST[3]. In order to apply our model at the discourse level, we formulate the hypothesis that *functionally equivalent* structures may be realised by higher level units, such as paragraphs or sentence groups. However, we claim that the status (core or satellite) of a given constituent at the discourse level can not be immediately deduced from its status at the sentence level. In particular, we will observe in excerpt 3 that a constituent can act as a topical core with respect to its hosting sentence, while acting as a satellite with respect to a wider discourse unit.

4 Discourse-Level Manifestations Of Composite Topics

A typical example of discourse-level composite topic involves discourse frames as described in (Charolles 1997):

> ② **[{ {** Dans l'enseignement primaire$_{P1}$, on assiste à une forte diminution du taux de retard scolaire dans les années 80.**}**$_{U1}$ Cette baisse est en partie attribuable à la réduction du nombre d'élèves par classe, qui [...]**}**$_{S1}$ **{**Dans le secondaire$_{P2}$, on assiste au contraire à une augmentation sensible du taux de retard. Celle-ci est principalement imputable à [...]**}**$_{S2}$ **]**$_{S0}$

$\Im(S_1, S_0)$ = LE RETARD SCOLAIRE ⊷ (DANS LE PRIMAIRE, [1980;1990])

$\Im(S_2, S_0)$ = LE RETARD SCOLAIRE ⊷ (DANS LE SECONDAIRE)

In this case, the passage is structured by two discourse frames, introduced by P_1 and P_2. This configuration can be seen as "ideal" in the sense that the persistence mode of each topical constituent is *typical of its topical function*: the core is the object of a referential chain, while satellites benefit from the scope peculiar to left-dislocated adverbials. In this situation, sentence and discourse level configurations are analogous, and each topical constituent plays an identical role with respect to its hosting sentence and to higher-level segments. For example, in the case of the previous excerpt, we have $\Im(U_1,U_1) = \Im(U_1,S_1) = \Im(U_1,S_0)$. For this reason, the framing theory plays a central role in our approach of discourse topics.

However, discourse frames are only one possible way to express composite topics in discourse, several other possible configurations being discussed in (Bilhaut & Enjalbert 2005). Moreover, as mentioned above, some configurations may appear to be in conflict with the informational structure of sentences considered individually. This is the case in the following example, which is a slightly modified version of the excerpt 2:

> ③ § **{ {** L'enseignement primaire$_{(P1)}$ a connu une forte diminution du taux de retard scolaire ces dernières années.**}**$_{U1}$ [...] **}**$_{S1}$ Dans le secondaire$_{P2}$, [...]

Let E_1 and E_2 be the referents of P_1 and P_2. In this version, S_1 is no more a discourse frame introducer: "l'enseignement primaire" occurs as the subject of its hosting clause. However, E_2 echoes E_1 as in excerpt 2, and the whole passage is still organised with the aim of opposing the two levels of the educational system in relation with LE RETARD SCOLAIRE (now E_0). Thus, E_1 alone can not be considered as an acceptable representation of the aboutness of S_1, and although it acts as a the topical core in U_1 considered in isolation, i.e. $\Im(U_1,U_1) = E_1 \multimap$ (CES DERNIÈRES ANNÉES), it acts as a satellite with respect to S_1: $\Im(U_1,S_1) = E_0 \multimap (E_1)$.

3 Non-topical expressions such as P_3 in the example above may also be of some interest regarding composite topics, but this issue will not be discussed here.

5 Implementation

As stated before, our approach to aboutness and discourse topics was originally motivated by information retrieval concerns, and has been developed to sustain an automatic analysis. An effective analyser based on this model has been implemented, using the LinguaStream platform[4] (Widlöcher & Bilhaut 2005). The obtained system is able to process structured documents in order to obtain a hierarchical thematic segmentation of the text as well as symbolic representations of the corresponding composite topics.

5.1 Main principles

One notable fact about the proposed method is that it is bootstrapped by the detection of the topical satellites and the analysis of the resulting discourse segmentation. The topical cores are analysed subsequently, with respect to the previously identified textual segments. Another notable fact is that our approach combines linguistic resources with numerical methods, *à la* (Ferret & al. 2001). This approach is particularly valuable in our case since we are simultaneously interested in fine-grained constructs such as dislocations, that are only accessible to "linguistic" methods, and in higher-level phenomena such as lexical cohesion, that are partially accessible to quantitative methods.

Yet another particularity of the presented approach is to rely simultaneously on two kinds of resources: generic discourse patterns on one hand, and domain-related knowledge on the other hand. The former are considered specific to a given language, and are given in the form of DSDL grammars[5]. The latter is specific to a domain, and is given in the form of *semantic axes*, which may be machine-learned from corpora as described in (Bilhaut & Enjalbert 2005). This approach allows us to take into account phrases that may play, as exemplified in the previous sections, a significant role in the organisation of the discourse that relates to a particular domain, as topical satellites.

5.2 Satellite-based segmentation

The first stages of the processing stream consist in the identification of lexical or phrasal units that should be considered as potential discourse structure markers with respect to the considered domain, relying on three kinds of resources:

- a language-specific - but domain independent - lexicon of *cue-phrases*;
- a set of language-specific syntactic patterns that aim at detecting characteristic constructs (now called *pseudo-cue-phrases*) that may also play a significant role in the discourse structure, such as discourse frame introducers or cleft expressions;
- a set of domain-specific *semantic axes* that refer to domain-specific groups of concepts that will usually play a satellite role[6].

The segmentation step relies on a set of language-specific discourse configurations, which forms a kind of partial textual grammar. Three configuration classes can be identified, depending on the formal and/or semantic nature of the criteria that determine them.

(i) *Explicit configurations* describe structures that are *integrally* marked by surface forms. This is for example the case of a discourse frame sequence such as in excerpt 2. In this case, the detection of cue-phrases or pseudo-cue-phrases is sufficient to proceed to the segmentation, and no domain-specific knowledge intervenes.

(ii) *Mixed configurations* describe structures that are *partially* marked, requiring some domain-related knowledge to be properly detected. For instance, this is the case in excerpt 3, where a frame introducer echoes another constituent that does not appear in a neutral position. In this case, the system looks for a constituent of a known semantic axis that appears after a cue-phrase or within a pseudo-cue-phrase, and looks for other items of the same axis if such an element is actually encountered.

(iii) *Implicit configurations* describe structures where explicit marks are totally absent (or where no explicit mark has been detected, for the matter of automatic analysis). Such a passage may be segmented provided it contains several items of the same semantic axis. It should be noted that although this case remains problematic (more precise heuristics still have to be determined), mixed configurations already cover many cases that *would not* be handled properly using a cue-phrase-based approach, since a mixed configuration may contain *no cue-phrase* and just one pseudo-cue-phrase that will allow other implicit marks to be properly detected.

4 http://www.linguastream.org

5 DSDL (Discourse Structure Description Language) is a discourse-targeted formalism introduced in (Widlöcher & Bilhaut 2005) and implemented in the LinguaStream platform.

6 Each semantic axis may be given in the form of a static list of lexical entries or in the form of a local unification grammar, for example for temporal or spatial expressions.

5.3 Identification of topical cores

After the segmentation step, the system proceeds to the identification of the topical cores of the obtained segments, in order to obtain complete representations of their composite topics. For the time being, the cores are obtained using a classical quantitative method based on the so-called *tf·idf* factor. For each word w_i in a segment s_j, the coefficient W_{ij} is computed as $tf_{ij} \cdot log(N/df_i)$, where tf_{ij} stands for the number of occurrences of w_i in s_j, df_i for the number of segments containing w_i, and N for the total number of segments in the document.

In our case we need to compute some form of distributional salience for complex terms, and not single words, but the above factor can not be directly applied to phrases since they do not follow the same redundancy rules as single words, at least when considering mid-sized units such as sentence groups or paragraphs. Indeed, due to various phenomena such as terminological reduction, multiples occurrences of complex phrase are marginal in such units, and their salience can not be evaluated distributionally. For that reason, the weight of a phrase p_i in a segment s_j is computed afterwards, as the sum of the weight of its constituents (which also favours longer phrases).

Using this factor, we are able to determine the most distributionally salient term(s) within a given segment, which provides some approximation of its discourse topic. In the applicative context of IR, we consider the *k* highest ranked as an acceptable representation of the topical core of a given segment. The *k* number is arbitrary, and is linked to the usual compromise between precision (low values of *k*) and recall (high values). It should be noted that our approach does not depend on the choice of this distributional method for topical core identification. On the contrary, it would be interesting to consider approaches driven by linguistic models (related to syntactic and informational structures, anaphora, centring...).

6 Conclusion

The point of view adopted here is *functional*, and we conceive the terms of topical core and satellite as referring to *discursive functions*. From this perspective, these concepts may be expected to apply to a variety of discursive configurations, from which a few instances were given in this paper. A wider set of patterns has been identified during our corpus study, and have been formalised using declarative languages under the LinguaStream platform. In combination with domain-related knowledge and numerical methods, these patterns are exploited by a system that detects composite topics automatically, and produces a segmentation as well as semantic annotations, as detailed in (Bilhaut & Enjalbert 2005).

Future work is firstly related to the evaluation of the obtained results, which is in itself a highly complex task when it comes to semantic and discourse analyses. Two kinds of evaluation are planned, restricting to geographical information. We are currently working on an *intrinsic* evaluation which aims at evaluating the quality of the segmentation in comparison with human-annotated documents. An *extrinsic* evaluation will also be conducted, relying on a search engine that has been developed on the notion of composite topics, in order to evaluate the overall gain brought by the system as perceived by a panel of real-world users.

References

(Asher 2004) N. Asher. Discourse topics. In *Theorical Linguistics*, 30:2-3, pp. 163–201, 2004.

(Bilhaut 2003) F. Bilhaut, T. Charnois, P. Enjalbert and Y. Mathet. Passage extraction in geographical documents. In *New Trends in Intelligent Information Processing and Web Mining*, Zakopane, Poland, 2003.

(Bilhaut 2005a) F. Bilhaut. *La notion de thème composite pour l'analyse thématique automatique du discours*. GREYC Research Report, 2005.

(Bilhaut & Enjalbert 2005) F. Bilhaut and P. Enjalbert Discourse Thematic Organisation Reveals Domain Knowledge Structure. In *Proceedings of IICAI'05*, Pune, India (t.b.p.).

(Chafe 1976) W.L. Chafe. "Giveness, constrastiveness, definiteness, subjects, topics and point of view", *Subject and Topic*, Academic Press, C. Li ed., 1976.

(Charolles 1997). M. Charolles. L'encadrement du discours – Univers, champs, domaines et espaces. In *Cahiers de recherche linguistique*, 6, pp. 1-60, 1997.

(Ferret & al. 2001) O. Ferret, B. Grau, J.-L. Minel, S. Porhiel Repérage de structures thématiques dans des textes. In *Actes de TALN'01*, Tours, France, 2001.

(van Dijk 1977) T.A. Van Dijk. *Text and context: explorations in the semantics and pragmatics of discourse*, London ; New York : Longman, 1977.

(Dik 1989) S.C. Dik. *The Theory of Functional Grammar*, Dordrecht: Foris Publications, 1989.

(Halliday & Hasan 1976) M. Halliday and R. Hasan. *Cohesion in English*, Longman, 1976.

(Lambrecht 1994) K. Lambrecht. *Information structure and sentence form: Topic, focus, and the mental representation of discourse referents*, Cambridge University Press, 1994.

(Widlöcher & Bilhaut 2005) A. Widlöcher and F. Bilhaut. La plate-forme LinguaStream : un outil d'exploration linguistique sur corpus. In *Actes de TALN'05*, Dourdan, 2005.

Visual Parser Builder

Dimitar Blagoev
Student
Plovdiv University "Paisii Hilendarski"
gefix@pu.acad.bg

George Totkov
Assoc. Prof. Dr.
Plovdiv University "Paisii Hilendarski"
totkov@pu.acad.bg

Abstract

The parser is an essential tool to every language researcher. Most parsers aim at speed using rather simple grammars and finite state-like automata (Gold Parser). Some allow using a programming language to extend the functionality of the grammar (YACC, ALE) but the need for programming skills arises. Others let you create extended grammars with visual graph editors (Intex) but lack the flexibility of a programming language. In this paper a visual parser building tool ILI is introduced. It allows the creation of complex grammars using simple visual elements and simple statements. The purpose is not fast parsing but ease of grammar creation.

1 Introduction

In computational linguistics finite-state automata and transducers are used to carry out different types of analysis: sentence boundary recognition (Silberztein 93), superficial syntax analysis, (Mohri 95), speech recognition (Mohri 97), discovering grammatical patterns in texts (Schiller 96), part-of-speech tagging (Roche & Schabes, 97) etc. Regular expressions (RE) and finite-state transducers (Daciuk 98; Karttunen et al. 97; Mohri 97; Noord & Gerdemann, 99) are the appropriate level of abstraction for thinking about finite-state languages and finite-state relations. In order to be able to experiment with such complex finite-state operations a number of compilers are provided (Noord 97; Watson 95). In (Noord & Gerdemann, 99) the regular expression operations provided by the compilers and the possibilities to create new regular expression operators are discussed. The benefits of such an extendible regular expression compiler are illustrated with a number of examples. An extension to finite state transducers is presented (Noord & Gerdemann, 01), in which the atomic symbols are replaced with predicates over the symbols. This extension is motivated by the observation that transducers with predicates generally have fewer states and fewer transitions. In this paper we present an extension in which the states and the number of transitions can be considerably less than in the traditional transducers.

Ordinary parsers see their grammars as static. They allow the user to enter a grammar either using some text form (Gold Parser[1]) or by visual editing (Intex[2]), then analyze the grammar and transform it into some other 'compiled' form that is faster to parse but often only a computer can understand. By restricting the grammar (for example to be context-free or regular expression) and defining some concrete rules that state how a grammar can be created, the parser can either convert the grammar to DFA or to special tables which makes the parsing process very fast but makes the grammar static. Parsers that use such tables are YACC[3] and Gold Parser (both are Look-Ahead LR parsers).

These restricted grammars may be very good for parsing simple well-defined formal languages, but more complex languages are very hard to define using such methods. A threadbare language is the $a^n b^n c^n$, which cannot be defined with context-free grammars. Although it does seem very artificial, it simply uses the property that in each context-free grammar every rule can be applied no matter what the surrounding. And in natural languages this is not the case. Even computer languages when described with a context-free grammar can (in many cases) only check the syntax of a program. The real work must be done by another program (compiler) which analyzes the result received from the parser and checks the semantics of the input text (program). This semantic checker is one level above the parser and is generally custom-made for each language using some popular programming language.

Also very often some algorithm is found that can generate or can help generate a substantial subset of a language (computer or natural), but the available parser would not allow it to be entered without modifying beyond recognition (or more often - at all). Try defining the $x_1^n x_2^n x_3^n \ldots x_m^n$ language (where $x_{n+1}=f(x_n)$).

[1] GOLD Parsing System - A Free, Multi-Programming Language, Parser – http://www.devincook.com/GOLDParser/ (1 June 2005)

[2] Intex - MSH Ledoux - http://msh.univ-fcomte.fr/intex/ (15 Aug. 2005)

[3] YACC: Yet Another Compiler-Compiler – http://dinosaur.compilertools.net/yacc/ (1 June 2005)

2 Existing Systems

The CLaRK system (Simov et al. 01), which development started under the Tübingen-Sofia International Graduate Programme in Computational Linguistics and Represented Knowledge (CLaRK), is a XML-based and incorporates regular cascaded grammars and the XPath[4] language. The grammars are composed of context-sensitive regular expressions (with left and right conditions) and can have variables and constraints over them, but no functions to work with them. The variables in the REs describe the substring matched by its definition when first initialized, and then matches to the same substring if used again. They can be used to form the result text. No functions are available to modify these variables.

Intex (Silberztein 93; Silberztein 99) is a linguistic development environment with a built-in parser, statistics and more. It can parse standard regular expressions or finite-state automata. As an extension the graph editor allows the inclusion of another graph as a sub-graph. Recursive graphs are allowed, however the building of a finite-state transducer of such graphs is not always possible (which results in slower parse speed). Predicates are allowed over word-by-word basis (by using the available POS tags).

ALE[5] is a system that integrates phrase structure parsing, semantic-head-driven generation and constraint logic programming with typed feature structures as terms. It employs a bottom-up, all-paths dynamic chart parser.

YACC and Bison are a type of parsers generators that convert a grammar description for a LALR(1) context-free grammar into a C program to parse that grammar. They create very fast parsers, but require programming skills if a more complex grammar is needed.

Gold Parsed Builder is another tool like YACC, with a nice graphical user interface for editing the grammars and exploring the created LALR tables. Unlike YACC/Bison however it does not generate C parsers. Instead it generates only the information needed – the tables and DFA – and another program called engine is used to do the actual parsing. Currently engines exist for more than 10 programming languages. Unfortunately because of this, no additional custom code can be added to enhance the functionality of the grammars (like in YACC/Bison – a custom C function).

3 Basic ILI Grammar Concepts

The ILI system has built-in virtual machine and a simple script language. On the surface its classes look very much like a finite-state automata, but the opportunities given by the built-in script language can be easily seen. It offers the user a tool to quickly create simple and average grammar, while allowing the experienced user to invent complex and reusable grammars that can be in turn used by a another user.

3.1 Parameters

Compiled grammars do not have any control over the parsing process. It is of course due to the fact that they are transformed into a static data that can only be interpreted in one way, predefined by the parser. This restriction forces that a grammar that accepts all lines in the input text that contain the current date must be edited at least once every day. And a grammar that accepts all lines that contain a date and time that is no older than one hour must be changed every hour. And this change is not straightforward – sometimes a sizeable part of the grammar must be rewritten. If such grammar could however be described as a function which takes one or more parameters this would make it a little more complex and big but will eliminate the need for editing every time.

In Figure 1 two classes are shown. The first node of class **A** accepts the a^n language and counts the length L (=n). The second node calls class **B**, giving as parameters L, the start symbol 'b' and the end symbol 'f'. With these parameters the recursive class **B** will accept the $b^nc^nd^ne^nf^n$ language. And with the change of only one character – the end symbol – class **A** can accept for example the $a^nb^n\ldots z^n$ language.

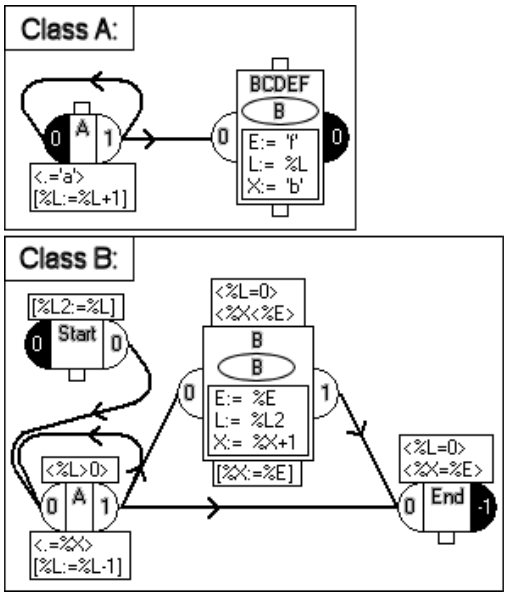


Figure 1: Classes for the $a^nb^nc^nd^ne^nf^n$ language

3.2 Head Movement Control

Normal parsers have full control over the movement of the reading head during parsing. This is because the used parsing algorithm explicitly defines when and how the head must be moved. If however the grammar author thinks of a way to make the parsing process much faster (by using a Boyer-Moore-like search or just by searching for some pattern in reverse order or in middle-first order) the parser would either not allow for such grammar to be entered or will transform it into its internal structures and will use the algorithm defined by the parser class.

[4] XML Path Language (XPath) – http://www.w3.org/TR/xpath (1 June 2005)

[5] Attribute-Logic Engine (ALE) – http://www.cs.toronto.edu/~gpenn/ale.html (15 Aug. 2005)

An ILI class has full control over the input head. Each node has input wait and output wait characteristics. The input wait specifies how the head is moved before processing the node and the output wait – after it has been processed. Both values can be negative. Furthermore the head pointer can be modified in a script statement like any other value.

3.3 Information Accumulation and Reuse

Often during the parsing of natural languages some parts are ambiguous when first recognized but the information that can disambiguate them can be near or anywhere in the text. An example is when searching for abbreviations – if an abbreviation is found at the end of a sentence it may or may not be considered as such (if it is not in the list of known abbreviations). But if later the same abbreviation is found in the middle of a sentence (followed by a small letter) it would also mean that most probably the first occurrence is also an abbreviation. For this simple case it is possible to just do the analysis in two steps – first to search for possible candidates and add them to the list of known abbreviations, and then do the whole analysis again finding more disambiguated abbreviations. But when a basic anaphora resolution is needed it would be useful if the grammar could accumulate information that eventually may be needed in the near text and then use it when ambiguous structures are found. For example to temporary store the unknown personal names whose gender can be identified by the context of the text, and use that fresh information for eventual anaphora resolution.

3.4 Objects

The normal nonterminal is a good way to define something that contains one or more other things. An example for nonterminal can be the word. But normal parsers can not give their grammars the ability to look inside each found nonterminal. Natural language parsers like Intex overcome this by tagging the words with simple one-letter characteristics and allowing the grammars to use these characteristics when a particular word group is wanted. But when a graph that tries to identify NPs is used by a graph that searches for VPs the latter must have some information about the found NP in order to found correct VPs. Graphs in Intex however cannot carry additional information (characteristics). This is so because even if such information does exist the used grammar definition and parser would not give any way of using it. To take advantages of objects and their data means to create grammars in which each transition may or may not be followed depending on the results received during the parsing and eventually manipulating the data and creating new one. A context-free grammar cannot do this. A context-sensitive grammar cannot do this. A script interpreter can do this.

Of course if for example the NP has only one characteristic it can be divided into a different graph for each possible value. But this is not the case. And for two or more characteristics the number of graphs that would be needed is the product of all possible values of all characteristics.

4 ILI Elements

4.1 Node

A node can be used to represent a simple state. But normal states in finite automata can not do anything besides remembering that the parser has reached them (and they forget that too once all transitions are followed).

In ILI the node not only says where to go next, but also:

- what conditions must be met upon entering and/or leaving the node
- what is done when entering and/or leaving the node
- how many characters to skip from the input before entering / after leaving the node (may be negative) (used to control the head)
- what class (if any) is used to analyze the data (a subclass)
- what parameters to be passed to the subclass if one is used
- what is its name - the place in the class where the result of a successful match is stored

4.2 Class

A class represents one graph (automata). Every grammar contains one or more classes. When ran, each class walks through the input text gathering all possible ways to go from its start state (node) to the end. Every nondeterministic automata can be represented using one class. But one class is not enough. Many parsers follow the Backus-Naur Form allowing one definition (automata) to be used many times thus significantly reducing both grammar size and ease of building. Each class in ILI can be regarded as a definition of a rule. But unlike the definitions in BNF, classes can accept input parameters and return what they have found. These parametric definitions let the user focus on the grammar as a whole. The "Regular expression", "Number in interval" and "Database query" classes are just few examples.

Each class has a *run at* property which instructs the parser when to start searching for matches of the given class. For the time being it can be set to "never", "once" or "at every character".

The class also stores the data of every last match returned by the nodes' subclasses. This allows the conditions and statements that the nodes contain to be more powerful, allowing late evaluation, predicates, etc.

4.3 Parser

The parser is relatively simple. It uses breadth first search to find all matches in the input text.

Because the classes are nondeterministic it is possible one initiating of a class to return more than one matches. This requires the parser to be able to trace two or more

paths inside one class, simultaneously, without messing up the memory of the different threads. To accomplish this, the parser creates a new instance of the class every time it is run or a branch in the path is reached. To make the processing more efficient three-layer memory architecture is used.

Each class has these three layers of variables:

- *Local Variables*: always copied, accessible only from one instance. Represent the memory of one path. Can be used for calculations that do not depend on other paths' results. Eliminates the risk of mixing up the variables of different running instances.
- *Intermediate Variables*: copied when a class run is initiated, accessible from every descendant of the first instance (when the class is initiated). Can be used to synchronize the active paths of one class run. Useful to implement the (A without B) structure.
- *Global Variables*: never copied, accessible from all instances.

The simple logic of the VM code and the ability to call external functions from it allows custom external classes to be embedded into the system. Such classes can be "Morphological Analyzer", "Database Query", "Web Page Download" and more. These classes and the results they produce can then be used by the grammars to perform much more complex analysis or even to perform complex tasks.

5 ILI Integrated Development Environment

The interface of the system (Figure 2) is simple. Multiple input files and grammars can be opened simultaneously with the results produced by previous parsings. An editable library contains classes that can be used in other grammars.

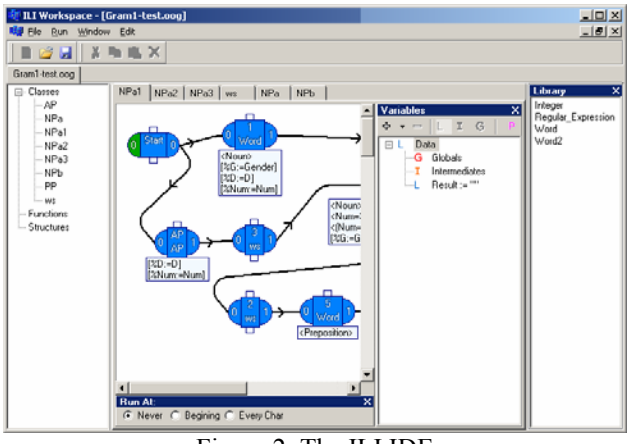


Figure 2: The ILI IDE

The grammars are edited visually. For each grammar the list of classes is shown. Each class can then be edited visually like a graph (see Figure 1). New nodes can be added and subclasses can be inserted with drag-and-drop. All properties of the nodes are edited in-place. All defined variables that the class uses are shown and edited in a tree structure.

After a parsing is complete a *Result Viewer* (Figure 3) appears which offers:

- Input text viewing with accepted areas highlighted
- Tree-style view of all found matches and their data
- Concordance view
- Saving the results for further analysis

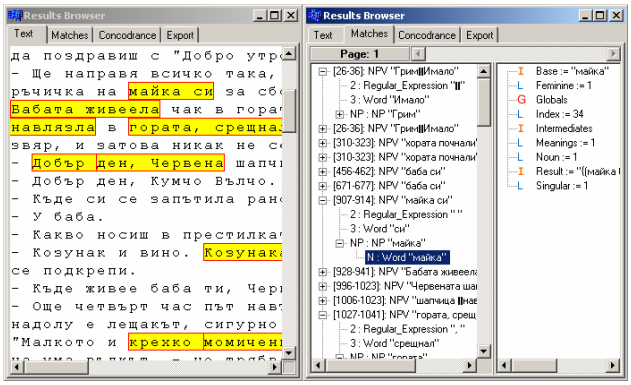


Figure 3: Result viewer

6 Comparison

Unlike other parsers with graphical front-end, ILI has a built-in virtual machine and a complex variable handling. This gives more flexibility and more control over the parsing. The $a^n b^n c^n$ language for example can be represented in many ways. It can be defined as three nodes for each character and one occurrence comparison at the end yielding a complexity of $O(M*n^3)$ where M is the length of the input. Or can be represented as a class with 15 nodes, 25 transitions and 25 statements but with lower complexity of O(M).

Table 1 shows a general comparison between GoldParser, Intex, CLaRK, ALE and ILI.

Feature (Expressive power)	Gold	Intex	CLaRK	ALE	ILI
Regular Expressions	√	√	√	√	√
Context-free grammars	√	√	√	√	√
Context-sensitive grammars	×	×	√	√	√[6]
Variables	×	×	√	√	√
Graphs (Classes)	×	√	×	×	√
XSL Transformations	×	×	√	×	×
Easily compare grammar outputs	×	×	√	√	√
Can parse binary files	×	√	×	×	√
User-friendly Visual Editor	×	√	×	×	√
Build-in script language	×	×	×	×	√
Word characteristics	×	√	×	√	√

[6] Because ILI classes can control the reading head, it is easy to move it to check for left / right conditions and then return it to a saved place.

Subclasses	√	√	×	√[7]	√
Recursive Subclasses	√	√	×	√	√
Classes can carry data	×	×	×	√	√
Symbol based characteristics	×	√	×	×	×
Structure (Object) based characteristics	×	×	√	√	√
Functions	×	×	×	√	√
Subclass Parameters	×	×	×	×	√
Head movement control	×	×	×	×	√
Easy access to other (predefined) classes (rules)	×	×	√	×	√
Parse-time modifiable classes and functions	×	×	×	×	√
Execution of external functions and classes	×	×	×	×	√
External Database Connectivity	×	×	×	×	√

Table 1: General features comparison

An ILI grammar can not only search for strings or classes but also execute commands during this process. For example if the classes "Find file in Directory", "Read File", "Add Watermark" and "Save File" are implemented, the user could write a grammar that adds watermarks to all pictures in all subdirectories of a given directory. Another grammar could download whole web sites or just a particular part of a given web site. If speech recognition and text-to-speech engines are added, a well defined semantic grammar could implement question answering.

7 Applications

In SLOG (Totkov & Angelova, 03), a system for Bulgarian speech synthesis, the module that does segmentation, transforms dates and numbers etc., morphological analysis and annotation is realized with the ILI technique. By using cascaded regular expression grammars, ILI classes and external functions all different parts of this analysis are connected in one grammar.

The ILI IDE could be used in the fields of language engineering (creation and testing of language grammars), data analysis, data conversion, etc.

The characteristics of the ILI classes allow complex parameterized patterns to be created and then reused.

Future plans include adding additional modules for statistics, web download, sub-parsings as well as wizards for creating various other grammars and rules. Possible future applications include data mining and automated information retrieval from the internet.

References

(Daciuk 98) J. Daciuk, *Incremental Construction of Finite-state Automata and Transducers and their Use in Natural Language Processing,* Ph.D. thesis, Technical University of Gdansk, 1998.

(Karttunen et al. 97) L. Karttunen, T. Gaal, A. Kempe, *Xerox Finite-State Tool*. Technical Report, Xerox Research Centre Europe, Grenoble, June 1997.

(Mohri 95) M. Mohri, *On Some Applications of Finite State Automata Theory to Natural Language Processing,* Natural Language Engineering 1 (1), 1995.

(Mohri 95) M. Mohri, *Finite-State Transducers in Language and Speech Processing*, Computational Linguistics 23 (2), 1997.

(Noord & Gerdemann, 01) G, Noord, D. Gerdemann, *Finite State Transducers with Predicates and Identity*, Grammars 4(3), 2001.

(Noord & Gerdemann, 99) G. Noord, D. Gerdemann, *An Extendible Regular Expression Compiler for Finite-state Approaches in Natural Language Processing,* in Proceedings of WIA'99, 1999.

(Roche & Schabes, 97) E. Roche, Y. Schabes (eds.), *Finite-State Language Processing,* MIT Press Cambridge, Massachusetts, 1997.

(Schiller 96) A. Schiller, *Multilingual Finite-State Noun Phrase Extraction*. ECAI Workshop on "Extended Finite State Models of Language", Budapest, 11-12 August 1996.

(Silberztein 93) M. Silberztein, *Dictionaires électroniques et analyse automatique de textes: le système INTEX*, Masson, Paris, 1993.

(Silberztein 99) M. Silberztein, *INTEX: a Finite State Transducer toolbox*, in Theoretical Computer Science #231:1, Elsevier Science, 1999.

(Simov et al. 01) K. Simov, Z. Peev, M. Kouylekov, A. Simov, M. Dimitrov, A. Kiryakov, *CLaRK - an XML-based System for Corpora Development*, in Proceedings of the Corpus Linguistics Conference, 2001.

(Totkov & Angelova, 03) G. Totkov, V. Angelova, *On Bulgarian Text to Speech System,* in Proceeding of the International Conference ICT&P, Varna, Bulgaria, 2003.

(Watson 95) B. Watson, *Taxonomies and Toolkits of Regular Language Algorithms*, University of Technology Eindhoven, the Neatherlands, 1995.

[7] Rules can be used to describe how subtypes are integrated into more general types.

Lexicalization of Grammars with Parameterized Graphs

Olivier Blanc and **Matthieu Constant**
Université of Marne-la-Vallée
5, bd Descartes
Champs-sur-Marne
77 454 Marne-la-Vallée, France
{oblanc,mconstant}@univ-mlv.fr

Abstract

This paper is about the use of large coverage syntactic lexicon for text parsing. Our work focuses on the construction of a lexicalized unification grammar using the fine-grained syntactic information encoded in the lexicon-grammar tables built at LADL (France). We present a method to generate this grammar from a hand-built meta-grammar composed of parameterized graphs. For each lexical item of our lexicon, a specialized grammar is generated by resolving the parameters referring to syntactic properties encoded in the lexicon-grammar tables. We also show that our method can be adapted to a more complex lexicon in the form of relational tables.

1 Introduction

Over the past ten years, interest in the development and use of Language Resources (LR) have increased dramatically and become a global concern. This interest is not confined to corpora, but extends to lexicons and grammars. For instance, as interaction between descriptive linguistics and language engineering is growing, Natural Language formalisms are now being adapted to the interaction between lexicon and grammars such as LTAG (Schabes et al. 1988; Abeillé 2002; XTAG Group Research) and related frameworks (Carroll et al. 1998) or HPSG (Pollard et al. 1994).

Our goal is to develop a robust syntactic parser dealing with real texts. This involves the construction of a fine-grained lexicalized grammar. In this paper, we present a method inspired by Roche (1993) to build such a grammar semi-automatically by using large-coverage lexicon-grammar resources (Gross 1994) and a system of parameterized graphs.

This paper will be preliminary devoted to a brief description of the Language Resources used (section 2) and then a detailed introduction to our grammar formalism (section 3). The last sections (4 and 5) will focus on the lexicalization process and some extensions.

2 Language Resources

Over the last thirty years, the informal network RELEX of laboratories in the domain of Linguistics and Computational Linguistics (http://infolingu.univ-mlv.fr), has been constructing hand-built lexical resources in several languages (French, English, Portuguese, Spanish, German, Korean, Thai, ...). Especially, their effort focused on the construction of exhaustive syntactic dictionaries in the framework of the lexicon-grammar methodology initiated by Gross (1975). The lexical entries are predicative elements, either verbs, nouns or adjectives (simple words or multiword expressions). For each predicate, a set of syntactic properties is systematically examined such as:

- number and nature of the arguments (e.g. complemental clause, infinitive, human noun phrase, ...),
- appropriate prepositions,
- accepted transformations (e.g. passivation, argumental alternation, pronominalization, etc.),
- some co-reference resolutions.

All these properties are encoded into syntactic dictionaries in the form of tables called lexicon-grammar tables. Each row corresponds to a lexical value and each column corresponds to a syntactic property. A boolean value at the intersection of a row and a column indicates whether a given lexical entry verifies a syntactic property. Each table gathers predicative elements that have some syntactic similarities according to definitional criteria (Gross 1975). An example of a lexicon-grammar table is given in figure 1[1]; it represents a subset of French verbs with the definitional construction *N0 V que P* (N0 V that S)[2],

[1] A true value is represented by the symbol + (- for false)

[2] These verbs have a noun phrase as subject and are followed by a complemental clause

such as the verb *empêcher* (to prevent).

The French lexicon-grammar currently contains 15,000 simple verbs and 10,000 predicative nouns and adjectives. In addition, there is a dictionary of frozen sentences (composed of 30,000 entries). This linguistic work is still in progress.

N0=Nhum	N0=Nnr	N0=queP	N0=V1W	entry	N0 V	N1=quePind	quePind=deVOW	N1=quePsubj	quePsubj=deVOW	quePsubj=VOW	(N1)(deV-infW)	queP=Ppv	N1=Nhum	N1=N-hum	N1=le fait que P
+	-	-	-	détester	-	-	-	+	+	+	+	-	+	+	+
+	+	+	+	empêcher	-	-	-	+	-	-	+	+	-	+	-
+	+	+	+	encenser	+	-	-	+	-	-	+	-	+	+	+
+	-	-	-	envier	-	-	-	+	-	-	+	-	+	+	+
+	-	-	-	estimer	-	-	-	+	-	-	+	-	+	+	+
+	+	+	+	exalter	-	-	-	+	-	-	+	+	+	+	+
+	-	-	-	exécrer	-	-	-	+	+	+	+	-	+	+	+
+	-	-	-	fêter	-	-	-	+	-	-	+	-	+	+	+
+	-	-	-	flétrir	-	-	-	+	-	-	+	-	+	+	+
+	-	-	-	fustiger	-	-	-	+	-	-	+	-	+	+	+
+	-	-	-	haïr	-	-	-	+	+	+	+	-	+	+	+
+	-	-	-	honnir	-	-	-	+	+	-	+	-	+	+	+

Figure 1: sample of a lexicon-grammar table

3 Decorated RTN as a grammatical formalism

Our current research focuses on the exploitation of those accurate and systematic subcategorization descriptions and transformational properties encoded in the lexicon-grammar tables for large coverage text parsing. For this purpose, we are currently constructing a lexicalized unification grammar for French, which is generated semi-automatically from the syntactic tables using the methods described in the next section.

Our grammar is a syntagmatic grammar represented by a Recursive Transition Network (RTN) (Woods 1970) augmented with feature structure constraints. The different realizations of each syntactic constituent of the grammar are described in recursive finite state automata; those descriptions are decorated with functionnal equations that help formalize various linguistic phenomena such as the agreement between two constituents or the extraction of a grammatical item and long distance dependencies.

This formalism is actually very close to the Lexical Functional Grammar model (LFG) (Bresnan 1982), both models being equivalent from the point of view of their descriptive and computational capacity. The main difference is that, in our case, context-free rules are replaced by linguistic descriptions encoded in finite-state graphs.

Many phrases such as semi-frozen expressions (e.g. time adverbials, numerical determiners, ...) or named entities frequently occur in texts and exhibit lexical and syntactic local constraints that can be easily described in the form of finite state graphs (Silberztein 1994; Gross 1997). Such local grammars permit efficient recognition and can be integrated well as part of our whole grammar framework with RTN-parsing as a basis.

Moreover, the representation of syntactic constituents into recursive finite state automata allow a grammar writer to relate with ease syntactic constructions which are considered transformationally equivalent, like passivation, argument alternation, nominalization of a finite clause.

We believe such transformations cannot be considered as general syntactic rules but are strongly related with some specific lexical elements. Thus, in this context, each transformation must be described on a case by case basis for each predicative element, as described in the tables. The complexity of such systematic description can be greatly reduced by the use of parameterized graphs as will be shown in the next section.

For instance, the graph in figure 2 represents different realizations of French clauses having as main predicate, the verb *empêcher* (to prevent) as described in the lexicon-grammar table given in figure 1. On the left, we describe the possibility to have the subject as a NP or a sentential complement like a subjunctive clause introduced by the conjunction *que* (that) or an infinitive:

> *(Lea+Que Lea ait quitté Max+Boire du café) empêche Luc de dormir.*
> ((Lea+That Lea left Max+Drinking coffee) prevents Luc from sleeping)

The right part of the graph presents the possible realizations of the second argument which is a predicative NP (SN in French) or a subjunctive clause. The bottom path describes the possibility of raising the subject of the *que*-complement clause in position of direct object:

> (1) a. *Le soleil empêche que Luc travaille*
> = b. *Le soleil empêche Luc de travailler*
> (The sun prevents Luc from working)

In our formalism, labels prefixed with a colon (such as <:SN>, <:P> or <:V>) are non-

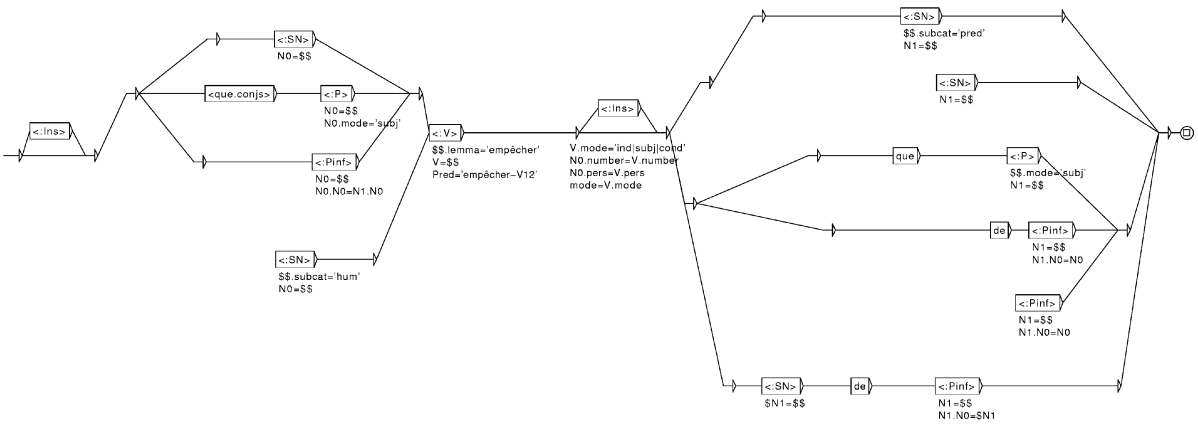


Figure 2: sentence constructions anchored by the verb *empêcher*

terminal symbols referring to syntactic constituents described in other graphs. For example, the label <:V> in the center of the figure refers to a graph describing the verbal complex of the sentence (which is the verb *empêcher* that might be modified by some adverbs, or modal and aspectual auxiliaries). Finally, the functional equations are given under the boxes and permit among others

- to verify the agreement in number and person between the verb and its subject (e.g. *N0.number=V.number*),
- to resolve some co-references, by identifying the subject of the infinitive (e.g. *N1.N0=N0*),
- and to identify the semantic predicate of the sentence with its arguments, while verifying that their natures are compatible with its subcategorization properties (e.g. *$$.subcat='hum'*[3]).

The result of the sentence analysis consists of a syntactic tree associated with a feature structure which contains all that information. Figure 3 is a simplified version of the feature structure resulting from the parsing of sentence 1.a and presents the semantic predicates with their essential arguments identified in the text.

$$\begin{bmatrix} CAT : P \\ Pred : empecher \\ N0 : \begin{bmatrix} CAT : SN \\ head : soleil \end{bmatrix} \\ N1 : \begin{bmatrix} CAT : Pinf \\ Pred : travailler \\ N0 : \begin{bmatrix} CAT : SN \\ head : Luc \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Figure 3: simplified version of the feature structure obtained by parsing sentence 1.a with the automaton given in figure 2

4 Construction of a Lexicalized Grammar

We are currently building a lexicalized grammar for French using the formalism described above. This grammar is semi-automatically generated from lexicon-grammar tables. The construction of specialized grammars for each predicative element requires the construction of meta-grammars by hand. A meta-grammar is associated with a table and is composed of a set of parameterized graphs.

Each parameterized graph describes a syntactic consituent (finite or infinite clause, clause missing an extraposed element, etc.) whose predicate element is a variable which will be instantiated during the lexicalisation stage. Informally, a meta-grammar (i.e. the set of parameterized graphs associated with a table) can be seen as the specialized grammar for an abstract entry of the table, that would verify all the properties encoded. Each path is identified with a parameter referring to the property encoded in the corresponding table. A parameter has the following format *@X@*, where X is the name of the column referring to a syntactical property.

Once the meta-grammar of a table is con-

[3]Symbol $$ refers to the feature structure associated with the item in the box above

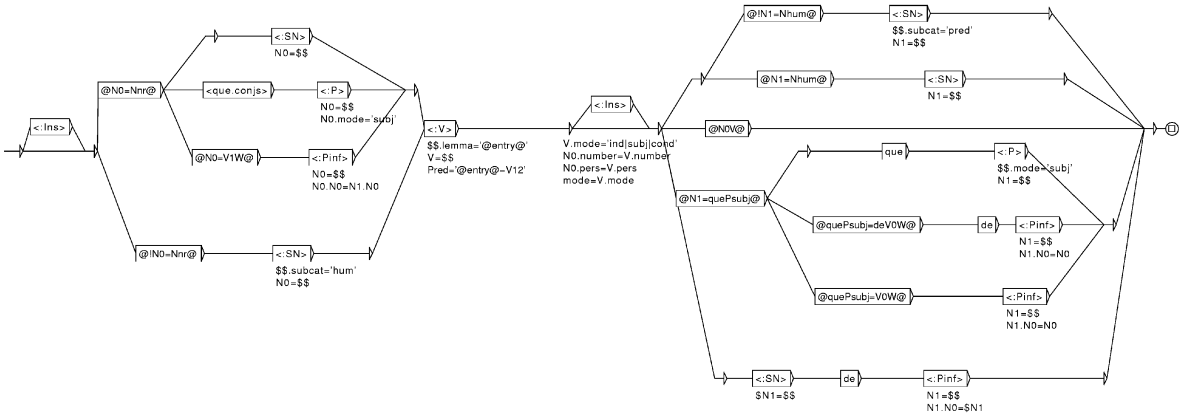


Figure 4: parameterized graph of declarative sentences for the table given in figure 1

structed, for each lexical entry, the generation process creates a specialized grammar where only the paths corresponding to the properties verified by the entry are kept. When the properties are not verified, the corresponding paths are removed. Columns can also contain textual value; in this case, the parameter is replaced by this value. It is also possible to negate a parameter: *@!X@* means that the paths corresponding to the property X are kept only if the value is false. For instance, figure 4 presents one of the parameterized graphs associated with the table given in figure 1. The lexicalized graph in figure 2 specialized for the verb *empêcher* has been generated from it. For instance, parameter *@N1=Nhum@* refers to the column indicating if the transitive complement can be a human Noun Phrase; parameter *@entry@* refers to the column providing the graphical form of the verb and parameter *@N0V@* refers to the column indicating whether this verb accepts the direct object ellipsis.

Note that it is theoretically possible to automatically produce the meta-grammars from the tables. However, this process is not straightforward because some syntactic properties encoded in the tables are specific to few tables only, and the meaning of a property can vary from a table to another. Moreover, some properties aren't explicitely encoded because they are accepted (or rejected) uniformally for all the verbs in a table. So we decided to write for each table, its associated meta-grammar manually.

Once the lexicalized graphs are automatically generated, we compute the union of the graphs for each syntactic constituent. Then, epsilon-transition removal, determinization and minimization are computed to obtain a grammar optimized for parsing. The construction of the whole lexicalized grammar for French is a long process. At this stage of our work, we only achieved the convertion of 17 tables (15 tables of verbs and 2 tables of nouns) which is about 15% of the whole set of tables and represent 2468 lexical entries. In its current state, the grammar, obtained from 137 parameterized graphs, contains 38,000 states and 70,000 transitions.

5 Extensions

5.1 Relational Tables

Standard syntactic dictionaries are in the form of simple tables. Nevertheless, it is sometimes more convenient to use *relational tables* to avoid duplication: for instance, this method has been used to represent time adverbials (Maurel 1990), geographical locative prepositional phrases (Constant 2003). A system of relational tables is composed of a set of tables (which includes a main table) and a set of relations between these tables. A relation is a special property that refers to a set of other properties in another table. This type of dictionaries, though similar to the standard ones, cannot be used straightforwardly in the lexicalization process described above and needs slightly different parameterized graphs. Actually, as information is split into multiple tables, a parameter should not only refer to a syntactic property (a column) but instead to the sequence of relations needed to reach the information pointed by the parameter. More detailed explanations can be found in (Constant 2003).

5.2 Meta-meta-grammars

The construction of the whole lexicalized grammar involves the construction of a parameterized graph for each type of constituents for each lexicon-grammar table. This process is costly because it requires many manual duplications. A more convenient way to deal with this would be to generate automatically every parameterized graphs related to a table from the same source. This source could be another kind of parameterized graph, let's call it meta-parameterized graph. The process of generation of the parameterized graphs from such a meta-meta-grammar requires a special table. Each row correspond to a type of constituent to be built, each column describes a property of those constituents such as the verbal tense, or the non-existence of a complement. Another approach using higher-level parameterized graphs has been studied in (Paumier 2003).

6 Conclusion

The need for fine-grained linguistic descriptions for parsing has become a reality with the development of more and more effective parsers. In this paper, we presented a method for interfacing a large-coverage syntactic dictionary with a grammar. We are currently using this method for the construction of a large-coverage unification grammar for French. It has been designed to deal with other languages studied within the lexicon-grammar framework. We think that it could be also adapted to other linguistic description frameworks such as the Proton (Eynde et al. 2001) or COMLEX Syntax (Grishman et al. 1994) projects.

References

(Abeillé 2002) Abeillé, Anne, 2002. *Une grammaire électronique du français*, CNRS Editions, Paris.

(Bresnan 1982) Bresnan, Joan, 1982, *The Mental Representation of grammatical relations*, MIT Press.

(Carroll et al. 1998) Carroll, John, Nicolas Nicolov, Olga Shaumyan, Martine Smets and David Weir, 1998, *LexSys Project*, Proceedings of the 4th International Workshop on Tree-adjoining Grammars and Related Frameworks (TAG+'98), Philadelphia, USA, pp.29-33

(Constant 2003) Constant, Matthieu, 2003, *Grammaires locales pour l'analyse automatique de textes : Méthodes de construction et outils de gestion*, Thèse de doctorat, Université de Marne la Vallée.

(Eynde et al. 2001) Eynde, Karel van den and Piet Mertens, 2001, *La syntaxe du verbe, l'approche pronominale et le lexique de valence PROTON* Preprint 174, Departement of Linguistics, K.U.Leuven

(Grishman et al. 1994) Grishman, Ralph, Catherine Macleod and Adam Meyers, 1994, *Comlex Syntax: building a computational lexicon*, Proceedings of the 15th conference on Computational linguistics, Kyoto, Japan, pp.268-272

(Gross 1975) Gross, Maurice, 1975, *Méthodes en syntaxe*, Hermann, Paris.

(Gross 1994) Gross Maurice, 1994, *Constructing Lexicon-grammars*, In Computational Approaches to the Lexicon, Atkins and Zampolli (eds.), Oxford Univ. Press, pp. 213-263

(Gross 1997) Gross Maurice, 1997, *The construction of Local Grammars*, in E. Roche and Y. Schabes Eds., *Finite State Language Processing*, Cambridge, Mass., MIT Press, pp. 329-352

(Maurel 1990) Maurel Denis, 1990, *Adverbes de date : étude préliminaire à leur traitement automatique*, Lingvisticae Investigationes, XIV:1, John Benjamins, pp 31-63

(Paumier 2003) Paumier Sébastien, 2003, *De la reconnaissance de formes linguistiques à l'analyse syntaxique*, Thèse de doctorat, Université de Marne-la-Vallée.

(Pollard et al. 1994) Pollard C. and I.A. Sag, 1994, *Head-Driven Phrase Structure Grammar*, University of Chicago Press and CSLI Publications.

(Roche 1993) Roche, Emmanuel, 1993, *Analyse syntaxique transformationnelle du français par transducteurs et lexique-grammaire*, Thèse de Doctorat, Paris, Université Paris 7.

(Schabes et al. 1988) Schabes, Yves, Anne Abeillé and Aravind K. Joshi, 1988, *Parsing strategies with 'lexicalized' grammars: Application to tree adjoining grammars*, In Proceedings of the 12 International Conference on Computational Linguistics (COLING'88), Budapest, Hungary, August 1988.

(Silberztein 1993) Silberztein, Max D., 1993, *Dictionnaires électroniques et analyse automatique de textes. Le système INTEX*, Paris, Masson, 234 p.

(Woods 1970) Woods, W.A., 1970, *Transition Network Grammars for Natural Language Analysis*, in Communications of the ACM, Vol 13:10

Unsupervised Knowledge-Free Morpheme Boundary Detection

Stefan Bordag *sbordag@informatik.uni-leipzig.de

Abstract

A new algorithm is presented which performs fully unsupervised basic morphologic analysis in any desired language without prior knowledge of that language. The algorithm detects morpheme boundaries and can also be modified to perform other tasks, e.g. clustering of word forms of the same lemma and the classification of the found morphemes. The primary aim is to reach maximum precision, so that the output can be used in a postprocessing machine learning step to increase recall. The algorithm is based on cooccurrence measures and letter successor variety and does not use any complex or computationally intense methods such as LSA. Consequently it is fast, efficient and scales well.

1 Introduction

This paper describes first results of a study of unsupervised, knowledge-free and therefore language-independent acquisition of morphology. This topic involves many different goals such as finding a segmentation of word forms into their morphemes, clustering different word forms of the same lemma, providing declensional and conjugational classes, extracting alternation rules etc. The identification of valid morpheme boundaries can be considered as the first step of the analysis. This step can be further divided into two parts: First, the identification of morphemes with a precision as high as possible, no matter how low the recall. Second, enlargement of this knowledge by common machine learning methods preferably without loss of too much precision. In a kind of circular feedback mechanism, this combined knowledge can be used in a repeated first step in order to find more knowledge.

This paper focuses on the first part, the identification of morpheme boundaries (also called morphology segmentation of word forms). Both the identified morphemes as well as the method itself can be used to produce a clustering of word forms of the same lemma as a side effect with a quite high precision. Another possible application of this method is the classification of the found morphemes into prefixes, stems and suffixes.

A brief evaluation for German and English is given but will be expanded in future work on an improved version of the algorithm.

1.1 Related work

Knowledge-free morphology segmentation has been the aim of several algorithms based on different approaches. Most of the methods can be divided into three general approaches: the minimum description length (MDL) model (first used in this context by (Brent *et al.* 95) and (Kazakov 97)), semantic based (Schone & Jurafsky 01) and letter successor variety based model (Harris 55). They all make use of very different mechanisms and so it might be possible to combine them in order to further boost their good results. The approach described in this study is directly based on the letter successor variety method but also makes use of context similarity. Therefore this approach can be viewed as one of the first attempts to merge such methods.

1.1.1 Minimum description length

One of the first successful knowledge-free algorithms is based on expectation maximization (Goldsmith 00). The initial algorithm cuts each word at one position based on a probability and the lengths of hypothesized stems and affixes. It then attempts to generalize various words into signatures (classes of words that have the same morphology). The quality of the algorithm improved when the minimum description length model (see (de Marcken 95)) was included, which has already been used in such a context by (Brent *et al.* 95) but also directly as a fitness function in a genetic algorithm approach (Kazakov 97). MDL represents a kind of balance between over- and undergeneration of stemming rules: the optimum is the most compressed representation of the data (the words) in the sense as to use the least necessary number of word forms and signatures at

*University of Leipzig

the same time. This removes all free parameters which makes a rather elegant solution. This method only considers the list of distinct words at any point and thus it has an (unknown) upper bound of quality since in a language certain things cannot be explained by any kind of frequency. Notably, Goldsmith's approach has since been used as the baseline algorithm for other algorithms to be compared against.

Another approach from the category of minimum description length based algorithms is the one introduced by (Creutz 03). Adding maximum likelihood (ML) and later the Hidden Markov Model (Creutz & Lagus 05) for a classification of the found morphemes the authors constructed an improved version of a segmentation algorithm: it randomly segments words and then measures how well the segmentation fits into the incrementally built knowledge base. This algorithm seems to be specialized on agglutinative languages such as Finnish and it tends to overgenerate slightly in other cases. Moreover it needs information on the length and frequency distributions of morphemes of that language (thus it is not entirely knowledge-free). Another enhancement worth mentioning was proposed by (Argamon *et al.* 04) by adding a recursive component to the analysis which while keeping steady results for morphology-poor languages might improve results on morphology-rich languages.

1.1.2 Semantic context

An entirely different approach has been taken by (Schone & Jurafsky 01) who included the semantic context of the words to be segmented into their segmentation algorithm. First a list of affix candidates is generated by simply counting frequencies. Using these candidates it is possible to generate a list of possible other word forms of the same lemma for each input word such as *listening* and *listen*. Second, latent semantic indexing (LSA) (Deerwester *et al.* 90) is used in order to find out whether the generated pairs of words are semantically similar according to the corpus used (that is, whether they appear in similar contexts).

1.1.3 Letter successor variety

The oldest and seemingly least successful approach to date is the *letter successor variety* method (Harris 51). The idea is to count the amount of different letters encountered after (or before, respectively) a part of a word and to compare it to the counts before and after that position. Morpheme boundaries are then likely to occur at sudden peaks or increases of that value. Parameters of this approach can be varied (Harris 55) but on the whole it has not yet been successfully employed for morpheme segmentation, see also (Hafer & Weiss 74) and (Frakes 92), because when applied to the whole list of distinct word forms, the 'noise' from too many different possibilities renders the results nearly useless. The method has also been used for the generation of 'good' candidate lists for postprocessing machine learning steps for morpheme segmentation by (Déjean 98), but unfortunately the authors do not mention the quality of the results they obtained by this method.

1.1.4 Corpus vs. word list

Another possible distinction of the existing approaches for morphology segmentation can be based on whether they make use of the list of word forms only (and eventually their frequencies) such as Linguistica (Goldsmith 01) or (Déjean 98). The work of (Baroni 03) can be included here, too. It is neither based upon the minimum description length model nor on any of the other possibilities. Baroni uses the cooccurrence of potential morphemes as an information source about the morphemes themselves. The other possibility is to include context information on the word level from the raw text in one way or another such as the approach taken by (Schone & Jurafsky 01). This kind of classification also shows that the methods used have at least two independent components which means that such algorithms might be able to boost each other's performance.

2 Context Similarity

The approach described in this study represents a combination of using context information (albeit in a different way from that described by (Schone & Jurafsky 01)) and the *letter successor variety* (LSV) idea described by (Harris 51). The idea is that the letter successor variety used on the plain list of word forms has to put up with too much noise from irregularities and other sources. However, a list of word forms that all have one or more kinds of syntactic information in common (such as gender, case, number) would make the noise for the LSV method manageable. This kind of approach would, of course, work even better, if it were possible to generate such a list for

each word. For the word *running*, the list would contain such word forms as *swimming*, *walking* or *diving*.

In order to obtain such a list of word forms with the same syntactic information for a given input word form, it is necessary to reflect on the possibilities of language. Whichever language is considered, syntagmatic relations will always hold between word forms standing immediately next to each other. For example, it is very probable that after the verb *goes*, any kind of lexicalized direction information will appear, such as *home*, *to* or *out*. On the other hand, in front of such direction information tokens, all kinds of verbs are likely to occur. Some or many of them will also have the same grammatical markers as the input word form such as *runs*, *walks* or *jumps*. These word forms are crucial for further analyses because they are morphologically similar to the input word.

Therefore, the first step is to compute all neighbouring word forms of a given word form A. At that point it is useful to discriminate mere frequent coappearance from statistically significant cooccurrence. This can be done by a multitude of methods and in this case the log-likelihood (Dunning 93) has been chosen because in other experiments (Bordag 05) it has proven to be one of the best measures.

The typical (left or right) neighbours of the word form A, along with their significances as found by applying this or another significance formula, can be represented as a vector $\overrightarrow{A_n}$ (the index n means neighbour) in the assumed vector space of word forms. The second step consists of comparing pairs of word forms based on their neighbourhood vectors. This can be done by simply counting the number of common words in the vector or using a distance or similarity measure such as the dice coefficient. Again, from other experiments (Bordag 05) the dice coefficient along with the simple counting proved to perform best and thus simple counting has been chosen. Thus, for any given word A it is possible to retrieve a vector $\overrightarrow{A_s}$ of most similar words to A. $\overrightarrow{A_s}$ then 'contains' all words that usually have similar left or right neighbours as A.

For the example word *running* given above, the most similar words are *run (108), using (99), runs (71), working (70), operating (70), moving (67), getting (65)*. The value 99 means that in the used corpus there are 99 different word forms appearing significantly often to the left and the right of both the words *running* and *using*.

3 Letter Successor Variety on Context Similarity Vectors

The second step of the algorithm described in this study is based on LSV and takes the context similarity vectors of the previous step as input. There is currently one free (probably language independent) parameter required at this point. From the context similarity vectors, only the 150 most similar words are kept for further processing. This is because keeping all similar words reintroduces some of the noise as discussed in section 1.1. Another noteworthy detail is that from this point on, all remaining word forms in the vector are treated as a set of words (dubbed similarity set) without any ranking. Later, it might be possible to introduce optimizations of the core algorithm by taking the similarity ratings into account.

In the next paragraphs, the German word form *glückliche* (*happy*) will be used as an example since its morphological segmentation includes two suffixes: *glück-lich-e*. The *-lich* derives an adjective from the noun *Glück* (happiness) and the *-e* is an ambiguous inflectional ending. One of its meanings is female gender. In the examples given, the character # marks the beginning and end of the word form.

The LSV based algorithm computes several values **for each transition between characters** of a given word form: The left and the right letter successor variety, the overlap factor, the bigram score and the multiletter score. Finally all of them are combined in order to produce a score for each transition. A high score translates into a morpheme boundary by means of a threshold. The complete computation of the example is additionally depicted in Table 1.

Computing the **letter successor variety** is done as follows: Count the number of different letters encountered after a given string from the beginning (or the end, respectively). In our example, after the string *#glü* only 1 distinct letter could be found in the similarity set, although 4 word forms began with this string. The same can be done for the other direction: before the string *liche#*, there were 7 different letters encountered out of a total of 15 word forms ending with that string.

Computing the **overlap factors**: In order to detect that a smaller string is part of a larger morpheme, it is possible to count how many of the words seen with the suffix *-iche#* (17 of 150) in the example have also been seen with *-liche#* (15 of 150), see also (Déjean 98). In this case, this is a strong evidence that *-iche#* is part of *-liche#* and thus a weight of $15/17 = 0.88$ will be computed for the transition at this place: *glück-liche*.

A problem with the overlap factor is that in some languages there are phonemes which are realized through a succession of several letters. This is the case for the *th* in English or for *sch* in German. This means that after the **single 'letter' *sch*** in, for example, *schlimme* (bad), 7 different letters after *sch* can be observed. The overlap factor at this point is wrongly 1.0, because of all 18 words which begin with *#sc-* 18 also begin with *#sch-*. But there are only 18 out of the 150 words which begin with the multiletter *sch* at all, thus the overlap factor should in fact be $18/150 = 0.12$ (at the same time expressing the uncertainty of making a decision at this point, because only 18 out of 150 words begin with the same letter). In order to detect this, the hypothesis has been made that the multiletters generally have a far higher frequency than other bi- or trigrams. This can be expressed as the multiletter weight.

In order to compute the **multiletter weight**, a ranking of all bi- and trigrams can be produced in order to distribute weights to each ngram between 1.0 (to the highest ranked) and 0.0 (to the lowest ranked). Using the weight of either the bigram or the trigram to the left (or respectively to the right), it is possible to take either the frequency count of the string of the transition one to the left or two to the left, depending on whether the weight of the bi- or the trigram is higher. This overlap factor can then be taken into account in the form of a weighted average. In the example in Table 1 for the transition *li-ch*, this means that to the right there is the frequency based bigram weight of 0.6 for the bigram *ch*. This is larger than the trigram weight 0.5 for *che*. Therefore the final right overlap factor is the weighted average $(36/39 * 1.0 + 36/129 * 0.6)/(1.0 + 0.6) = 0.68$ instead of the initial $36/39 = 0.92$. Over 3% of the German words begin with *sch* and almost all of them were wrongly analyzed to have this prefix. After applying multiletter weights, almost none of them were falsely analyzed with only minor changes to the analyses of other cases.

Since this algorithm does not (yet) take the distributions of the potential suffixes into account , it has a bias towards analyzing frequent strings at the end of a word as suffixes. This and other effects can lead for a word like *Barbier* to be falsely analyzed as *Barbi-er* because *-er* is one of the most frequent suffixes in German. However, in this case the bigram *ie* is divided, which in fact is a multiletter. Thus, the bigram frequency based ranking can be reused in order to distribute a **bigram weight**. The weight states how improbable it is for a division to occur in that bigram where 0.9, for example, means that it is extremely improbable for a division to occur at that place. In table 1 for example the value 0.7 in the row of bigram weight means that it is quite improbable to divide the string *ch*.

Another consequence of the missing distributions of affixes at this stage is that an overestimation of common strings encountered at the beginning and the end of word forms can occur. Therefore, a trivial uncertainty weight has temporarily been introduced which weights down all short strings (uni- and bigrams) at the beginning or end of word forms. In the experiments conducted for the evaluation it was arbitrarily chosen to be 0.3 for unigrams and 0.6 for bigrams at the ends of the word forms. Thus, for the word form *rote* the final score for *rot-e* would be 11.2, but since *-e* is a unigram at the end of a word, the score would be $11.2 * 0.3 = 3.36$. This mechanism removed all wrong and approximately half of the correct affix boundary identifications at the beginnings or ends of word forms. The remaining half of correct boundary identifications will still be enough in order to induce a learning of these affixes in a postprocessing machine learning step.

The missing distributions of affixes can later be added in a postprocessing step in order to refine the results. This bootstrapping process is subject to further research.

The final **right score** for any transition n can then be computed quite easily (and in the same manner as the **left score**). It consists of the multiplication of the initially obtained LSV with the averaged overlap factor and the inverse bigram weight. In the example in Table 1, for the transition *glückli-che* the right score would be computed as follows: the initial LSV is 4, the weighted over-

the input word:	#	g	l	ü	c	k	l	i	c	h	e	#
LSV from left:			6	2	1	1	1	1	1	1	2	
LSV from right:			2	1	1	2	7	2	4	3	16	
freq. left string:	150	16	5	4	4	4	4	4	4	4	4	
freq. right string:			3	3	3	4	15	17	36	39	129	150
multil. bigram left:				0.0	0.0	0.0	0.1	0.0	0.2	0.2	0.6	
multil. trigram left:					0.0	0.0	0.0	0.0	0.0	0.1	0.3	
multil. bigram right:		0.0	0.0	0.0	0.1	0.0	0.0	0.2	0.6	0.2		
multil. trigram right:		0.0	0.0	0.0	0.0	0.0	0.1	0.3	0.5			
bigram weight:			0.0	0.0	0.0	0.1	0.0	0.2	0.2	0.7	0.3	
final score left:			0.2	0.4	0.8	0.9	1.0	0.8	0.8	0.2	0.4	
final score right:			2.0	1.0	0.8	0.5	**6.3**	0.7	2.2	0.2	**3.0**	

Table 1: Depicting the LSV based algorithm for the example of the German word form *glück-lich-e*. Weights were rounded and the given scores and weights refer to the transition to the left of the letter.

lap factor $(36/39*1.0+36/129*0.6)/(1.0+0.6) = 0.68$ and the bigram weight is 0.2. Thus, the result is $4*0.68*(1.0-0.2) = 2.2$. After applying the same method (but left to right changed) in order to compute the final left score, the final overall score for a given transition is the maximum of either the left or the right score.

The final scores can then be interpreted as representing morpheme boundaries. There are various possibilities to interpret such scores. In this first prototype, a simple threshold has been introduced, as another free parameter. All final scores above the threshold are considered to mark morpheme boundaries and the words are then segmented using these boundaries.

The difference between the final left and right score can be used in order to classify the morphemes. If the right score is higher than the left score, then the morpheme discovered to the right is probably a suffix and a prefix otherwise. This topic is subject to further research and an evaluation is not yet available. I am inclined to try to use a model such as described in (Creutz & Lagus 05) for a more proficient tagging of the categories.

3.1 Clustering of word forms of the same lemma

The simple detection of affixes described in the previous section can be used for the clustering of word forms belonging to the same lemma. In fact, this task can be reformulated as a retrieval task: for a given input word form A, retrieve all word forms of the same lemma.

The first step is to identify and remove the affixes of a given word form based on the detected morpheme boundaries. In the example *glückliche* the stem *glück* remains. The second step is to remove these and all trailing affixes from all words in the context. Thus, if the suffix *-lich-* was detected, then the removal of it and all trailing from the word form *glüklichen* results in the string *glück*. The removals are only temporary in order to detect which word forms have the same stem.

After retrieving all word forms with the same stem, the initial word forms can be printed out as a result of the word form retrieval algorithm. Additionally, since the use word form set contains only contextually similar words, it is also possible to print out all word forms whose stems differ only in one letter, which might help to detect stem alternations. This is a further side effect of the algorithm which has to be investigated in detail.

4 Evaluation

There are almost as many different evaluation methods as there are algorithms for any of the tasks of morpheme identification, morphologic segmentation and lemma to word form clustering. Since one of the two main goals of the algorithm described in this paper is to produce correct morpheme segmentations, an evaluation will be provided which measures the accuracy and recall of finding proper morpheme boundaries. However, most algorithms such as (Goldsmith 01) and (Schone & Jurafsky 01) provide an evaluation which measures the accuracy and productiveness of word form retrieval. Therefore the precision and recall numbers provided below cannot be compared to the evaluations of the cited algorithms. In general, it would be necessary to organize a standardized evaluation framework such as SENSEVAL 2 (SENSEVAL 01) for

the word sense disambiguation task. This framework should comprise several corpora of raw text of typologically distinct languages along with a list of both morphologically correct segmentations for that language and lemmatizations. Since I will give an evaluation based on the German and English language and the gold standard used will be the German and the English part of CELEX (Baayen *et al.* 95), it is, for example, quite difficult to tell the relation between this algorithm and the one described by (Creutz & Lagus 04), which was tested on Finnish and English. Providing more complete and comparable evaluations as well as the word form clustering algorithm will be the focus and direct consequence of this work.

As mentioned above, the languages used to evaluate the algorithm were German and English. I used the corpora available from (Quasthoff 98). The German part contained about 24 million sentences and the English corpus contained 13 million sentences. The gold standard from which information about word form stems and correct morphology segmentation has been acquired was CELEX (Baayen *et al.* 95). The computation of the neighbourhood cooccurrences and the similarities based on them takes up by far the most computation time (several days on a modern PC) due to the huge amounts of sentences. The computation of similarity has been optimized so that not every word was compared with every other: cues from sentence cooccurrences have been used in order to single out candidates of words which might have some neighbour cooccurrences in common. The computation time of the LSV based algorithm once the similarity data is available is negligible. The evaluation was performed on the most frequent 20.000 word forms.

In the evaluation, the overlap between the manually tagged morpheme boundaries and the computed ones is measured. Precision is the number of found correct boundaries divided by the total number of found boundaries. Recall is the number of found correct boundaries divided by the total number of boundaries present in the gold standard. Thus, a complex word analyzed by the algorithm as *ent-zünde-t* would have one correctly detected boundary for the prefix *ent-* and one wrongly detected boundary, because the correct analysis would be *ent-zünd-et*. But according to CELEX both are wrong because this word is not analyzed in CELEX. Such cases were, of course, excluded from the evaluation. Table 2 gives an overview of both precision and recall for three different threshold settings as well as the most frequent prefixes and suffixes found in the data.

	German	English	Prefixes	Suffixes
threshold t=3			ver-	-en
Prec.	75.59	61.80	be-	-e
Rec.	44.83	29.02	ge-	-t
threshold t=4			Ver-	-er
Prec.	79.88	62.97	Be-	-ung
Rec.	32.48	21.00	un-	-s
threshold t=6			ein-	-es
Prec.	83.19	66.02	Bundes-	-lich
Rec.	15.24	11.31	aus-	-te

Table 2: Precision and recall of morpheme boundary detection for various threshold settings for both corpora and the most frequent pre- and suffixes for the German corpus only.

As can be seen, precision cannot be raised much by increasing the threshold, but recall decreases significantly when doing so. An error analysis shows that over 50% of 'errors' according to CELEX were not errors and most of the other errors are at least arguable. For example, in most languages the gender marking is being considered as a suffix. In German, because of the absence of neutrum and masculinum suffixes, the femininum suffix *-e* is not considered to be a suffix even if there are word forms with the same stem and without this 'suffix', such as *Schule* and *schulisch*. Consequently, all occurrences of the femininum suffix are marked as wrong according to CELEX.

Even worse, the rather low 61.80% − 66.02% precision of the English evaluation results from the fact that the algorithm would always analyze words like *lured* as *lur-ed* instead of *lure-d* according to CELEX. Another way to see this, however, is that there is a deletion which occurs because the past tense *-ed* would produce a double *ee* in the word form *lureed* which is phonologically unsound. Over 58% of all the errors of the English evaluation are due to this problem - thus if it would not count as a mistake (by adding some kind of deletion detection algorithm, perhaps based on semantics), then precision would be around 85%. Since there are more examples like this (*plopp-ed* according to the algorithm and wrong according to CELEX but this is an addition again due to phonological reasons), the es-

timated precision of the algorithm in general lies somewhere between $90\% - 95\%$. Other common sources of 'errors' are words of foreign origin, especially Latin words in the two evaluated languages.

5 Conclusions

This study presents a method which performs morpheme analyses of word forms of a given language based on a corpus of raw text. The results shown are competitive and it has been shown that the algorithm can be improved in many different ways. Possible enhancements include iterative applications of the algorithm while utilizing knowledge such as affix frequency distributions or the affixes found in earlier steps.

One particular advantage of the described algorithm is that the intermediary results, namely the sets of similar words based on neighbourhood cooccurrences, can be used to explain the results obtained by the algorithm. In fact, the algorithm works in such a way that the decisions it makes are grounded directly in the rules of morphology such as e.g. that a morpheme is a unit which can be replaced by another morpheme in order to produce another existing word form. Therefore, if the algorithm makes a seemingly wrong decision (according to CELEX) such that e.g. the word *Virologe* has the suffix *-ologe*, it is possible to produce the following explanation from the available data: Not only do all words appearing in similar contexts have this suffix (e.g. *Biologe*), but also there are almost no words that have the suffix *-loge* without the preceding *o*. The few words remaining (e.g. *Kataloge* (catalogues)) usually have no similarity (using a word similarity algorithm) with the words ending in *-ologe*. This makes the algorithm a suitable tool for a linguistic analysis of the morphology of an unknown language.

References

(Argamon *et al.* 04) Shlomo Argamon, Navot Akiva, Amihood Amir, and Oren Kapah. Efficient unsupervized recursive word segmentation using minimun desctiption length. In *Proceedings of Coling 2004*, Geneva, Switzerland, 2004.

(Baayen *et al.* 95) R. Harald Baayen, Richard Piepenbrock, and Léon Gulikers. *The CELEX lexical database (CD-ROM)*. Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, http://www.ldc.upenn.edu/Catalog/ CatalogEntry.jsp?catalogId=LDC96L14, 1995.

(Baroni 03) Marco Baroni. Distribution-driven morpheme discovery: A computational/experimental study. *Yearbook of Morphology*, pages 213–248, 2003.

(Bordag 05) Stefan Bordag. Algorithms extracting linguistic relations and their evaluation. In *preparation*, 2005.

(Brent *et al.* 95) Michael Brent, Sreerama K. Murthy, and Andrew Lundberg. Discovering morphemic suffixes: A case study in MDL induction. In *5th International Workshop on Artificial Intelligence and Statistics*, Ft. Lauderdale, Florida, 1995.

(Creutz & Lagus 04) Mathias Creutz and Krista Lagus. Induction of simple morphology for highly inflecting languages. In *Proceedings of 7th Meeting of the ACL Special Interest Group in Computational Phonology (SIGPHON)*, pages 43–51, Barcelona, July 2004.

(Creutz & Lagus 05) Mathias Creutz and Krista Lagus. Unsupervised morpheme segmentation and morphology induction from text corpora using morfessor 1.0. In *Publications in Computer and Information Science, Report A81*. Helsinki University of Technology, March 2005.

(Creutz 03) Mathias Creutz. Unsupervised segmentation of words using prior distributions of morph length and frequency. In *Proceedings of ACL-03, the 41st Annual Meeting of the Association of Computational Linguistics*, pages 280–287, Sapporo, Japan, July 2003.

(de Marcken 95) Carl de Marcken. The unsupervised acquisition of a lexicon from continuous speech. Memo 1558, MIT Artificial Intelligence Lab, 1995.

(Deerwester *et al.* 90) Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshmann. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

(Déjean 98) Hervé Déjean. Morphemes as necessary concept for structures discovery from untagged corpora. In D.M.W. Powers, editor, *NeMLaP3/CoNLL98 Workshop on Paradigms and Grounding in Natural Language Learning, ACL*, pages 295–299, Adelaide, January 1998.

(Dunning 93) T. E. Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74, 1993.

(Frakes 92) William R. Frakes. *Stemming Algorithms*, chapter 8, pages 131–160. Frakes und Baeza-Yates, 1992.

(Goldsmith 00) John Goldsmith. Linguistica: An automatic morphological analyzer. In Arika Okrent and John Boyle, editors, *The Proceedings from the Main Session of the Chicago Linguistic Society's Thirty-sixth Meeting*, Chicago, 2000. Chicago Linguistic Society.

(Goldsmith 01) John Goldsmith. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27(2):153–198, 2001.

(Hafer & Weiss 74) Margaret A. Hafer and Stephen F. Weiss. Word segmentation by letter successor varieties. *Information Storage and Retrieval*, 10:371–385, 1974.

(Harris 51) Zellig S. Harris. *Structural Linguistics*. University of Chicago Press, Chicago, 1951.

(Harris 55) Zellig S. Harris. From phonemes to morphemes. *Language*, 31(2):190–222, 1955.

(Kazakov 97) Dimitar Kazakov. Unsupervised learning of naïve morphology with genetic algorithms. In A. van den Bosch, W. Daelemans, and A. Weijters, editors, *Workshop Notes of the ECML/MLnet Workshop on Empirical Learning of Natural Language Processing Tasks*, pages 105–112, Prague, Czech Republic, April 1997.

(Quasthoff 98) Uwe Quasthoff. Projekt: Der Deutsche Wortschatz. In Gerhard Heyer and Christian Wolff, editors, *Tagungsband zur GLDV-Tagung*, pages 93–99, Leipzig, March 1998. Deutscher Universitätsverlag.

(Schone & Jurafsky 01) Patrick Schone and Daniel Jurafsky. Language-independent induction of part of speech class labels using only language universals. In *Workshop at IJCAI-2001*, Seattle, WA., August 2001. Machine Learning: Beyond Supervision.

(SENSEVAL 01) SENSEVAL. *Second International Workshop on Evaluating Word Sense Disambiguation Systems*. Toulouse, France, http://www.sle.sharp.co.uk/senseval2/, 5-6 July 2001.

An interactive environment for creating and validating syntactic rules

Panagiotis Bouros*, Aggeliki Fotopoulou, Nicholas Glaros
Institute for Language and Speech Processing (ILSP),
Artemidos 6 & Epidavrou,
GR-151 25, Athens, Greece
{pbour,afotop,nglaros}@ilsp.gr

Abstract

Syntactic analysis is a key component in many Natural Language Processing applications. This is especially true when considering advanced spelling checkers, where the usage of contextual rules at the syntax level can significantly increase the spelling error detection and correction capability of such systems. The advantage of the contextual approach over the isolated-word approach becomes more clear in morphologically rich languages, in which it is very likely that a spelling error free word can, in fact, represent a misspelled word within a given context. In such cases, even a minimal set of syntactic rules can be proved very effective in obtaining high spelling performance levels. However, determining a consistent set of rules for spelling checking purposes is not always a straightforward task. In this paper, we design and implement an interactive linguistic environment for managing the grammatical and syntactic resources of an advanced spelling checker system for Greek.

1 Introduction

Checking human free text has always been a very important and challenging issue to address. There is a lot of work already done for lexical analysis of text in order to identify and tag, using dictionaries, the lexical units contained in a text.

This word-by-word approach is quite efficient for the automatic check of spelling errors, which render a word totally invalid or non-existent. This type of spelling errors is most prominent in languages with poor morphology. However, in highly inflectional languages, it is very common that a spelling error in a lexical type produces another lexical type, which is valid on its own. For example, in the sentence: "I listens to the music.", there are no misspelled words on their own, yet, the syntax is still incorrect, because the verb type, according to the subject, should be "listen".

Clearly, the latter type of spelling errors is totally missed out by the word-by-word approach as well as by all spelling checkers that rely on it. On the contrary, this is precisely not the case when a rule-based syntactic analysis of every phrase of the text being checked (phrase-by-phrase approach) is employed. Resolving this kind of spelling errors takes more than simply going through a lexicon to match a given token. This leads us to advanced spelling systems, the design and implementation of which is still challenging and necessary for morphologically rich languages.

Building advanced spellers, based on statistical approaches, may require the use of a corpus in order to extract n-grams (Knight 99), (Beaujard & Jardino 99) (in most cases up to 3-grams) and then apply statistical models to compute the occurrence probability of the n-grams and of the corresponding parent sentence. Of course, if the lexical pattern of a sentence is correct, but never occurred before, there lies the problem of mischaracterizing it as incorrect. This problem is only partially addressed by smoothing techniques.

On the other hand, the fundamentals of a syntactic analysis framework are a morphological lexicon and a set of syntactic rules. Each rule of the set defines a number of word environments, i.e. grammatical patterns, which are formed by acceptable combinations of grammatical categories. In this manner, after tagging the words of a given sentence, the checking procedure attempts to verify the presence of the defined grammatical patterns on specific segments of the tagged sentence, thus concluding on possible rule-violations owning to spelling errors.

The work presented in this paper is directly connected with the syntactic analysis. In particular, we tackle the problem of creating, managing, monitoring and testing syntactic rules from within an easy and user-friendly interactive environment. For this purpose, we have designed and implemented a special tool for the graphical, most of all, creation of rules for the advanced

* Current affiliation is National and Kapodistrian University of Athens (NKUA), Department of Informatics and Telecommunications.

spelling checker of (ILSP) (Symfonia) (Stathis & Carayannis 99) and, moreover, for monitoring their application and interaction on existing text corpora. Symfonia employs a context-based spelling check technique, in addition to the isolated word-based approach. Cases where words sound similarly but are spelt differently, e.g. /δόsis/ "δόσης" noun feminine (nominative of plural or genitive of singular) : payment and /δόsis/ "δώσεις" verb (2nd person of singular in Future Simple or 2nd person of singular in Subjunctive) : give, and in which the spelling depends on the grammatical identity of the word, can be resolved.

The rest of the paper is organized as follows. Section 2 discusses the objectives of the proposed environment, while section 3 presents its architecture. Section 4 describes the working environment of the tool and lists its functional features. Section 5 demonstrates a real world scenario of using the tool. Finally, in section 6 some concluding remarks and prompts for further work are given.

2 Objectives - Specifications

The main purpose of the work presented in this paper is to provide a supportive environment for fast generating a consistent set of syntactic rules optimized for advanced spelling checking processes. Through a user-friendly interface, this tool allows language specialists to create, view, edit, real-time test monitor and validate syntactic rules, while leaving them out of the underlying computer programming technicalities.

As far as rule generation and editing is concerned, the environment provides a graphical rule representation mechanism. We consider that a tree graphical representation is suitable for presenting the word environments, the decision and generally the context of a syntactic rule. Moreover, in order for the tool to be speller technology-independent, we provide an XML (xml)-based mechanism for storing the rule tree representations. Furthermore, the tool automatically transcribes the user-defined rules into ready-to execute speller code (according to the speller being targeted), thus, providing a test-bed for the fast generation of a robust syntax analyzer.

By means of rich enough monitoring information, the system enables the user to evaluate the application of rules either individually or in combination with other user-specified rules. Emphasis is given on the production of a detailed report depicting the lexical analysis of the text, as well as details on the application of the user selected subset of rules, in order to identify or handle potential misusage, conflicts etc.

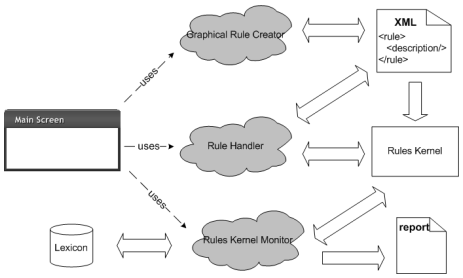


Figure 1: System architecture

3 Architecture

Figure 1 illustrates the architecture of the implemented tool. Each syntactic rule created by the Graphical Rule Creator is stored in an XML document and integrated in the Rules Kernel. The Rules Kernel is an extension of the kernel used by Symfonia speller with extra features for supporting insertion, handling and monitoring of the rules' application. Graphical Rule Creator is also used for editing and updating a syntactic rule. Furthermore, in order to provide additional handling functionality on the Rules Kernel, we have introduced the Rule Handle component.

Finally, the Rules Kernel Monitor is responsible for testing and reporting on the usage of a subset of the rules, integrated into the kernel, across real unformatted text. The monitor procedure relies on the speller's built-in lexicon for the lexical analysis and on the Rules Kernel for syntactic analysis, in order to generate a detailed report.

4 Working Environment - Functionalities

Figure 2 displays the main screenshot of the implemented tool, being the first window interacting with the user. This window consists of the list of rules that are integrated into the kernel. For every rule, its name, description and status are indicated. The status of a rule is either *enabled* or *disabled*, meaning that can be either taken into account by the monitor procedure or not. Moreover, all functionalities of the developed system are available through the menu options and the toolbar icons of this window. A detailed presen-

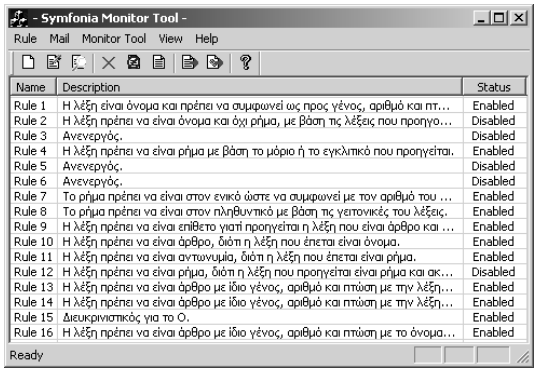


Figure 2: Main screen

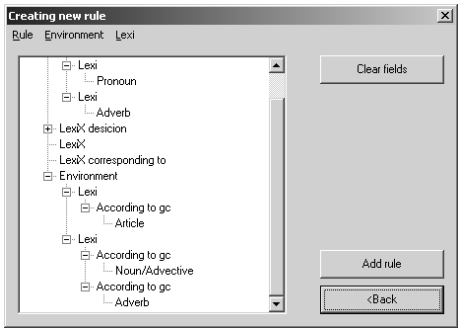


Figure 3: Rule tree

tation of the working environment and the implemented functionalities can be found in (Bouros 05).

4.1 Rule Handling

Rule handling mainly pertains to the management of the Rules Kernel component. Thus, it permits addition of new rules, editing of the definition and of the status of an existing rule or simply its removal from the kernel. All these changes are reflected in the list of Figure 2.

1. **Create a new rule**. In order to create a new syntactic rule the user takes advantage of the rule graphic tree representation presented in Figure 3. Each rule is focused on a single lexi[1] called LexiX.

[1] The term *lexi* (lexis in plural) is used in this paper to denote the set of grammatical characteristics of a word - On the other hand *words* are simply the tokens of a sentence.

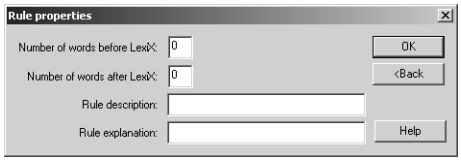


Figure 4: Specifying rule properties

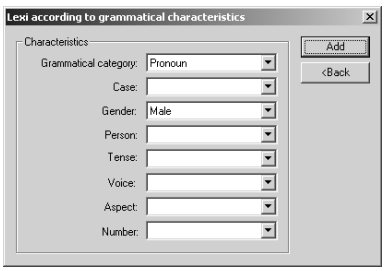


Figure 5: Specifying lexi's grammatical characteristics

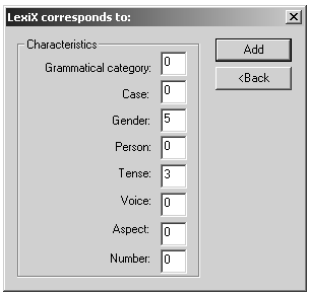


Figure 6: Specifying LexiX correspondence

First of all, the user should provide the description and the explanation of the rule. Explanation can contain parameters, denoted by $x for the LexiX or $+/- number, for a specific word of a sentence. These parameters are replaced by the corresponding words during the rule usage. User should also specify the number of words contained in rule environment before and after the LexiX position. The above rule properties are specified in the rule properties dialog depicted in Figure 4.

Next the user defines the valid combinations of grammatical characterizations, i.e. lexis, for LexiX, as well as the lexi which the new rule should conclude to. The definition of grammatical characteristics of each lexi is done through the dialog in Figure 5. The user can also restrict the application of the rule to a specific set of words. Through the dialog in Figure 6, the user can specify the adjacent words whose grammatical characteristics will be inherited to LexiX.

Finally, the user can specify the alternative environments of the new syntactic rule. Each environment is a set of lexis defined by their

grammatical characteristics (using Figure 5 dialog). The number of lexis contained in each environment must be equal to the total number of words specified in the rules properties dialog in Figure 4.

After completing the definition of the syntactic rule, the user integrates the new rule into the kernel. This is an automatic operation, which also constructs the XML rule file.

2. **Edit an existing rule**. The procedure of editing an existing rule is alike to the one of creating a new rule. Editing starts after the system has parsed the XML rule file and reproduced the tree representation of the rule (Figure 3). The user can modify the rule properties, characteristics and alternative environments and then choose to update the Rules Kernel and the corresponding XML file.

3. **Remove an existing rule**. Removal of an existing rule can be done through the respective menu option or toolbar icon located in the main screen (Figure 2).

4. **Disable/enable an existing rule**. By default, the status of a new rule is set to *enabled*. The status can be altered from the main screen in Figure 2 either to *disabled* or *enabled*.

5. **Export of existing rules**. Apart from XML format, a single or the entire set of the syntactic rules can be exported in a high level programming language code. The user has the option from within the environment to e-mail the resulted source code to the programmers group of the targeted syntactic speller.

4.2 Monitor

Efficient syntactic rules-based spell checking leads to the problem of generating and choosing syntactic rules that on the one hand optimize the performance of the spelling checker engine and on the other constitute a consistent set of rules. In trying to resolve this problem, there are many cases when a rule or a number of rules should be checked against a different set of rules, for identifying and minimizing potential rules conflicts and insufficiencies.

For this purpose, the system provides a monitor functionality for the evaluation of Rules Kernel

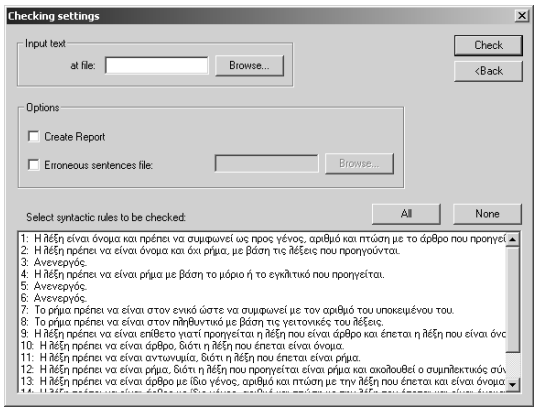


Figure 7: Checking procedure settings

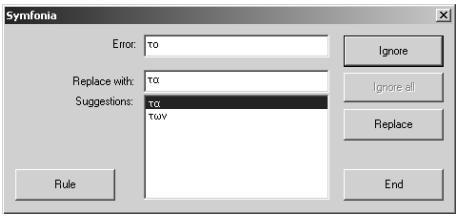


Figure 8: Interactive check dialog

while being on text documents. The system also takes advantage of the lexicon of (Symfonia) in order to perform the additional grammatical and lexical analysis required.

Rules checking can be done either interactively or automatically. In the first case, the user has to select one of the automatically generated system suggestions that attempt to correct the syntax error encountered. In the second case, the system by default adopts the first suggestion.

Nevertheless, in both cases the starting point is the same. Figure 7 presents the settings dialog of the checking procedure. In this dialog the user specifies the input text containing the sentences that should be checked and the set of syntactic rules that will be used, by picking them out from the rules list on the bottom of the dialog. The list contains all the rules integrated into the Rules Kernel except from the rule checking for simple spelling errors. This is a check that always takes place. Moreover the user can choose if the system will produce a report of the check and a document containing the erroneous sentences. In the latter option, the name of the output document should also be specified.

After having specified the settings, the rules checking begins. The procedure stops when an error is encountered and when in interactive mode.

The user is informed about the spelling mistake by Figure 8 dialog. This dialog is identical to the one used in the Symfonia advanced spelling checker. It denotes the misspelled word and proposes a number of alternative words. The user can either ignore this error or all of its subsequent occurrences, or replace the misspelled word or simply choose to end the checking procedure. In addition, the user can read the explanation of the rule used to detect the error.

A report regarding the checking of the document is produced at the end of the procedure if the user has requested so. The information contained in a report file is sentence-wise organized. In the beginning of the document, there is a list of the rules selected in Figure 7 to be taken into account. Then, for each sentence of the input document and for each error detected, a section is given that contains the grammatical analysis of the sentence words: lemma and grammatical category, the rules used in the checking of this sentence and the one that identified the error. In addition, the report lists the alternatives words proposed by the rule that detected the error, and also in case of an interactive check, it denotes the action of the user taken place in Figure's 8 dialog.

5 Real-World scenario

Let us assume that we wish to solve the ambiguity between the greek words for "more" and "which": "πιο" and "ποιο". Although these two words have the same phonetic transcription /pjo/, the first one is an adverb and the second is a pronoun. We create a syntactic rule with the following environment:

Lexi1 LexiX Lexi2

If LexiX is characterized by the ambiguity "πιο" - "ποιο" and Lexi1 is an article and Lexi2 is either an adjective or a noun or an adverb, then LexiX is an adverb, i.e. "πιο". Figure 3 illustrates the rule tree representing the created rule.

The previous rule resolves the ambiguity by rendering LexiX as an adverb. We can also define another rule for specifying that LexiX should be a pronoun, i.e. "ποιο". The environment of the required rule would be:

LexiX Lexi1 Lexi2 Lexi3 Lexi4 Lexi5

LexiX is "ποιο" if Lexi1 is an article, Lexi2 an adnoun, Lexi3 a noun, Lexi4 a particle and Lexi5 a verb. In addition, some or all of Lexi1, Lexi2, Lexi3 and Lexi4 can be missing.

6 Conclusion

Designing highly robust proofing tools for inflectional languages (Amaral *et al.*) is still an open issue. One fundamental approach to address this problem is to use grammar and syntax rules-based checking on a phrase-by-phrase basis. This, in turn, leads us to the problem of generating and choosing syntactic rules that not only optimize the performance of the spelling checker engine, but they also constitute a consistent set of rules. To this end, a purely linguistic tool was developed that lets language knowledgeable but computer programming unaware people to devise, build and test in real-time spelling checking processes whatever grammar and syntax rules they like, by means of graphical tree representations. At the same time plenty of monitoring information is provided by user-friendly interface in all phases of every syntactic rule life-cycle. Testing of the rules on large text corpora is also supported. The tool was implemented for the Greek language and for the Symfonia speller of ILSP. The environment has proven its value (e.g. rapid rule creation, efficient identification of potential rules conflicts etc.) after having thoroughly tested and evaluated by ILSP linguists group. Further work can be focused in converting the tool to a platform that can accommodate other spellers and support other morphologically rich languages.

References

(Amaral *et al.*) Carlos Amaral, Helena Figueira, Afonso Mendes, Pedro Mendes, and Claudia Pinto. A Workbench for Developing Natural Language Processing Tools.

(Beaujard & Jardino 99) Christel Beaujard and Michele Jardino. Classification of not labelled words by statistical methods. *Mathematics, Informatics and Social Science*, (147):7–23, 1999. in french.

(Bouros 05) Panagiotis Bouros. Technical report, Symfonia Monitor Tool manual, 2005. in greek.

(ILSP) ILSP. Institute of Language and Speech Processing, http://www.ilsp.gr.

(Knight 99) Kevin Knight. A Statistical MT Tutorial Workbook. prepared in connection with the JHU summer workshop, April 1999.

(Stathis & Carayannis 99) C. Stathis and G. Carayannis. title (in greek). In *2nd ELETO Conference: Hellenic Language and Terminology*, 1999. in greek.

(Symfonia) Symfonia. An intelligent spelling checker, http://www.ilsp.gr/correct.html.

(xml) xml. Extended Markup Language, http://www.w3.org/XML/.

A Core Model of Systemic Linguistic Analysis

Sylviane Cardey and **Peter Greenfield**
Centre Tesnière, Université de Franche-Comté
30 rue Mégevand, F-25030 Besançon cedex, France
{sylviane.cardey,peter.greenfield}@univ-fcomte.fr

Abstract

This paper presents a core model of systemic linguistic analysis involving materialising the relation between a linguistic system's component variant and canonical systems. The model described is a relation between equivalence relations over each of these component systems. The precise structure of this relation and the component systems enables objective evaluation required in safety critical applications such as case based validation, traceability and the verification that a given linguistic analysis core instance is well formed.

1 Introduction

Systemic linguistic analysis, which is due to Sylviane Cardey (Cardey 87), is based on the postulate that a language can be segmented into individual systems based on the observation that such systems influence each other. All levels of language analysis can be described in this way (lexis, syntax, morpho-syntax, semantics, morpho-semantics and others) (Cardey & Greenfield 02; Cardey & Greenfield 05). Systemic linguistic analysis has been applied to many applications such as disambiguated parts of speech tagging - the Labelgram system (Cardey & Greenfield 03), machine translation of 'far' language couples (including anaphoric reference and zero anaphor processing) (Cardey et al. 03; Cardey et al. 04a), grammar checking and correcting (including noun phrase identification) (Cardey et al. 04c), and for safety critical applications where evaluation ability is required in the form of validation and traceability as in controlled languages, for example cockpit alarm message vocabulary (Spaggiari et al. 05) and also in the machine translation of medical protocols (Cardey et al. 04b). An initial study determining the modelling constraints has been reported in (Greenfield 03).

In this paper we present a model of the core of systemic linguistics which involves the system of variants and canonicals.

2 Systemic linguistic analysis

Systemic linguistics methodology consists in analysing a linguistic system in component systems as follows:

- Sv: a system representing the variants;
- Sc: another system which is recognisably canonical;
- Ss: a 'super' system which puts the two systems Sv and Sc in relation with each other.

2.1 An example linguistic system

The example linguistic system that we describe in this paper concerns the doubling or not of the final consonant in English words before the endings -ed, -ing, -er, -est, -en. For example we observe the variants 'modeling' and 'modelling' for the canonical form 'model'. This system, *Doubling_or_not*, comprises the following component systems:

- $Sv_{Doubling\_or\_not}$, the words concerned in their derived or inflected form, the variants; e.g. 'modeling', 'modelling', 'frolicked'
- $Sc_{Doubling\_or\_not}$, the words concerned in their basic form, that is their canonical form; e.g. 'model', 'frolic'
- $Ss_{Doubling\_or\_not}$, the super system relating systems $Sv_{Doubling\_or\_not}$ and $Sc_{Doubling\_or\_not}$.

3 Establishing a systemic linguistics analysis

This requires modelling system Ss. To do so for some application, the linguist establishes two categorisations:

i. Firstly a 'non-contextual' (nc) categorisation of the canonical forms in relation with the variant forms in isolation, the context being limited to just the canonical and variant forms themselves. For *Doubling_or_not*, this categorisation can thus only depend on the form of the words concerned, this being an aspect of morphology.
ii. Secondly an 'in-context' (ic) categorisation of the canonical forms in relation with the variant forms in terms of the linguistic contexts of the variant forms. The systemic analysis reveals precisely what other internally related linguistic systems are involved.

For systems Ss, Sv and Sc, let S be a set structure modelling super system Ss, let V be the set of variant forms, C the set of canonical forms, and let VC be the binary ordered relation between V and C corresponding to system Ss.

Each of the above two categorisations, 'nc' and 'ic', can be modelled by a partition on VC; we have P_{nc} and P_{ic}. Given that we have partitions, from the fundamental theorem on equivalence relations, it follows that there exist two corresponding equivalence relations E_{nc} and E_{ic} on VC. Each equivalence class in respectively E_{nc} and E_{ic} corresponds to a distinct categorisation or case. We model system Ss, the super system relating systems Sv and Sc, by means of the ordered binary relation S between the equivalence relations E_{nc} and E_{ic}, and similarly S^{-1} between E_{ic} and E_{nc}.

We can subsequently model functions for finding the canonical element(s) corresponding to a variant element and vice-versa or others such as finding the (name of the) canonical equivalence class for a variant element as for example in parts of speech tagging. Furthermore, because we have a precise structure for S, we can verify that a given linguistic analysis representation is well formed. In respect of equivalence relations and when the linguistic

domain consists of strings, we note that every finite automaton induces a right invariant equivalence relation on its input strings (Hopcraft & Ullman 69, pp. 28-30).

3.1 Establishing the partitions

We now turn to how to establish the partitions P_{nc} and P_{ic}. Here the linguist can adopt either a proof theoretic or a model theoretic approach – or indeed combine these (Greenfield 97). The former is allied to the development of an algorithm, the latter being case (truth table) based. In any case, if the goal is an automated functional application, an eventual algorithm is typically necessary; see for example (Humby 73; Dial 70).

We illustrate an algorithmic analysis with *Doubling_or_not*. Using a binary divide and conquer approach the linguist determines an algorithm for each of P_{nc} and P_{ic}; many equivalent forms of source representation are available and have been implemented, including automated translation between them (Cardey & Greenfield 92). As to which partition P_{nc} or P_{ic} to start with and even whether it is possible or feasible to sequence the establishment of the two partitions depends on various factors such as:

- What prior knowledge is available. For example existing classifications as for example a parts of speech tag set;
- The simplicity or otherwise of organising observations including their extraction. For example in machine translation and in concept mining, concepts which will constitute the canonical forms are themselves often revealed during the analysis process at the same time as the contexts indicating their presence as variants in the language.

3.2 The non-contextual analysis

Figure 1 shows a representation of a result of this analysis, that is, system Sc *Doubling_or_not*.

System Sc *Doubling_or_not* – Non-contextual (nc) analysis			
Conditions			
Id	**Condition text**		
vc	word with final consonant in English taking -ed, -ing, -er, -est, -en		
vcd	Doubling of the final consonant		
k	The words terminating in –ic take –ck		
Operators			
Id	**Operator text**		
N	No doubling of the consonant		
D	Doubling of the consonant		
K	The words terminating in –ic take –ck		
Algorithm with justifications			
Line #	**Level**	**Condition->Operator**	**Variant ->Canonical**
0	0	vc -> N	'feeling -> 'feel
1	1	vcd -> D	'runner -> 'run
2	2	k -> K	'frolicked -> 'frolic

Figure 1: Representation of Sc*Doubling_or_not*

The conditions and the operators are abstraction predicates over *VC*. The algorithm's entry condition id is vc, the abstraction predicate for the set *VC*, this being the linguistic domain under analysis. The algorithm is represented in a particular fashion, due to Sylviane Cardey (Cardey 87) and which suits the partitioning. The algorithm is shown in Figure 2 in conventional 'if then (else) fi' representation.

if condition vc is true
then **if** condition vcd is true
then **if** condition k is true
then operator K
else operator D
fi
else operator N
fi
fi

Figure 2: Conventional representation of algorithm (nc)

Here it is to be observed that the 'ifs' other than the outermost are of the type 'if then else fi'. This is because in general we require that systemic linguistic analyses be exhaustive but not over-generative; the algorithm covers exactly the linguistic domain *VC* under analysis. This 'if then (else) fi' structure is equivalent to a binary tree rooted by a single noded unary tree; hence as the algorithm representation in Figure 1 shows, the number of lines equals the number of nodes (condition appearances) equals the number of leaves (operator appearances).

We can write the model theoretic model of Sc with, for convenience, each interpretation in the same order as in the algorithm. Here the conditions and operators are predicates (Figure 3) where the items in bold correspond to those in the associated algorithm line. We formulate the model as a single proposition, hence the disjunction of the interpretations.

0. **vc** $\wedge \neg$ vcd $\wedge$ **N** $\wedge \neg$ D $\wedge \neg$ K $\vee$
1. vc $\wedge$ **vcd** $\wedge \neg$ k $\wedge$ $\neg$ N $\wedge$ **D** $\wedge \neg$ K $\vee$
2. vc $\wedge$ vcd $\wedge$ **k** $\wedge$ $\neg$ N $\wedge \neg$ D $\wedge$ **K**

Figure 3: Model theoretic model of Sc*Doubling_or_not*

In any interpretation we observe that only one operator is positive. However, there exist linguistic systems with more than one variant for a given interpretation of the conditions; for example for the system of the plural of the French adjective, the canonical (singular) form 'austral' has variants (plurals) 'australs' and 'austraux'. Thus for the 'austral' example there are two interpretations in the model but with condition formulae that are equal.

Concerning the algorithm and its model:

- the model can be generated from the algorithm;
- in general there are many functionally identical algorithms that can be generated from a model (the conditions and operators resting unchanged). Let P_N be the number of algorithms that can be generated from a model with N conditions, where all the 2^N possible interpretations are present. We have

  $P_N = N \times (P_{N-1})^2$

  where $P_2 = 2$ (Humby 73, pp. 32-34).

  In consequence alternative functionally identical algorithms can be generated to meet specific needs such as speed optimisation in automated applications.

Let *X* and *Y* be sets and their predicates of abstraction be x and y respectively. We have:

- x $\wedge$ y corresponds to: $X \cap Y$
- x $\wedge \neg$ y corresponds to: $X \setminus Y$

where \ is set difference. From the model, set expressions can thus be formulated corresponding to the conditions component of each interpretation as shown in Figure 4. Here, the set abstractions of the conditions are in italic

capitals, and a Dewey Decimal based notation is used to identify each model interpretation's conditions component.

Non-contextual (nc) analysis			
Algorithm		Set name	Set formulation
Line #	Level		
0.	0	*VCnc*.0	***VC*** \ *VCD*
1.	1	*VCnc*.0.0	*VC* ∩ ***VCD*** \ *K*
2.	2	*VCnc*.0.0.0	*VC* ∩ *VCD* ∩ ***K***

Figure 4: Set formulation of conditions components (nc)

The sets *VCnc*.0, *VCnc*.0.0 and *VCnc*.0.0.0 partition the set *VC*. (Let X, Y, Z be sets; partition is defined as: $\{X, Y\}$ partition $Z \Leftrightarrow X \cap Y = \varnothing \wedge X \cup Y = Z$.) We observe:

- The intersection of the sets *VCnc*.0, *VCnc*.0.0 and *VCnc*.0.0.0 is the empty set:
  $\cap\{$ *VCnc*.0, *VCnc*.0.0, *VCnc*.0.0.0$\} = \varnothing$
- The union of the sets *VCnc*.0, *VCnc*.0.0 and *VCnc*.0.0.0 is the set *VC*:
  $\cup\{$ *VCnc*.0, *VCnc*.0.0, *VCnc*.0.0.0$\} =$ *VC*

Thus P_{nc} = {*VCnc*.0, *VCnc*.0.0, *VCnc*.0.0.0}. Being a partition, the algorithm has determined an equivalence relation E_{nc} over *VC*, each of the sets *VCnc*.0, *VCnc*.0.0 and *VCnc*.0.0.0 is an equivalence class, and the number of equivalence classes, that is the index of the equivalence relation, i_{nc}, is 3 (the number of lines in the algorithm):

$\# P_{nc} = 3$

Being a partition, with each equivalence class being associated directly with a line in the algorithm allows us to include a justification for each class, that is, case; these are shown in Figure 1 in the column **Variant -> Canonical**. In *Doubling_or_not*, the apostrophes in the justifications are English stress indicators.

Including such case justifications assists evaluation processes such as validation and is essential in safety critical applications. In automated applications, automated case based regression validation testing can be implemented. Such case justifications, precisely because they are case based, can serve as the basis for evaluation benchmarks.

Now consider the sets that are defined during the execution of the non-contextual algorithm (Figure 5).

Non-contextual (nc) analysis			
Algorithm		Set name	Set formulation
Line #	Level		
0.	0.	*VC'nc*.0	***VC***
1.	1.	*VC'nc*.0.0	*VC* ∩ ***VCD***
2.	2.	*VC'nc*.0.0.0	*VC* \ *VCD* ∩ ***K***

Figure 5: Sets that are defined during the execution of the algorithm (nc)

The sets so defined form a collection of proper sub-sets (Figure 6).

Non-contextual (nc) analysis	
Algorithm line #	Parent set ⊃ Line set
1.	*VC'*.0 ⊃ *VC'*.0.0
2.	*VC'*.0.0 ⊃ *VC'*.0.0.0

Figure 6: Proper sub-set structure (nc)

3.3 The in-context analysis

The in-context (ic) analysis for deriving Sv follows the same approach as the non-contextual analysis giving Sc, but with the addition that the Sv thus revealed is put in relation with Sc, thus resulting in the derivation of the super-system Ss - here $Ss_{Doubling\_or\_not}$ (Figure 7). This representation has the same structure as Figure 1: the representation of $Sc_{Doubling\_or\_not}$, and provides the same capabilities, for example case based justifications. Figure 7 also shows the dynamic tracing of an application of the analysis, with one of the possible solutions of the particular problem of whether the variant **model+ing** is spelt **modelling** or **modeling**; the conditions, the algorithm branches visited, and the justifications are shown variously as **true**, **false** and *undefined* (i.e. not visited). Inspection of the conditions and operators in Figure 7 shows that whilst the operators concern solely morphology, the conditions range over phonetics, phonology, lexis, morphology, syntax, morpho-syntax, semantics, morpho-semantics, register (regional variation). Furthermore certain conditions can be automated simply, for example lexical conditions naming lexical items and the morphological conditions 'terminated by ...'.

$Ss_{Doubling\_or\_not}$ - In-context (ic) analysis			
Conditions			
Id	**Condition text**		
vc	**word with final consonant in English taking –ed, -ing, -er, -est, -en**		
a	**word of a syllable of the form C-V-C**		
b	**terminated by C—V--C or by C-V(pronounced)-V(pronounced)-C**		
c	**last syllable accented**		
d	**terminated by –l or –m**		
e	**used in England**		
f	*"(un)parallel"*		
g	*"handicap, humbug"*		
h	*"worship, kidnap"*		
i	*terminated par –ic*		
j	*"wool"*		
Operators			
Id	**Operator text**		
N	**No doubling of the consonant**		
D	**Doubling of the consonant**		
K	**The words terminating in –ic take –ck**		
Algorithm with justifications			
Line #	**Level**	**Condition->Operator**	**Variant -> Canonical**
0	**0**	**vc -> N**	**'feeling -> 'feel**
1	**1**	**a -> D**	**'runner -> 'run**
2	**1**	**b -> N**	**'answerer -> 'answer**
3	**2**	**c -> D**	**dis'tiller -> dis'til**
4	**2**	**d -> N**	**'modeling -> 'model**
5	**3**	**e > D**	**'modelling -> 'model**
6	**4**	**f -> N**	**(un)'paralleled -> (un)'parallel**
7	*2*	*g -> D*	*'handicapped -> 'handicap*
8	*2*	*h -> N*	*'worshiped -> 'worship*
9	*3*	*e -> D*	*'worshipped -> 'worship*
10	*2*	*i -> K*	*'frolicked -> 'frolic*
11	*1*	*j -> N*	*'woolen -> 'wool*
12	*2*	*e -> D*	*'woollen -> 'wool*

Figure 7: Representation of $Ss_{Doubling\_or\_not}$

The index i_{ic} of the equivalence relation, E_{ic}, determined by the algorithm, is 13.

Figure 8 shows the model theoretic model of the super system $Ss_{Doubling\_or\_not}$.

0.	**vc** ∧ ¬ a ∧ ¬ b ∧ ¬ j ∧	**N** ∧ ¬ D ∧ ¬ K ∨
1.	vc ∧ **a** ∧	¬ N ∧ **D** ∧ ¬ K ∨
2.	vc ∧ ¬ a ∧ **b** ∧ ¬ c ∧ ¬ d ∧¬ g ∧ ¬ h ∧¬ i ∧¬ D ∧ ¬ K ∧ **N** ∨	
3.	vc ∧ ¬ a ∧ b ∧ **c** ∧	¬ N ∧ **D** ∧ ¬ K ∨
4.	vc ∧ ¬ a ∧ b ∧ ¬ c ∧ **d** ∧ ¬ e ∧	**N** ∧ ¬ D ∧ ¬ K ∨
5.	vc ∧ ¬ a ∧ b ∧ ¬ c ∧ d ∧ **e** ∧ ¬ f ∧	¬ N ∧ **D** ∧ ¬ K ∨
6.	vc ∧ ¬ a ∧ b ∧ ¬ c ∧ d ∧ e ∧ **f** ∧	**N** ∧ ¬ D ∧ ¬ K ∨
7.	vc ∧ ¬ a ∧ b ∧ ¬ c ∧ ¬ d ∧ **g** ∧	¬ N ∧ **D** ∧ ¬ K ∨
8.	vc ∧ ¬ a ∧ b ∧ ¬ c ∧ ¬ d ∧ ¬ g ∧ **h** ∧ ¬ e ∧	**N** ∧ ¬ D ∧ ¬ K ∨
9.	vc ∧ ¬ a ∧ b ∧ ¬ c ∧ ¬ d ∧ ¬ g ∧ h ∧ **e** ∧	¬ N ∧ **D** ∧ ¬ K ∨
10.	vc ∧ ¬ a ∧ b ∧ ¬ c ∧ ¬ d ∧ ¬ g ∧ ¬ h ∧ **i** ∧	¬ N ∧ ¬ D ∧ **K** ∨
11.	vc ∧ ¬ a ∧ ¬ b ∧ **j** ∧ ¬ e ∧	**N** ∧ ¬ D ∧ ¬ K ∨
12.	vc ∧ ¬ a ∧ ¬ b ∧ j ∧ **e** ∧	¬ N ∧ **D** ∧ ¬ K

Figure 8: Model theoretic model of $Ss_{Doubling\_or\_not}$

Figure 9 shows an extract of the set formulation of the condition components of the in-context model's interpretations.

In-context (ic) analysis			
Algorithm		**Set name**	**Set value**
Line #	**Level**		
0.	0	*VC*.0	***VC****\A\B\J*
5.	3	*VCic*.0.1.1.0	*VC\A∩B\C∩D∩**E**\F*
12.	2	*VCic*.0.2.0	*VC\A\B∩J∩**E***

Figure 9: Set formulation (extract) of condition components (ic)

The sets *VCic*.0, ... *VCic*.0.2.0 partition the set *VC*:

– The intersection of these sets is the empty set:
  ∩{ *VCic*.0, *VCic*.0.0, *VCic*.0.1, *VCic*.0.1.0, *VCic*.0.1.1, *VCic*.0.1.1.0, *VCic*.0.1.1.0.0, *VCic*.0.1.2, *VCic*.0.1.3, *VCic*.0.1.3.0, *VCic*.0.1.4, *VCic*.0.2, *VCic*.0.2.0 } = ∅
– The union of these sets is the set *VC*:
  ∪{ *VCic*.0, *VCic*.0.0, *VCic*.0.1, *VCic*.0.1.0, *VCic*.0.1.1, *VCic*.0.1.1.0, *VCic*.0.1.1.0.0, *VCic*.0.1.2, *VCic*.0.1.3, *VCic*.0.1.3.0, *VCic*.0.1.4, *VCic*.0.2, *VCic*.0.2.0 } = *VC*

Figure 10 shows an extract of the sets that are defined during the execution of the in context algorithm at the point where the last condition has the value true.

In-context (ic) analysis			
Algorithm		**Set name**	**Set formulation**
Line #	**Level**		
0.	0	*VC′*.0	***VC***
5.	3	*VC′*.0.1.1.0	*VC \ A ∩ B \C ∩D ∩ **E***
12.	2	*VC′*.0.2.0	*VC \ A \ B ∩ J ∩ **E***

Figure 10: Sets defined during the execution of the algorithm (ic) (extract)

The sets thus defined form a collection of nested sets. A collection of nonempty sets is said to be *nested* if, given any pair X, Y of the sets, either X ⊆ Y or X ⊇ Y or X and Y are disjoint. (In other words, X ∩ Y is either X, Y or ∅.) (Knuth 75, pp. 309, 314). The non-contextual analysis of *Doubling_or_not* also resulted in a collection of nested sets, but there were no disjunctions. (Disjunctions can occur in non-contextual analyses; for example in disambiguating applications such as semantic hierarchies in machine translation (Cardey et al. 03) and ambiguous tag sets in disambiguating parts of speech taggers (Cardey & Greenfield 03)). Thus for a systemic linguistics analysis representation to be well formed, two constraints must be met: proper sub-setting and disjunction.

3.3.1 Proper sub-setting

The sets defined during the execution of the in-context algorithm form a collection of proper sub-sets (Figure 11).

In-context (ic) analysis	
Algorithm line #	**Parent set ⊃ Line's set**
1.	*VC′*.0 ⊃ *VC′*.0.0
5.	*VC′*.0.1.1 ⊃ *VC′*.0.1.1.0
12.	*VC′*.0.2 ⊃ *VC′*.0.2.0

Figure 11: Proper sub-set structure (ic) (extract)

3.3.2 Disjunction

The sets defined during the execution of the in-context algorithm at the same (nesting) level and with common parent set are mutually disjoint. For example from line 0 of the algorithm:

disjoint <*VC′*.0.0, *VC′*.0.1, *VC′*.0.2>

To show this, it is necessary to show that:

(*VC* ∩ ***A***) ∩ (*VC* \ *A* ∩ ***B***) ∩ (*VC* \ *A* \ *B* ∩ ***J***) = ∅

It is sufficient to show that:

(*VC* ∩ ***A***) ∩ (*VC* \ *A* ∩ ***B***) = ∅

We have:

(*VC* ∩ ***A***) ∩ (*VC* \ *A* ∩ ***B***) = (*VC* ∩ ***A***) ∩ ((*VC* \ *A*) ∩ ***B***) = (*VC* ∩ ***A***) ∩ (*VC* \ *A*) ∩ ***B***

But (*VC* ∩ ***A***) ∩ (*VC* \ *A*) = ∅. Therefore:

(*VC* ∩ ***A***) ∩ (*VC* \ *A* ∩ ***B***) = ∅ ■

3.4 Formulation of the super system

Figure 12 illustrates the formulation of the super system $S_{Doubling\_or\_not}$ as the ordered binary relation *S* between the equivalence relations E_{nc} and E_{ic}.with the materialisation of its associated graph.

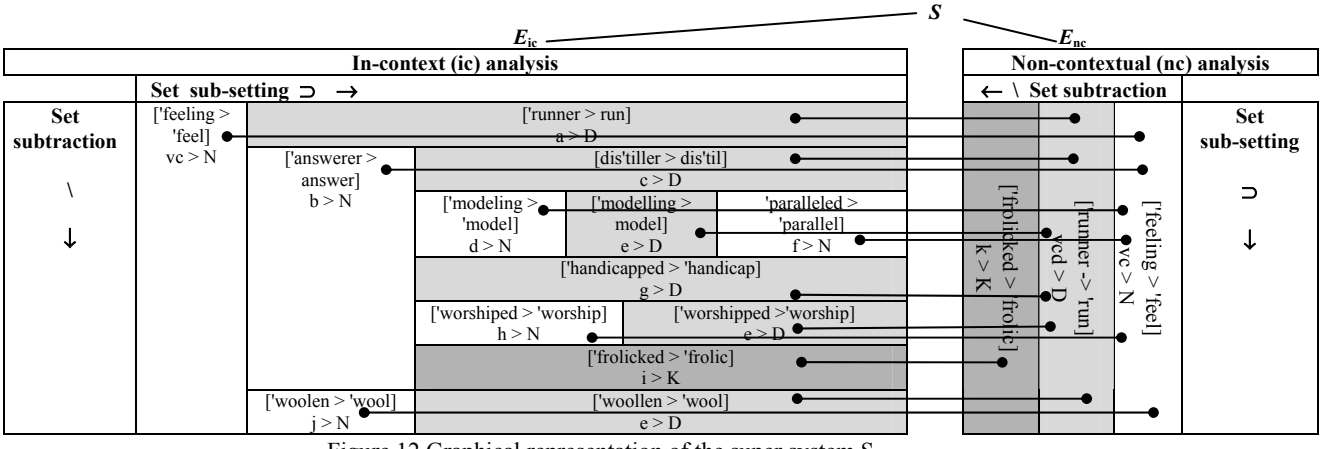


Figure 12 Graphical representation of the super system $S_{Doubling\_or\_not}$

4 Optimisation considerations

The model described presents optimisation possibilities in variously processing speed, space and verification. The nest of sets defined during the execution of the in-context algorithm at the same (nesting) level and with common parent set are formed by a process of progressive set differences, for *Doubling_or_not*, see Figure 13.

Algorithm		Set name	Set formulation
Ligne #	Level		
3	2	*VC'*.0.1.0	$VC \setminus A \cap B \cap \boldsymbol{C}$
4	2	*VC'*.0.1.1	$VC \setminus A \cap B \setminus C \cap \boldsymbol{D}$
7	2	*VC'*.0.1.2	$VC \setminus A \cap B \setminus C \setminus D \cap \boldsymbol{G}$
8	2	*VC'*.0.1.3	$VC \setminus A \cap B \setminus C \setminus D \setminus G \cap \boldsymbol{H}$
10	2	*VC'*.0.1.4	$VC \setminus A \cap B \setminus C \setminus D \setminus G \setminus H \cap \boldsymbol{I}$

Figure 13: Progressive set difference process

This progressive set difference process which has as result set subtraction is safe in functional terms, we have:

disjoint <*VC'*.0.1.0, *VC'*.0.1.1, *VC'*.0.1.2, *VC'*.0.1.3, *VC'*.0.1.4>

but for a given analysis there may be variously certain redundant operations or indeed no such need of them at all. For *Doubling_or_not*, linguistic inspection of the abstraction conditions a, b, j shows that the sets of the nest *A*, *B*, *J* are necessarily mutually disjoint; not only is no subtraction necessary but there is no explicit algorithmic sequencing necessary for the conditions a, b and j. However for the algorithm condition sequence <c, d, g, h, i>, whilst sets *D*, *G*, *H*, *I* are mutually disjoint, algorithmically, condition c delivering set *C* must for linguistic reasons precede any sequence of d, g, h, i.

There exist linguistic analysis situations where the conditions in a nest level are such that their abstracted sets are in any case mutually disjoint; no explicit set subtraction is required. An example is the use of the form of words for raw parts of speech tagging (Cardey et al. 97), which coupled with string based set sub-setting leads to highly space efficient intentionally intensive raw parts of speech dictionaries which can also process host language oriented neologisms. The choice of nest level condition sequencing can thus depend on external criteria, such as speed optimisation in the case of automated applications. Furthermore, in such automated applications, mechanical verification that a representation instance is well formed in respect of proper sub-setting and disjunction is possible (Robardet 03).

5 Conclusion

Confronted with the increasing need for quality natural language processing applications in particular in the domain of safety critical systems, we need to ensure that linguistic analysis models are accessible not only to linguists and computer scientists, but also that they can be objectively evaluated and be subject to quality management. We have presented a core model of systemic linguistic analysis which is intended to meet this goal, and have described the materialising of the relation between a linguistic system's component variant and canonical systems. The model described is a relation between equivalence relations over each of these component systems. The precise structure of this relation enables case based validation, traceability and the verification that a given linguistic analysis's representation is well formed. Furthermore for certain linguistic analysis situations where automation is possible, case based regression validation and well formed representation verification can be done by machine.

References

(Cardey 87), Cardey, S. *Traitement algorithmique de la grammaire normative du français pour une utilisation automatique et didactique*, Thèse de Doctorat d'Etat, Université de Franche-Comté, France, June 1987.

(Cardey et al. 97) Cardey, S., El Harouchy, Z., Greenfield, P., *La forme des mots nous renseigne-t-elle sur leur nature ?*. In Actes des 5èmes journées scientifiques, Réseau Lexicologie, Terminologie, Traduction, LA MEMOIRE DES MOTS, Tunis, 22-23 September 1997. Collection "actualité scientifique" de l'AUPELF-UREF, pp. 305-313.

(Cardey et al. 03) Cardey, S., Greenfield, P., Hong, M-S., *The TACT machine translation system: problems and solutions for the pair Korean – French*, Translation Quarterly, No. 27, The Hong Kong Translation Society, Hong Kong, 2003, pp. 22-44.

(Cardey et al. 04a) Cardey, S., Alsharaf, H., Greenfield, P., Shen, Y., (2004), *Problems and Solutions in Machine Translation Involving Arabic, Chinese and French*, Proceedings of the International Conference on Information Technology, ITCC 2004, April 5-7, 2004, Las Vegas, Nevada, USA, IEEE Computer Society, Vol 2, pp. 293-297

(Cardey et al. 04b) Cardey, S., Greenfield, P., Wu, X., *Designing a Controlled Language for the Machine Translation of Medical Protocols: The Case of English to Chinese*, Proceedings of AMTA-2004 The 6th Conference of the Association for Machine Translation in the Americas, Georgetown University, Washington DC, USA, September 28 - October 2, 2004, Springer-Verlag: LNAI 3265: Machine Translation: From Real Users to Research, ISBN 3-540-23300-8, pp.37-47.

(Cardey et al. 04c) Cardey, S., Vienney, S., Greenfield, P., *Systemic analysis applied to problem solving*, Proceedings of EsTAL 2004, Alicante, Spain, October 20-22, 2004, Springer-Verlag – LNAI 3230, ISBN 3-540-23498-5, pp. 431-441.

(Cardey & Greenfield 92) Cardey, S., Greenfield, P., *The `Studygram' Natural Language Morphology System: A First Step to an Intelligent Tutoring Platform for Natural Language Morphology*. In Proceedings of the UMIST ICALL workshop, CTI Centre for Modern Languages, University of Hull, UK, ISBN 0 9520183 0 6, 1992, pp. 42-59.

(Cardey & Greenfield 02) Cardey, S., Greenfield, P., *Systemic Language Analysis and Processing*. To appear in the proceedings of the Second Conference of the FNLP, Language, Brain and Computation: Venice, October 3-5, 2002 (Benjamins).

(Cardey & Greenfield 03), Cardey, S., Greenfield, P. *Disambiguating and Tagging Using Systemic Grammar*, Proceedings of the 8th International Symposium on Social Communication, Santiago de Cuba, January 20-24, 2003, Actas I, pp. 559-564.

(Cardey & Greenfield 05) Cardey, S., Greenfield, P., *Systemic Linguistics with Applications*, In Proceedings of the 9th International Symposium on Social Communication, Santiago de Cuba, January 24-28, 2005, Actas II, pp. 649-653.

(Dial 70), Dial, R.,B., *Algorithm 394. Decision Table Translation*, Communications of the ACM, Vol. 12, No. 9, September 1970, pp. 571-572.

(Greenfield 03) Greenfield, P., *An initial study concerning the basis for the computational modelling of Systemic Grammar*. BULAG N° 28, 2003, ISBN 2-84867-042-8, pp. 83-95.

(Greenfield 97) Greenfield, P., *Exploiting the Model Theory and Proof Theory of Propositional Logic in the Microsystem Approach to Natural Language Processing*. In BULAG N° 22, ISSN 0758 6787, 1997, pp. 325-346.

(Hopcraft & Ullman 69) Hopcroft, J.E., Ullman, J.D., *Formal languages and their relation to automata*, Addison-Wesley Publishing Company, 1969.

(Humby 73) Humby, E., *Programs from decision tables*, Macdonald/American Elsevier, 1973, ISBN 0 356 04126 3/ISBN 0 444 19569 6.

(Knuth 75) Knuth, D. *The art of computer programming*, Second edition, Volume 1 / Fundamental Algorithms, Addison-Wesley Publishing Company, Reading, Massachusetts, 1975.

(Robardet 03) Robardet, G., *Vérification automatisée des règles de contexte du logiciel Labelgram*, Mémoire de maîtrise, Sciences du langage mention TAL, Besançon, 2003.

(Spaggiari et al. 05) Spaggiari, L., Beaujard, F., Cannesson, E. *A controlled language at Airbus*. To appear in the number "Machine translation, controlled languages and specialised languages", Lingvisticae Investigationes, Benjamins, 2005.

A corpus-based model for bridging anaphora resolution in Italian

Tommaso Caselli* and **Irina Prodanof†**
*†Dipartimento di Linguistica, Università degli Studi di Pavia,
Corso Strada Nuova, 65 27100 Pavia
†Istituto di Linguistica Computazionale Pisa, Consiglio Nazionale delle Ricerche
Via Moruzzi, 1 56124 Pisa
{cobweb80@yahoo.it } {irina.prodanof@ilc.cnr.it}

Abstract

The aim of this work is to provide a corpus-based model for the resolution and analysis of bridging anaphors expressed by full definite noun phrases (FDNPs) of the form "definite article + N" in Italian. The model developed is an unification of the *Discourse Representation Theory* –DRT- (Kamp-Reyle 1993) and the *Centering Theory* (Grosz et alii 1995). The framework thus developed has been tested on a corpus of newspaper articles which provided both interesting results on the occurrence of this kind of anaphoric strategy in Italian and necessary corrections to the model.

1 Introduction

The aim of this work is to provide a corpus-based model for the resolution and analysis of bridging anaphors expressed by full definite noun phrases (FDNPs) of the form "definite article + N" in Italian.

Kleiber (1999) defines bridging anaphora as a "type of indirect textual reference whereby a new referent is introduced as an anaphoric not of but via the referent of an antecedent expression" [Kleiber 1999: 339].

A trend in linguistic theories, which had counterparts in computational frameworks, tends to emphasise the idea that FDNPs are a matter of the global discourse focus i.e. they are used to retrieve a referent which is no more accessible or to construct a conceptual representation which uniquely identifies a referent. On the contrary, empirical studies provided evidence to Sidner's (1979) hypothesis that bridging FDNPs are different from other occurrences of anaphoric FDNPs, since, in the process of identification of their antecedents, they are more sensitive to the local focus. In addition to this, bridging FDNPs trigger an "*inferential presupposition*" of the kind "the [N_1] R [N_2]", where N_1 represents the FDNP and R is the inferential relation or bridge the interpreter has to perform to correctly interpret the occurrence of the N_2, i.e. the bridging anaphor[1].

Kleiber (1999) identifies a sort of semantic restrictions on what kinds of FDNPs can enter a bridging relation. Recalling the notion of functional nouns[2], he identifies two very general, language independent factors which work in the mechanism of the relation between the referents involved in a bridging anaphor: a condition of alienation and the principle of ontological congruence. A bridging description can be conceived of as a FC2 with implicit argument. This type of semantic definites introduces the referent by sole means of the sortal predicate N, without semantic subordination to another individual. In other words, the head noun looks as semantically autonomous or alienated.

Next to these semantic restrictions, a couple of pragmatic constraints can be identified. We propose to use the following pragmatic restrictions on

[1] The R relation can be thought as deriving from Chierchia's (1995) compositional semantics of FDNPs.

[2] By functional nouns we intend NPs denoting a non-ambiguous interpretation, or a functional concept (FC) (see Löbner 1985).

inferencing: an Effort Condition and a Plausibility Condition.

We consider bridging as a by-product of computing the discourse structure[3] which is a necessary precondition for discourse interpretation.

Three pragma-cognitive dimensions for the interpretation of bridging anaphors can be identified: a lexical-semantic dimension, a co-textual dimension and a contextual or extra-linguistic dimension. These three levels are organized like Chinese boxes and will have a role both in modelling the framework and in the corpus study.

2 The Model

In the development of the framework some aspects must be considered, in particular:

- the role of discourse structure;
- the semantic nature of FDNPs as anaphoric expressions;
- the empirically based claim that FDNPs are not primarily anaphoric.

The model we propose tries to answer these problems by integrating *DRT* (Kamp-Reyle 1993) with some aspects of the *Centering Theory* (Grosz et alii 1995).

The use of a DRT-like framework offers a good solution to the problem of discourse structure representation. DRT is an attempt to represent the incremental nature of interpretation of discourse connected with semantic cohesiveness. The information of each sentence contributes to the construction of the DRS by developing a process- and context-oriented perspective on semantics.

Another great advantage of this framework is represented by its adaptability since the three dimensions of bridging can be integrated into the model and eliminate the main shortcoming of DRT analysis of anaphoric FDNPs, i.e. the process of accommodation.

Centering Theory (Grosz et alii 1995, , being both a theory of salience and of local discourse coherence, tries to overcome some shortcomings of DRT. In particular, it predicts a positive correlation between salience of the antecedent and bridging anaphor, as already demonstrated by Poesio (2003).

In addition to this, the use of a Centering based framework helps us in providing a solution also to the problem of discourse segmentation. The look-back for the antecedent of a bridging is to be performed either in the same sentence or in the immediate previous one. If no anchor is found, we have to extend the discourse segment to a maximum of five sentences back[4]. To avoid ambiguity in the recognition of the probable anchors, the entities of the universe of discourse are ranked according to Centering parameters (Cb>Cp>Cf) and searched accordingly for a positive match.

A decay process is integrated into the framework, so that when we try to integrate a DRS into the main DRS, the entities at disposal for the identification of the would-be anchor of a bridging FDNP above the horizontal line of the main DRS are only those of the previous discourse segment or DRS.

The model thus developed predicts a positive correlation between entities in the local focus and bridging FDNPs. Moreover, the restriction to the most salient entities - the previous backward-looking centres (Cbs) and preferred centres (Cps) of each discourse segment or DRS - should increase precision in the identification of the right anchor, reducing ambiguity. A further advantage of this proposal is that the representation of the discourse focus is

[3] As suggested by Asher-Lascarides (1998).

[4] The choice of a span of five sentences is arbitrary but it tries to represent the local focus of discourse, to which bridging FDNPs are sensitive for the identification of their anchor.

connected to the representation of discourse.

3 Testing the Model.

To verify the validity of the theoretical model we performed a test on a corpus of 17 randomly chosen articles from the Italian financial newspaper "*il Sole-24 Ore*". The classification task has been performed using processing requirements, i.e. the information needed to process a FDNP[5].

A total of 1412 FDNPs has been identified. Table 1 shows the distribution for each class adopted:

CLASS	NUMBER OF ITEMS	PERCENTAGE
First Mention	869	61.15%
Direct Anaphora	170	12.03%
Bridging	299	21.17%
Idiom	25	1.62%
Doubt	49	3.47%
Total	1412	100%

Table 1 – Results for each class of FDNPs.

The figures provide further support to theoretical claims and results of previous empirical studies[6] that FDNPs are not primarily anaphoric, since more than 60% of the occurrences are instances of First Mention.

A first interesting observation about the class of Bridging is that they represent the 63.88% (299/469) of all anaphoric FDNPs, suggesting that bridging is a more productive strategy in Italian with respect to other languages, i.e. English.

Five classes of bridging FDNPs are identified (Table 2). These classes corresponds to the three sources, or pragma-cognitive dimensions, which may give rise to a bridging anaphor:

CLASS	NUMBER OF ITEMS	PERCENTAGE
Lexical	119	39.79%
Event	18	6.02%
Rhetorical Relation	27	9.03%
Discourse Topic	26	8.69%
Inferential	109	36.45%
Total	299	100%

Table 2 - Distribution of Bridging FDNPs.

Further evidence to Sidner's (1979) hypothesis that bridging FDNPs are sensitive to the local focus has been found.

LOOK-BACK OF THE ANCHOR	NUMBER OF ANCHORS	PERCENTAGE
0/1 SENTENCE	181	68.30%
2/3 SENTENCES	58	21.88%
4/5 SENTENCES	26	9.82%
Total	265	100%

Table 3 - The look-back between anchors and bridging FDNPs.

As Table 3 shows, almost 70% of the anchors may be found either in the current sentence or in the immediate previous one. Similar results can be obtained for the class of Rhetorical Relation with implicit anchor.

If we exclude the occurrences of the class of Event and those instances of Rhetorical Relation which can be reduced to argument-event relation, 221 anchors are nominal entities i.e. NPs, and 56 are proper names of individuals or of organizations.

An analysis of the syntactic structures of these anchors provides interesting results. In particular, we found that the majority of the anchors are definite NPs. A high number of them (34.03%) are NPs of postmodifying PPs. The hypothesis of a positive correlation between Cbs/Cps and anchors is in part confirmed (119/221 of the anchors have been Cbs or Cps of previous sentences). The remaining anchors are realized by other elements in the forward-looking centre (Cf) list. This means that:

- knowledge of the local focus is necessary but not sufficient to determine the anchor of a bridging description;

[5] Vieira-Poesio's (2000) heuristics have been used as point of departure, which have been adapted and enlarged.

[6] Löbner's (1985), Vieira-Poesio (2000), Fraurud (1990).

- limiting the search of probable anchors Cbs and Cps might be very useful, since it can only increase the precision of the process of resolution.

The ranking of the elements in the Cf list according to the grammatical role suggests a preference of anchors in Oblique position (33.03%) over Indirect Object (6.03%).

Other interesting data are provided by the analysis of the bridging FDNPs. As far as their syntactic structure is concerned, we found similar results to those obtained for the anchors, since a high number of them (more than 40%) are noun head of postmodifying PPs of First Mention definites. The results for the syntactic structure correlate with those obtained for the grammatical roles, where 44.96% of bridging FDNPs in the corpus are in oblique position. Only 27% are in subject position, followed by 18.13% which are in indirect object position, and finally 9.39% are in direct object position.

3.1 Revision of the Model.

The core structure of the model has been confirmed but the data suggested modifications.

The main shortcoming of the model is represented by the lack of a global focus tracking mechanism. The need of such a mechanism is also suggested by the relative high number of bridging FDNPs classified as Discourse Topic, which cannot be recognized using the framework as it is.

The solution to this problem is provided by the corpus itself. Newspaper articles have headlines whose role is both to catch the reader's attention and to inform the reader in an efficient and effective way.

We can consider the referential entities introduced in titles as meta-textual objects. If so, we can consider titles as a meta-level of the discourse/text. If the meta-level function of titles is accepted, we are in a position to claim that the objects which are in titles are entities with a special status. Thus, they must be accessible to all DRS when integrated into the main DRS. This means that above the horizontal line of the main DRS, and after the entities of the previous segment, we should find the entities of the title. The presence of these entities represents, then, the tracking focus mechanism which lacked from the framework (Figure 1).

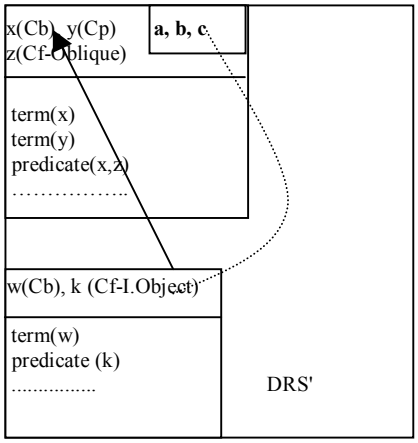


Figure 1- The model revisited

To integrate DRS' into the main DRS, we try to match the elements which form the universe of discourse first with the discourse referents of the previous segment, and if this procedure fails we try to match them with those of the title. Such a strategy also has interesting counterparts for the use of computational lexical resources for a computer-based resolution of bridging anaphora.

The other modification to the model concerns the ranking order of the probable anchors. Although empirical support to the hypothesis of a preference for Cbs and Cps position is provided, the high number of anchors in Oblique position with respect to those in Indirect Object position suggests that the elements of the Cf list should be ranked accordingly, as already stated.

4. Conclusion and Future Work.

The most innovative aspect of this work is represented by the proposal to put together in the same model two different approaches, namely Kamp-Reyle's (1993) *Discourse Representation Theory* and Grosz et alii's (1995) *Centering Theory*.

The use of a DRT-like framework offers a good solution to the problem of discourse structure representation.

The presence of Centering represents a solution to the problem of a focusing mechanism for selecting discourse referents. Despite the observation that FDNPs are a matter of the global discourse focus, we have shown that bridging FDNPs are heavily influenced by the local focus, as about 70% of the anchors can be found either in the current sentence or in the immediate previous one. A centering-based analysis of the sentences which form a discourse segment increases the precision in the identification of the right anchor avoiding problems of multiple anchoring. The ranking of anchors according to their grammatical role suggests that knowledge of the local focus is necessary but not sufficient to determine the anchor of a bridging FDNPs since only 53.84% of the anchors have been Cbs or Cps of previous sentences. Another contribution of the Centering framework is that entities in the Cf list should be ranked, and not searched randomly.

A great advantage of this framework is represented by its adaptability. The focus tracking mechanism introduced in the model after the corpus-study did not change the core aspect of the framework and its functioning, but provides a solution for the class of Discourse Topic bridging.

The use of processing requirements in the classification task of the bridging FDNPs suggests a preference for a shallow processing approach. Instead of proposing specific hand coded knowledge for each type of discourse we propose the use of existing lexical resources like WordNet and PAROLE/ SIMPLE/CLIPS. Such a proposal is supported by the results obtained in the corpus for the class of bridging, where about 40% of the links, or relations, where based on lexical-semantics.

The use of these resources has two main advantages:

- it makes the system domain independent;
- due to their organisation, they can be thought of as reflecting the different dimensions involved in the process of resolution of bridging anaphors.

References

Asher, N and A. Lascarides.1998, Bridging, in *Journal of Semantics* vol. 15 (1): 83-113.

Caselli, T. 2005, *Lexical Anaphora: a corpus-based model and an XML annotation scheme for bridging anaphora in Italian*. M.A. Thesis, University of Pavia.

Chierchia, G.1995, *Dynamics of Meaning: anaphora, presuppositions and the Theory of Grammar*, University of Chicago Press, Chicago.

Fraurud, K.1990, Definiteness and the Processing of Noun Phrase in Natural Discourse in *Journal of Semantics*, vol. 7 (4): 395-433.

Grosz, B. et alii.1995, Centering: A framework for modelling the local coherence of discourse, in *Computational Linguistics*, vol. 21 (2): 202-225.

Kamp, H and U. Reyle.1993, *From Discourse to Logic: Introduction to Model-theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*, Kluwer Academic Publishers, Dordrecht.

Kleiber, G.1999, Associative anaphora and part-whole relationship: the condition of alienation and the principle of ontological congruence, in *Journal of Pragmatics*, vol. 31: 339-362.

Korzen, I.2003, Anafora associativa: aspetti lessicali, testuali e contestuali, in *Atti del XXXIV Congresso Internazionale di studi della Società di Linguistica Italiana (SLI). Firenze 19-21 ottobre 2000*, Maraschino, N. and T. Poggi Salani (eds)Bulzoni, Roma.

Krahmer, E and P.Piwek.2000, *Varieties of Anaphora*, (eds), Course Notes, ESSLLI00, Birmingham, August 11-23 2000.

Lenci, A. et alii.2003, SIMPLE: plurilingual semantic lexicons for natural language processing", in *Linguistica Computazionale, Computational Linguistics in Pisa, Special Issue*, A. Zampolli, N. Calzolari, L. Cignoni (eds.), voll. XVI-XVII: 323-352.

Löbner, S.1985, Definites, in *Journal of Semantics* vol. 4: 297-326.

Poesio, M.2003, Associative FDNPs and salience, in *Proceedings of the EACL Workshop on Computational Treatments of Anaphora*, Budapest.

Ruimy, N.et alii.2003, "The PAROLE model and the Italian syntactic lexicon", in *Linguistica Computazionale, Computational Linguistics in Pisa, Special Issue*, A. Zampolli, N. Calzolari, L. Cignoni (eds.), voll. XVI-XVII: 793-820.

Sidner, C.L.1979, Towards a computational theory of definite anaphora comprehension in English discourse, Ph.D. Thesis, MIT.

Vieira, R. and M. Poesio.2000, An Empirically-Based System for Processing FDNPs, in *Computational Linguistics*, vol. 26 (4): 539-593.

Disambiguating Coordinations Using Word Distribution Information

Francis Chantree[1] **Adam Kilgarriff**[2] **Anne de Roeck**[1] **Alistair Willis**[1]
[1]The Open University, Milton Keynes, U.K.
[2]Lexical Computing Ltd, Brighton, U.K.
[1]{F.J.Chantree,A.DeRoeck,A.G.Willis}@open.ac.uk; [2]adam@lexmasterclass.com

Abstract

In this paper we present some heuristics for resolving coordination ambiguities. This type of ambiguity is one of the most pervasive and challenging. We test the hypothesis that the most likely reading of a coordination can be predicted using word distribution information from a generic corpus. The measures that we use are: the relative frequency of the coordination in the corpus, the distributional similarity of the coordinated words, and the collocation frequency between the coordinated words and their modifiers. The heuristics that we present based on these measures have varying but useful predictive power. They also take into account our view that many ambiguities cannot be effectively disambiguated, since human perceptions vary widely.

1 Introduction

Coordination ambiguity is a structural (i.e. syntactic) ambiguity. Compared with other structural ambiguities, e.g. prepositional phrase (PP) attachment ambiguity, it has received little attention in the literature. This is despite the fact that coordinations are known to be a "pernicious source of structural ambiguity in English" (Resnik 99). Our work is novel in that we use several types of word distribution information to disambiguate coordinations of any type of word, and in that we acknowledge that some ambiguities are too ambiguous to be judged reliably. This latter point is an important consideration, as providing readings for such ambiguities would be misleading and potentially dangerous.

We test the hypothesis that the preferred reading of a coordination can be predicted using word distribution information from a generic corpus. To do this we present three heuristics. These use the relative frequency of the coordination in the corpus, the distributional similarity of the coordinated words, and the collocation frequency between the coordinated words and a modifier. All the heuristics use information generated by the Sketch Engine[1] (Kilgarriff *et al.* 04) operating on the British National Corpus[2] (BNC).

The examples that we investigate contain a single coordination which incorporates two phrases and a modifier, such as in the phrase:

old boots and shoes,

(where *old* is the modifier). Applying our heuristics to this phrase, we find firstly that *boots and shoes* appears relatively often in the corpus. Secondly, *boots* and *shoes* are shown to have strong distributional similarity, suggesting that *boots and shoes* is a syntactic unit. Both these factors suggest that coordination takes place before the modifier *old* takes scope. Thirdly, the collocation frequency of *old* and *boots* is not significantly greater than that of *old* and *shoes*, suggesting that it is not likely that only *boots* is modified by *old*. All three heuristics agree therefore that coordination takes place before the modifier takes scope.

We test our hypothesis for text drawn from requirements engineering. This is a very suitable domain as ambiguity is recognised as being a serious and potentially costly problem (Gause & Weinberg 89). For instance, a system might be built incorrectly due to a requirement being read in a way that was unintended.

We have built and tagged a corpus of requirements specification documents, from which we extract a collection of sentences and phrases containing coordination ambiguities. We identify preferred readings for each of these by means of ambiguity surveys, in which we obtain human judgements on each example. This forms our evaluation dataset. We then apply our heuristics to the dataset to see if they can automatically replicate the consensus human judgements.

In this paper we first discuss the coordination ambiguity problem and research related to our own, and then outline how we create our evaluation dataset. We then describe our empirical research, beginning with methodology that is

[1]http://www.sketchengine.co.uk

[2]http://natcorp.ox.ac.uk

Researchers	Recall (%)	Baseline Precision (%)	Precision (%)	Precision % points above base line (%)	F-Measure $\beta = 0.25$ (%)	F-Measure % points above base line (%)
(Agarwal & Boggess 92)	n/a	n/a	82.3	n/a	n/a	n/a
(Goldberg 99)	n/a	64	72	18	n/a	n/a
(Resnik 99) (unweighted)	66.0	66.0	71.2	5.2	70.9	3.5
(Resnik 99) (weighted)	69.7	44.9	77.4	32.5	76.9	30.5

Table 1: Performances of other researchers

generic to all our heuristics, followed by a description of each heuristic, and ending with an evaluation of our results. Lastly, we offer our conclusions and present some ideas for future work.

2 Coordination Ambiguity

Coordination ambiguity can occur whenever coordinating conjunctions are used, and it is a pervasive problem in English as coordinating conjunctions are common. Together, *and* and *or* account for approximately 3% of the words in the BNC, and they account for the great majority of coordinating conjunctions. We confine our investigations to *and*, *or* and *and/or*. Words and phrases of all types can be coordinated (Okumura & Muraki 94). The external modifier can also be a word or phrase of almost any type, and it can appear before or after the coordination.

In an example from our dataset:

> *Assumptions and dependencies that are of importance*

the external modifier *that are of importance* applies either to both the *assumptions* and the *dependencies* or to just the *dependencies*. Because of the order in which the words are connected, we refer to the former case as *coordination-first*, and to the latter as *coordination-last*[3]. We concentrate on coordinations of this type where two syntactic readings are possible.

3 Related Research

There has not been a large amount of research on coordination ambiguity in English in the NLP community, and what has been carried out has been quite diverse. The results of the researchers discussed below are summarised in Table 1.

[3]Other terminology can be used, e.g. *low attachment* and *high attachment*, depending on where the coordinated phrase furthest from the modifier attaches in the parse tree (Goldberg 99).

Agarwal and Boggess present an algorithm that attempts to identify which phrases are coordinated by coordinating conjunctions (Agarwal & Boggess 92). Using the machine-readable Merck Veterinary Manual as their dataset, they achieve an accuracy of 81.6% for the conjunctions *and* and *or*. Their method matches parts of speech and case labels of the head words of the coordinated phrases. Pre-conjunction phrases are popped off a stack until a match with the post-conjunction phrase is found. Their method is a straightforward and potentially useful way of matching candidate coordinated phrases, but it does not deal adequately with ambiguity arising from modifier attachment.

Goldberg uses unsupervised learning to determine the attachment of noun phrases in ambiguous coordinations (Goldberg 99). She simplifies the text using a chunker, and then extracts the headwords of the coordinated phrases. Her test data, which is unannotated, includes a lot of noise. Also, as her method is a simple re-implementation of a PP-attachment method (Ratnaparkhi 98), it does not model information, such as word similarity, that is useful for coordination disambiguation. Goldberg's system correctly predicts with an accuracy of 72% the annotated attachments of her development set drawn from the Wall Street Journal.

Using an unweighted heuristic, Resnik investigates the role of semantic similarity in resolving coordination ambiguities involving nominal compounds of the form *noun1 and noun2 noun3* (Resnik 99). (Note that this is not the same as the distributional similarity which we use.) He looks up the nouns in WordNet and determines which of the classes that subsume them both has the highest information content. Without any backoff strategy, this procedure results in 71.2% precision and 66.0% recall of the correct human dis-

ambiguations in his dataset drawn from the Wall Street Journal.

Using a weighted heuristic, Resnik adds an evaluation of the selectional association between the nouns to his semantic similarity evaluation (Resnik 99). He also restricts his dataset to coordinations of the form *noun0 noun1 and noun2 noun3*. Improved precision of 77.4% and 69.7% recall is achieved. We believe that this heuristic's high performance is in no small part due to the highly specific dataset being used, allowing for more measurements of similarity to be factored in. The results are interesting, but we feel that a useful disambiguation heuristic should be able to cope with less constrained data.

Some research on disambiguating uncoordinated noun compounds using corpus information bears similarity to our own. Lauer provides a synopsis of some approaches to the binary decision problem of disambiguating the bracketings in compounds of the form *noun1 noun2 noun3* (Lauer 95). He reports a maximum accuracy of 81%, above a baseline of 67%, using a hand-disambiguated dataset drawn from a popular encyclopedia. Lauer shares our opinion that some linguistic constructions are too ambiguous to be assigned a reading with confidence, and as a result he excludes 11% of sentences from his dataset.

4 Developing an Evaluation Dataset

4.1 Human Judgements

Ambiguity is context-, speaker- and listener-dependent, and so there are no absolute criteria for judging it. Therefore, we capture human judgements about the ambiguity of the sentences in our surveys in order to form our evaluation dataset. Rather than rely upon the judgement of one human reader, we take a consensus from multiple readers. Such an approach is known to be very effective albeit quite expensive (Berry *et al.* 03).

4.2 The Ambiguity Surveys

The sentences in our ambiguity surveys are drawn from our corpus of requirements specifications. Sentences — or non-sentential titles, bullet points etc — that contain coordinating conjunctions are identified. We do not use all the sentences containing a coordination that we find. Sentences containing coordinations which are syntactically unambiguous are identified, by hand, and discarded. A breakdown of the sentences we use by the part of speech of the head word of the coordinated phrases is given in Table 2. A breakdown by the part of speech of the external modifier is given in Table 3.

Head Word	% of Total	Example from Surveys
Noun	85.5	Communication and performance requirements
Verb	13.8	Proceed to enter and verify the data
Adjective	0.7	It is very common and ubiquitous

Table 2: Breakdown of sentences by head word type (head words are underlined)

Modifier	% of Total	Example from Surveys
Noun	46.4	(It) targeted the project and election managers
Adjective	23.2	.... define architectural components and connectors
Prep	15.9	Facilitate the scheduling and performing of works
Verb	5.8	capacity and network resources required
Adverb	4.4	(It) might be automatically rejected or flagged
Relative Clause	2.2	Assumptions and dependencies that are of importance
Number	0.7	zero mean values and standard deviation
Other	1.4	increased by the lack of funding and local resources

Table 3: Breakdown of sentences by modifier type (modifiers are underlined)

In total, we extracted 138 suitable coordination constructions and showed each one to 17 judges. They were asked to judge whether each coordination was to be read coordination first, coordination last or "ambiguous so that it might lead to misunderstanding". In the last case, the coordination is then classed as an *acknowledged ambiguity* for that participant. Clearly, the dividing line between what would and what would not lead to misunderstandings is elusive. We take the view that, by using a sufficiently large number of judges, rogue interpretations are not accorded undue significance. Then we use ambiguity thresholds to account for, to whatever extent we desire, the varying differences in opinion that occur.

5 Empirical Study

5.1 Methodology

Here we introduce the metrics, ranking cut-offs and ambiguity thresholds that we use to get the most predictive and appropriate results from our data.

For each heuristic, the number of true positives is the number of coordinations for which the heuristic predicts the consensus result deter-

mined by the surveys, taking the ranking cut-off and ambiguity threshold into consideration. Precision for each heuristic is the number of true positives divided by the total number of positive results achieved by that heuristic. Recall for each heuristic is the number of true positives divided by the number of coordinations which that heuristic should have judged positively.

Precision is much more important to us than recall: we wish each heuristic to be a reliable indicator of how any given coordination should be read, rather than a catch-all technique. (Ultimately, we envisage using each heuristic as one of a large suite of techniques which will disambiguate many coordinations with good precision. Good recall may thereby be achieved if the heuristics have complementary coverage.) We use a weighted f-measure statistic, based on van Rijsbergen's e-measure (vanRijsbergen 79), to combine precision and recall:

$$F\text{–}Measure = \frac{(1+\beta) * Precision * Recall}{\beta^2 * Precision + Recall}$$

A weighting of $\beta = 0.5$ is commonly used to ensure that true positives are not obtained at the expense of also obtaining too many false positives. We use a weighting of $\beta = 0.25$, even more strongly in favour of precision. We aim to maximise the f-measure for all of our heuristics.

We employ 10-fold cross validation, which is an accurate and efficient way of ensuring that data is considered uniformly and that the resulting statistics are not biased (Weiss & Kulikowski 91). Our dataset is first randomly sorted to remove any bias caused by the order in which the sentences were collected. Then it is split into ten equal parts. Nine of the parts are concatenated and used for training to find the optimum ranking cut-off and ambiguity threshold for each heuristic. The heuristics are then run on the heldout tenth part using those cut-offs and ambiguity thresholds. This procedure is carried out for each heldout part, and the performances on all the heldout parts are then averaged to give the performances of the heuristics.

The results that we use from the Sketch Engine, for all three heuristics, are in the forms of rankings. We use rankings, rather than actual measures of frequency or similarity, as it is suggested that they are a more accurate measure for analysis based on word distribution — see for example (McLauchlan 04). For each heuristic, in order to maximise its performance, a ranking cut-off is chosen, and rankings below that cut-off are not considered. The cut-off is found experimentally for each fold in the cross-validation exercise. For each of the three heuristics, the optimum cut-off is in fact found to be the same for all 10 folds.

We also determine different ambiguity thresholds for each heuristic in order to maximise its performance, (although a non-optimal threshold may in fact be preferred by a user). These are not always the same for each of the 10 folds of any heuristic. The ambiguity threshold is the minimum level of certainty that must be reflected by the consensus of survey judgements. Let us say that a particular coordination has been judged by 65% of the judges in the surveys to be coordination-first, and we are using a heuristic that predicts coordination-first readings. Then, if the ambiguity threshold is 60% the consensus judgement will be considered to be coordination-first, whereas it will not if the ambiguity threshold is 70%. It must be noted that this can significantly change the baseline — the percentage of true positives found if all coordinations are considered to be (in this case) coordination-first.

5.2 BNC and the Sketch Engine

All our heuristics use information generated by the Sketch Engine with the BNC as its data source. The BNC is a modern corpus containing over 100 million words of English. It is collated from a variety of sources, including some that share specialist terminology with our chosen domain.

The Sketch Engine accepts input of lemmatised verbs, nouns and adjectives. We use two of the key facilities offered by the Sketch Engine: a word sketch facility giving information about the frequency with which words are found collocated with each other, and a thesaurus giving distributional similarity between words.

The word sketch facility, rather than looking at an arbitrary window of text around a word, finds the correct collocations for the word by use of grammatical patterns (Kilgarriff *et al.* 04). Head words of coordinated phrases can therefore be found with some certainty. Parameters for minimum frequency, minimum salience and maximum number of matches can be entered. We use a minimum frequency of 1 and a minimum salience of 0 throughout, to ensure that we get results even for unusual words.

The Sketch Engine's thesaurus is a distributional thesaurus in the tradition of (Sparck-Jones 86) and (Grefenstette 94); it measures similarity between any pair of words according to the number of corpus contexts they share. The corpus is parsed and all triples comprising a grammatical relation and two collocates, (eg ⟨*object, drink, wine*⟩ or ⟨*modifier, wine, red*⟩) are identified. Contexts are shared where the relation and one collocate remain the same, so ⟨*object, drink, wine*⟩ and ⟨*object, drink, beer*⟩ count towards the similarity between wine and beer. Shared collocates are weighted according to the product of their mutual information, and the similarity score is the sum of these weights across all shared collocates, as in (Lin 98). Distributional thesauruses are especially suitable for analysis of coordinations. For instance, words which have opposite meaning, such as *good* and *bad*, are often coordinated, and such words often have strong distributional similarity.

5.3 Coordination-Matches Heuristic

One approach to finding the most likely reading of a coordination, using a generic corpus, is to find out if that coordination occurs within that corpus. Our hypothesis here is that if a coordination in our dataset is found within the corpus, then that coordination is likely to be a syntactic unit and a coordination-first reading is the most likely.

Using the Sketch Engine, we search the BNC for each coordination in our dataset. This is done using the word sketch facility's list of words that are conjoined with *and* or *or*. Each head word is looked up in turn. The ranking of the match of the second head word with the first head word may not be the same as the ranking of the match of the first head word with the second head word. This is because of the difference in overall frequency of the two words. We use the higher of the two rankings. We find that considering only the top 25 rankings is a suitable cut-off. An ambiguity threshold of 60% is found to be the optimum for all ten folds in the cross-validation exercise.

For the example from our dataset:

Security and Privacy Requirements,

the highest of the two rankings of *Security* and *Privacy* in the word sketch facility's *and/or* lists is 9. This is in the top 25 rankings, and so the heuristic yields a positive result. Of the 17 survey judges, 12 judged this ambiguity to be coordination-first — 1 judged it to be coordination-last and 4 judged it to be ambiguous — which is a certainty of 12/17 = 70.5%. This is over the ambiguity threshold of 60%, so the heuristic always yields a true positive result on this sentence.

Averaging for all the ten folds, the heuristic achieves 43.6% precision, 64.3% recall and 44.0% f-measure. However, the baselines are low, given the relatively high ambiguity threshold, giving 20.0 % points precision and 19.4 % points f-measure above the baselines.

5.4 Distributional-Similarity Heuristic

Our hypothesis here is that if two coordinated head words in our dataset display strong distributional similarity, then the coordinated phrases are likely to be a syntactic unit and a coordination-first reading is therefore the most likely. This is an idea suggested by Kilgarriff (Kilgarriff 03).

For each coordination, the lemmatised head words of both the coordinated phrases are looked up in the Sketch Engine's thesaurus. The ranking of the match of the second head word with the first head word may not be the same as the ranking of the match of the first head word with the second head word. We use the higher of the two rankings. We find that considering only the top 10 matches is the best cut-off for our purposes. An ambiguity threshold of 50% produces optimal results for 7 of the folds, while 70% is optimal for the other 3.

For the example from our dataset:

processed and stored in database,

the verb *process* has the verb *store* as its second ranked match in the thesaurus, and vice versa. This is in the top 10 matches, so the heuristic yields a positive result. Of the 17 survey judges, only 1 judged the ambiguity to be coordination-first — 11 judged it to be coordination-last and 5 judged it to be ambiguous — which is a certainty of 1/17 = 5.9%. This is below both the ambiguity thresholds used by the folds, so the heuristic's performance on this sentence always yields a false positive result.

Averaging for all the ten folds, the heuristic achieves 50.8% precision, 22.4% recall and 46.4% f-measure. Again the baselines are quite low, giving 11.5 % points precision and 5.8 % points f-measure above the baselines.

Heuristic	Recall (%)	Baseline Precision (%)	Precision (%)	Precision above Base line (%)	F-Measure $\beta = 0.25$ (%)	F-Measure above Base line (%)
1: Coordination-Matches	64.3	23.6	43.6	20.0	44.0	19.4
2: Distributional-Similarity	22.4	39.3	50.8	11.5	46.4	5.8
3: Collocation-Frequency	35.3	22.1	40.0	17.9	37.3	14.1
Combination of 1 & not 3	64.3	23.6	47.1	23.5	47.4	22.9

Table 4: Performance of our heuristics

5.5 Collocation-Frequency Heuristic

The third heuristic differs from the other two in that it predicts coordination-last readings, and in that it involves the modifiers of the coordinated phrases in our dataset. The hypothesis here is that if a modifier is shown to be collocated, in a corpus, much more frequently with the coordinated head word that it is nearest to than it is to the further head word, then it is more likely to form a syntactic unit with only the nearest head word. This implies that a coordination-last reading is the most likely.

Using the Sketch Engine's word sketch facility's collocation lists, we find the frequencies in the BNC with which the modifier in each sentence is collocated with the coordinated head words. There are lists for most relationships that a word can have with a modifier. We experimented with using as a cut-off the ratio of the collocation frequency with the nearest head word to the collocation frequency with the further head word. However, the optimal cut-off is found to be when there were no collocations between the modifier and the further head word, and any non-zero number of collocations between the modifier and the nearest head word. An ambiguity threshold of 40% produces optimum results for 8 of the folds, while 70% is optimal for the other 2.

For the example from our dataset:

project manager and designer,

project often modifies *manager* in the BNC but never *designer*. The heuristic therefore yields a positive result. Of the 17 survey judges, 8 judged this ambiguity to be a coordination-last reading — 4 judged it to be coordination-first and 5 judged it to be ambiguous — which is a certainty of 8/17 = 47.1%. This is over the ambiguity threshold of 40% but under the threshold of 70%. On this sentence, the heuristic therefore yields a true positive result for 8 of the folds but a false positive result for 2 of them.

Averaging for all the ten folds, the heuristic achieves 40.0% precision, 35.3% recall and 37.3% f-measure. The baselines are low, giving performances of 17.9 % points precision and 14.1 % points f-measure above the baselines.

5.6 Other Heuristics Considered

We experimented using heuristics based on the lengths of the coordinated phrases and the number agreement of coordinated nouns. The hypothesis was that disparities in either of these two factors would suggest that the coordination was not a syntactic unit and that a coordination-first reading was therefore not likely. We also tested a simple metric of semantic similarity, based on the closeness of the coordinated head words to their lowest common ancestor in hierarchies of WordNet hypernyms. However, these three heuristics demonstrated only very weak predictive power.

5.7 Evaluation

Table 4 summarises our results. These are not directly comparable with the results of the researchers presented in Table 1. This is because of the absence of some statistics in the published results of those researchers, and because we consider that highly ambiguous coordinations cannot be judged accurately and consistently by humans. On one hand, our use of ambiguity thresholds, which implement this consideration, makes the task easier by restricting the target set to relatively clear-cut examples. On the other hand the task is more difficult as there are fewer examples to find. The worth of the ambiguity thresholds is shown, however, in the improvements in performance that they give over the baselines. Our precision and f-measure in terms of percentage points over the baseline, except for the distributional-similarity heuristic, are encouraging.

We combine the two most successful heuristics, as shown in the last line of Table 4. These results are achieved by saying that a coordination-

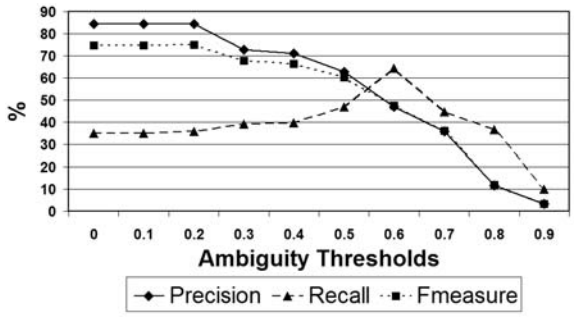


Figure 1: Heuristics 1 and 3 combined

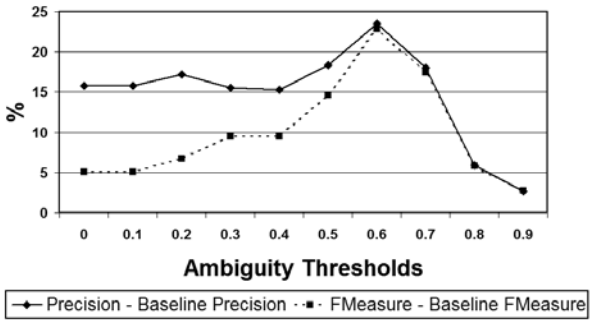


Figure 2: Heuristics 1 and 3 combined: percentage points above baselines

first reading is predicted if the coordination-matches heuristic gives a positive result and the collocation-frequency heuristic gives a negative result. This gives the best performance of all. Figure 1 shows the precision, recall and f-measure for this combination of heuristics, at different ambiguity thresholds. As can be seen, high precision and f-measure can be achieved with low ambiguity thresholds. At these thresholds, even highly ambiguous coordinations are judged to be either coordination-first or -last. However, as can be seen in Figure 2, as percentage points above the baselines, these performances are relatively modest. The combination of heuristics performs best, relative to the baseline, when the ambiguity threshold is set at 0.6, aided by the high recall at this level.

Users of our technique can choose not to use the optimal ambiguity threshold. They choose whatever threshold they feel to be appropriate, considering the linguistic abilities of the people who will read the resulting documents and the importance that they give to ambiguity as a potential threat. Figure 3 shows the proportions of ambiguous and non-ambiguous interpretations at different ambiguity thresholds. It can be seen that none of the

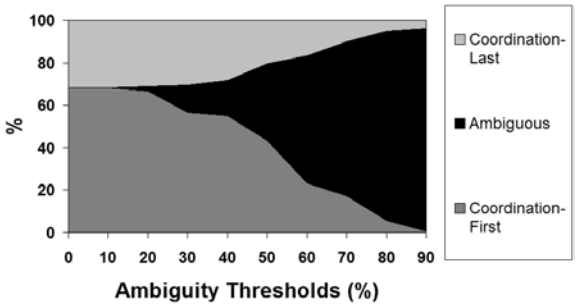


Figure 3: Proportions of ambiguous and non-ambiguous readings at different thresholds

coordinations are judged to be ambiguous with an ambiguity threshold of zero - a dangerous situation. At the other end of the spectrum, an ambiguity threshold of 90% results in almost everything being considered ambiguous - a situation which will waste users' time. From the former of these extremes to the other, the numbers of coordination-first and coordination-last readings are increasingly judged to be ambiguous at an approximately equal rate.

6 Conclusions

We conclude from our research that a surprising number of coordinations in a specialised corpus can also be found in a generic corpus. As a result, our heuristic for predicting that those former coordinations are to be read coordination first is the most effective and useful of the three which we present here.

We conclude that strong association between a modifier and the nearest coordinated head word — in comparison to the association with the further coordinated head word — indicates that those two words form a syntactic unit before the coordination takes place, and that a coordination-first reading is therefore less likely. We also conclude from the performance of our collocation frequency heuristic, that surprising numbers of those syntactic units which occur in our specialised corpus can also be found in a generic corpus.

We conclude from the performance of our distributional-similarity heuristic, that distributional similarity between head words of coordinated phrases is only a weak indicator that they form syntactic units leading to coordination-first interpretations. It might be concluded that this is due to the poor recall achieved by this heuristic, and it might still be the case

that this heuristic could be used in conjunction with other coordination-first predicting heuristics which have wider coverage. Currently, however, using this heuristic in conjunction with the other heuristics produces negligible improvements.

The improved performance obtained when we combined our two most successful heuristics shows that combining such predictors is beneficial. Overall, we conclude that word distribution information can be used effectively to indicate preferred readings of coordination ambiguities, particularly when they are not overly ambiguous. We have shown that this is achievable regardless of the type of words that are coordinated, and regardless of the type of word that modifies them.

We have found that people's judgements can vary quite widely. In addition to the acknowledged ambiguity that occurs when people judge a coordination to be ambiguous, there is also *unacknowledged ambiguity*. This occurs when various people have different interpretations of a sentence or phrase, but each of them thinks that theirs is the only possible interpretation of it. This is potentially more dangerous than acknowledged ambiguity: it is not noticed and it therefore doesn't get resolved. Unacknowledged ambiguity is measured as the number of judgements in favour of the minority non-ambiguous choice, over all the non-ambiguous judgements. The average unacknowledged ambiguity over all the examples in our dataset is 15.3%. Note that unacknowledged ambiguity is automatically included in the consensus judgement for each sentence.

7 Further Work

This paper is part of wider research into notifying users of ambiguities in text and informing them of how likely they are to be misunderstood by readers of the text. We intend to look at improving the heuristics that we have tested, and combining them with others in a manner which gives greater coverage and good precision. We will be testing heuristics based on morphology, typography and word sub-categorisation. Of interest to us in this further work is the analytical method of Okumura and Muraki, which incorporates three feature sets for analysing the parallelism of coordinated phrases (Okumura & Muraki 94).

At present in our dataset, although unacknowledged ambiguity generally occurs together with acknowledged ambiguity, thereby reducing its danger, we consider that it may be interesting to investigate whether unacknowledged ambiguity has any particular characteristics.

References

(Agarwal & Boggess 92) Rajeev Agarwal and Lois Boggess. A simple but useful approach to conjunct identification. In *Proceedings of the 30th conference on Association for Computational Linguistics*, pages 15–21. Association for Computational Linguistics, 1992.

(Berry *et al.* 03) Daniel M. Berry, Erik Kamsties, and Michael M. Krieger. From contract drafting to software specification: Linguistic sources of ambiguity, 2003. A Handbook.

(Gause & Weinberg 89) Donald C. Gause and Gerald M. Weinberg. *Exploring requirements: quality before design.* Dorset House, New York, 1989.

(Goldberg 99) Miriam Goldberg. An unsupervised model for statistically determining coordinate phrase attachment. In *Proceedings of the 37th conference on Association for Computational Linguistics*, pages 610–614. Association for Computational Linguistics, 1999.

(Grefenstette 94) Gregory Grefenstette. *Explorations in Automatic Thesaurus Discovery.* Kluwer Academic Publishers, 1994.

(Kilgarriff 03) Adam Kilgarriff. Thesauruses for natural language processing. In *Proceedings of NLP-KE*, pages 5–13, Beijing, China, 2003.

(Kilgarriff *et al.* 04) Adam Kilgarriff, Pavel Rychly, Pavel Smrz, and David Tugwell. The sketch engine. In *Proceedings of EURALEX 2004*, pages 105–116, 2004.

(Lauer 95) Mark Lauer. Corpus statistics meet the noun compound: some empirical results. In *Proceedings of the 33rd conference on Association for Computational Linguistics*, pages 47–54, Morristown, NJ, USA, 1995. Association for Computational Linguistics.

(Lin 98) Dekang Lin. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics*, pages 768–774. Association for Computational Linguistics, 1998.

(McLauchlan 04) Mark McLauchlan. Thesauruses for prepositional phrase attachment. In *Proceedings of CoNLL-2004*, pages 73–80. Boston, MA, USA, 2004.

(Okumura & Muraki 94) Akitoshi Okumura and Kazunori Muraki. Symmetric pattern matching analysis for english coordinate structures. In *Proceedings of the 4th Conference on Applied Natural Language Processing*, pages 41–46. Association for Computational Linguistics, 1994.

(Ratnaparkhi 98) Adwait Ratnaparkhi. Unsupervised statistical models for prepositional phrase attachment. In *Proceedings of the 17th International Conference on Computational Linguistics*, pages 1079–1085, 1998.

(Resnik 99) Philip Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11:95–130, 1999.

(Sparck-Jones 86) Karen Sparck-Jones. *Synonymy and semantic classification.* Edinburgh University Press, 1986.

(vanRijsbergen 79) C. J. van Rijsbergen. *Information Retrieval.* Butterworths, London, U.K., 1979.

(Weiss & Kulikowski 91) Sholom M. Weiss and Casimir A. Kulikowski. *Computer systems that learn: classification and prediction methods from statistics, neural nets, machine learning, and expert systems.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1991.

Pattern Ambiguity and its Resolution in English to Hindi Translation

Niladri Chatterjee Shailly Goyal Anjali Naithani
Department of Mathematics
Indian Institute of Technology Delhi
Hauz Khas, New Delhi - 110 016, India
{niladri_iitd, shailly_goyal}@yahoo.com

Abstract

A common belief about natural language translation is that sentences of similar structure in the source language should have translations of similar structure in the target language also. This paper shows that this assumption does not hold well always. At least eleven different patterns exist in the Hindi translation of English sentences having the main verb "have" (or any of its declensions). Traditionally such variations are termed as "translation divergence". However, typically a study of divergence considers some standard translation pattern for a given input sentence structure. A translation is said to be a divergence if it deviates from the standard pattern. However, this is not the case with the above-mentioned sentence structures as no standard translation pattern can be assumed for these cases. We term this ambiguity as "pattern ambiguity". In this ongoing work we propose a rule-based scheme to resolve the ambiguity using word senses.

1 Introduction

Natural language translation between two languages almost inevitably suffers from ambiguities of various types, such as, lexical ambiguity, semantic ambiguity, syntactic ambiguity (Dorr *et al.* 99). Typically, all these ambiguities are related to deciphering the inherent meaning of the source language sentence. However, with respect to English to Hindi translation a different type of ambiguity can be observed (Goyal *et al.* 04). Here, the problem is not in understanding the sense of the sentence, but in deciding the correct structure of the translation in Hindi. For illustration, consider:

> Ram has a pen ∼ *ram* (Ram) *ke pass* (near to) *ek* (one) *kalam* (pen) *hai* (is).
>
> Ram has fever ∼ *ram* (Ram) *ko* (to) *bukhaar* (fever) *hai* (is).

Despite the structural similarity of the above two English sentences their Hindi translations are structurally very different. This creates a different type of ambiguity to the translator, which we term as "pattern ambiguity". Note that pattern ambiguity is different from "translation divergence" (Dorr 93). Divergence occurs when the translation of a sentence deviates from some standard translation structure. But, pattern ambiguity does not assume any such standard structure. Rather, this ambiguity occurs when corresponding to different input sentences of the same structure different translation patterns are observed. Obviously, statistical techniques are incapable of resolving this ambiguity, and deep semantic analysis of source language sentences is needed to resolve this ambiguity.

With respect to English to Hindi translation we notice that the presence of pattern ambiguity is most prominent in dealing with English verbs. In particular, as many as eleven different translation patterns have been observed in the translation of English sentences where the main verb is "have" or some of its declensions. In this work, we propose a rule based scheme that takes into account senses of the underlying English verbs, and other constituent words of the sentence to resolve the ambiguity.

2 Translation Patterns of Different English Verbs to Hindi

In English often a single verb is used to convey different senses. For example, according to WordNet 2.0[1], the verb "run" has 41 senses, "call" has 28 senses, "take" has 42 senses. But, almost for each of these senses, a specific verb exists in Hindi. The use of the appropriate Hindi verb can be determined by identifying the sense in which the English verb is used. This helps in resolving pattern ambiguity for these verbs.

However, most interesting observation can be made with respect to the English verb "have". Although the number of possible senses for "have" is relatively less (only 19, as per WordNet 2.0), we have obtained as many as 11 trans-

[1] http://wordnet.princeton.edu/

lation patterns for sentences where "have" (or its declensions) is the main verb of the sentence. Further, depending upon the situation, there are variations in the verb used, or the case-ending used, or sometimes even in the overall translation structure. This makes pattern ambiguity to be a serious problem for English to Hindi translation while translating sentences of this type. Below we describe the different translation patterns that we observed in dealing with the English verb "have".

Translation Pattern P1 Here genitive case ending (*ka, kii, ke*) is used to convey the sense of the "have" verb. For example:

The school has good name ∼*vidyaalay* (school) *kaa* (of) *achchhaa* (good) *naam* (name) *hai* (is)

Which of the genitive case endings (i.e. *kaa, ke, kii*) will be used in a given sentence depends upon the number and gender of the object.

Translation Pattern P2 Here, the object and its pre-modifying adjective in the English sentence are realized in Hindi as the subject and subjective complement (SC), respectively. The subject of English sentence is realized as possessive case of the subject of the Hindi translation. For example,

Gita has beautiful hair ∼ *Gita* (Gita) *ke* (of) *baal* (hair) *sundar* (beautiful) *hain* (are)

Translation Patterns: P3 & P4 Here instead of genitive postposition, postpositions "*ke paas*" and "*ko*" are used, respectively. Section 1 shows one example of each P3 and P4.

Translation Pattern: P5 Here the postposition "*mein*" is used for conveying the sense of the verb "have". For example:

This city has a museum ∼ *iss* (This) *shahar* (city) *mein* (in) *ek* (a) *sangrahaalay* (museum) *hai* (is)

Translation Pattern: P6 Here, instead of "*mein*", another postposition "*par*" is used. Consider, for example, the following:

The tiger has stripes ∼ *baagh* (tiger) *par* (on) *dhaariyan* (stripes) *hain* (are)

Translation Pattern: P7 Here, the object of the English sentence is realized as an adjectival SC. For illustration:

She has grace ∼ *vah* (she) *aakarshak* (graceful) *hai* (is)

Despite the obvious differences all the above-mentioned patterns have one common feature: the main verb of the Hindi sentence is "*hai*" or any of its declension (*hain, thaa, the, thii, thiin*) which means "to be". But patterns P8 and P9, given below, illustrate cases when some other verb is used as the main verb instead of "*hai*" (or its declension).

Translation Pattern: P8 Pattern P8 occurs if the main verb of the Hindi translation is obtained from the object of the English sentence. For illustration, consider the following example:

Gita has regards for old men ∼ *Gita buzurgon* (old men) *kii* (of) *izzat* (respect) *kartii hai* (does)

The main verb of the Hindi sentence is *izzat karnaa*, which comes from the object "regards". In this respect, one may note that Hindi verbs are often made of a noun followed by a commonly-used verb. The verb "*izzat karnaa*" (to respect), is an example of this type.

Translation Pattern: P9 This is similar to pattern P8, but here the verb is not obtained from the object. Rather, a completely new verb is introduced in the Hindi translation. For example,

I had tea. ∼ *maine* (I) *chai* (tea) *pee* (drank)

But,

I had rice. ∼ *maine* (I) *chaawal* (rice) *khaaye* (ate)

Evidently, the verb of the translated sentence is obtained from the *sense* in which the verb "have" is used in the English sentence.

Translation Pattern: P10 If the English sentence has a component in the form of adjunct, then a variation in the translation may be noticed. For example, consider the two sentences

(a) Ram has two rupees.

(b) Ram has two rupees in his pocket.

While the translation of the first one is "*Ram ke pass do rupayaa hain*", the translation of the second one is "*Ram ki* (Ram's) *zeb* (pocket) *mein* (in) *do* (two) *rupay* (Rupees) *hain* (are)".

Translation Pattern: P11 This pattern is observed if, along with the subject, verb and object, the sentence has an infinitive verb phrase. For example,

My children had me buy the car ∼ *mere* (my) *bachchon ne* (children) *mujhse* (me) *gaadi* (car) *kharidvaayai* (buy)

Such a large variety of translation patterns pose great difficulty for any MT system, as it is required to take a decision regarding the pattern that will be most suitable for a given input sentence. In this work we study if a rule-based scheme can be developed to resolve this ambiguity.

3 How to Design Rules?

One may observe that translation patterns P10 and P11 are associated with specific English sentence structures. Hence obtaining rules for identifying these translation patterns are simpler. Lack of space prohibits us to discuss these rules in this paper. The sentence structure for rest of the patterns is <SVO>. Hence resolving pattern ambiguity requires detailed investigation of translation patterns P1 to P9. In this respect the following is observed.

3.1 Inadequacy of Subject/Object

We first observe that neither the subject nor the object of the sentence alone is sufficient to determine the translation pattern of the sentence. Table 1 and 2 illustrate this point. These examples highlight the inadequacy of the subject/object in determining the translation pattern.

English sentence	Hindi Translation	Translation Pattern
`Mohan has a good brain`	*Mohan kaa dimaag achchhaa hai*	P1
`Mohan has a good pen`	*Mohan ke paas ek achchhii kalam hai*	P3
`Mohan has high fever`	*Mohan ko tej bukhaar hai*	P4
`Mohan had a sweet apple`	*Mohan ne meethaa seb khaayaa*	P9

Table 1: Translation Patterns for Same Subject

English sentence	Hindi Translation	Translation Pattern
`Sita has flowers`	*Sita ke paas phool hain*	P3
`The tree has flowers`	*ped par phool hain*	P6
`The vase has flowers`	*phooldaan mein phool hain*	P5
`Meera has flowers in her home`	*Meera ke ghar mein phool hain*	P10

Table 2: Translation Patterns for Same Object

3.2 Rules Based on Senses of "Have"

WordNet 2.0 has been used to decide upon the sense of the "`have`" verb. Our observations in this regard are as follows.

a) Use of the verb "`have`" to convey senses numbered 5 (cause to move), 10 (be confronted with), 11 (experience), 13 (cause to do) and 19 (have sex with) is very rare.

b) Identification of translation pattern for eight senses (viz., 6, 8, 9, 12, 14, 15, 17 and 18) can be done by using their senses. As in all these cases only a single translation pattern can be observed (which in some cases is a mixed pattern!).

c) For sense numbers 1, 2, 3, 4, 7 and 16 more than one translation pattern is observed. Hence in these cases, the sense of "`have`" is not sufficient, and finer rules are required to determine the possible translation patterns of a given sentence

Table 3 summarizes our findings in this regard. This observation was made on the basis of our manual analysis of 6000 sentences with "`have`" as the main verb. We first worked on 2000 sentences, and corroborated our findings on the basis of the remaining. All the patterns obtained so far are given in Table 3.

The above observation suggests that even the sense of the verb is not enough to resolve the pattern ambiguity. For further investigation we took the help of WordNet's Lexicographer files. The lexicographer file information helps in identifying the *selectional restriction* (Allen 95) of subject's/ object's semantics of a sentence.

3.3 Rules Based on Lexicographer Files

Lexicographer files in WordNet 2.0 are the files containing all the synonyms logically grouped on the basis of syntactic category. For example, the file *noun.act* contains nouns that describe any act or action, *noun.animal* is a file containing nouns that are animals. According to WordNet, noun has 26 different senses. Corresponding to these senses there are 26 lexicographer files. Pronouns can be taken care of under these categories primarily as *noun.person*, or some other cases depending upon the context. We used these lexicographer files for designing rules for translation patterns. Further there can be imperative sentences where the subject "`you`" is silent (e.g. `Have this book.`). Thus we have 27 possibilities for subjects whereas 26 possibilities for objects for dealing with word sense disambiguation of "`have`".

Upon studying the subjects and objects of our database sentences a 27×26 matrix has been constructed. The matrix suggests the translation patterns obtained for different subject-object combinations. In our example base we found no sentences in which the subjects are one of *noun.motive*, *noun.phenomenon*, *noun.process*, *noun.feeling*, *noun.possession* and *noun.relation*. Also, there are no sentences in which the objects are *noun.motive* or *noun.relation*. So we

Sense Number	Definition (As given by WordNet 2.0)	Translation Pattern	Example sentence	Translated sentence
1	have or possess, either in a concrete or abstract sense	P1	Rita has two daughters.	*Rita ki do betiyaan hain*
		P3	She has a degree from IIT.	*us ke paas IIT kii degree hai*
2	have as a feature	P1	The dog has a tail.	*kutte kii ek poonch hai*
		P2	She has beautiful eyes	*uskii aankhen sundar hain*
		P5	The car has an airbag.	*gaadi mein airbag hai*
		P6	The tree has flowers.	*ped par phool hain.*
		Mixed P1 and P8	Ravi has a good grasp of subject.	*Ravi kii vishay par achchhii pakad hai.*
3	of mental or physical states or experiences	P2	Mita has an idea.	*Mita ke paas ek upaay hai*
		P3	Ram has sympathy for the poor.	*Ram ko gariibon ki liye shaanubhutti hai*
		P8	She has regards for her father.	*vah apne pitaa kii izzat kartii hai.*
		P9	She had a difficult time.	*usne mushkil samay bitaayaa.*
4	have ownership or possession of	P1	Hemu has three houses.	*Hemu ke teen ghar hain*
		P3	Mohan has a car.	*Mohan ke paas ek gaadii hai*
6	serve oneself to, or consume regularly	P9	I had an apple.	*maine ek seb khaayaa*
7	have a personal or business relationship with someone	P1	He has an assistant.	*us kaa ek sahaayak hai*
		P3	This professor has a research scholar.	*iss professor ke paas ek gaveshi hai*
8	organize or be responsible for	P1	John has a meeting.	*John kii ek meeting hai*
9	have left	P3	Meera has two years left.	*Meera ke paas do saal bache hain*
12	Suffer from; be ill with	P4	Paul has fever.	*Paul ko bukhaar hai*
14	receive willingly something given or offered	P9	Please have this gift.	*kripayaa yeh uphaar lein*
15	get something; come into possession of	P9	I have a letter from a friend.	*mujhe ek mitr kaa patr milaa.*
16	undergo (as of injuries and illnesses)	Mixed P1 and P8	Rama had a fracture	*Ram kii haddii tootii.*
		Mixed P4 and P8	His father had a heart attack.	*uske pitaa ko hradayaaghaat huaa*
17	achieve a point or goal	P9	Sachin had a century.	*Sachin ne shatak banaayaa.*
18	give birth (to a newborn)	P9	My wife had a baby boy yesterday.	*kal meri patnii ne ladkee ko janam diyaa.*

Table 3: Rules for Translation patterns for different senses of "`have`"

Subject	Object	Pattern Observed
noun.artifact	noun.artifact	P1 - 67, P2 - 35, P5 - 36, P6 - 45
Noun.group	noun.act	P1 - 34, P2 - 9, P4 - 8, P5 - 18
Noun.group	noun.attribute	P1 - 18, P2 - 7, P3 - 8, P5 - 17
noun.person	noun.act	P1 - 51, P2 - 34, P3 - 22, P4 - 8, P6 - 6, P8 - 16, P9 - 25
noun.person	noun.artifact	P1 - 25, P2 - 10, P3 - 35, P5 - 10, P10 - 24
noun.person	noun.attribute	P1 - 56, P2 - 34, P3 - 12, P4 - 4, P5 - 56, P6 - 23, P7 - 59, P8 - 13, P10 - 6
noun.person	noun.body	P1 - 15, P2 - 6, P3 - 6, P5 - 10, P8 - 14, P9 - 7
noun.person	noun.cognition	P1 - 35, P2 - 24, P3 - 35, P4 - 23, P5 - 25, P7 - 12, P9 - 8
noun.person	noun.communication	P1 - 24, P2 - 34, P3 - 29, P4 - 4, P5 - 15
noun.person	noun.feeling	P1 - 16, P3 - 6, P4 - 35, P5 - 25, P7 - 27
noun.location	noun.group	P1 - 7, P2 - 5, P5 - 24, P6 - 7
noun.person	noun.person	P1 - 17, P2 - 3, P3 - 4, P9 - 2
noun.person	noun.possession	P1 - 40, P3 - 16, P8 - 16, P9 -6, P10 - 13
noun.person	noun.state	P1 - 24, P2 - 35, P3 - 18, P4 - 16, P5 - 26, P6 - 8, P7 - 17, P8-25, P9 - 16
noun.person	noun.time	P1 - 7, P2 - 7, P3 - 13, P8 - 13

Table 4: Densely Occupied Cells

Subject Sense	Object Sense	Pattern
noun.act	noun.state	P1
noun.act	noun.substance	P5
noun.animal	noun.cognition	P2
noun.animal	noun.substance	P6
noun.group	noun.quantity	P1
noun.group	noun.substance	P3
noun.plant	noun.phenomenon	P8
noun.plant	noun.state	P5
Imperative sentence	noun.act	P9

Table 5: Singly Occupied Cells

discarded these columns and rows from the matrix. Thus the final matrix has $21 \times 24 = 504$ cells. A thorough scrutiny of the matrix reveals the following:

Case 1. Out of the 504 cells, 297 cells are empty i.e. no example has been found for corresponding combination of subject and object, e.g., subject is *noun.attribute* and object is *noun.animal*. Evidently, for these 297 situations no translation rules need to be formed.

Case 2. The simplest case is when we found one entry in each cell. There are 85 (out of 504) cells which have only one entry. For these subject-object combinations pattern ambiguity can be resolved easily. Some of these combinations are given in Table 5.

Case 3. We further observe that for some of the subjects, there are only two or three possible translation patterns irrespective of the object used. For example, if the subject is *noun.act* then pattern observed are P1 or P5. Similarly for some objects, only a limited number of patterns are possible. For example, if object is *noun.shape* then possible translation patterns are P5 or P6.

Case 4. Further, there exists some subject-object combinations with only two or three entries. For instance, if subject is *noun.artifact* and object is *noun.communication* then pattern observed are P5 or P6.

The advantage of observing Cases 3 and 4 is that to resolve ambiguity the system need not explore all the 11 possibilities. Rather, it may furnish two or three translations of the sentence and obtain user feedback. There is also scope of learning by the MT system, as it handles more cases of a particular type.

Case 5. However, there are 15 cells that are very dense, i.e. for these combinations of subject and object, the number of possible translation patterns is quite large. Table 4 shows these cells, the possible translation patterns, and the number of observations. Pattern ambiguity cannot be resolved for these sentences, since for each of the 15 cases a large number translation patterns are possible.

The question therefore arises whether pattern ambiguity in translating English sentences with "`have`" as its main verb is completely resolvable. We tried to capitalize on all possible sentential information, yet we have not been able to find a foolproof solution. So far, we could resolve pattern ambiguity for about 75% of cases (out of about 4000 sentences) using the above scheme. We feel that the only way it may be resolvable is by analyzing the context. But creating a large database containing appropriate context information as well as having "`have`" sentences is not easy task. Currently we are looking into this aspect.

4 Concluding Remarks

This paper defines the term "pattern ambiguity" that is observed in translation from English to Hindi. This ambiguity is particularly prominent and not yet fully resolvable for sentences whose main verb is "`have`" or its declensions. The primary reason behind this ambiguity is that Hindi does not have a verb that is equivalent in sense to the English "`have`" verb. However, not only Hindi, many other languages (e.g. Bengali, Hausa[2]) do not have any possessive verbs. We feel that this study will be helpful for studying translation patterns of such languages as well.

In this work we have used verb senses and subject-object senses separately. We feel that the problem may be dealt with at a more granular level by considering these two senses together for a given input sentence. Presently we are focusing our investigations to that direction.

References

(Allen 95) James Allen. *Natural Language Understanding*. Benjamin Cummings, California, 2nd edition, 1995.

(Dorr 93) Bonnie J. Dorr. *Machine Translation: A View from the Lexicon*. MIT Press, Cambridge, MA, 1993.

(Dorr *et al.* 99) Bonnie J. Dorr, Pamela W. Jordan, and John W. Benoit. *Advances in Computers*, volume 49, chapter A Survey of Current Research in Machine Translation, pages 1–68. Academic Press, London, 1999.

(Goyal *et al.* 04) Shailly Goyal, Deepa Gupta, and Niladri Chatterjee. A study of Hindi translation pattern for English sentences with "have" as the main verb. In *ISTRANS-2004*, pages 46–51, New Delhi, India, 2004.

[2]http://www.humnet.ucla.edu/humnet/aflang/Hausa/Hausa_online_grammar/grammar_frame.html

Partial Forest Transfer for Spoken Language Translation

Tetsuro CHINO and **Satoshi KAMATANI**
Knowledge Media Laboratory
Corporate Research & Development Center
Toshiba Corporation
1, Komukai Toshiba-Cho, Saiwai-Ku,
Kawasaki, 212-8583, Japan
tetsuro.chino@toshiba.co.jp

Abstract

In this paper, a new method for Spoken Languages Translation (SLT) is proposed. First, we developed, 1) a robust grammar for Spoken Language (SL) and, 2) a robust forest parser. Then, we proposed, 3) a new method what we call "Partial Forest Transfer (PFT)" that enables us to apply linguistic and/or semantic knowledge to the ambiguous candidates coinstantaneously, without any loss ascribable to unfounded hypothesis. We also describe the implemented experimental Japanese to English SLT System. The preliminary evaluation results (parsing accuracy : 97.1% within 2.43 best parses, improvements in SLT capability : +32.9%) demonstrates high validity of proposed methods.

KEYWORDS: Spoken Language Translation, Syntactic Forest, Transfer.

1 Introduction

There are strong demands on practicable spoken language translation (SLT). But some burdensome characteristics of spoken languages (SL), such that, (a) different vocabularies, (b) syntactic ill-formedness/disfluency, (c) paralinguistic phenomena, (d) high context/situation dependency, other than stemmed from automatic speech recognition (ASR), and speech synthesis (SS), cause hardness for conventional NLP for written languages (WL) to treat SL appropriately. Since we regard the robust processing technology as one of the most promising key to overcome these problems, we have developed the technologies described in the rest of this paper,

2 Japanese Spoken Language Analysis

2.1 Robust Grammar for Japanese SL

Figure 1 shows our CFG rules fundamentally designed to capture the clause structure (C) of Japanese SL. In addition, we introduced the ill-formedness marker ($weak$) on each category that represents a syntactically ill-formed or incomplete substructure [1]. Figure 2 is the syntactic structure

S	$\rightarrow$	C			
S	$\rightarrow$	$comp\ \ C$	$comp$	$\rightarrow$	$C\ \ marker$
C	$\rightarrow$	VP	PP	$\rightarrow$	$N\ \ P$
VP	$\rightarrow$	$PP\ \ VP$	PP	$\rightarrow$	PP_{weak}
VP	$\rightarrow$	V	PP_{weak}	$\rightarrow$	N

Figure 1: Example of Robust CFG Rules

for an example of spoken Japanese sentence with two missing particles denoted as (ϵ).

君 kimi (you)	(ϵ)	作った tukutta (make)	のは nowa (thing)	どこ doko (where)	(ϵ)	しまった. simatta (store)
(*Where did you store the thing you made?*)						
$S($	$comp($	$C($	$VP($	$PP($	$PP_{weak}($	$N($[君/you]) ϵ))
				$VP($	V([作った/make]))))	
		$marker$[のは/thing]))				
	$C($	$VP($	$PP($	$PP_{weak}($	N([どこ/where]) ϵ))	
			$VP($	V([しまった/store]))))))		

Figure 2: Example of Syntactic Structure for ill-formed Japanese SL

2.2 Robust Forest Parser

A robust grammar is necessary to accept spontaneous SL. But, a robust grammar often leads a huge number of parses in parsing process. So, it is next to impossible to get the optimum parse first, in sequential parsing process guided by the robust grammar. Consequently, we utilized a forest parsing method to get all possible parses expressed in a syntactic forest. We developed a robust forest parser that is an extension of GLR parser (Tomita 91). Figure 3 shows the algorithm of our parser. This parser enables us to get the *preferred forest* that consist of only and all syntactically preferred parses.

Figure 4 shows the *accepted forest* (left) and the *preferred forest* (right) for the example of spoken Japanese sentence. In this case, while the *accepted forest* contains 1,537 parses analyzed by our robust grammar, our robust forest parser extracts

[1] These rules enabes us to capture the linguistic phenomena of "*Joshi − Ochi*"(omission of case marker/particle) that often occur in Japanese SL.

1. Perform GLR parsing normally. (*accepted forest*).
2. Estimate well-formedness of each parse in the Forest.
   (a) Penalize on all *vertices* with *weak*.
   (b) Aggregate the possible minimum penalty for each *subforest*, in a bottom up fashion.
3. Extract *preferred forest*.
   (a) Select *vertices* with the lowest penalty in each packed *subforest* as the preferred *vertices*, in an iterative top-down fashion.
   (b) Extract the *preferred forest* consisting of only (all) preferred *vertices*.

Figure 3: Algorithm of Robust Forest Parsing

the preferred forest with only 66 parses. The ambiguity is successfully reduced to less than one twenty thirds.

スープ	の	おいしい	レストラン	を	予約	し
suupu	no	oisii	resutoran	wo	yoyaku	si
(soup)		(delicious)	(restaurant)		(reserve)	(do)

なく	ちゃ	いけ	ない	ん	でしょ.
naku	tya	ike	nai	n	desho
(not)	(if)	(good)	(not)		(*tag question*)

(*I should reserve a restaurant serves delicious soup, shouldn't I ?*)

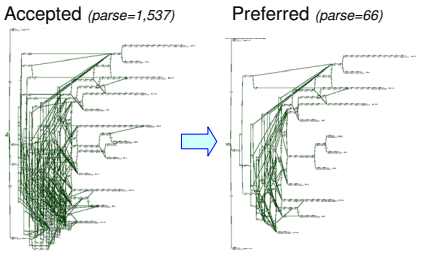


Figure 4: Example of *accepted* and *preferred forest*

3 Partial Forest Transfer (PFT)

In this section we propose a new processing method for spoken language translation.

3.1 Transfer

The "transfer" mechanism is widely and successfully utilized in conventional Rule based Machine Translation (RBMT). In addition, we have insightful linguistic resources for conventional RBMT continuously refined by agelong efforts. Many differences between SL and WL prevents direct usage of these resources. But there are small differences between SL and WL within clauses. So we try to translate SL as follows. 1) Extract the clause structures of SL by the robust forest parser with robust grammar. Then, 2) transfer the forest by getting the best of the conventional linguistic resources and transfer mechanisms, to translate within clauses of SL.

But, conventional transfer cannot treat ambiguous structures. So we propose PFT mechanism to subdue the difficulties and ambiguities of SL, by extending transfer to convert a syntactic forest to another syntactic forest directly.

3.2 Partial Forest

We introduce a notion of *partial forest* that is a partial specification of syntactic forests, defined recursively.

A *partial forest* is a triple $< t, L, c >$. Where, t is a *top node* that specifies of a *top vertex* of *subforest* f. L is an ordered list of *nodes* dominated by t such that the span of L covers some part of span of f,without any overlaps nor gaps. c is a positional constraint that specifies positional constraint on the span of L in the span of f.

A *node* is, a *leaf node* that specifies a leaf of syntax forest, or an *internal node* that specifies an *internal vertex*, or a *partial forest*.

The internal structures between each *top node* and their constituents are excluded in the *partial forest*, since each *top node* do not have to dominate their constituents (*nodes*) directly. In addition, the internal structure of each *internal node* is still a forest, so the ambiguity will be preserved. These discriminating characteristics of *partial forest* enables us to handle ambiguous syntactic forests containing many possible parse candidates in one structure.

We refer to a *node* (include *top node*) as *trigger* that will be a variable (of PFT Rules) to be assigned to some *vertex* in the forest.

In each layer of *partial forests*, between the *top node* and *nodes* in their ordered list has a (direct or indirect) *dominant* relation.

3.3 PFT Rules

PFT rules are a set of declarative rewriting rules for syntactic forests. A PFT rule is a pair of a Matching Pattern (MP) and a Target Pattern (TP). An MP is expressed as a *partial forest*, with the structural preconditions on the forest to be applied the rule. A TP is structural template

to be referred to reconstruct structure of the forest.

Figure 5 shows examples of *PFT Rule*, where the diagram left to the arrow denotes the MP, and the diagram right denotes the TP, in each rule.

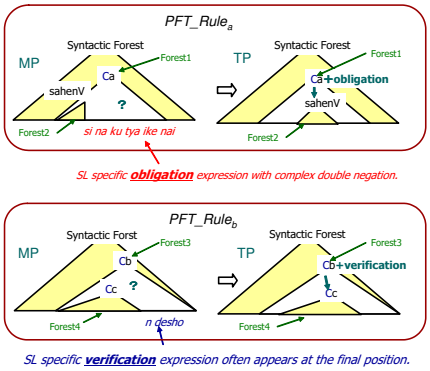


Figure 5: Examples of PFT Rule

The MP of *PFTRulea* expresses the following preconditions on the forests. A *subforest* $Forest_1$ under the *vertex1* labeled with category C_a exists in the forest. And, the $Forest_1$ consist of, and just covered with[2], the sequence of another *subforest* $Forest_2$ dominated by a *vertex2* labeled with category *sahenV*, and a sequence of *leaves* "si/na/ku/tya/ike/nai". On the other hand, the TP of *PFT Rule*$_a$ expresses the following reconstruction for the forest. Add an *attribute* (obligation) on the *vertex1*, and link $Forest_2$ under the *vertex1*. (Then, remove all other substructures within $Forest_1$, and their dependents[3].) Since the linguistic substructure specified by the MP of *PFT Rule*$_a$, forms a complex double negation, it is relatively hard to treat by conventional NLP for WL. But, this rule captures a fixed Japanese SL specific expression, and convert the forest into more simple forest with an adequate *attribute* (obligation) to interpret it.

The *PFT Rule*$_b$ captures a substructure under a clause (C_c) consist of another clause (C_d) and a sequence of leaves "n/desho" at the rightmost position of the forest. This *PFT Rule*$_b$ enables to capture Japanese SL specific expressions for verification, and convert it into more simple forest with adequate *attribute*.

[2]Since the *positional constraint* (not shown in the figure) was *whole*.

[3] *Vertices* that cannot exist in the forest without some other *vertices*.

3.4 PFT Algorithm

Given a set of PFT rules R_{all}, and a forest f, proposed PFT mechanism convert the structure of f, according as the algorithm shown in Figure 6. The term *assignments* means a substitution of particular *vertex*(*assignee*) in the forest into one *trigger* in the PFT rule. The term *binding* means a set of *assignments* for all *triggers* in one PFT rule.

1. Given a $forest\ f$.
2. $Loop_1$: (Repeat)
   - 2.1. Gather all candidate $rules\ r \in R \subseteq R_{all}$ where f meets $precondition$ of r.
   - 2.2. if $\mid R \mid = 0$ then exit $Loop_1$.
   - 2.3. $Loop_2$: For each candidate $rule\ r_i \in R$.
     - 2.3.1. Gather all $triggers\ t \in T_i$ from r_i
     - 2.3.2. $Loop_3$: For each $trigger\ t_{ij} \in T_i$.
       - 2.3.2.1. Gather all possible *assignments* $< t_{ij}, v > \in A_{ij}$, where v is a *vertex* within f, and v is possible *assignee* of t_{ij}.
       - 2.3.2.2. if $\mid A_{ij} \mid = 0$ then remove r_i from R, and exit $Loop_3$.
     - 2.3.3. if $\mid R \mid = 0$ then exit $Loop_1$.
     - 2.3.4. Create all double of, candidate $rule\ r_i$, and combinations of *assignments*, as possible *bindings* $B = \bigcup_{r_i \in R}(< r_i, AL_i >)$. where $AL_i = (\prod_j A_{ij})$
   - 2.4. $Loop_4$: For each possible $binding\ b_k \in B$.
     - 2.4.1. Estimate $applicability$ of b_k on f.
     - 2.4.2. if b_k is not *applicable* on f then remove b_k from B.
   - 2.5. if $\mid B \mid = 0$ then exit $Loop_1$
   - 2.6. Select the *optimum binding* $b_o \in B$ based on *selection criteria*.
   - 2.7. Apply b_o on f (Reconstruct f based on b_o).
3. Output f as the *transferred forest*.

Figure 6: Algorithm of Partial Forest Transfer

3.4.1 Applicability of Bindings

Given a possible binding $b_{ik} =< r_i, al_k >$, where $al_k \in AL_i$ is one combination of possible *assignments*. The *binding* b_{ik} is *applicable*, if for all *vertices* as all *assignees* of all *triggers* in r_i, 1) all positional constraints in r_i is satisfied, and 2) all *dominant* relations in r_i is satisfied, 3) there is at least one parse in the current forest f that contains all *vertices* as all *assignees* for all *assignments* $a \in al_k$.

3.4.2 Optimum Binding Selection

The *optimum binding* is selected from *applicable bindings* set, based on predetermined *selection criteria*. The *selection criteria* concerns, the width of the span for *top node* in the forest, the sorts of restriction on the *positional constraints*, the number of *triggers*,

and the degree of *partial specifications* in MP, and so on.

3.4.3 Reconstruction of Forest

Given a *forest* f_0, and an *optimum binding* b_o, the PFT algorithm reconstructs f_0 as follows. 1) Extract all *subforests* f dominated by the *vertex* that is *assignee* of *trigger* in TP of r_o. 2) Remove all *vertices* under *top node* except *top node*. 3) Remove all *vertices* that depend on the removed *vertices* in step 1 and 2. Then, 4) reconstruct f recursively based on the TP of r_o, by adding specified *attributes*, and link specified *subforest* f, onto the *vertex* that is the *assignee* of *top node* in each layer of TP.

In this process, bootless substructures are removed from the syntactic forest, so the ambiguity of the forest be reduced for corollary.

3.5 Example

Figure 7 shows an example of the *optimum bindings* for the *preferred forest* shown in Figure 3. In this case, the *PFT* $Rule_b$ was used in the first *optimum binding*, and the *PFT* $Rule_a$ was used in the next cycle (of $Loop_1$).

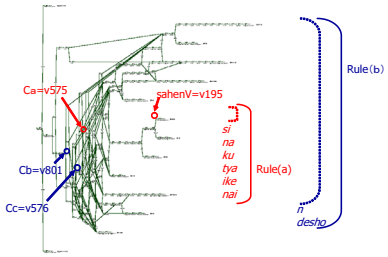


Figure 7: Example of Optimum Binidings

After the reconstruction based on these two *optimum bindings*, the *preferred forest* is converted into the *transferred forest* shown in Figure 8. This result is not a tree but still a forest that includes 3 parses. The PFT mechanism can transfer a forest to another forest directly, and can preserve uncertain ambiguity.

This *transferrd forest* is almost same syntactic structure for the Japanese WL that has same meaning for the inputted Japanese SL. In addition, this result comes with the appropriate *attributes* (*i.e.* obligation and verification) to translate exactly as intended in the original SL.

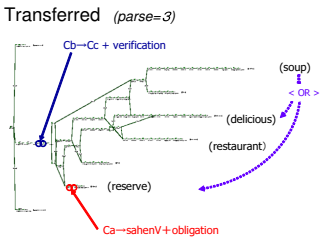


Figure 8: Example of *transferred forest*

4 Forest Dependency Analysis

We also developed the Forest Dependency Analysis (Kamatani *et al.* 05) to get semantically most preferable *SL Dependency Structure* based on semantic preferences. Figure 9 shows the overview of this method. The semantic preferences are learned from a large corpora (7M sentences from newspapers for 7 years, (Mainichi 02)), via 28M of cooccurrence pair, and 100 hidden classes, by an extended method of (Torisawa 01). Figure 10 shows sample result.

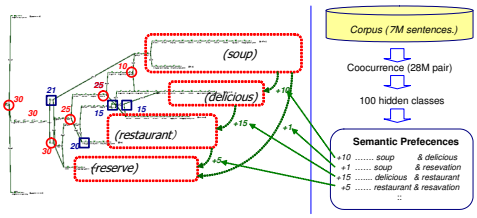


Figure 9: Outline of Forest Dependency Analysis

5 Japanese to English SLT System

We have implemented an experimental Japanese to English SLT system. We will give an overview of this system in this section.

5.1 System Configuration

Figure 11 shows the configuration of implemented SLT system. The *Source Language* provided from ASR is parsed by the Robust Forest Parser (section 2.2), work with Robust Grammar (section 2.1) and the *preferred forest* is extracted.

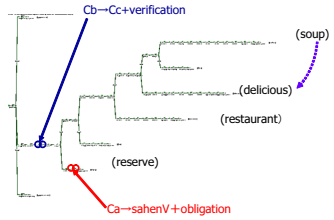


Figure 10: Example of SL Dependency Structure

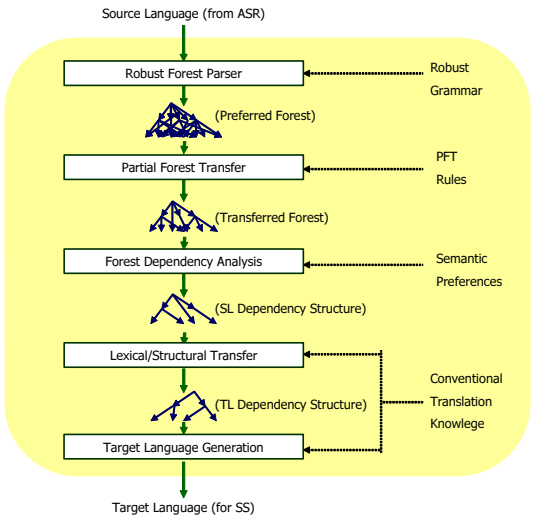


Figure 11: SLT System Configuration

Then, the *preferred forest* is converted by the Partial Forest Transfer (section 3) mechanism work with PFT rules (section 3.5), into *transferred forest*. In the next stage, semantically preferred *SL Dependency Structure* (tree) is extracted by the Forest Dependency analysis (section 4) work with semantic preferences, After that, the semantically preferred *SL Dependency Structure* is translated into *Target Language* by the Lexical/Structural Transfer with the Target Language Generation by utilizing the conventional translation knowledge, via *TL Dependency Structure*. Finally, translated *Target Language* is sent to Speech Synthesis.

5.2 Preliminary Evaluation

We perform the following preliminary evaluation by three Test Sets.

- Test Sets: (conversation corpus, open data)

  Set1 = [3,033 sentences, 8.2 words/sent.]
  Set2 = [70 sentences, 8.3 words/sent.]
  Set3 = [199 sentences, 12.3 words/sent.]

- Coverage:

  All sentences were accepted, for Set1. The average number of parse were 163.6 in the *accepted forests*, 4.64 in the *preferred forests*, 3.06 in the *transferred forests*.

- Accuracy:

  The 97.1% of final forests include the correct parse, for Set2. The average number of parse was 2.43.

- Validity:

  The 32.9% of sentences that cannot be translated by conventional MT for WL, are successfully translated by the SLT system, for Set3.

6 Discussion

The result shown in the previous section demonstrates that, 1) our robust forest parser and robust grammar have wide coverage, and can accept Japanese SL robustly, and 2) can extract the best parses in high accuracy within small number of candidates, and 4) proposed PFT mechanism has high validity for SLT application.

In addition, proposed PFT mechanism has an advantage that it enables apply syntactic and/or semantic knowledge to all possible parses in the forest at same time, in early stage of process, without any loss ascribed from uncertain hypothesis that often necessary in the conventional approaches.

7 Conclusions and Future Works

In this paper a new processing method "Partial Forest Transfer" for spoken language translation is proposed. In addition, an experimental implementation of SLT system are also described.

The evaluation results demonstrates high performance and high validity of proposed methods for spoken language translation application.

The future works are, 1) minimization of influence of ASR failures in SLT system, and 2) utilization of paralinguistic clues.

References

(Kamatani *et al.* 05) S. Kamatani, T. Chino, and K. Kimura. Syntax forest based syntactic and dependency analysis towards robust spoken language processing. In *Proc. PACLING '05*, Hino, Japan, 2005.

(Mainichi 02) Mainichi. Mainichi shinbun (newspaper) cd-rom, 1995,1996,1997,1999,2000,2001,2002. (7 years).

(Tomita 91) M. Tomita. *Generalized LR Parsing*. Kluwer Academic Publishers, MA, USA, 1991.

(Torisawa 01) K. Torisawa. An supervised method for canonicalization of japanese postopositionals. In *Proc. of (NLPRS) '01*, pages 211–218, Tokyo, Japan, 2001.

RAPID METHODS FOR OPTIMAL TEXT SELECTION

Rahul Chitturi,Sebsibe H Mariam, Rohit Kumar
Speech Lab, LTRC
IIIT- Hyderabad
INDIA
rahul_ch@students.iiit.net,{sebsibe,rohit}@iiit.net

Abstract

The performance of a speech recognition system depends mainly on the training data In this paper we select an optimal set of sentences for training the speech recognition systems, from a huge text corpus that is available for a language. Greedy solution is the best-known solution for optimal text selection [1]. But the time it takes to obtain the optimal text is in the order of days. Using the approach that is described in this paper, optimal text can be obtained in a few hours. Also, the use of selection of words instead of sentences is intimated, to reduce the size of optimal text. This algorithm was tested on four Indian languages: Hindi, Telugu, Tamil, and Marathi and results were compared with that of the greedy algorithm. This algorithm can be even used to test various selection criteria for optimal text selection in a short amount of time. The problems in generating optimal text for training speech recognition systems are also discussed.

1 Introduction

Designing the speech corpus is one of the key issues in building high quality automatic speech recognition or synthesis systems. The quality of an automatic speech recognition system is implicitly related to the data that it is trained on. For training a computer with speech data, there should be a good coverage of basic units of the speech in the data.

The basic units of speech that are considered in this work are diphones. Logically the accuracy of an ASR system will be good if triphones are considered rather than diphones. But for covering each triphone, the data that has to be collected would become very large and triphones are relatively inefficient decompositional units due to the large number of frequently occurring patterns. Moreover, since a triphone unit spans an extremely short time-interval, such a unit is not suitable for integration of spectral and temporal dependencies [10]. The phone coverage is not generally considered, as the aim of this algorithm is to generate the optimal text for large vocabulary speech recognition systems, all the phones can be covered in the order of tens of sentences. So, we used diphones to get the optimal set of utterances. A speech recognition system can be directly built using this generated text.

The amount of data that has to be collected depends on the number of speakers and the number of sentences that are supposed to be spoken. For the speaker independent recognition systems, data from a large number of speakers is needed for training. From the speaker.s point of view, it is more comfortable to read small set of utterances. The speaker cannot be expected to read a long sentence in a single stretch. Also there is more probability to get a wrong alignment with long sentences [11]. Based on the specifications of number of users and their comfort, the requirements of the training data for a speech recognition system [7],[3] and the type of application that is going to be built, the ranking criteria is decided. For example if we are building a speech recognition system for Railway Domain, the words which are more probable to be used in that domain is given higher rank than the other sentences. This would train the ASR specific to that domain.

As the main aim of this algorithm is to select the optimal data, the sentence that adds redundant data should be given a negative score. Since our aim is to select all the diphones optimally, selecting a sentence to cover a rare diphone adds redundant data, because that sentence will include the frequent diphones are highly probable to occur in the already selected sentences. Selecting redundant data also increases the time for processing. Instead, if a word is selected for that diphone, that diphone will be covered in a small utterance. So, in our case, the speakers were given 30 different sentences and 15 different words each.

A fast optimal selection is highly desirable in

building Speech Recognition Systems. Many a time, it takes several days to get this optimal text. Even for a small change in the selection criteria, one will have to wait for another stretch of days, leading to greater delay.The tradeoff of building a fast optimal text generator using our approach is that the quality of this data decreases minutely. The comparison of this algorithm and the greedy algorithm is shown in the section 4.

2 Algorithm

Many researchers use the greedy algorithm for speech corpus design [5]. The best-known algorithm is the greedy algorithm as applied to set covering problem [9]. In this greedy approach a sentence is taken from the corpus that has the maximum features of our requirements. The main feature for the optimal text selection would be generally, selecting the sentence that has the maximum number of diphones or some rank or cost based on the type of the diphones that sentence has. In each iteration the algorithm selects a sentence that has maximum number of diphones and removes that sentence from the corpus. The diphones that are already covered are marked. In the next iteration the next sentence that has the maximum number of new diphones is selected. One can give more weightage to the diphone that occurs rarely. As a result of this only one sentence is selected in one iteration which would take a considerable amount of time. So, if hundreds of lines are to be selected one can imagine the time it takes.

2.1 Basic Algorithm

The algorithm that is implemented in this paper is threshold based automatic algorithm. In this a threshold is set on the number of new diphones that it has to cover for selecting a sentence. The maximum and the minimum number of words that a sentence can have, is fixed in the beginning. The first line is selected in the beginning. A new sentence is selected if the number of new diphones is greater than certain threshold. Initially this threshold will be initialized to a big value of the order of tens. Threshold initialization is highly dependent on the type of the problem we are dealing with. This value has to be set by the user based on the experience. So, at a single stretch a good number of sentences can be selected meeting the requirement of the maximum coverage of diphones in minimum amount of time. In the next few iterations, the decrease in the threshold should be high so as to speed up the process. Then the threshold is decreased moderately till 60-75% of the total diphones are covered. Then the threshold is decreased slowly till the threshold becomes one. Generally in most of the languages the threshold becomes 1 when the diphone coverage is around 85%. The number of words that should be collected can be fixed if a large number of words have to be selected. Then sentences should be selected even though the threshold becomes one. For the rest of the diphones which are not selected either because of the threshold or fixing the minimum number of words, words are selected.

2.2 Application of Algorithm

This algorithm can be applied for collecting speech data for any language independent of any factors. Since time is the major factor in determining the effective usage of the system, we can follow two approaches. For a rapid application of the algorithm, where there is no need of any specific requirements or features that affect the speech recognition system, the approach described in section [2.2.1] can be followed. If there are some requirements for the speech recognition like for example, the phones of some vowels or consonants should be given more importance, or some kind of ranking, then approach described in section [2.2.2] can be followed.

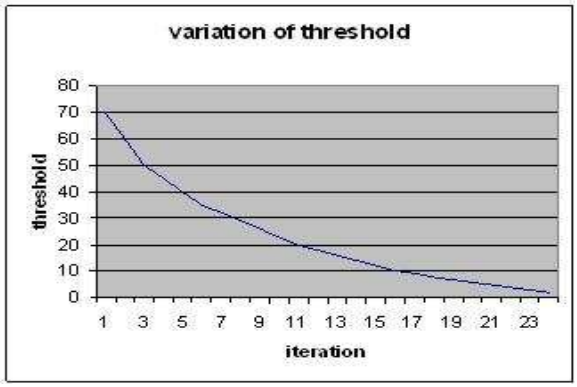


Figure 1: Variation of threshold

2.2.1 Approach for building rapid applications

Using the algorithm [2.1], a set of sentences is collected covering all the diphones in that language. But if sufficient amount of coverage of

each diphone is needed then, another set of sentences should be selected excluding the previous sentences that are already selected, covering all the diphones. Since the sentences that are already selected will not be selected, some diphones will be missed in those sentences. But as the words are collected for unselected diphones, the words from the original corpus should be selected. A hash is kept for each diphone, which contains the words from the corpus which have that diphone. Since for rare diphones there may not be many words, some words may be repeated.

2.2.2 Approach for building specific applications

Since the ranking of the sentences has some problems with respect to the time constraints and the structure of the sentences that will be selected, first the sentences which are phonetically rich can be obtained from the corpus using this algorithm, and then the ranking can be done after that. As this algorithm takes less amount of time, first some large number of sentences can be selected. For example, 10% of the entire corpus can be selected using this approach. Basically, a small corpus which is the representative of the whole corpus is obtained using this algorithm. In the algorithm [2.1], a small modification can be done that 100% diphone coverage can be attained, since a large number of sentences are selected, there is no need of selecting any words.

3 Benefits

To generate a set covering all the diphones, generally this algorithm iterates not more than 25-30 times. This greatly decreases the amount of time that is taken by the previous algorithm which iterates hundreds of times depending on the number of sentences in the corpus, corpus.s structure, and the structure of the language that is being modeled.

The number of sentences that are selected with this algorithm becomes approximately half the number of sentences that are selected by the greedy algorithm. This is mainly because of the additional words which are selected by this algorithm. So, the speech of both the sentences and the words should be collected. A speaker feels more comfortable reading a set of sentences and words, rather than a set of same number of sentences.

4 Tests and Results

We call our algorithm as "Threshold algorithm". Table 1 gives the details of the diphones in all the four languages in our corpus Table 2 shows the

Language	Diphones
Telugu	3240
Tamil	1820
Marathi	1778
Hindi	3138

Table 1: Diphones

comparison of time for the 4 languages with respect to the greedy algorithm Taking the greedy.s

Language	Greedy (Sec)	Threshold (Sec)
Telugu	66034	681
Tamil	133465	9732
Marathi	22493	1127
Hindi	87398	4490

Table 2: Time Performance

result as 100% the following is the performance graph for the time taken to select the set of sentences covering all diphones. It is assumed that

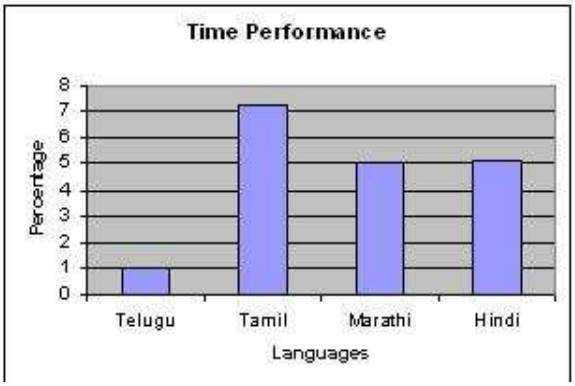


Figure 2: Time performance

there are approximately 20 words in each sentence. So, words collected in the threshold approach are converted into sentences. Actually, if only sentences have to be selected then this approach selects negligible number (usually 5-10) of more sentences than the greedy algorithm. This is the tradeoff with this algorithm.

Table 3 shows the comparison of number of sentences selected for the 4 languages for getting a set of sentences for covering all diphones.

Taking the greedy.s result as 100% the following is the performance graph based on the number

Language	Greedy (sentences)	Threshold (sentences)
Telugu	1268	443
Tamil	698	275
Marathi	681	265
Hindi	1082	476

Table 3: Text Performance

of sentences selected to get a set of sentences covering all diphones.

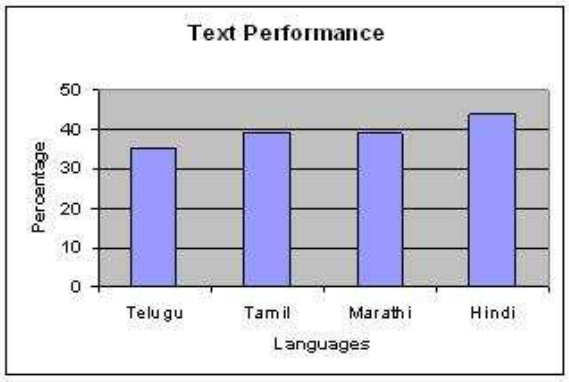


Figure 3: Text Performance

5 Acknowledgements

Thanks to our guide Mr. S.P.Kishore, LTI, CMU.

This work is sponsored by Hewlett Packard Labs, Bangalore, India with the collaboration of Carnegie Melon University, USA and IIIT-Hyderabad, India.

We are very thankful to each one of the members of our team (J.Natraj, C.Balakrishna, GopalaKrishna, Jithendra, and Gaurav Somani) who worked on this.

6 Future Work

This paper discussed several methods for optimal text selection for speech recognition in rapid domains, where time taken and the number of sentences selected are the criteria. There can be many other criteria that a system can use to generate task dependent speech systems. For such systems, we can build some other algorithms depending on the situation.

7 References

1] Vanten, J P. H. and Buchsbaum, A. L., .Methods for optimal text selection ", Proc. of Eurospeech, p. 553-556, Rhodes, Greece, 1997.

2] Beutnagel, M. and Conkie, A., .'Interaction of Units in a Unit Selection Database", Proc. of Eurospeech, Budapest,Hungary, 1999, p. 1063-1066.

3] Ms, B., .' Rare Events and Closed Domains: Two Delicate Concepts in Speech Synthesis ", Proc. of 4th ISCA Speech Synthesis Workshop, Scotland, 2001.

4] van Santen, J P. H., .Combinatorial issues in text-tospeech synthesis", Proc. of Eurospeech, p. 2511-2514, Rhodes, Greece, 1997.

5] Frans, H. and Board, .The Greedy Algorithm and its Application to the Construction of a Continuos Speech Database", Proc. of LREC, Las Palmas de Gran Canaria, Spain, 2002.

6] Frans, H. and Board, .Design of an Optimal Continuous Speech Database for Text-To-Speech Synthesis Considered as a Set Covering Problem ", Proc. of Eurospeech, Aalborg, Denmark, 2001.

7] Gauvain, J.F., Lamel, L.F., and Eskenazi, M.," Design Considerations and Text Selection for BREF, a Large French Read Speech Corpus ", Proc. of ICSLP, Kobe, Japan, 1990.

8] Zhu, W., Zhang, W., Shi,Q., Chen, F., Li, H., Ma, X. And Shen, L.,"Corpus Building for Data-Driven TTS System", Proc. of the IEEE TTS 2002 Workshop, Santa Monica, USA 2002.

9] T.Cormen, C. Leiserson and R. Rivest. Introduction to Algorithms. The MIT Press, Cambridge, Massachusetts, 1990.

10]A. Ganapathiraju et al, "Syllable - A Promising Recognition Unit for LVCSR," Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop, pp. 207-214, Santa Barbara, California, USA, December 1997.

11] Integrated Natural Spoken Dialogue System of Jijo-2 Mobile Robot for Office Services Toshihiro Matsui et al, Nobuyuki Otsu Proc. of AAAI-99, Florida, July 1999

Towards large-scale, open-domain and ontology-based named entity classification

Philipp Cimiano and **Johanna Völker**
Institute of Applied Informatics and Formal Description Methods
University of Karlsruhe

Abstract

Named entity recognition and classification research has so far mainly focused on supervised techniques and has typically considered only small sets of classes with regard to which to classify the recognized entities. In this paper we address the classification of named entities with regard to large sets of classes which are specified by a given ontology. Our approach is unsupervised as it relies on no labeled training data and is open-domain as the ontology can simply be exchanged. The approach is based on Harris' distributional hypothesis and, based on the vector-space model, it assigns a named entity to the contextually most similar concept from the ontology. The main contribution of the paper is a systematic analysis of the impact of varying certain parameters on such a context-based approach exploiting similarities in vector space for the disambiguation of named entities.

1 Introduction and Related Work

Named Entity Recognition (NER) systems have typically considered only a limited number of classes. The MUC named entity task (Hirschman & Chinchor 97), for example, distinguishes three classes: PERSON, LOCATION and ORGANIZATION, and the CoNLL[1] task adds one more: MISC, while the ACE framework[2] adds two more: GPE (geo-political entity) and FACILITY. Further, it has often been shown that it is relatively easy to recognize the PERSON and ORGANIZATION classes due to certain regularities, which renders MUC-like named entity recognition tasks even easier.

In this paper we propose a more challenging task, i.e. the classification of named entities with regard to a large number of classes specified by an ontology or more specifically by a concept hierarchy. Our approach aims at being open-domain in the sense that the underlying ontology and the corpus can be replaced. In our view this aim can only be accomplished if one resorts to an unsupervised system since providing labeled training data for a few hundred concepts as we consider in our approach is often unfeasible. Some researchers have addressed this challenge and have considered a larger number of classes. (Fleischman & Hovy 02) for example have considered 8 classes: ATHLETE, POLITICIAN/GOVERNMENT, CLERGY, BUSINESSPERSON, ENTERTAINER/ARTIST, LAWYER, DOCTOR/SCIENTIST and POLICE. (Evans 03) considers a totally unsupervised scenario in which the classes themselves are derived from the documents. (Hahn & Schnattinger 98) consider an ontology with 325 concepts and (Alfonseca & Manandhar 02) consider 1200 WordNet synsets. In our approach we consider an ontology consisting of 682 concepts.

Named entity recognition and classification has been so far mainly concerned with supervised techniques, the obvious drawback here being that one has to provide labeled training data for each domain and set of classes (compare (Sekine *et al.* 98; Borthwick *et al.* 98; Bikel *et al.* 99; Zhou & Su 02; G. Pailouras & Spyropoulos 00; Isozaki & Kazawa 02; Chieu & Ng 03; Hendrickx & van denBosch 03)). However, when considering hundreds of concepts as possible tags, a supervised approach requiring thousands of training examples seems quite unfeasible. On the other hand, the use of handcrafted resources such as gazetteers or pattern libraries (compare (Maynard *et al.* 03)) will also not help as creating and maintaining such resources for hundreds of concepts is equally unfeasible. Interesting and very promising are approaches which operate in a bootstrapping-like fashion, using a set of seeds to derive more training data such as the supervised approach using Hidden Markov Models in (Niu *et al.* 03) or the unsupervised approach in (Collins & Singer 99).

In this paper we present an unsupervised approach which - as many others - is based on Harris' distributional hypothesis, i.e. that words are semantically similar to the extent to which they share syntactic contexts. There have been many approaches in NLP exploiting this hypothesis, the most influential probably being the work of (Grefenstette 94) on automatic thesaurus construction as well as of (Pereira *et al.* 93) on building hierarchical clusters of nouns, the work of (Hindle 90) on discovering groups of (semantically) similar nouns as well as the work of (Yarowsky 95) and (Schuetze 98) on Word Sense disambiguation/discrimination. In particular some researchers have considered using syntactic collocations for named entity recognition (cf. (Cucchiarelli & Velardi 01) and (Lin 98)). More recently, several researchers have addressed the problem of classifying a new term into an existing ontology (Agirre *et al.* 00; Pekar & Staab 02; Alfonseca & Manandhar 02; Widdows).

In this paper we investigate the impact of using different feature weighting measures and various similarity measures described in (Lee 99). Further, to address data sparseness problems we examine the influence of (i) anaphora resolution in the hope that it will yield more context information as speculated in (Grefenstette 94) (ii)

[1] http://cnts.uia.ac.be/conll2003/ner/

[2] http://www.itl.nist.gov/iaui/894.01/tests/ace/phase1/index.htm

downloading additional textual material from the Web as in (Agirre *et al.* 00) and making use of the structure of the concept hierarchy or taxonomy in calculating the context vectors for the classes as in (Resnik 93), (Hearst & Schütze 93) or (Pekar & Staab 02). The paper is organized as follows: first, we present our data set in Section 2 and describe our evaluation measures as well as present a few baselines for the task showing its complexity in Section 3. In section 4 we analyze the impact of varying the above mentioned parameters step by step starting with a window-based approach as a baseline. Before concluding we also discuss the results of our approach with respect to other systems performing a similar task.

2 Data Set

Our data set consists of 1880 texts containing destination descriptions downloaded from *http://www.lonelyplanet.com/destinations*. In order to create an evaluation standard, we asked two test persons to annotate the named entities of 30 randomly selected texts with the appropriate concept from a given ontology. They used a pruned version of a tourism ontology developed within an information retrieval project at our site. The original ontology consisted of 1043 concepts, but we removed some irrelevant concepts beforehand in order to facilitate the task for the annotators, resulting in an ontology with 682 concepts. In what follows, we will refer to these annotators as A and B. Annotator A actually produced 436 annotations and subject B produced 392. There were 277 named entities that were annotated by both subjects. For these 277 named entities, they used 59 different concepts and coincided in 176 cases, the agreement thus being 63.54%. The categorial agreement on these 277 named entities measured by the Kappa statistic was 63.48% (cf. (Carletta 96)), which allows to conclude that the annotation task is overall more or less well defined but that the agreement between humans is far from perfect. A system selecting a concept for a given named entity at random would thus be correct in 0.15% cases, which already shows the difficulty of the task. We evaluate our system on the named entities annotated by both subjects as described in the following section. It is important to emphasize however that we totally abstract here from the actual recognition of named entities in the sense that the input to our system is a set of named entities to be assigned to the appropriate class.

3 Evaluation

As mentioned in (Collins & Singer 99), the task in named entity recognition is to learn a function from an input string (a proper name) to its class. In particular our aim is to learn a function f_S which approximates the functions f_A and f_B specified by both annotators. We assume that these functions are given as sets $C_X := \{(e,c)|e \in dom(f_X) \wedge f_X(e) = c\}$, where e is the named entity in question, c is the concept it has been assigned to and $dom(f)$ is the domain of a function f. While f_A and f_B are total functions, f_S is a partial one as our system does not always produce an answer. In fact, if the distributional similarity between the entity to be tagged and all the concepts in the ontology is minimal, then the system will give no answer. Thus it is not only important to measure the recall, but also the precision of the system. We thus evaluate the system with the standard measures of Precision, Recall and F-Measure, i.e.

$$P_A = \frac{|C_A \cap C_S|}{|C_S|} \quad P_B = \frac{|C_B \cap C_S|}{|C_S|} \quad P = \frac{P_A + P_B}{2}$$
$$R_A = \frac{|C_A \cap C_S|}{|C_A|} \quad R_B = \frac{|C_B \cap C_S|}{|C_B|} \quad R = \frac{R_A + R_B}{2}$$
$$F_A = \frac{2 * P_A * R_A}{P_A + R_A} \quad F_B = \frac{2 * P_B * R_B}{P_B + R_B} \quad F = \frac{F_A + F_B}{2}$$

As named entities can be tagged at different levels of detail and there is certainly not only one correct assignment of a concept, we also consider how close the assignment of the system is with respect to the assignment of the annotator by using the *Learning Accuracy* originally introduced by (Hahn & Schnattinger 98). However, we consider a slightly different formulation of the Learning Accuracy in line with the measures defined in (Maedche *et al.* 02). Both measures are in fact equivalent, the only difference is that we measure the distance between nodes in terms of edges – in contrast to nodes in Hahn's version – and we do not need any case distinction taking into account if the classification was correct or not. The Learning Accuracy is defined as follows:

$$LA(a,b) := \frac{\delta(top,c) + 1}{\delta(top,c) + \delta(a,c) + \delta(b,c) + 1}$$

where $c = lcs(a,b)$ is the least common subsumer of concepts a and b as defined in (Maedche *et al.* 02).

4 Experiments

As mentioned above, our approach is in line with Harris' distributional hypothesis and other approaches in which the context of a phrase is used to disambiguate its sense (Yarowsky 95; Schuetze 98) or class (Lin 98) or to discover other semantically related terms (Hindle 90). As other approaches, we also adopt the one-sense-per-discourse assumption (Gale *et al.* 92), i.e. we do not perform any word sense disambiguation. Our algorithm thus assigns an instance represented by a certain context vector $\vec{i}$ to the concept corresponding to the most similar vector $\vec{c}$. The algorithm is basically as follows:

classify(set of instances I, corpus t, set of concepts C){
    foreach c in C {
        $\vec{v_c}$ = getContextVector(c,t);
    }
    foreach foreach c in C {
        doFeatureWeighting($\vec{v_c}$);
    }
    foreach i in I {
        $\vec{v_i}$ = getContextVector(i,t);
        class(i)=$maxarg_c$ sim($\vec{v_c}$,$\vec{v_i}$);
    }
    return class;
}

Though most approaches represent the context of a phrase as a vector, there are great differences in which features are used ranging from simple word windows (Yarowsky 95; Schuetze 98) to syntactic dependencies extracted with a parser (Hindle 90; Pereira *et al.* 93; Grefenstette 94). We start our analysis by comparing window-based techniques with using pseudo-syntactic dependencies extracted by using regular expressions over part-of-speech tags. Furthermore, we analyze the impact of using different similarity and feature weighting measures. As they were found to perform particularly well in (Lee 99), we use the following similarity measures: the *cosine* and *Jaccard* measures, the *L1 norm* as well as the *Jensen-Shannon* and the *Skew* divergence. Further, we weight the features according to different measures. In particular, we use the following measures:

$$Conditional(n, feat) = P(n|feat) = \frac{f(n,feat)}{f(feat)}$$

$$PMI(n, feat) = P(n|feat)\, log\, \frac{P(n|feat)}{P(n)}$$

$$Resnik(n, feat) = S_R(feat)\, PMI(n, feat)$$

where $S_R(feat) = \sum_{n'} P(n'|feat)\, log \frac{P(n'|feat)}{P(n')}$.

Furthermore, $f(n, feat)$ is the number of occurrences of a term n with feature $feat$, $f(feat)$ is the number of occurrences of the feature $feat$ and $P(n)$ is the relative frequency of a term n compared to all other terms. The first information measure is simply the conditional probability of the term n given the feature $feat$. The second measure $PMI(n, v)$ is the so called *pointwise mutual information* and was used by (Hindle 90) for discovering groups of similar terms. The third measure is inspired by the work of (Resnik 93) and introduces an additional factor $S_R(n, feat)$ which takes into account all the terms appearing with the feature in question. In particular, the factor measures the relative entropy of the prior and posterior (given the feature) distributions of n and thus the 'selectional strength' of a given feature.

4.1 Using Word Windows

In a first experiment we used the n words to the left and right of a certain word of interest excluding so called stopwords and without trespassing sentence boundaries. Here n is the so called window size. The advantage of such an approach is that no preprocessing is necessary to extract context information. However, it also has the drawback of making context vectors larger than when using syntactic dependencies thus making the similarity calculation less efficient (cf.(Grefenstette 94)). We implemented this approach in order to verify if syntactic dependencies actually perform better in our setting. We varied the similarity measure, the feature weighting strategy as well as experimented with the three different window sizes: 3, 5 and 10 words. thus producing 5 * 4 * 3 = 60 runs of the similarity-based classification algorithm. Due to space limitations we do not present all the results. The best result was an F-Measure of 19.7% and a Learning Accuracy of 57.78%. It was achieved when using the Skew divergence as similarity measure, PMI as feature weighting measure and a window size of 10.

4.2 Using pseudo-syntactic dependencies

Instead of merely using the words occurring within a given window size before and after the word in question, we also experimented with using pseudo-syntactic dependencies. These dependencies are not really syntactical as they are not obtained from parse trees, but from a very shallow method consisting in matching certain regular expression over part of speech tags. The motivation for doing this is the observation in (Grefenstette 94) that the quality of using word windows or syntactic dependencies for distributional analysis depends on the rank or frequency of the word in question. In this line, our intention is to make a compromise between using word windows and syntactic dependencies extracted from parse trees. Our pseudo-syntactic dependencies are surface dependencies extracted by matching regular expressions. In what follows we list the syntactic expressions we use and give a brief example of how the features, represented as predicates, are extracted from these expressions:

- adjective modifiers, i.e. *a nice city* → nice(city)
- prepositional phrase modifiers, i.e. *a city near the river* → near_river(city) and city_near(river), respectively
- possessive modifiers, i.e. *the city's center* → has_center(city)
- noun phrases in subject or object position. i.e. *the city offers an exciting nightlife* → offer_subj (city) and offer_obj(nightlife)
- prepositional phrases following a verb, i.e. *the river flows through the city* → flows_through(city)
- copula constructs i.e. *a flamingo is a bird* → is_bird(flamingo)
- verb phrases with the verb *to have*, i.e. *every country has a capital* → has_capital(country)

Consider for example the following discourse:

Mopti is the biggest city along the Niger with one of the most vibrant ports and a large bustling market. Mopti has a traditional ambience that other towns seem to have lost. It is also the center of the local tourist industry and suffers from hard-sell overload. The nearby junction towns of Gao and San offer nice views over the Niger's delta.

Here we would extract the following concept vectors:

city: biggest(1)
ambience: traditional(1)

center: of_tourist_industry(1)
junction towns: nearby(1)
market: bustling(1)
port: vibrant(1)
tourist industry: center_of(1), local(1)
town: seem_subj(1)
view: nice(1), offer_obj(1)

and the following ones for named entities:

Mopti: is_city(1), has_ambience(1)
San: offer_subj(1)
Gao: junction_of(1)
Niger: has_delta(1)

Table 1 shows the results for the version of the classification algorithm making use of the pseudo-syntactic dependencies using the different similarity and feature weighting measures (Standard). The best result was an F-Measure of 19.58% and a Learning Accuracy of 60.03%. The fact that the F-Measure is slightly worse is definitely compensated by a higher Learning Accuracy. Furthermore, as the length of the vectors is much smaller and thus the computation of the similarities faster, we conclude that using the pseudo-syntactic dependencies is an interesting alternative and present the results of further modifications to our algorithm with respect to the version using these sort of dependencies.

4.3 Dealing with Data Sparseness

4.3.1 Using Conjunctions

In order to address the problem of data sparseness we exploit conjunctions of named entities in the sense that if two named entities appear linked by the conjunctions 'and' or 'or', we count any occurrence of a feature with one of the named entities also as an occurrence of the other. As the results in Table 1 show, this simple heuristic improves the results of our approach considerably. The top results are F-Measures of 22.8% (Cosine), 22.57% (L1 norm) and 22.57% (Skew divergence) with a Learning Accuracy of 61.23%, 61.4% and 62.7%, respectively.

4.3.2 Exploiting the Taxonomy

An interesting option discussed in (Resnik 93), (Pekar & Staab 02) and (Hearst & Schütze 93) is to take into account the taxonomy of the underlying ontology to compute the context vector of a certain term by taking into account the context vectors of its hyponyms. This is in fact a delicate issue as some studies have shown that this doesn't work while other have shown the contrary. We adopt here a conservative strategy and take only into account the context vectors of direct hyponyms to compute the vector of a certain term. In fact, the context vector of a hypernym will be the sum of the context vectors of all its direct hyponyms. We assume a one-to-one mapping between nouns and concept labels, thus considering the hyponyms of all possible concepts for a given label. We will refer to this as the 'standard' version. However, the aggregated vectors can also be normalized. In fact, we experiment with the two possibilities also discussed in (Pekar & Staab 02): (i) standard normalization of the vector or (ii) calculating its centroid (compare (Pekar & Staab 02) and (Hearst & Schütze 93)). In the latter the only difference is that we create an average vector by dividing through the number of direct hyponyms. As the results in Table 1 show, only the version with the centroid method did indeed yield better results, while with the standard (no vector normalization) and the category method (standard vector normalization) did actually make the results worse. The best result with the centroid method was an F-Measure of 23.02% and a Learning Accuracy of 64.11%.

4.3.3 Anaphora Resolution

As another approach to overcome the problem of data sparseness we explored the impact of anaphora resolution on the task of named entity recognition. Based on MINIPAR (cf. (Lin 93)) and the work by (Lappin & Leass 94) we implemented an algorithm for identifying intrasentential antecedents of 3rd person personal and possessive pronouns which replaces each (non-pleonastic) anaphor by the grammatically correct form of the corresponding antecedent as shown in the following examples:
*The port capital of Vathy is dominated by **its** fortified Venetian harbour.* →
*The port capital of Vathy is dominated by **Vathy's** fortified Venetian harbour.*
*Holiday hooligans used to head to nearby Benitses, until **it** was ruined, so now **they** head north to cut a swathe through the coastline's few remaining unspoilt coves and fishing villages.*→
*Holiday hooligans used to head to nearby Benitses, until **Benitses** was ruined, so now **the hooligans** head north to cut a swathe through the coastline's few remaining unspoilt coves and fishing villages.*

Moreover, in order to improve the detection of pleonastic occurrences of *it*, we use a modified set of patterns developed by (Dimitrov 02). Although our implementation seems to perform a bit worse than the one by Lappin and Leass (maybe due to the very noisy data set) the evaluation yielded a remarkable precision of about 0.79 and a recall of approximately 0.7.

As shown by Table 1 the use of anaphora resolution even improves the results we obtained by exploiting the taxonomy leading to an F-Measure of 23.82% and a Learning Accuracy of 65.04% (Skew divergence).

4.3.4 Downloading Documents from the Web

Since named entities tend to occur less often than common nouns representing possible classes, they are to a particularly high degree affected by the problem of data sparseness. We address this issue by downloading from the web a set of at most 20 additional documents D_i for each named entity i. Moreover, in order make sure that each $d \in D_i$ belongs to the correct sense of i we compare d with all documents in the original corpus containing at least one occurrence of i. The decision whether to keep d or not is made by creating bag-of-words style vectors rep-

	Cosine		Jaccard		L1		JS		Skew	
	F	LA	F	LA	F	LA	F	LA	F	LA
Standard										
Frequency	13.29%	55.77%	1.4%	29.99%	15.62%	59.45%	2.56%	39%	14.45%	59.41%
Conditional	16.78%	58.47%	1.4%	29.99%	18.65%	59.31%	6.29%	41.86%	17.02%	58.71%
PMI	19.11%	58.93%	1.4%	29.99%	17.72%	57.29%	5.13%	40.25%	**19.58%**	**60.03%**
Resnik	15.38%	56.33%	1.4%	29.99%	18.18%	58.91%	4.9%	38.12%	19.35%	60.44%
Conjunctions										
Frequency	18.51%	61.25%	11.54%	44.22%	18.28%	63.58%	10.16%	52.06%	21.9%	65.19%
Conditional	20.77%	60.87%	11.54%	44.22%	21.9%	63.27%	11.06%	43.46%	22.12%	63.41%
PMI	**22.8%**	**61.23%**	11.54%	44.37%	**22.57%**	**61.4%**	10.84%	42%	**22.57%**	**62.7%**
Resnik	21.22%	60.32%	11.54%	44.37%	22.12%	61.71%	10.61%	43.1%	22.35%	62.92%
Conjunctions + Ontology										
Freuqency	5.42%	63.12%	11.09%	44.93%	5.42%	66.82%	10.61%	51.18%	5.42%	65.82%
Conditional	5.64%	64.04%	11.09%	44.93%	5.64%	64.46%	10.84%	46.09%	5.64%	64.99%
PMI	6.32%	64.17%	11.09%	44.81%	5.87%	63.59%	10.61%	43.59%	5.87%	63.43%
Resnik	5.42%	62.52%	11.09%	44.81%	5.87%	62.78%	11.06%	44.88%	5.87%	63.39%
Conjunctions + Ontology (Category)										
Frequency	10.16%	47.84%	11.09%	44.93%	13.77%	55.78%	10.61%	51.18%	14.67%	59.79%
Conditional	3.16%	42.84%	11.09%	44.93%	5.42%	49.7%	10.84%	46.09%	6.77%	58.04%
PMI	5.87%	45.76%	11.09%	44.81%	9.71%	44.03%	1.36%	38.65%	7.9%	53.71%
Resnik	5.19%	43.16%	11.09%	44.81%	6.55%	49.14%	0.9%	34.92%	6.32%	59.06%
Conjunctions + Ontology (Centroid)										
Frequency	22.35%	63.57%	11.09%	44.93%	23.02%	63.27%	10.61%	51.18%	13.54%	62.63%
Conditional	22.12%	61.05%	11.09%	44.93%	22.8%	62.53%	10.84%	46.09%	**23.02%**	**64.11%**
PMI	22.12%	60.66%	11.09%	44.81%	22.8%	61.72%	10.38%	42.33%	19.86%	63.47%
Resnik	20.99%	60.62%	11.09%	44.81%	22.12%	61.89%	10.61%	43.39%	21.9%	64.33%
Conjunctions + Ontology (Centroid) + Anaphora Resolution										
Frequency	22.25%	64.8%	10.59%	42.8%	23.15%	65.45%	10.11%	49.12%	15.28%	65.17%
Conditional	22.7%	62.19%	10.59%	42.8%	23.37%	63.92%	11.01%	45.58%	**23.82%**	**65.04%**
PMI	22.92%	61.69%	10.59%	43.1%	23.6%	63.32%	11.24%	43.6%	18.88%	64.49%
Resnik	22.25%	61.06%	10.59%	43.1%	23.37%	63.42%	10.36%	43.16%	23.37%	64.69%
Conjunctions + Ontology (Centroid) + Web Crawling										
Frequency	25.4%	65.43%	12.1%	51.01%	24.4%	64.22%	6.25%	45.61%	9.07%	64.68%
Conditional	25.6%	64.46%	12.1%	51.01%	25.81%	64.43%	3.63%	39.72%	**26.21%**	**65.91%**
Mutual	25.6%	63.94%	10.08%	50.4%	25.81%	63.72%	3.43%	23.63%	12.1%	64.31%
Resnik	24.4%	61.9%	10.08%	50.4%	24.6%	62.41%	1.81%	20.17%	25.2%	65.18%

Table 1: Results for pseudo-syntactic dependencies

resentations for each of the involved documents, computing their cosine and only considering the document if the similarity is over an experimentally determined threshold of 0.2. Table 1 shows that this way of extending the corpus with documents from the web considerably improves all previous results. With the Skew divergence we achieved an F-Measure of 26.21% and a Learning Accuracy of 65.91%.

4.3.5 Postprocessing

Finally, we also examine a postprocessing step in which the k best answers of the system (ranked according to their corresponding similarities from highest to lowest) are checked for their statistical plausibility on the Web. For this purpose, inspired by the work of (Markert *et al.* 03), for each named entity e and the top k answers $c_1, .., c_k$ we generate the following Hearst-style (Hearst 92) pattern strings and count their occurrences on the Web by using the Google Web API:

1. $\pi(c_i)$ such as e
2. e and other $\pi(c_i)$
3. e or other $\pi(c_i)$
4. $\pi(c_i)$, especially e
5. $\pi(c_i)$, including e

where $\pi(w)$ is the result of looking up the plural form of the word w in a lexicon containing inflected forms and their corresponding lemmas. Furthermore, the number of hits of the above pattern string are normalized by dividing through the number of hits of the underlined parts. At the end, that answer of the k best is chosen which maximizes this coefficient. We experimented with different values for k, i.e. 3, 5 and 10. This extension is furthermore efficient as we only need to generate $k + 1$ queries to the Google Web API for each named entity. Table 2 gives the results of this step when postprocessing the results produced with the versions of our system using anaphora resolution and crawling documents from the Web. The results show that the F-Measures increase considerably when using our postprocessing step. The best result is an F-Measure of 32.6% with a precision of 36.82%, a recall of 29.34% and a Learning Accuracy of 68.87% for the version of our system crawling the Web.

4.3.6 Discussion

The best result of our approach is an F-Measure of 32.6% which is more than 32 points points above the naive baseline of F=0.15%, almost 20 points over the majority-class-baseline of F=12.64% and 12.9 points over the word-window-based approach approach with a window size of 10 (F=19.7%). When considering this best version of our approach, the precision is 36.82% and the recall 29.34%. In order to compare our results with systems performing a similar task, we compare our recall as well as Learning Accuracy with the one of the systems in Table 3. In fact, our recall value corresponds to the accuracy values of the other approaches. (Fleischman & Hovy 02) for example

k	k=3				k=5				k=10			
	F	P	R	LA	F	P	R	LA	F	P	R	LA
AR	29.15%	38.46%	23.47%	71.04%	28.7%	37.87%	23.1%	71%	30.72%	40.53%	24.73%	71.71%
WC	30.58%	34.54%	27.44%	67.71%	30.78%	34.77%	27.62%	68.52%	**32.6%**	36.82%	29.24%	**69.87%**

Table 2: Results of the postprocessing step on the A(naphora) R(esolution) and the W(eb) C(rawling versions))

make use of a supervised approach and extract n-grams for training several classifiers. (Evans 03) computes hypernym vectors for each entity by using the Google API and clusters instances on the basis of these, thus considering a totally unsupervised scenario in which the classes themselves are derived from the data. (Alfonseca & Manandhar 02) present a similar approach to ours relying on distributional similarity and achieve the best results using verb-object dependencies as features, while (Hahn & Schnattinger 98) present an elaborated qualification calculus for reasoning about the quality of different hypothesis. The systems thus rely on different assumptions, learning paradigms as well as number of classes, such that they are not directly comparable. The conclusions which can be drawn from Table 3 are that (i) obviously the task is the harder the more classes are considered and (ii) our approach fits very well from a quantitative point of view into the landscape of systems performing a similar – but not equivalent – task. Considering the most similar systems, it is worth mentioning that our results are worse than the ones of (Hahn & Schnattinger 98), which however consider half as many concepts and furthermore assume a perfect syntactic and semantic analysis as well as an elaborated DL concept hierarchy. On the other hand we achieve much better results than (Alfonseca & Manandhar 02), but they also consider a larger number of classes. SemTag (Dill *et al.* 03) also considers a large amount of classes from the TAP ontology, but assumes that the possible classes or tags for each instance are known in advance. Thus, the system effectively performs sense disambiguation with respect to a much smaller set of classes per instance.

System	#concepts	Rec/Acc	LA
MUC	3	>90%	n.a
Fleischman et al.	8	70.4%	n.a.
Evans	2-8	41.41%	n.a.
Hahn et al. (Baseline)	325	21%	67%
Hahn et al. (TH)	325	26%	73%
Hahn et al. (CB)	325	31%	76%
BEST	682	29.24%	69.87%
Alfonseca et al.	1200	17.39%	44%

Table 3: Comparison of results

5 Conclusion

We have addressed the problem of tagging named entities with regard to a large set of concepts as specified within a given concept hierarchy. In particular we have presented an approach relying on Harris' distributional hypothesis as well as on the vector-space model and assigning a named entity to that concept which maximizes the contextual similarity with the named entity in question. The aim has not been to present a fully fledged system performing this task, but to investigate the impact of varying a number of parameters. In this line we have shown that the pseudo-syntactic dependencies we have considered are an interesting alternative to window-based approaches as they yield a higher Learning Accuracy and also allow a more efficient computation of the similarities. To address the typical data sparseness problems one encounters when working with corpora, we have examined the impact of (i) exploiting conjunctions, (ii) factoring the underlying taxonomy into the computation of the concept vectors as in (Pekar & Staab 02), (iii) getting additional context by applying an anaphora resolution algorithm developed for this purpose and (iv) additionally downloading additional documents from the World Wide Web as in (Agirre *et al.* 00), showing that with the correct settings all these techniques improve the results of our approach both in terms of F-Measure and Learning Accuracy. Finally, we have also presented a postprocessing step by which the system's k most highly ranked answers are checked for their statistical plausibility on the Web, which notably improves the results of the approach. In general, the best results were achieved using the conditional probability as feature weighting strategy and the Skew divergence as similarity measure, thus confirming the results obtained in (Lee 99).

Acknowledgements: We would like to acknowledge support from the EU-IST project SEKT (IST-2003-506826) as well as the SmartWeb project, funded by the German Ministry for Education and Research.

References

(Agirre *et al.* 00) E. Agirre, O. Ansa, E. Hovy, and D. Martinez. Enriching very large ontologies using the WWW. In *Proceedings of the Workshop on Ontology Construction of the ECAI*, 2000.

(Alfonseca & Manandhar 02) E. Alfonseca and S. Manandhar. Extending a lexical ontology by a combination of distributional semantics signatures. In *Proceedings of the 13th EKAW*, 2002.

(Bikel *et al.* 99) D.M. Bikel, R.L. Schwartz, and R.M. Weischedel. An algorithm that learns what's in a name. *Machine Learning*, 34(1-3):211–231, 1999.

(Borthwick *et al.* 98) A. Borthwick, J. Sterling, E. Agichtein, and R. Grishman. Exploiting diverse knowledge sources via maximum entropy in named entity recognition. In *Proc. of the Sixth ACL Workshop on Very Large Corpora*, 1998.

(Carletta 96) J. Carletta. Asessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2):249–254, 1996.

(Chieu & Ng 03) H.L. Chieu and H.T. Ng. Named entity recognition with a maximum entropy approach. In

Proceedings of the Seventh Conference on Natural Language Learning (CoNLL-2003), pages 160–163, 2003.

(Collins & Singer 99) M. Collins and Y. Singer. Unsupervised models for named entity classification. In *Proc. of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 1999.

(Cucchiarelli & Velardi 01) A. Cucchiarelli and P. Velardi. Unsupervised named entity recognition using syntatic and semantic contextual evidence. *Computational Linguistics*, 27(1):123–131, 2001.

(Dill *et al.* 03) S. Dill, N. Eiron, D. Gibson, D. Gruhl, R. Guha, A. Jhingran, T. Kanungo, S. Rajagopalan, A. Tomkins, J.A. Tomlin, and J.Y. Zien. SemTag and Seeker: bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the 12th International Conference on the World Wide Web (WWW'03)*, pages 178–186, 2003.

(Dimitrov 02) M. Dimitrov. A light-weight approach to coreference resolution for named entities in text. Unpublished M.Sc. thesis, University of Sofia, 2002.

(Evans 03) R. Evans. A framework for named entity recognition in the open domain. In *Proceedings of RANLP*, pages 137–144, 2003.

(Fleischman & Hovy 02) M. Fleischman and E. Hovy. Fine grained classification of named entities. In *Proceedings of COLING*, 2002.

(G. Pailouras & Spyropoulos 00) V. Karkaletsis G. Pailouras and C.D. Spyropoulos. Learning decision trees for named-entity recognition and classification. In *Proceedings of the ECAI Workshop on Machine Learning for Information Extraction*, 2000.

(Gale *et al.* 92) W. Gale, K. Church, and Y. Yarowsky. One sense per discourse. In *Proceedings of the 4th DARPA Speech and Natural Language Workshop*, pages 233–237, 1992.

(Grefenstette 94) G. Grefenstette. *Explorations in Authomatic Thesaurus Construction*. Kluwer, 1994.

(Hahn & Schnattinger 98) U. Hahn and K. Schnattinger. Towards text knowledge engineering. In *Proceedings of AAAI'98/IAAI'98*, 1998.

(Hearst & Schütze 93) M.A. Hearst and H. Schütze. Customizing a lexicon to better suit a computational task. In *Proceedings of the ACL SIGLEX Workshop on Acquisition of Lexical Knowledge from Text*, 1993.

(Hearst 92) M.A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of COLING*, 1992.

(Hendrickx & van denBosch 03) I. Hendrickx and A. van den Bosch. Memory-based one-step named-entity recognition: Effects of seed list features, classifier stacking, and unannotated data. In *Proceedings of CoNLL-2003*, pages 176–179, 2003.

(Hindle 90) D. Hindle. Noun classification from predicate-argument structures. In *Proceedings of the Annual Meeting of the ACL*, pages 268–275, 1990.

(Hirschman & Chinchor 97) L. Hirschman and N. Chinchor. Muc-7 named entity task definition. In *Proceedings of the 7th Message Understanding Conference (MUC-7)*, 1997.

(Isozaki & Kazawa 02) H. Isozaki and H. Kazawa. Efficient support vector classifiers for named entity recognition. In *Proceedings of COLING*, 2002.

(Lappin & Leass 94) S. Lappin and H.J. Leass. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561, 1994.

(Lee 99) L. Lee. Measures of distributional similarity. In *Proceedings of the 37th Annual Meeting of the ACL*, pages 25–32, 1999.

(Lin 93) D. Lin. Principle-based parsing without overgeneration. In *Proceedings of the Annual Meeting of the ACL*, pages 112–120, 1993.

(Lin 98) D. Lin. Using collocation statistics in information extraction. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, 1998.

(Maedche *et al.* 02) A. Maedche, V. Pekar, and S. Staab. Ontology learning part one - on discovering taxonomic relations from the web. In *Web Intelligence*. Springer, 2002.

(Markert *et al.* 03) K. Markert, N. Modjeska, and M. Nissim. Using the web for nominal anaphora resolution. In *EACL Workshop on the Computational Treatment of Anaphora*, 2003.

(Maynard *et al.* 03) D. Maynard, K. Bontcheva, and H. Cunningham. Towards a semantic extraction of named entities. In *Proceedings of RANLP*, 2003.

(Niu *et al.* 03) C. Niu, W. Lei, J. Ding, and R.K. Srihari. A bootstrapping approach to named entity classification using successive learners. In *Proceedings of the 41st Annual Meeting of the ACL*, pages 335–342, 2003.

(Pekar & Staab 02) V. Pekar and S. Staab. Taxonomy learning: factoring the structure of a taxonomy into a semantic classification decision. *Proceedings of COLING*, 2002.

(Pereira *et al.* 93) F. Pereira, N. Tishby, and L. Lee. Distributional clustering of english words. In *Proceedings of the 31st Annual Meeting of the ACL*, pages 183–190, 1993.

(Resnik 93) P. Resnik. *Selection and Information: A Class-Based Approach to Lexical Relationships*. Unpublished PhD thesis, 1993.

(Schuetze 98) H. Schuetze. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123, 1998.

(Sekine *et al.* 98) S. Sekine, R. Grishman, and H. Shinnou. A decision tree method for finding and classifying names in japanese texts. In *Proceedings of the Sixth ACL Workshop on Very Large Corpora*, 1998.

(Widdows) D. Widdows. Unsupervised method for developing taxonomies by combining syntactic and statistical information. In *Proceedings of HLT/NAACL*.

(Yarowsky 95) D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings od the Annual Meeting of the ACL*, pages 189–196, 1995.

(Zhou & Su 02) G. Zhou and J. Su. Named entity recognition using an hmm-based chunk tagger. In *Proceedings of the 40th Meeting of the ACL*, pages 473–480, 2002.

Text Semantic Similarity, with Applications

Courtney Corley and **Andras Csomai** and **Rada Mihalcea**
Department of Computer Science
University of North Texas
{corley,ac0225,rada}@cs.unt.edu

Abstract

In this paper, we present a knowledge-based method for measuring the semantic-similarity of texts. Through experiments performed on two different applications: (1) paraphrase and entailment identification, and (2) word sense similarity, we show that this method outperforms the traditional text similarity metrics based on lexical matching.

1 Introduction

Measures of text similarity have been used for a long time in applications in natural language processing and related areas. One of the earliest applications of text similarity is perhaps the vectorial model in information retrieval, where the document most relevant to an input query is determined by ranking documents in a collection in reversed order of their *similarity* to the given query (Salton & Lesk 71). Text similarity has been also used for relevance feedback and text classification (Rocchio 71), word sense disambiguation (Lesk 86), and more recently for extractive summarization (Salton *et al.* 97b), and methods for automatic evaluation of machine translation (Papineni *et al.* 02) or text summarization (Lin & Hovy 03).

The typical approach to finding the similarity between two text segments is to use a simple lexical matching method, and produce a similarity score based on the number of lexical units that occur in both input segments. Improvements to this simple method have considered stemming, stop-word removal, part-of-speech tagging, longest subsequence matching, as well as various weighting and normalization factors (Salton *et al.* 97a). While successful to a certain degree, these lexical similarity methods fail to identify the *semantic* similarity of texts. For instance, there is an obvious similarity between the text segments *I own a dog* and *I have an animal*, but most of the current text similarity metrics will fail in identifying any kind of connection between these texts. The only exception to this trend is perhaps the latent semantic analysis (LSA) method (Landauer *et al.* 98), which represents an improvement over earlier attempts to use measures of semantic similarity for information retrieval (Voorhees 93), (Xu & Croft 96). LSA aims to find similar terms in large text collections, and measures similarity between texts by including these additional related words. However, to date LSA has not been used on a large scale, due to the complexity and computational cost associated with the algorithm, and perhaps also due to the "black-box" effect that does not allow for any deep insights into why some terms are selected as similar during the singular value decomposition process.

In this paper, we explore a knowledge-based method for measuring the semantic similarity of texts. While there are several methods previously proposed for finding the semantic similarity of words, to our knowledge the application of these word-oriented methods to text similarity has not been yet explored. We introduce an algorithm that combines the word-to-word similarity metrics into a text-to-text semantic similarity metric, and we show that this method outperforms the simpler lexical matching similarity approach, as measured in two different applications: (1) paraphrase and entailment identification, and (2) word sense similarity.

2 Measuring Text Semantic Similarity

Given two input text segments, we want to automatically derive a score that indicates their similarity at *semantic* level, thus going beyond the simple lexical matching methods traditionally used for this task. Although we acknowledge the fact that a comprehensive metric of text semantic similarity should take into account the relations between words, as well as the role played by the various entities involved in the interactions

described by each of the two texts, we take a first rough cut at this problem and attempt to model the semantic similarity of texts as a function of the semantic similarity of the component words. We do this by combining metrics of word-to-word similarity and language models into a formula that is a potentially good indicator of the semantic similarity of the two input texts.

2.1 Semantic Similarity of Words

There is a relatively large number of word-to-word similarity metrics that were previously proposed in the literature, ranging from distance-oriented measures computed on semantic networks, to metrics based on models of distributional similarity learned from large text collections. From these, we chose to focus our attention on six different metrics, selected mainly for their observed performance in natural language processing applications, e.g. malapropism detection (Budanitsky & Hirst 01) and word sense disambiguation (Patwardhan *et al.* 03), and for their relatively high computational efficiency.

We conduct our evaluation using the following word similarity metrics: Leacock & Chodorow, Lesk, Wu & Palmer, Resnik, Lin, and Jiang & Conrath. Note that all these metrics are defined between concepts, rather than words, but they can be easily turned into a word-to-word similarity metric by selecting for any given pair of words those two meanings that lead to the highest concept-to-concept similarity. We use the WordNet-based implementation of these metrics, as available in the WordNet::Similarity package (Patwardhan *et al.* 03). We provide below a short description for each of these six metrics.

The **Leacock & Chodorow** (Leacock & Chodorow 98) similarity is determined as:

$$Sim_{lch} = -\log \frac{length}{2 * D} \quad (1)$$

where *length* is the length of the shortest path between two concepts using node-counting, and D is the maximum depth of the taxonomy.

The **Lesk** similarity of two concepts is defined as a function of the overlap between the corresponding definitions, as provided by a dictionary. It is based on an algorithm proposed in (Lesk 86) as a solution for word sense disambiguation.

The **Wu and Palmer** (Wu & Palmer 94) similarity metric measures the depth of the two concepts in the WordNet taxonomy, and the depth of the least common subsumer (LCS), and combines these figures into a similarity score:

$$Sim_{wup} = \frac{2 * depth(LCS)}{depth(concept_1) + depth(concept_2)} \quad (2)$$

The measure introduced by **Resnik** (Resnik 95) returns the information content (IC) of the LCS of two concepts:

$$Sim_{res} = IC(LCS) \quad (3)$$

where IC is defined as:

$$IC(c) = -\log P(c) \quad (4)$$

and $P(c)$ is the probability of encountering an instance of concept c in a large corpus.

The next measure we use in our experiments is the metric introduced by **Lin** (Lin 98), which builds on Resnik's measure of similarity, and adds a normalization factor consisting of the information content of the two input concepts:

$$Sim_{lin} = \frac{2 * IC(LCS)}{IC(concept_1) + IC(concept_2)} \quad (5)$$

Finally, the last similarity metric we consider is **Jiang & Conrath** (Jiang & Conrath 97), which returns a score determined by:

$$Sim_{jnc} = \frac{1}{IC(concept_1) + IC(concept_2) - 2 * IC(LCS)} \quad (6)$$

2.2 Language Models

In addition to the semantic similarity of words, we also want to take into account the *specificity* of words, so that we can give a higher weight to a semantic matching identified between two very specific words (e.g. *collie* and *sheepdog*), and give less importance to the similarity score measured between generic concepts (e.g. *go* and *be*). While the specificity of words is already measured to some extent by their depth in the semantic hierarchy, we are reinforcing this factor with a corpus-based measure of word specificity, based on distributional information learned from large corpora.

Language models are frequently used in natural language processing applications to account for

the distribution of words in language. While word frequency does not always constitute a good measure of word importance, the distribution of words across an entire collection can be a good indicator of the *specificity* of the words. Terms that occur in a few documents with high frequency contain a greater amount of discriminatory ability, while terms that occur in numerous documents across a collection with a high frequency have inherently less meaning to a document. We determine the *specificity* of a word using the inverse document frequency introduced in (Sparck-Jones 72), which is defined as the total number of documents in the corpus, divided by the total number of documents that include that word. In the experiments reported in this paper, we use the British National Corpus to derive the document frequency counts, but other corpora can be used to the same effect.

2.3 Semantic Similarity of Texts

Provided a measure of semantic similarity between words, and an indication of the word specificity, we combine them into a measure of text semantic similarity, by pairing up those words that are found to be most similar to each other, and weighting their similarity with the corresponding specificity score.

We define a *directional* measure of similarity, which indicates the semantic similarity of a text segment T_i *with respect to* a text segment T_j. This definition provides us with the flexibility we need to handle applications where the directional knowledge is useful (e.g. entailment), and at the same time it gives us the means to handle bidirectional similarity through a simple combination of two unidirectional metrics.

For a given pair of text segments, we start by creating *sets* of open-class words, with a separate set created for nouns, verbs, adjectives, and adverbs. In addition, we also create a set for cardinals, since numbers can also play an important role in the understanding of a text. Next, we try to determine pairs of similar words across the sets corresponding to the same open-class in the two text segments. For nouns and verbs, we use a measure of semantic similarity based on WordNet, while for the other word classes we apply lexical matching[1].

[1]The reason behind this decision is the fact that most of the semantic similarity measures apply only to nouns and verbs, and there are only one or two relatedness metrics that can be applied to adjectives and adverbs.

For each noun (verb) in the set of nouns (verbs) belonging to one of the text segments, we try to identify the noun (verb) in the other text segment that has the highest semantic similarity ($maxSim$), according to one of the six measures of similarity described in Section 2.1. If this similarity measure results in a score greater than 0, then the word is added to the set of similar words for the corresponding word class WS_{pos}[2]. The remaining word classes: adjectives, adverbs, and cardinals, are checked for lexical similarity with their counter-parts and included in the corresponding word class set if a match is found.

The similarity between the input text segments T_i and T_j is then determined using a scoring function that combines the word-to-word similarities and the word specificity:

$$sim(T_i, T_j)_{T_i} = \frac{\sum_{pos}(\sum_{\mathbf{w}_k \in \{WS_{pos}\}} (maxSim(\mathbf{w}_k) * idf_{\mathbf{w}_k}))}{\sum_{\mathbf{w}_k \in \{T_{i_{pos}}\}} idf_{\mathbf{w}_k}} \quad (7)$$

This score, which has a value between 0 and 1, is a measure of the directional similarity, in this case computed with respect to T_i. The scores from both directions can be combined into a bidirectional similarity using a simple product function:

$$sim(T_i, T_j) = sim(T_i, T_j)_{T_i} \times sim(T_i, T_j)_{T_j} \quad (8)$$

3 A Walk-Through Example

We illustrate the application of the text similarity measure with an example. Given two text segments, as shown in Figure 1, we want to determine a score that reflects their semantic similarity. For illustration purposes, we restrict our attention to one measure of word-to-word similarity, the **Wu & Palmer** metric.

First, the text segments are tokenized, part-of-speech tagged, and the words are inserted into

[2]All similarity scores have a value between 0 and 1. The similarity threshold can be also set to a value larger than 0, which would result in tighter measures of similarity.

Text Segment 1: The jurors were taken into the courtroom in groups of 40 and asked to fill out a questionnaire.

- Set_{NN} = {juror, courtroom, group, questionnaire}
  Set_{VB} = {be, take, ask, fill}
  Set_{RB} = {out}
  Set_{CD} = {40}

Text Segment 2: About 120 potential jurors were being asked to complete a lengthy questionnaire.

- Set_{NN} = {juror, questionnaire}
  Set_{VB} = {be, ask, complete}
  Set_{JJ} = {potential, lengthy}
  Set_{CD} = {120}

Figure 1: Two text segments and their corresponding word class sets

their corresponding word class sets. The sets obtained for the given text segments are illustrated in Figure 1.

Starting with each of the two text segments, and for each word in its word class sets, we determine the most similar word from the corresponding set in the other text segment. As mentioned earlier, we seek a WordNet-based semantic similarity for nouns and verbs, and only lexical matching for adjectives, adverbs, and cardinals. Table 3 shows the word semantic similarity scores computed starting with the first text segment.

Text 1	Text 2	maxSim	IDF
jurors	jurors	1.00	5.80
courtroom	jurors	0.30	5.23
questionnaire	questionnaire	1.00	3.57
groups	questionnaire	0.29	0.85
were	were	1.00	0.09
taken	asked	1.00	0.28
asked	asked	1.00	0.45
fill	complete	0.86	1.29
out	–	0	0.06
40	–	0	1.39

Table 1: Wu & Palmer word similarity scores for computing text similarity with respect to text 1

Next, we use equation 7 and determine the semantic similarity of the two text segments with respect to text 1 as 0.6702, and with respect to text 2 as 0.7202. Finally, the two figures are combined into a bidirectional measure of similarity, calculated as 0.4826 based on equation 8.

Although there are a few words that occur in both text segments (e.g. *juror*, *questionnaire*), there are also words that are not identical, but closely related, e.g. *courtroom* found similar to *juror*, or *fill* which is related to *complete*. Unlike traditional similarity measures based on lexical matching, our metric takes into account the semantic similarity of these words, resulting in a more precise measure of text similarity.

4 Application 1: Paraphrase and Entailment Recognition

To test the effectiveness of the text semantic similarity measures, we use them to automatically identify if two text segments are paraphrases of each other. We use the Microsoft paraphrase corpus (Dolan *et al.* 04), consisting of 4,076 training and 1,725 test pairs, and determine the number of correctly identified paraphrase pairs in the corpus using the text semantic similarity measure as the only indicator of paraphrasing. The paraphrase pairs in this corpus consist of two text segments labeled with a unique identifier, which were automatically collected from thousands of news sources on the Web over a period of 18 months. The pairs were manually annotated by human judges who determined if the two sentences in a pair were "semantically equivalent".

In addition, we also evaluate the measure using the PASCAL corpus (Dagan *et al.* 05), consisting of 1,380 test–hypothesis pairs with a directional entailment (580 development pairs and 800 test pairs). The text segment pairs in this data set are assigned with a unique identifier and a *true* or *false* label, indicating if the test sentence entails the hypothesis or not.

For each of the two data sets, we conduct two evaluations, under two different settings: (1) An unsupervised setting, where the decision on what constitutes a paraphrase (entailment) is made using a constant similarity threshold of 0.25 across all experiments; and (2) A supervised setting, where the optimal threshold and weights associated with various similarity metrics are determined through learning on training data. In the latter case, we use a voted perceptron algorithm (Freund & Schapire 98)[3], trained on examples formed with numerical features consisting of the value of all the similarity measures.

[3]Classification using this algorithm was determined optimal empirically through experiments.

Metric	Acc.	Prec.	Rec.	F
Semantic similarity (knowledge-based)				
J & C	0.693	0.722	0.871	0.790
L & C	0.695	0.724	0.870	0.790
Lesk	0.693	0.724	0.866	0.789
Lin	0.693	0.716	0.887	0.792
W & P	0.690	0.702	0.921	0.800
Resnik	0.690	0.690	0.964	0.804
Combined	**0.700**	0.719	0.893	0.796
Baselines				
Vectorial	0.654	0.716	0.795	0.753
Random	0.513	0.683	0.500	0.578

Table 2: Text semantic similarity for paraphrase identification (unsupervised)

Metric	Acc.	Prec.	Rec.	F
Semantic similarity (knowledge-based)				
J & C	0.573	0.543	0.908	0.680
L & C	0.569	0.543	0.870	0.669
Lesk	0.568	0.542	0.875	0.669
Resnik	0.565	0.541	0.850	0.662
Lin	0.563	0.538	0.878	0.667
W & P	0.558	0.534	0.895	0.669
Combined	**0.583**	0.561	0.755	0.644
Baselines				
Vectorial	0.528	0.525	0.588	0.555
Random	0.486	0.486	0.493	0.489

Table 3: Text semantic similarity for entailment identification (unsupervised)

Metric	Acc.	Prec.	Rec.	F
Semantic similarity (knowledge-based)				
Lin	0.702	0.706	0.947	0.809
W & P	0.699	0.705	0.941	0.806
L & C	0.699	0.708	0.931	0.804
J & C	0.699	0.707	0.935	0.805
Lesk	0.695	0.702	0.929	0.800
Resnik	0.692	0.705	0.921	0.799
Combined	**0.715**	0.723	0.925	0.812
Baselines				
Vectorial	0.665	0.665	1.000	0.799
Most frequent	0.665	0.665	1.000	0.799

Table 4: Text semantic similarity for paraphrase identification (supervised)

Metric	Acc.	Prec.	Rec.	F
Semantic similarity (knowledge-based)				
L & C	0.583	0.573	0.650	0.609
W & P	0.580	0.570	0.648	0.607
Resnik	0.579	0.572	0.628	0.598
Lin	0.574	0.568	0.620	0.593
J & C	0.575	0.566	0.643	0.602
Lesk	0.573	0.566	0.633	0.597
Combined	**0.589**	0.579	0.650	0.612
Baselines				
Most frequent	0.500	0.500	1.000	0.667
Vectorial	0.479	0.484	0.645	0.553

Table 5: Text semantic similarity for entailment identification (supervised)

We evaluate the text similarity metric built on top of the various word-to-word metrics introduced in Section 2.1. For comparison, we also compute two baselines: (1) A random baseline created by randomly choosing a true or false value for each text pair; and (2) A vectorial similarity baseline, using a cosine similarity measure as traditionally used in information retrieval, with *tf.idf* weighting. For comparison, we also evaluated the corpus-based similarity obtained through LSA; however, the results obtained were below the lexical matching baseline and are not reported here.

For paraphrase identification, we use the bidirectional similarity measure, and determine the similarity with respect to each of the two text segments in turn, and then combine them into a bidirectional similarity metric. For entailment identification, since this is a directional relation, we only measure the semantic similarity with respect to the *hypothesis* (the text that is entailed).

We evaluate the results in terms of accuracy, representing the number of correctly identified true or false classifications in the test data set. We also measure precision, recall and F-measure, calculated with respect to the *true* values in each of the test data sets.

Tables 2 and 3 show the results obtained in the unsupervised setting, when a text semantic similarity larger than 0.25 was considered to be an indicator of paraphrasing (entailment). We also evaluate a metric that combines all the similarity measures, including the lexical similarity, using a simple average, with results indicated in the **Combined** row.

The results obtained in the supervised setting are shown in Tables 4 and 5. The optimal combination of similarity metrics and optimal threshold are now determined in a learning process performed on the training set. Under this setting, we also compute an additional baseline, consisting of the most frequent label, as determined from the training data.

5 Application 2: Word Sense Similarity

As a second evaluation testbed for the measures of semantic similarity, we considered another application, namely the unsupervised clustering of the WordNet senses (Miller 95). The goal of this application is to reduce the often criticized fine granularity of the WordNet sense inventory (Palmer & Dang 05), by merging the senses of a given word based on the similarities of the corresponding glossary definitions. Comparative evaluations are performed by comparing the quality of the sense clusters obtained with different measures of definition semantic similarity, versus simpler methods based on lexical similarity.

The sense clustering process proceeds as follows. First, we create a similarity matrix for all the possible senses of a word, based on the pairwise similarities of the corresponding glossary definitions. Next, an unsupervised agglomerative average link clustering algorithm (Jain & Dubes 88) is used to find the sense clusters (groupings). The clustering algorithm starts with the set of all word senses, and considers every sense as an individual cluster. Then, at every iteration, it merges the two most similar clusters and recalculates the similarity matrix. Finally, once the clustering process stops, the resulting set of sense clusters can be used as a coarse-grained sense inventory for the given word.

An important aspect of any agglomerative clustering algorithm is the stopping criterion that interrupts the clustering process, thus preventing the construction of a single large cluster. In the current implementation, the clustering stops when the similarity of the two most similar synsets falls below the median of all pairwise similarities (excluding self similarities), as measured in the initial state. A similar stopping criterion was successfully used in other algorithms for word sense clustering, e.g. (Chklovski & Mihalcea 03).

To evaluate the sense clustering algorithm, we use a set of 42 highly ambiguous verbs from WordNet, and their corresponding manually created sense clusters. This data set was constructed by trained linguists and lexicographers at University of Pennsylvania, as part of a larger sense clustering project [4].

The quality of the generated sense clusters is measured using two metrics considered standard in clustering evaluation, namely *purity* and *entropy* (Zhao & Karypis 01), computed for the automatically generated sense groupings relative to the gold standard clusters.

The *entropy* shows how the various gold-standard clusters are distributed within each automatically generated cluster. The entropy of a cluster S_r of size n_r is defined as:

$$E(S_r) = -\frac{1}{\log q} \sum_{i=1}^{q} \frac{n_r^i}{n_r} \log \frac{n_r^i}{n_r}$$

where q is the number of clusters in the gold standard, and n_r^i is the number of senses from the gold standard cluster i found in cluster r. The overall entropy of a clustering solution is calculated as the weighted sum of the entropy values of individual clusters, where each cluster is weighted by the number of synsets forming the cluster, and the overall sum is normalized with the total number of synsets.

The *purity* of an automatically generated cluster is defined as the fraction represented by the largest cluster of senses from the gold standard assigned to that cluster:

$$P(S_r) = \frac{1}{n_r} max_i(n_r^i)$$

The overall purity of a clustering solution is the weighted sum of the individual cluster purities.

To evaluate the measures of text similarity, we perform comparative evaluations of sense clustering solutions using semantic similarity metrics computed between glossary definitions. We use the measures introduced in Section 2.1, plus an additional **Combined** measure, which computes the arithmetic average of the individual metrics. The baseline once again is represented by the traditional measure of lexical similarity. We also calculate a random baseline, where the definition similarities are determined as random numbers between 0 and 1.

Table 6 shows the purity and entropy values of the sense clusters obtained with the semantic similarity metrics, as well as the two baselines. Note

[4] Martha Palmer, personal communication.

Metric	Entropy	Purity
Sense Clustering		
Lin	**0.163**	**0.835**
Resnik	0.176	0.827
W & P	0.174	0.824
J & C	0.186	0.823
Lesk	0.186	0.819
L & C	0.186	0.816
Combined	**0.163**	0.833
Baseline		
Lexical Similarity	0.191	0.818
Random	0.248	0.761

Table 6: Sense clustering results using semantic similarities of gloss definitions

that all evaluation measures take values between 0 and 1, with smaller entropy values and higher purity values representing clusters of higher quality (a perfect match with the gold standard would be represented by a set of clusters with entropy of 0 and purity of 1).

All the semantic similarity measures lead to sense clusters that are better than those obtained with the lexical similarity measure. The best results are obtained with the **Lin** metric, for an overall entropy of 0.163 and a purity of 0.835, at par with the **Combined** measure that combines together all the individual metrics. It is also worth noting that the random baseline establishes a rather high lower bound: 0.248 entropy and 0.761 purity. Considering these values as the origin of a 0–100% evaluation scale, the improvements brought by the best semantic similarity measure (Lin) relative to the lexical matching baseline translate into 11.6% and 6.7% improvement for entropy and purity respectively. These figures are competitive with previously published sense clustering results (Agirre & Lopez 03), (Chklovski & Mihalcea 03),

6 Discussion and Conclusions

For the task of paraphrase recognition, incorporating semantic information into the text similarity measure increases the likelihood of recognition significantly over the random baseline and over the vectorial similarity baseline. In the unsupervised setting, the best performance is achieved using a method that combines several similarity metrics into one, for an overall accuracy of 70.0%. When learning is used to find the optimal combination of metrics and optimal threshold, the highest accuracy of 71.5% is obtained by combining the semantic similarity metrics and the lexical similarity together.

For the entailment data set, although we do not explicitly check for entailment, the directional similarity computed for textual entailment recognition does improve over the random and vectorial similarity baselines. Once again, the combination of similarity metrics gives the highest accuracy, measured at 58.3%, with a slight improvement observed in the supervised setting, where the highest accuracy was measured at 58.9%. Both these figures are competitive with the best results achieved during the PASCAL entailment evaluation (Dagan *et al.* 05).

For the word sense similarity application, the clusters obtained using the semantic similarity metrics have higher purity and lower entropy as compared to those generated based on the simpler lexical matching measure. The best clustering solution is obtained with the **Lin** similarity metric, for an entropy of 0.163 and a purity of 0.835, which represent a clear improvement with respect to the lexical matching baseline, taking also into account the competitive lower bound obtained through random clustering.

Although our method relies on a bag-of-words approach, as it turns out the use of measures of *semantic* similarity improves significantly over the traditional lexical matching metrics[5]. We are nonetheless aware that a bag-of-words approach ignores many of important relationships in sentence structure, such as dependencies between words, or roles played by the various arguments in the sentence. Future work will consider the investigation of more sophisticated representations of sentence structure, such as first order predicate logic or semantic parse trees, which should allow for the implementation of more effective measures of text semantic similarity.

[5]The improvement of the combined semantic similarity metric over the simpler lexical similarity measure was found to be statistically significant in all experiments, using a paired t-test ($p < 0.001$).

References

(Agirre & Lopez 03) E. Agirre and O. Lopez. Clustering WordNet word senses. In *Proceedings of Recent Advances In NLP (RANLP 2003)*, September 2003.

(Budanitsky & Hirst 01) A. Budanitsky and G. Hirst. Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures. In *Proceedings of the NAACL Workshop on WordNet and Other Lexical Resources*, Pittsburgh, June 2001.

(Chklovski & Mihalcea 03) T. Chklovski and R. Mihalcea. Exploiting agreement and disagreement of human annotators for word sense disambiguation. In *Proceedings of Recent Advances In NLP (RANLP 2003)*, September 2003.

(Dagan *et al.* 05) I. Dagan, O. Glickman, and B. Magnini. The PASCAL recognising textual entailment challenge. In *Proceedings of the PASCAL Workshop*, 2005.

(Dolan *et al.* 04) W.B. Dolan, C. Quirk, and C. Brockett. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th International Conference on Computational Linguistics*, Geneva, Switzerland, 2004.

(Freund & Schapire 98) Y. Freund and R.E. Schapire. Large margin classification using the perceptron algorithm. In *Proceedings of the 11th Annual Conference on Computational Learning Theory*, pages 209–217, New York, NY, 1998. ACM Press.

(Jain & Dubes 88) A.K. Jain and R.C. Dubes. *Algorithms for clustering data*. Prentice Hall, Englewood Cliffs, N.J., 1988.

(Jiang & Conrath 97) J. Jiang and D. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the International Conference on Research in Computational Linguistics*, Taiwan, 1997.

(Landauer *et al.* 98) T. K. Landauer, P. Foltz, and D. Laham. Introduction to latent semantic analysis. *Discourse Processes*, 25, 1998.

(Leacock & Chodorow 98) C. Leacock and M. Chodorow. Combining local context and WordNet sense similiarity for word sense disambiguation. In *WordNet, An Electronic Lexical Database*. The MIT Press, 1998.

(Lesk 86) M.E. Lesk. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the SIGDOC Conference 1986*, Toronto, June 1986.

(Lin & Hovy 03) C.Y. Lin and E.H. Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of Human Language Technology Conference (HLT-NAACL 2003)*, Edmonton, Canada, May 2003.

(Lin 98) D. Lin. An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning*, Madison, WI, 1998.

(Miller 95) G. Miller. Wordnet: A lexical database. *Communication of the ACM*, 38(11):39–41, 1995.

(Palmer & Dang 05) M. Palmer and H. T. Dang. Making fine grained and coarse-grained sense distinctions, both manually and automatically. *Natural Language Engineering*, 1, 2005.

(Papineni *et al.* 02) K. Papineni, S. Roukos, T. Ward, and W. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, Philadelphia, PA, July 2002.

(Patwardhan *et al.* 03) S. Patwardhan, S. Banerjee, and T. Pedersen. Using measures of semantic relatedness for word sense disambiguation. In *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*, Mexico City, February 2003.

(Resnik 95) P. Resnik. Using information content to evaluate semantic similarity. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Montreal, Canada, 1995.

(Rocchio 71) J. Rocchio. *Relevance feedback in information retrieval*. Prentice Hall, Ing. Englewood Cliffs, New Jersey, 1971.

(Salton & Lesk 71) G. Salton and M.E. Lesk. *Computer evaluation of indexing and text processing*, pages 143–180. Prentice Hall, Ing. Englewood Cliffs, New Jersey, 1971.

(Salton *et al.* 97a) G. Salton, , and A. Bukley. Term weighting approaches in automatic text retrieval. In *Readings in Information Retrieval*. Morgan Kaufmann Publishers, San Francisco, CA, 1997.

(Salton *et al.* 97b) G. Salton, A. Singhal, M. Mitra, and C. Buckley. Automatic text structuring and summarization. *Information Processing and Management*, 2(32), 1997.

(Sparck-Jones 72) K. Sparck-Jones. A statistical interpretation of term specificity and its applicatoin in retrieval. *Journal of Documentation*, 28(1):11–21, 1972.

(Voorhees 93) E. Voorhees. Using wordnet to disambiguate word senses for text retrieval. In *Proceedings of the 16th annual international ACM SIGIR conference*, Pittsburgh, PA, 1993.

(Wu & Palmer 94) Z. Wu and M. Palmer. Verb semantics and lexical selection. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, Las Cruces, New Mexico, 1994.

(Xu & Croft 96) J. Xu and W. B. Croft. Query expansion using local and global document analysis. In *Proceedings of the 19th annual international ACM SIGIR conference*, Zurich, Switzerland, 1996.

(Zhao & Karypis 01) Y. Zhao and G. Karypis. Criterion functions for document clustering: Experiments and analysis, 2001. Technical Report TR 01–40, Department of Computer Science, University of Minnesota.

Comparing methods for automatic acquisition of Topic Signatures

Montse Cuadros, Lluis Padro
TALP Research Center
Universitat Politecnica de Catalunya
C/Jordi Girona, Omega S107
08034 Barcelona
{cuadros, padro}@lsi.upc.es

German Rigau
IXA Group
Euskal Herriko Unibertsitatea
P.Manuel Irdiazabal, 1
20018 Donostia
rigau@si.ehu.edu

Abstract

The main goal of this work is to compare two methods for building Topic Signatures, which are vectors of weighted words acquired from large corpora. We used two different software tools, ExRetriever and Infomap, for acquiring Topic Signatures from corpus. Using these tools, we retrieve sense examples from large text collections. Both systems construct a query for each word sense using WordNet. The quality of the acquired Topic Signatures is indirectly evaluated on the Word Sense Disambiguation English Lexical Task of Senseval-2.

keywords: Topic Signatures, acquisition, Latent Semantic Indexing, Word Sense Disambiguation, Multilingual Central Repository, WordNet.

1 Introduction

Topic Signatures (TS) are word vectors related to a particular topic. Topic Signatures are built by retrieving context words of a target word from large text collections. They have been used in a variety of ways, such as in Summarization Tasks (Lin & Hovy 00), ontology population (Alfonseca *et al.* 04) or word sense disambiguation (Agirre *et al.* 00), (Agirre *et al.* 01). In fact, there is now available Topic Signatures for all WordNet (Fellbaum 98) nominal senses (Agirre & de laCalle 04).

This work presents a comparison of two different techniques for building Topic Signatures.

The first technique retrieve contexts using queries which consist of a set of literal words. Although these systems have been improved with several enhancements such as term weighting, authority linking, and ad-hoc heuristics to improve their performance, these lexical matching methods can be inaccurate because the queries are based on words instead of concepts. However, there are many ways to characterize a given concept.

The second technique uses Latent Semantic Indexing (LSI). LSI tries to overcome the problems of lexical matching by using statistically derived conceptual indexes instead of literal words for retrieval. This technique assumes that there is some underlying latent semantic structure in the data.

A Topic Signature, for our purposes, is a weighted vector of words related to a particular word sense. We tried two different systems for build Topic Signatures. The first one, ExRetriever (Cuadros *et al.* 04), is based on the first technique described above, and the second one, Infomap (Dorow & Widdows 03), is based on the second technique.

Our main goal with this study, as mentioned before, is to compare the performances of both methods for automatic TS acquisition. In order to perform this comparison, we evaluated the TS acquired by both systems in a specific task, the English-Lexical Sample task of Senseval-2.

For building the Topic Signatures for all the word senses of the Lexical Sample task of Senseval-2 we used BNC (British National Corpus).

This paper is organised as follows: In section 2, we explain in detail the software tools we use for the task, providing a brief explanation of Latent Semantic Indexing (LSI). In section 3, we explain the steps followed to construct the Topic Signatures and in section 4, the results of the indirect evaluation we carried out. Finally, in section 5 and 6, some concluding remarks and future work are provided.

2 Tools

2.1 ExRetriever

ExRetriever is a flexible tool to perform sense queries on large corpora (Cuadros *et al.* 04). ExRetriever characterises automatically each synset of a word as a query (mainly using: synonyms, hyponyms and the words of the definitions); and then, uses these queries to obtain sense examples (sentences) automatically from a large text collection. The current implementation of

ExRetriever accesses directly the content of the Multilingual Central Repository (MCR) (Atserias *et al.* 04) of the MEANING project which includes several WordNet versions. The system uses also SWISH-E[1] to index large collections of text such as SemCor (Miller *et al.* 93) or BNC. SWISH-E is a fast, powerful, flexible, free, and easy to use system for indexing collections of Web pages or other files. ExRetriever has been designed to be easily ported to other lexical knowledge bases and corpora, including the possibility to query search engines such as Google.

2.2 Infomap

The Infomap NLP Software package[2] uses a variant of Latent Semantic Indexing (LSI) on free-text corpora to learn vectors representing the meanings of words in a reduced vector-space known as Word-Space (Dorow & Widdows 03).

The Infomap software performs two basic functions: building models by learning them from a free-text corpus using certain learning parameters specified by the user, and searching an existing model to find the words or documents that best match a query according to that model.

2.2.1 Latent Semantic Indexing

Latent Semantic Indexing (LSI) allows to extract and represent the contextual meaning of words by statistical computations applied to a large corpus of text (Schtze 98). The underlying idea is that when reducing the dimensionality of the original word-space, similar words are projected closer to each other in the reduced space while dissimilar words are projected to distant locations. The reduced space is obtained using linear algebra methods, in particular, the Singular Value Decomposition (SVD). Part of the motivation for using SVD for word vectors is the success of LSI in information retrieval.

Latent Semantic Indexing maps the contextual relationships between words in terms of common usage across a collection of documents. LSI enables to understand how words relate to each other through the creation of a similarity measure, which reveals whether a given word or document is similarly used compared with another word or document.

[1] `http://swish-e.org`
[2] `http://infomap-nlp.sourceforge.net/`

3 Strategies for acquiring Topic Signatures

In order to evaluate the performance of both approaches, we designed a preliminary set of strategies for acquiring the Topic Signatures from BNC.

3.1 Acquisition Process

The acquisition process consist of the following steps:

1. Devise a particular strategy for query construction and apply the query construction schema to all the senses of a word.
2. Perform the sense queries on the BNC.
3. Collect the sense corpus.
4. Obtain a Topic Signature for each sense.

3.2 Query construction strategies

We have designed a few preliminary set of query construction strategies based on synonymy, hyponymy and hypernymy relationship of WordNet inspired by the work of (Leacock *et al.* 98).

- A) Monosemous strategy : (OR monosemous-words) the union set of all the synonym, hyponym and hyperonym monosemous words of a WordNet sense.
- B) Polysemous strategy : (OR polysemous-words) the union set of all the synonym, hyponym and hyperonym polisemous words of a WordNet sense.
- C) Monosemous and Polysemous strategy : (word AND (OR polysemous-words)) OR* (OR monosemous-words) the union set of all synonym, hyponym and hyperonym monosemous and polisemous words of a WordNet sense in such a way. OR* stands for a particular OR boolean function to express that there is at least one monosemous word or the word and one polysemous word.

We remove those words (monosemous or polysemous) appearing in more than one sense query, trying to construct the sense queries in such a way, that there is no overlapping words in different sense queries of the same word.

3.3 Construction of the Topic Signatures using ExRetriever

These queries have been applied to locate particular sentences of the BNC using ExRetriever. In that way, we are able to retrieve a set of examples for each word sense. In all cases, we remove all stop words from the corpus. Afterwards, we calculate the Mutual Information for each word in the sense corpus with respect to their synset using the formula (1).

$$MI(w,s) = log\frac{P(w \wedge s)}{P(w)P(s)} \quad (1)$$

Given a word w and their word sense s, $P(w \wedge s)$ represents the probability of appearing w in the s sense. $P(w)$ is the probability of occurring w in the BNC corpus, and $P(s)$ is the probability of a document (sentence) to belong to the s sense.

As an example, we will show the full process of obtaining a Topic Signature.

For example, a query of type C for the word *church#n* is constructed using ExRetriever as follows:

As WordNet 1.7 *church#n* has three senses, ExRetriever builds three different queries:

- sense 1: ((church and (christianity or protestant or religion)) or christian_church or catholic_church or coptic_church)
- sense 2: ((church and (abbey or basilica or cathedral)) or church_building or kirk or place_of_worship or house_of_prayer or house_of_god)
- sense 3: ((church and (service)) or church_service or religious_service or divine_service)

Once we construct each sense query, we use ExRetriever to gather all matching sentence examples from the BNC corpus. Afterwards, we calculate the Mutual Information of all the words appearing in the corpora obtained.

We have calculated the Topic Signatures for query A and C, in an improved method based on not taking account the case of the words and looking for the appearance of the exact compound-words in the gathered examples.

After this process, we obtain per each word sense, a word vector with weights (Topic Signatures). Table 1 presents some resulting words for sense 3 of *church#n* using the strategy A).

witness	2.229616	context	2.411937
burial	2.517298	husband	2.517298
participants	2.517298	sermon	2.517298
service	2.715123	adapted	2.715123
adults	2.715123	afternoon	2.922763
agenda	2.922763	arranged	2.922763
attracted	2.922763	audible	2.922763
augment	2.922763	award	2.922763

Table 1: Example of a Topic Signature obtained with ExRetriever

3.4 Construction of the Topic Signatures using Infomap

Infomap only allows AND and ANDNOT operator and does not consider the OR operator. For this reason, the queries have been modified slightly. We use the same words that we used when querying with ExRetriever but we remove all the operators (by default Infomap uses the AND operator).

After building a model with the corpus, the *associate* command of Infomap can return both a list of the words or the documents best matching the query, in descending order of relevance. Using this option provided by Infomap, once we have the queries, we obtain the list of weighted words that in this experiment we consider the Topic Signature of the query. Table 2 presents the resulting words for sense 3 of *church#n* using the strategy C) with higher relevance.

service	0.776187	anglican	0.651298
church	0.776186	services	0.651127
clergy	0.718070	tower	0.651071
hymns	0.695500	st	0.650787
peterś	0.695215	congregational	0.648595
episcopal	0.689341	congregation	0.647037
presbyterian	0.685548	priest	0.644656
cathedral	0.685220	memorial	0.644652
churches	0.683878	charters	0.642540
royal	0.673297	worship	0.637472
parish	0.671534	bishop	0.634107
pastoral	0.670789	volunteer	0.629541
maryś	0.666601	...	

Table 2: Example of a Topic Signature obtained with Infomap

4 Indirect evaluation on Word Sense Disambiguation

In order to measure the quality of the acquired TS by these two different approaches, we performed an indirect evaluation by using the acquired Topic Signatures (TS) for a Word Sense Disambiguation (WSD) task. In particular, the Senseval-2

English Lexical Sample task. We used this evaluation framework instead of the the one provided by Senseval-3 because in this case, the verbal part was not directly annotated using WordNet senses.

The TS are applied to all the examples of the test set of the Senseval-2 using a simple word overlapping (or weighting) counting. That is, the program calculates the total number of overlapping words between the Topic Signature and the test example. The sense having higher counting (or weighting) is selected for that particular test example. In table 3, we can see an example of the evaluation test corresponding to sense 3 of *church#n*. As we can see, in bold there are some words that appear in the Topic Signatures for sense 3 obtained using Infomap.

In table 4 appears a summary of the results of this indirect evaluation. This table presents the results for each type of query construction strategy (either A, B or C), each system (either Infomap or ExRetriever), and with several levels of sense granularity (either fine or coarse). In this table, P stands for Precision, R for Recall and F1 for F1 measure.

The best figures are obtained by using the Infomap method with occurrences, which is not surprising due to the LSI effect (39.1 precision and recall for fine grained granularity). In table 4, we present the official results of the Senseval-2 of those systems declared to be unsupervised. When comparing with those systems, Infomap would score second while ExRetriever fourth getting as a reference the recall in fine-grained. Looking at literature, (Agirre & Martinez 04), UNED-LS-U unsupervised method is considered semi-supervised. This approach, uses some heuristics rely on the bias information available in Semcor. The distribution of senses is used to discard low-frequency senses.

In table 4, we present the results of the queries for each system based on POS, and we can see that the best query for each POS always rely on A, the only difference is that sometimes the best result uses the occurrence or the weight measure method. We have put the results of the improved methods for ExRetriever. If we had used the best method for each part of speech, we had improved our results achieving a precision of 31.5, a recall of 29.7 and a f1 of 30.57 which would imply to be one position over in the 4 results for ExRetriever. Otherwise, Infomap would improve not very significantly, we would get a precision and recall of 39.3, that would mean that we would be in the same position.

As expected, regarding the query construction strategy, in general it seems that strategy A (Monosemous strategy), is better than C (Monosemous and Polysemous strategy) and B (Polysemous strategy), which is the one with the lowest results. We also obtain similar figures with respect occurrences vs. weights methods: using Infomap we obtain slightly better figures for occurrences while when using ExRetriever the best results appear for weights.

5 Conclusions

We presented some experiments using two software tools to compare the automatic acquisition of Topic Signatures for word senses. Our Evaluation Framework has been the English Lexical Sample task of Senseval-2. We have focus on the Senseval-2 task because it uses the synsets of WordNet 1.7 for each part of speech, and then is more reliable to our experiments because our queries are build with WordNet 1.7.

We can observe that using Infomap, the tool developed to work with vector models acquired from Corpus, we obtain promising results.

In order to improve the ExRetriever results we plan to filter out those words that seem to be very common in all senses, for example, Named Entities, Multi Words Expressions, etc. or keeping those words that have a common domain or any other semantic relation in common.

Infomap vectors seem to be more accurate for obtaining good context words of an specific word sense. Furthermore, it seems that the results could improve largely varying different system parameters such as dimensionalty of the model, size of the Topic Signatures, etc.

We also plan to tune separately each part-of-speech.

6 Acknowledgements

This work is supported by the European Commission (MEANING IST-2001-34460). Our research group, TALP Research Center, is recognized as a Quality Research Group (2001 SGR 00254) by DURSI, the Research Department of the Catalan Government.

In developing measuring tools for the local **church** we are concerned with quality control as much as quantity performance, to use commercial language. Responsible leaders want to know how people are growing in their understanding of the Christian faith, whether relationships are deepening and extending throughout the **church-fellowship**, and to what extent the Christian presence is evident in the community outside. Such information cannot be gathered with such precision as numerical data, but it is essential that each area be investigated to ensure that there is a balance between **worship**, fellowship, learning, evangelism and **service**. Healthy organic growth is proportionate, with each area and function developing in relation to the other. Quality of <*head*> **church** <*head*> life can be measured in the following three ways

Table 3: Test example for noun church

		fine			**coarse**		
Method	Query	P	R	F1	P	R	F1
Infomap	A	**39.1**	**39.1**	**39.1**	**51.0**	**51.0**	**51.0**
occurrences	B	37.8	33.2	35.3	50.0	43.8	46.7
	C	37.8	33.2	33.2	50.0	43.8	46.7
Infomap	A	**39.1**	**39.1**	**39.1**	**50.7**	**50.7**	**50.7**
weights	B	38.4	32.8	35.4	49.9	42.7	46.02
	C	38.4	32.8	35.38	49.9	42.7	46.02
ExRetriever	A	**28.5**	**27.1**	**27.8**	**42.3**	**40.3**	**41.3**
occurrences	B	24.1	17.2	20.0	35.4	25.3	29.5
	C	21.7	21.3	21.5	36.6	36.0	36.3
ExRetriever	A	**28.9**	**27.2**	**28.02**	**41.9**	**39.3**	**40.6**
weight	B	22.6	15.9	18.67	33.0	23.2	27.3
	C	25.1	24.6	24.85	36.9	36.1	36.5

Table 4: Overall results of the systems using Senseval-2 with respect fine-grained and coarse-grained senses

Method	Query	Noun	Verb	Adj
Infomap	A	40.1	**32.2**	**53.3**
occurrences	B	34.26	29.47	51.29
	C	34.26	29.47	51.29
Infomap	A	**40.6**	31.7	53
weights	B	34.93	29.19	50.77
	C	34.93	29.19	50.77
ExRetriever	A	27.8	**28**	**27.03**
occurrences	C	25.3	17.1	22.79
ExRetriever	A	**34.6**	23.25	23.64
weights	C	32.45	18.2	23.39

Table 5: F1 related to each POS

References

(Agirre & de laCalle 04) E. Agirre and O. Lopez de la Calle. Publicity available topic signatures for all wordnet nominal senses. In *LREC'04*, pages 97–104, 2004.

(Agirre & Martinez 04) E. Agirre and D. Martinez. Unsupervised wsd based on automatically retrieved examples: The importance of bias. In *Proceedings of the EMNLP*, Barcelona, 2004.

(Agirre *et al.* 00) E. Agirre, O. Ansa, D. Martinez, and E. Hovy. Enriching very large ontologies with topic signatures. In *Proceedings of ECAI'00 workshop on Ontology Learning*, Berlin, Germany, 2000.

(Agirre *et al.* 01) E. Agirre, O. Ansa, D. Martez, and E. Hovy. Enriching wordnet concepts with topic signatures. In *Proceedings of the NAACL workshop on WordNet and Other lexical Resources: Applications, Extensions and Customizations*, Pittsburg, 2001.

(Alfonseca *et al.* 04) E. Alfonseca, E. Agirre, and O. Lopez de Lacalle. Approximating hierachy-based similarity for wordnet nominal synsets using topic signatures. In *Proceedings of the Second International Global WordNet Conference (GWC'04). Panel on figurative language*, Brno, Czech Republic, January 2004. ISBN 80-210-3302-9.

	fine			coarse		
Method	P	R	F1	P	R	F1
UNED - LS-U	40.2	40.1	40.15	51.8	51.7	51.75
Infomap - A Occ	**39.1**	**39.1**	**39.1**	**51.0**	**51.0**	**51.0**
ITRI - WASPS-Workbench	58.1	31.9	41.19	66.1	36.3	46.86
CL Research - DIMAP	29.3	29.3	29.3	36.7	36.7	36.7
ExRetriever - A weight	**28.9**	**27.2**	**28.02**	**41.9**	**39.3**	**40.56**
IIT 2 (R)	24.7	24.4	24.55	34.6	34.1	34.35
IIT 1 (R)	24.3	23.9	24.1	34.1	33.6	33.85
IIT 2	23.3	23.2	23.25	32.3	32.2	32.25
IIT 1	22	22	22	32.1	32	32.05

Table 6: Senseval-2 systems results for fine-grained and coarse-grained senses, in wining order

(Atserias *et al.* 04) Jordi Atserias, Luís Villarejo, German Rigau, Eneko Agirre, John Carroll, Bernardo Magnini, and Piek Vossen. The meaning multilingual central repository. In *Proceedings of the Second International Global WordNet Conference (GWC'04)*, Brno, Czech Republic, January 2004. ISBN 80-210-3302-9.

(Cuadros *et al.* 04) M. Cuadros, M. Castillo, G. Rigau, and J. Atserias. Automatic Acquisition of Sense Examples using ExRetriever. In *Iberamia'04*, pages 97–104, 2004.

(Dorow & Widdows 03) B. Dorow and D. Widdows. Discovering corpus-specific word senses. In *EACL*, Budapest, 2003.

(Fellbaum 98) C. Fellbaum, editor. *WordNet. An Electronic Lexical Database.* The MIT Press, 1998.

(Leacock *et al.* 98) C. Leacock, M. Chodorow, and G. Miller. Using Corpus Statistics and WordNet Relations for Sense Identification. *Computational Linguistics*, 24(1):147–166, 1998.

(Lin & Hovy 00) C. Lin and E. Hovy. The automated acquisition of topic signatures for text summarization. In *Proceedings of 18th International Conference of Computational Linguistics, COLING'00*, 2000. Strasbourg, France.

(Miller *et al.* 93) G. Miller, C. Leacock, R. Tengi, and R. Bunker. A Semantic Concordance. In *Proceedings of the ARPA Workshop on Human Language Technology*, 1993.

(Schtze 98) H. Schtze. Automatic word sense discrimination. In *Computational Linguistics*, 1998.

Discovering Topic Boundaries for Text Summarization Based on Word Co-occurrence

Gaël Dias* and Elsa Alves*†

*HULTIG, Department of Computer Science, University of Beira Interior, Covilhã, Portugal
ddg@di.ubi.pt
†GLINT, Department of Computer Science, New University of Lisbon, Lisbon, Portugal
elsalves@zmail.pt

Abstract

Topic Segmentation is the task of breaking documents into topically coherent multi-paragraph subparts. In particular, Topic Segmentation is extensively used in Text Summarization to provide more coherent results by taking into account raw document structure. However, most methodologies are based on lexical repetition that show evident reliability problems or rely on harvesting linguistic resources that are usually available only for dominating languages and do not apply to less favored and emerging languages. In order to tackle these drawbacks, we present an innovative Topic Segmentation system based on a new informative similarity measure based on word co-occurrences and evaluate it on a set of web documents belonging to a single domain.

1. Introduction

This paper introduces a new technique for improving access to information dividing lengthy documents into topically coherent sections. This research area is commonly called Topic Segmentation and can be defined as the task of breaking documents into topically coherent multi-paragraph subparts.

Topic Segmentation has extensively been used in Text Summarization where it serves as the basic text structure in order to apply sentence extraction and sentence compression techniques (Boguraev and Neff, 2000; Angheluta *et al.*, 2002; Farzindar and Lapalme, 2004). In this paper, we present an innovative Topic Segmentation system based on a new informative similarity measure that takes into account word co-occurrence in order to avoid the accessibility to existing linguistic resources such as electronic dictionaries or lexico-semantic databases. In particular, our architecture solves three main problems evidenced by previous research. First, systems based uniquely on lexical repetition show reliability problems (Hearst, 1994; Reynar, 1994; Sardinha, 2002) as common writing rules prevent from using lexical repetition. Second, systems based on lexical cohesion, using existing linguistic resources that are usually only available for dominating languages like English, French or German, do not apply to less favored and emerging languages (Morris and Hirst, 1991; Kozima, 1993). Third, systems that need previously existing harvesting training data (Beeferman *et al.*, 1997) do not adapt easily to new domains as training data is usually difficult to find or build depending on the domain being tackled. Instead, our architecture proposes a language-independent unsupervised solution, similar to (Phillips, 1985; Ponte and Croft, 1997), defending that Topic Segmentation should be done "on the fly" on any text thus avoiding the problems of domain, genre, or language-dependent systems.

In order to show the results of our system in real-world conditions, we propose an evaluation on a set of web documents belonging to a single domain unlike other methodologies that have been evaluated on (Choi, 2000)'s data set that relies on small texts of different domains within which lexical repetition is high. It is clear that this situation does not correspond to real-world conditions for Text summarization as documents to segment are usually from a same domain and do not use repetition.

This paper is divided into four sections. First, we show the weighting process of each word of the input text corpus. Second, we introduce our main contribution i.e. the informative similarity measure. Third, we define how subparts can be elected from the values of the informative similarity measure. And finally, we propose an evaluation on a real-world situation for Text Summarization.

2. Weighting Score

Our algorithm is based on the vector space model which determines the similarity of neighboring groups of sentences and places subtopic boundaries between dissimilar blocks. In our specific case, each

sentence in the corpus is evaluated in terms of similarity with the previous block of *k* sentences and the next block of *k* sentences.

The simplest form of the vector space model treats a document (in our case, a sentence or a group of sentences) as a vector whose values correspond to the number of occurrences of the words appearing in the document as in (Hearst, 1994). Although (Hearst, 1994) showed successful results with this weighting scheme, we strongly believe that the importance of a word in a document does not only depend on its frequency. Indeed, frequency can only be reliable for technical texts where ambiguity is drastically limited and word repetition largely used. But unfortunately, these documents are an exception in the global environment of the internet for example. According to us, two main factors must be taken into account to define the relevance of a word for the specific task of Topic Segmentation: its semantic importance and its distribution across the text. For that purpose, we propose a new weighting scheme based on three heuristics: the well-known *tf.idf* measure, the adaptation of the *tf.idf* measure for sentences, the *tf.isf*, and a new density measure that calculates the density of each word in the text.

2.1 The *tf.idf* Score

The basic idea of the *tf.idf* score (Salton *et al.*, 1975) is to evaluate the importance of a word within a document based on its frequency and its distribution across a set of documents. The *tf.idf* is defined in equation 1 where *w* is a word and *d* a document.

$$tf.idf(w,d) = \frac{tf(w;d)}{|d|} \times \log_2 \frac{N}{df(w)} \qquad (1)$$

However, not all relevant words in a document are useful for Topic Segmentation. For instance, relevant words appearing in all sentences will be of no help to segment the text into topics. For that purpose, we extend the idea of the *tf.idf* to sentences.

2.2 The *tf.isf* Score

The basic idea of the *tf.isf* score is to evaluate each word in terms of its distribution over the document. Indeed, it is obvious that words occurring in many sentences within a document may not be useful for Topic Segmentation purposes. So, we will define the *tf.isf* to evaluate the importance of a word within a document based on its frequency within a given sentence and its distribution across all the sentences within the document. The *tf.isf* score is defined in equation 2 where *w* is a word, *s* a sentence, s*tf(w; s)* the number of occurrences of *w* in *s*, |*s*| the number of words in *s*, *Ns* the number of sentences within the document and *sf(w)* the number of sentences in which the word *w* occurs.

$$tf.isf(w,s) = \frac{stf(w;s)}{|s|} \times \log_2 \frac{Ns}{sf(w)} \qquad (2)$$

However, we can push even further our idea of word distribution. Indeed, a word *w* occurring 3 times in 3 different sentences may not have the same importance in all cases. Let's exemplify. If the 3 sentences are consecutive, the word *w* will have a strong influence on what is said in this specific region of the text. On the opposite, it will not be the case if the word *w* occurs in the first sentence, in the middle sentence and then in the last sentence. For that purpose, we propose a new density measure that calculates the density of each word in a document.

2.3 The Word Density Score

The basic idea of the word density measure is to evaluate the dispersion of a word within a document. So, very disperse words will not be as relevant as dense words. In order to evaluate the word density, we propose a new measure based on the distance of all consecutive occurrences of the word in the document. We call this measure *dens* and is defined in equation 3.

$$dens(w,d) = \sum_{k=1}^{|w|-1} \frac{1}{\ln(dist(occur(k), occur(k+1)) + e)} \qquad (3)$$

For any given word *w*, its density *dens(w,d)* in document d, is calculated from all the distances between all its occurrences, |*w*|. So, *occur(k)* and *occur(k+1)* respectively represent the positions in the text of two consecutive occurrences of the word *w* and *dist(occur(k), occur(k+1))* calculates the distance that separates them in terms of words within the document. Thus, by summing their inverse distances, we get a density function that gives higher scores to highly dense words. As a result, a word, the occurrences of which appear close to one another, will show small distances and as a result a high density. On the opposite, a word, the occurrences of which appear far from each other, will show high distances and as a result a small word density.

2.4 The Weighting Score

The weighting score of any word in a document can be directly derived from the previous three heuristics by combining these three scores as in equation 4 where each score is normalized so that they can be combined.

$$weight(w,d) = \|tf.idf(w,d)\| \times \|tf.isf(w,s)\| \times \|dens(w,d)\| \qquad (4)$$

The next step of the application of the vector space model aims at determining the similarity of neighboring groups of sentences. For that purpose, it

is important to define an appropriate similarity measure. That is the objective of our next section.

3. Similarity Measure

There are a number of ways to compute the similarity between two documents. However, we show that classic similarity measures evidence problems in dealing with semantic information. Most similarity measures determine the distance between two vectors associated to two documents (i.e. Vector Space Model). However, when applying the classic similarity measures between two documents, only the identical indexes of the row vectors X_i and X_j are taken into account. However, this is not tolerable. Indeed, it is clear that both sentences (1) and (2) are similar although they do not share any word in common:

(1) *Ronaldo defeated the goalkeeper once more*.
(2) *Real Madrid striker scored again*.

The most interesting idea to avoid word repetition problems is certainly to identify lexical cohesion relationships between words. Indeed, systems should take into account semantic information that could, for instance, relate *Ronaldo* to *Real Madrid striker*. For that purpose, many authors have proposed to computationally identify these relationships (in particular, the synonym relation) using large linguistic resources such as Wordnet (Angheluta *et al.*, 2002), Roget's thesaurus (Morris and Hirst, 1991) or LDOCE (Kozima, 1993). However, these huge resources are only available for dominating languages and as a consequence do not apply to less favored languages. A much more interesting research direction is proposed by (Ponte and Croft, 1997) that propose a Topic Segmentation technique based on the Local Content Analysis (Xu and Croft, 1996), allowing substituting each sentence with words and phrases related to it. Our methodology is based on this same idea but differs from it as the word co-occurrence information is directly embedded in the calculation of the similarity between blocks of sentences thus avoiding an extra-step in the topic boundaries discovery. For that purpose, we propose a new informative similarity measure that includes in its definition the Equivalence Index Association Measure (*EI*) proposed by (Muller *et al.*, 1997) as in equation 5.

$$EI(w_1, w_2) = p(w_1 \mid w_2) \times p(w_2 \mid w_1) = \frac{f(w_1, w_2)^2}{f(w_1) \times f(w_2)} \qquad (5)$$

The frequency of co-occurrence $f(w_1, w_2)$ between w_1 and w_2 is calculated within a context window from a collection of documents. Our informative similarity measure is defined in equation 6 where $EI(W_{ik}, W_{jl})$ is the Equivalence Index value between W_{ik}, the word that indexes the vector of the document i at position k, and W_{jl}, the word that indexes the vector of the document j at position l.

$$Sij = \text{infosimba}(X_i, X_j) = \frac{\sum_{k=1}^{p}\sum_{l=1}^{p} X_{ik} \times X_{jl} \times EI(W_{ik}, W_{jl})}{\sqrt{\sum_{k=1}^{p}\sum_{l=1}^{p} X_{ik} \times X_{il} \times EI(W_{ik}, W_{il})} \times \sqrt{\sum_{k=1}^{p}\sum_{l=1}^{p} X_{jk} \times X_{jl} \times EI(W_{jk}, W_{jl})}} \qquad (6)$$

The next step of the application aims at placing subtopic boundaries between dissimilar blocks. For that purpose, we propose a detection methodology based on the standard deviation algorithm proposed by (Hearst, 1994).

4. Topic Boundary Detection

Different methodologies have been proposed to place subtopic boundaries between dissimilar blocks depending on the models used to determine similarity between blocks of sentences (Kozima, 1993; Hearst, 1994; Beeferman *et al.*, 1997; Ponte and Croft, 1997; Stokes, *et al.*, 2002). Taking as reference the idea of (Ponte and Croft, 1997) who take into account the preceding and the following contexts of a segment, we calculate the informative similarity of each sentence in the corpus with its surrounding pieces of texts i.e. its previous block of k sentences and its next block of k sentences. The basic idea is to know whether the focus sentence is more similar to the preceding block of sentences or to the following block of sentences. For that purpose, we propose a score for each sentence as (Beeferman *et al.*, 1997) compare short and long-range models. It is defined in equation 7.

$$ps(S_i) = \log_2 \frac{\text{infosimba}(S_i, X_{i-1})}{\text{infosimba}(S_i, X_{i+1})} \qquad (7)$$

In order to better understand the variation of the *ps* score, each time its value goes from positive to negative between two consecutive sentences, there exits a topic shift. We will call this phenomenon a downhill. In fact, it means that the previous sentence is more similar to the preceding block of sentences and the following sentence is more similar to the following block of sentences thus representing a shift in topic in the text. A downhill is simply defined in equation 8 whenever the value of the *ps* score goes from positive to negative between two consecutive sentences S_i and S_{i+1}.

$$downhill(S_i, S_{i+1}) = ps(S_i) - ps(S_{i+1}) \qquad (8)$$

However, not all downhills identify the presence of a new topic in the text. Indeed, only deeper ones must be taken into account. In order to automatically

identify these downhills, and as a consequence the topic shifts, we adapt the algorithm proposed by (Hearst, 1994) to our specific case. Downhills are topic boundaries if they satisfy the constraint expressed in equation 9 where c is a constant to be tuned and $\bar{x}$ is the average of all downhills and σ the standard deviation.

$$downhill(Si, Si+1) \geq \bar{x} + c\sigma \quad (9)$$

By applying this threshold, we obtain promising results for the discovery of topic boundaries for the specific case of web news segmentation. We illustrate these results in the next section.

5. Results

Topic Segmentation systems (Ferret, 2002; Xiang and Hongyuan, 2003) have usually been evaluated on (Choi, 2000)'s data set that represents the standard for evaluation. However, many authors have discussed the validity of this test corpus (Ferret, 2002; Xiang and Hongyuan, 2003) and proposed their own test corpus. Indeed, (Choi, 2000)'s data set, also called c99, evidences two major drawbacks: (1) it deals with segments of different domains and (2) lexical repetition is high within each segment. It is clear that the c99 corpus does not apply for an evaluation oriented towards Text Summarization. Indeed, in this case, the texts must cover a single domain and intra-segment lexical repetitions are not used as much as in the c99 corpus. However, it is likely that there exist inter-segment lexical repetitions which unease the process of boundary detection. By tackling this particular situation, we propose a new challenge compared to other works that have been proposed so far and use test corpora based on multi-domain and multi-genre segments as in (Ferret, 2002). In fact, the most similar experiment, to our knowledge, is the one proposed by (Xiang and Hongyuan, 2003) who use the *Mars* novel. However, their segments are 2650 words-long while we deal with segments around 100 words each. In fact, we aim at proposing a fine-grained system capable of finding topic boundaries with high precision in a single domain and in short texts. To our knowledge, such a challenge has never been attempted so far.

In order to evaluate our system, we propose an evaluation on a set of web documents about a unique domain using words as the basic textual information. In order to run our experiments, we built our own corpus by taking from two Portuguese soccer websites, a set of 100 articles of more or less 100 words each. Then, we built 10 test corpora by choosing randomly 10 articles from our database of 100 articles leading to 10 texts of around 1000 words-long[1].

A classical way of evaluating retrieval systems is to use Precision, Recall and F-measure. So, we show these results on our test corpus in Table 1.

	Measures	c=-1.5
T1	Precision	0,64
	Recall	0,78
	F-measure	0,70
T2	Precision	0,67
	Recall	0,67
	F-measure	0,67
T3	Precision	0,80
	Recall	0,89
	F-measure	0,84
T4	Precision	0,73
	Recall	0,89
	F-measure	0,80
T5	Precision	0,60
	Recall	0,67
	F-measure	0,63
T6	Precision	0,73
	Recall	0,89
	F-measure	0,80
T7	Precision	0,80
	Recall	0,89
	F-measure	0,84
T8	Precision	0,64
	Recall	0,78
	F-measure	0,70
T9	Precision	0,60
	Recall	0,67
	F-measure	0,63
T10	Precision	0,70
	Recall	0,78
	F-measure	0,74
Average	Precision	0,69
	Recall	0,79
	F-measure	0,73

Table 1. Quantitative Results

The results are surprisingly good considering the challenging task we were facing. Indeed, by using words as basic textual units, the average F-measure reaches 73% being Recall 79% and Precision 69%. After different tuning, the best results were obtained for c=-1.5. In any case, these global results hide most of the behavior of our system and a more detailed evaluation is needed. As (Reynar, 1994) evidences, Precision and Recall measures are overly strict. By taking into account only Precision and Recall, a hypothesized boundary close to a real segment boundary is equally detrimental to performance as one far from a boundary. This definitely should not be the case. As a consequence, we present, in Table 2, quantitative results by taking into account, as correct boundaries, all correct boundaries and all near misses with ± 1 sentence.

Precision	0,83
Recall	0,95
F-measure	0,89

Table 2. Estimated Results

We can see from these results that we would obtain 89% F-measure, which means that our system fails most correct topic for only one sentence.

[1] The chosen parameters of our experiments were the following: block size=2 sentences and EI window=10 words.

The results presented in this section are promising as we deal with a very difficult challenge which is working without any linguistic knowledge, on the basis of small mono-domain texts with many inter-segments lexical repetitions. As we said earlier, to our knowledge, such a challenge has never been attempted so far.

6. Conclusions and Future Work

In this paper, we proposed a language-independent unsupervised Topic Segmentation system based on word-co-occurrences that avoids the accessibility to existing linguistic resources such as electronic dictionaries or lexico-semantic databases. In particular, our architecture proposes a system that solves three main problems evidenced by previous research: systems based uniquely on lexical repetition that show reliability problems, systems based on lexical cohesion using existing linguistic resources that are usually available only for dominating languages and as a consequence do not apply to less favored and emerging languages and finally systems that need previously existing harvesting training data. Our evaluation has evidenced promising results showing an average F-measure of 73% being Recall 79% and Precision 69%. As immediate future work, we intend to test our system by integrating Multiword Units. Indeed, on-going results seem to lead to more accurate figures. The system and its evolutions will be available for download as a GPL license at the following address: http://asas.di.ubi.pt.

References

(Angheluta et al., 2002) Angheluta, R., De Busser, R., Moens, M-F. 2002. The Use of Topic Segmentation for Automatic Summarization. In Workshop on Text Summarization in Conjunction with the ACL 2002 and including the DARPA/NIST sponsored DUC 2002 Meeting on Text Summarization. July 11-12, Philadelphia, Pennsylvania, USA.

(Beeferman et al., 1997) Beeferman, D., Berger, A., and Lafferty, J. 1997. Text segmentation using exponential models. In Proceedings of the Second Conference on Empirical Methods in Natural Language Processing, 35--46.

(Boguraev and Neff, 2000) Boguraev, B. and Neff, M. 2000. Discourse segmentation in aid of document summarization. In Proceedings of Hawaii International Conference on System Sciences (HICSS- 33), Minitrack on Digital Documents Understanding, Maui, Hawaii. IEEE.

(Choi, 2000) Choi, F.Y.Y. 2000. Advances in Domain Independent Linear Text Segmentation. In Proceedings of NAACL'00, Seattle, April 2000. ACL.

(Cleuziou et al., 2003) Cleuziou G., Clavier V., Martin L. 2003. Une méthode de regroupement de mots fondée sur la recherche de cliques dans un graphe de cooccurrences. In Proceedings of the 5èmes rencontres Terminologie et Intelligence Artificielle), LIIA - ENSAIS ed., 179--182, Strasbourg, France.

(Farzindar and Lapalme, 2004) Farzindar, A. and Lapalme, G. 2004. Legal text summarization by exploration of the thematic structures and argumentative roles. In Text Summarization Branches Out Conference held in conjunction with ACL 2004, Barcelona, Spain, 27-38

(Ferret, 2002) Ferret, O. 2002.*Using Collocations for Topic Segmentation and Link Detection*. In Proceedings of COLING 2002, 19th International Conference on Computational Linguistics, August 24 - September 1, 2002, Taipei, Taiwan.

(Hearst, 1994) Hearst, M. 1994. Multi-Paragraph Segmentation of Expository Text, In Proceedings of the 32nd Meeting of the Association for Computational Linguistics, Las Cruces, New Mexico, June, 9--16.

(Kozima, 1993) Kozima, H. 1993. Text Segmentation Based on Similarity between Words. In Proceedings of the 31th Annual Meeting of the Association for Computational Linguistics (Student Session), Colombus, Ohio, USA, 286--288.

(Morris and Hirst, 1991) Morris, J. and Hirst, G. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text, Computational Linguistics 17(1): 21--43.

(Muller, 1997) Muller, C., Polanco, X., Royauté, J. and Toussaint, Y. 1997. Acquisition et structuration des connaissances en corpus: éléments méthodologiques. Technical Report RR-3198, Inria, Institut National de Recherche en Informatique et en Automatique.

(Phillips, 1985) Phillips, M. 1985. Aspects of Text Structure: An Investigation of the Lexical Organisation of Text, North Holland Linguistic Series, North Holland, Amsterdam.

(Ponte and Croft, 1997) Ponte J.M. and Croft W.B. 1997. Text Segmentation by Topic. In Proceedings of the 1st European Conference on Research and Advanced Technology for Digitial Libraries.120--129.

(Reynar, 1994) Reynar, J.C. 1994. An Automatic Method of Finding Topic Boundaries. In Proceedings of the 32th Annual Meeting of the Association for Computational Linguistics, Las Cruces, USA.

(Salton et al., 1975) Salton, G., Yang, C.S., and Yu, C.T. 1975. A theory of term importance in automatic text analysis. Amer. Soc. Inf. Sc~ 26, 1, 33--44.

(Sardinha, 2002) Sardinha, T.B. 2002. Segmenting corpora of texts. DELTA, 2002, 18(2), 273--286. ISSN 0102-4450.

(Stokes et al., 2002) Stokes, N., Carthy, J. and Smeaton, A.F. 2002. Segmenting Broadcast News Streams Using Lexical Chains. In Proceedings of 1st Starting AI Researchers Symposium (STAIRS 2002), volume 1. 145--154.

(Xiang and Hongyuan, 2003) Xiang, J. and Hongyuan, Z. 2003. Domain-independent Text Segmentation Using Anisotropic Diffusion and Dynamic Programming. In proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Toronto, Canada. pp.322—329.

(Xu and Croft, 1996) Xu, J. and Croft, W.B. 1996. Query Expansion Using Local and Global Document Analysis. In Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 4--11.

An advanced approach for English-Vietnamese syntactic tree transfer

Do Xuan Quang*, Nguyen Luu Thuy Ngan, Dinh Dien*****
(*) Department of Computer Science, University of Illinois at Urbana Champaign, USA
(**) Department of Computer Sciences, Tokyo University, Japan
(*) Faculty of Information Technology, University of Natural Sciences, HCMC, Vietnam
quangdo2@uiuc.edu

Abstract

This paper presents an advanced approach in learning syntactic tree transfer rules (STTRs) used in English-Vietnamese machine translation. In this approach we use fast transformation-based learning to train and extract effective STTRs from our parallel corpora which is parsed and annotated with some useful features in language. Essentially, FastTBL will extract the same rule set with which is extracted by original TBL but reducing in training time much. Basing on the advanced base line process and training model, this approach also makes significant increasing in transfer accuracy. There are two stages in our learning model, the first one is learning STTRs for the same level, and the second one is learning STTRs for the different level of parsed tree. Due to the variety of the two languages, we also include some other auxiliary actions into learning process in order to improve the quality of transferring. These stages will be described in the sections of the paper.

1 Introduction

The problem of matching sentence's structures from the source language to the target one is always complicated and exciting as well. Especially for the language pair which are different in typologies, such as flexional vs isolated languages (e.g. English vs Vietnamese), or flexional vs agglutinate languages (e.g. English vs Japanese) (Chu Mai N. et al., 1991). In general, structural transfer is the stage in which grammatical constituents are arranged in source language including deletion and insertion some functional parts of the sentences. STTRs are the ordered-rules which can transfer source sentences into new structures, which are suitable with grammatical rules in target language. In 1993, Brown et al. proposed the statistical machine translation (SMT) employing IBM models (Brown et al., 1993), which combines two stages of lexical transfer and word-order transfer into one step of translation. This approach can generate high quality of translation. However, the advantage in translation time is paid the penalty for learning time; especially it becomes difficult to get the optimal solution in enormous search space (Imamura et al., 2004).

Another approach for structural translation is example-based MT as work of Nagao (1984). In his work, frameworks were used to capture examples that retrieved from bilingual corpus database. In some cases, this approach will not work properly due to the conflict of frameworks appropriating with the structures in sentences.

Our approach can be considered as an extended work of Dien et al. (2003). However, in this paper we want to emphasize the reducing in learning time and the accuracy due to some differences in baseline tagging and learning process.. There are three major improvements in our work. The first one is that we employed the learning method FastTBL (Ngai and Radu, 2001) for training STTRs, instead the original TBL (Brill, 1993). The second one is that we employed two stages of training to extract STTRs solving the problems of syntactic transfer in the same levels and different levels in parsed tree. The features of these stages are quite different in nature. The third improvement is in the stage of baseline transferring. In this stage, we analyze the input sentence and determine the grammatical relation tags which are very useful in baseline transferring.

2 Syntactic tree

The input of our syntactic tree transfer module is the English pared tree. In our experience, we employ the parser called nlparser from Brown University (Charniak, 2000) and the grammatical labels used are from Penn Tree Bank[1] (Marcus et

[1] Grammatical labels used are from PTB and can be found at http://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

al., 1993). Figure 1 presents an original English parsed tree with lexical labels in the two languages. According to the evaluation of Charniak (2000), this parser acquires the accuracy about 90.1% of parsing English sentences (for the sentences' lengths less than 40 words). However, we will learn and control the mistakes of parsing by forcing the mistaken syntactic trees to be transfered into the right ones in target language. This ability is acquire by the learning method TBL (FastTBL in our case)

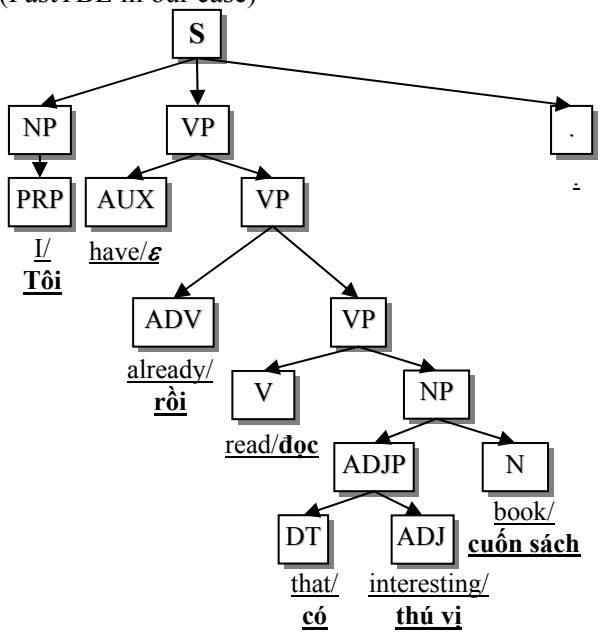


Figure 1: An English parsed tree with lexical labels in two languages. ε is null-translated.

3 FastTBL approach for syntactic tree transferring

The worst disadvantage of transformation-based learning is intolerably long in learning time, especially when it is used to train on large corpora. Fast TBL enable us to reduce training time significantly and get efficient rules due to the robust features of TBL. Thanks to the idea of Ngai and Radu (2001) in applying FastTBL for POS tagging and text chunking, we applied FastTBL as an advanced approach in extracting STTRs. However, we did not employ the fnTBL toolkit (Ngai and Radu, 2001) for our model because its corpus format and learning templates are not suitable for the feature of syntactic tree transfer from English to Vietnamese. The baseline transferring and learning engines will be described in the following sections.

3.1 Fast transformation-based learning

Figure 2 presents our approach using FastTBL learning model to acquire STTRs. The number 1 in the model represents for learning STTRs for the same level transferring (stage 1). The number 2 in the model represents for learning STTRs for the different one (stage 2). The reason why we have to employ two stages of transferring is that in the same level learning, we can not extract rules which can transfer one constituent from its phrase out to other phrases. It will be solved in the stage 2. Besides, two auxiliary tasks: deletion and insertion are also included in this stage.

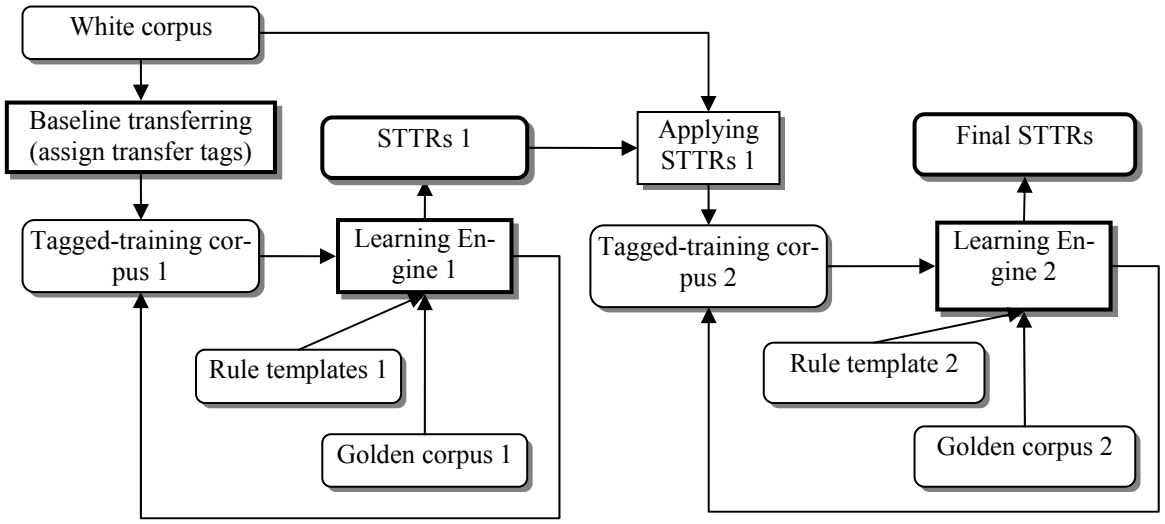


Figure 2: FastTBL learning model in our approach.

3.2 Baseline transferring for the model

We determine and classify all the grammatical relations available in the input sentence. It plays an important role in arranging parts. To do so, we use the principle-based parser called Minipar with the accuracy about 88% (Dekang Lin, 1994). Then we use the phrase information from English sentence to project to Vietnamese one based on the word alignment as the following steps:

- Mapping grammatical relations into Vietnamese sentence based on statistical maps.
- Constructing Vietnamese phrases from mapped relations from low to high levels.

The constructing is based on the principle of "Direct Corresponding Assumption" (Rebecca, 2002). This result is combined with parsed tree to have a new parsed tree for learning STTRs in the next stages.

Thanks to the algorithm of FastTBL, we modified it for our tasks.

Please refer the Fast TBL formalizing in the work of Ngai and Radu, 2001.

The alg. for learning same level transfer rules:

- Step 1: Generating the set of candidate rules $\Re$, computing score for each rule.
- Step 2: Choosing the best rule (highest score) b in $\Re$
- Step 3: Checking f(b) with training threshold α
  - If $f(b) < \alpha$, finish learning process.
  - Else, go to step 4.
- Step 4: For each sample s
  - If $C[s] \neq C[b(s)]$,
    - For each rule r in $\Re$
      - If $p_r(s)$ = true,
        - If $C[s] \neq T[s]$ and $t_r = T[s]$, decrease $good(r)$
        - Elseif $C[s] = T[s]$ and $t_r \neq T[s]$, decrease $bad(r)$
      - If $C[b(s)] \neq T[s]$,
        - For each rule template
          - Extracting rules for $b(s)$ and adding to $\Re$'
      - For each rule r
        - If $p_r(b(s))$ = true,
          - If $C[b(s)] \neq T[s]$ and $t_r = T[s]$, increase $good(r)$
          - Elseif $C[b(s)] = T[s]$ and $t_r \neq T[s]$, increase $bad(r)$
- Step 5: Computing score for rules in $\Re$', and moving these rules to $\Re$
- Step 6: Go to step 2

In the second stage of learning, we will extract STTRs to transfer a constituent from one part of the parsed tree to another part. The following algorithm has the ability to learn on one sample, a complete parsed tree, many times with one template for learning iterations.

- Step 1: Generating the set of candidate rules $\Re$, computing score for each rule.
- Step 2: Choosing the best rule (highest score) b in $\Re$ with the location τ
- Step 3: Checking $f(b)$ with training threshold α
  - If $f(b) < \alpha$, finish learning process.
  - Else, go to step 4.
- Step 4: For each parsed tree ψ
  - For each node t_ψ in ψ
    - If $C\lfloor t_\psi \rfloor \neq C\lfloor b(t_\psi) \rfloor$,
      - For each rule in $\Re$, with ω is the indicator
        - If $\omega = \tau$
          - For each node t_ψ in ψ
            - If $p_r(t_\psi)$ = true,
              - If $C[t_\psi] \neq T[t_\psi]$ and $t_r(t_\psi) = T[t_\psi]$, decrease $good(r)$
              - Elseif $C[t_\psi] = = T[t_\psi]$ and $t_r(t_\psi) \neq T[t_\psi]$, decrase $bad(r)$
    - For each node t_ψ in ψ
      - If $C[t_\psi] \neq T[t_\psi]$,
        - For each rule template
          - Extracting rule r_ψ for t_ψ
          - If $r_\psi \in \Re$, delete r_ψ in $\Re$
          - Adding r_ψ into $\Re$'
  - If $\Re' \neq \varnothing$, for each rule r in $\Re$'
    - Computing score for r
    - If r is needed to update and $f(r) > \alpha$
      - Moving r from $\Re$' to $\Re$
    - Else
      - Deleting r from $\Re$'
- Step 5: Go to step 2

4 Preparing training corpora and templates

4.1 For the same level (stage 1)

In this stage, the corpus consists of bilingual sentence pairs in English and Vietnamese is aligned at word level, mounted to parsed tree and disinte-

grated to discrete phrases. Each of these discrete phrases is a sample for training process.
For the rule templates, we choose the major features of linguistics to capture the factors making the structural differences in two languages. These features are word, POS, and the order of constituents in a phrase which we call index. We designed the templates from general to detailed ones.

4.2 For the different level (stage 2)

Because the features of the nature of learning in the first stage and second stage are quite different, we have to build golden corpus for stage 2 separately. For stage 2 the training corpus is the result from stage 1. The problem is that a parsed tree can have many samples for learning STTRs, therefore, we have to modify the learning algorithm as showed in above section to provide the algorithm the ability of learning many times on one sample with one template for every iteration of learning.
To determine the route from one place to another place in the parsed tree, we provided the learning engine the ability to establish the route from the source to the target place by determine the common parent for these places. After that, we assigned the relative identifiers for constituents in the route. Figure 3 present a part of parsed tree where the [ADVP → also/RB] had to move to the place as the first child node of [VP] (We used the parser of Charniak 2000 in this case. The one of Charniak 2005 is correct for this phrase and we do not need to move the constituent up.) We establish the route by assigning relative identifier for each constituent in different levels as showed by dash arrows.

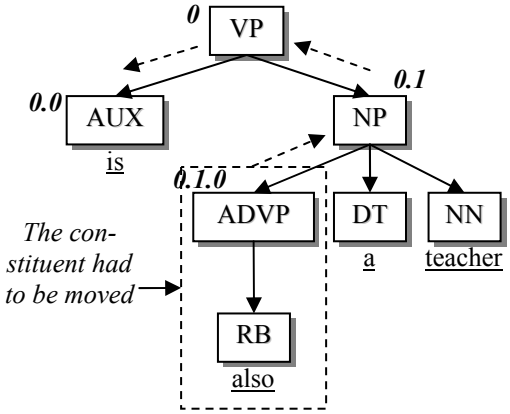


Figure 3: The route from ADVP to VP

In this stage, we also learn the deleting and inserting rules for Vietnamese sentence.

5 Experimences and evaluation

We would like to show our experimences in learning STTRs and the evaluation for our module as well. For convenience, we will compare the result of our work with the work of Dien et al. (2003) which employed TBL as the learning engine to extract transfer rules for English-Vietnamese language pair. We used the computer with the configuration: PIII-800, 256MB RAM

5.1 Experiment

The corpus used in our experiments is the CADASA corpus from the set of bilingual books "Come to the world of microcomputers" (CADASA Press). This corpus contains 8553 English-Vietnamese sentence pairs; the average length is 17 words per sentence. We use 8053 sentence pairs for training process and the rest 500 pairs are used for testing and evaluating.
For learning for same level transfer we have:

Number of sample	79.663
Number of template	12
Learning time	7h23m56s
Number of STTRs extracted	1427
Number of the first best rules	137

Table 1: Statistic for learning in stage 1

For learning for different levels transfer we have:

Number of sample	8053
The average length of sentences	17 words/sentence
Number of template	3
Learning time	9h27m6s
Number of STTRs extracted	817
Number of the first best rules	112

Table 2: Statistic for learning in stage 2

According to the accuracy for each stage, we keep the first 137 rules for the first stage and 112 rules for the second one. Total time for two stages of learning is 16h51m2s and total number of rule is 249. We can conclude that the rules extracted in our work are more efficient and take less time to transfer from English structure into Vietnamese one.

5.2 Evaluation

Actually, when translating an English sentence, we often assign corresponding Vietnamese meaning for English words in the sentence then we arrange constituents in the sentence into their orders in Vietnamese structure. We determine the minimal number of constituent needed to transfer to correct

order. According to this point of view, we used the following formula to compute the accuracy of transferring:

$$D_S = \frac{W - A}{W} \times 100\%$$

D_s: the accuracy of transferring for sentence S (%)
W: total number of word in sentence S
A: the minimal number of constituents in sentence S needed to transfer to have the correct sentence.

To determine the value of A, we apply the dynamic programming method.
This measurement is equivalent with the "key stroke" measurement (Chen et al., 1996).
Each "key stroke" consists of two actions: deleting and inserting a constituent. It is the cost to transfer a constituent to its correct position in the source sentence.

	Baseline - fixed rules	TBL-rules	Baseline - Grammatical relations	FastTBL-rules, only stage 1	FastTBL-rules, two stages
D_S	87,71%	93,88%	**88,36%**	93,72%	**94,05%**
Total cost	14960	11475	12665	12070	10880
Cost per sentence	29.92	22.95	25.33	24.14	21.76
Cost per word	1.76	1.35	1.49	1.42	**1.28**

Table 3: Evaluation of syntactic tree transfer modules

We employed our syntactic tree transfer module for the completed English-Vietnamese machine translation system. We used the same method of evaluation presented in work of Dien et al. (2003) to compute the cost which is use to edit a transferred sentence into the correct one. We tested on 500 sentences from CADASA. Each sentence has the average length 17 word/sentence. Table 3 compares the results from differences systems. The dark columns are ours.

6 Conclusion

This paper presents an advanced approach in learning STTRs using modified FastTBL algorithm. We evaluated the result objectively and also embedded our module into the English-Vietnamese MT system to test its performances. The result is quite good in both test and real conditions. Now, we are improving the learning process. When new data is added into the training corpus, the algorithm does not need to learn from beginning again. It will recognize the new data and extract rules only on it. Certainly, the rules extracted have to be computed the effects on whole corpus.

References

Brown, P., S. Della Pietra, V. Della Pietra, and R. Mercer. 1993. *The Mathematics of Statistical Machine Translation: Parameter Estimation*, Computational Linguistics, 19(2), pp. 263-311,

Chu Mai N., Nghieu Vu D., Phien Hoang T. 1991. *Cơ sở ngôn ngữ học và tiếng Việt*, publisher ĐH and GDCN, Hanoi,

D. Dien, N.L.T. Ngan, D.X. Quang and V.C.Nam. 2003. *A Hybrid Approach to Word Order Transfer in English to Vietnamese Machine Translation*, MT Summit IX, Louisiana, USA.

Dekang Lin. 1994. *PRINCIPAR---An Efficient, broad-coverage, principle-based parser*, COLING-94. pp. 42-488, Kyoto, Japan.

Eric Brill. 1993. *A corpus-based approach to language learning*, PhD-dissertation.

Eugene Charniak. 1996. *Tree-bank grammar*, AAAI/ IAAI, Vol. 2

Eugene Charniak, 2000. *A Maximum-Entropy-Inspired Parser*, NAACL 2000.

G. Ngai & R. Floriance. 2001. *Transformation-Based Learning in the Fast Lane*", NAACL 2001.

K. Imamura, H. Okuma, T. Watanabe, E. Sumita. 2004. *Example-based Machine Translation Based on Syntactic Transfer with Statistical Models*, COLING 2004

Keh-Yih Su, Ming-Wen Wu, Jing-Shin Chang. 1992. *A New Quantitative Quality Measure for Machine Translation System*, COLING-92, Nantes, France, pp. 433-439.

Kuang-Hua Chen & Hsin-His Chen. 1996. *A Hybrid Approach to Machine Translation System Design*, Computational Linguistics and Chinese Languge Processing. Vol.1.

M. Marcus, B. Santorini, M. Marcinkiewicz. 1993. *Building a large annotated corpus of English: the Penn Treebank*, Computational Linguistics 19, pp. 313-330

M. Nagao. 1984. *A framework of a mechanical translation between Japanese and English by analogy principle*, In. Art. and Human Intelligence, pp. 173–180.

Rebecca Hwa, Philip Resnik, Amy Weinberg, and Okan Kolak. 2002. *Evaluating Translational Correspondence using Annotation Projection*, ACL2002.

One Step Toward A Richer Model Of Unsupervised Grammar Induction

Heshaam Feili and **Gholam-Reza Ghassem-Sani***
Department of Computer Engineering
Sharif University of Technology
hfaili@mehr.sharif.edu, sani@sharif.edu

Abstract

Probabilistic Context-Free Grammars (PCFGs) are useful tools for syntactic analysis of natural languages. Availability of large Treebank has encouraged many researchers to use PCFG in language modeling. Automatic learning of PCFGs is divided into three different categories, based on the needed data set for the training phase: *supervised*, *semi-supervised* and *unsupervised*. Most current inductive methods are supervised, which need a bracketed data set in the training phase. However, lack of this kind of data set in many languages, has encouraged us to pay more attention to unsupervised approaches. So far, unsupervised approaches have achieved little success. By considering a history-based notion, we propose an extension of the inside-outside algorithm introduced by Lari and Young. Our experiments show that inducing more conditioned grammars improves the quality of the output grammar.

1 Introduction

Availability of large online corpora and fast computers has increased the efforts and interests in trying to automatically extract linguistic knowledge from text corpora (Brill, 1993). The success of part-of-speech tagging by using the Hidden Markov Model (HMM) (Church, 1988; Charniak, 1997b) also attracts the attention of computational linguistics to the lexical analysis, language modeling, and machine translation by various statistical methods (Feili & Sani 2004; Charniak, 1996).

Designing and refining a natural language grammar manually is a difficult and time-consuming task, which requires a large amount of skilled effort. A handcrafted grammar is usually not completely satisfactory and frequently fails to cover many unseen sentences. Automatic acquisition of grammars is a solution to this problem. With the increasing availability of large, machine-readable, parsed corpora such as Penn Treebank (Mitchell & Marcus, 1993), there have been numerous attempts to automatically derive a CFG grammar by using such corpora (Lari & Young, 1990; Pereira & Schabes, 1992).

There are different induction methods that can be classified into three categories based on the type of data that are needed (Thanaruk & Omkumaru, 1995). In the first category, which is called *supervised,* a full-parsed and tagged corpora such as Penn Treebank (Mitchell & Marcus, 1993) is used. The most successful methods of this category were presented by (Charniak, 1997a; Charniak, 1997b; Magerman, 1995; Collins, 1996), all of which used Penn Treebank. In the second category, which is called *semi-supervised*, only bracketed corpora without any label is used. Some semi-supervised methods were introduced in (Pereira & Schabes, 1992; Schabes et al, 1993). They improved the accuracy to the level of 90% in bracketing accuracy for sentences with an average length of 10 words.

In the third category, which is called *unsupervised grammar induction*; only tagged sentences without any bracketing information or other supervised information are used. Based on the Expectation Maximization (EM) algorithm, Lari and Young, proposed what they called the *Inside-outside* (IO) algorithm, that constructs a grammar from an unbracketed corpus (Lari & Young, 1990). This algorithm wills coverage towards a local optimum when used to iteratively re-estimate probabilities on a training corpus in a manner, which maximizes the likelihood of the training corpus, given the grammar (Holmes 1988). This method is so far one of the basic algorithms for unsupervised automatic learning of grammars (Briscoe & Waegner, 1992; Baker, 1979; Casacuberta, 1996). The IO algorithm requires a great deal of computation and produces a grammar in the *Chomsky normal form* and has been used in various unsupervised approaches of grammar induction (Amaya et al., 1999; Baker, 1979; Lari & Young, 1990; Pereira & Schabes, 1992). Also, Stolcke and Omohundro induced a small and artificial context free grammar with chunk-merge systems (Stolcke & Omohundro, 1994). The results of these approaches for completely unsupervised acquisition showed that they are generally ineffective. Later, Charniak describes some experiments running the EM algorithm from random starting points (Charniak, 1993), which produced widely varying grammar and Chen presents a Bayesian grammar induction method, which is followed by a post-pass using the IO algorithm (Chen, 1995).

There are also other works to improve the quality of the unsupervised induction methods by considering some limitation or additional information. Magerman and Weir use a distituent grammar to eliminate undesirable rules (Magerman & Marcus, 1990). Carroll and Charniak, restrict the set of non-terminals that may appear on the right hand side of rules with a given left hand side (Carroll & Charniak, 1992). The latest and the most promising class of unsupervised induction algorithms is distribution based, which uses distributional evidence to identify constituent structure. The

* This research has been partially funded by the Iranian Telecommunication Research Center (ITRC).

idea here is that sequences of words or tags that are generated by the same non-terminal will appear in similar contexts (Clark, 2001a; Clark, 2001b; Klein & Manning, 2005). Alignment Based Learning (ABL) is a learning paradigm that can be regarded as a distribution based method. It is based on the principle of substitutability, whereby two constituents are of the same type, and then they could be substituted (van Zaanen, 2000; van Zaanen, 2002; van Zaanen & Adriaans, 2001). Also, Adriaans presents EMILE, which initially used some aspects of supervision, but in later work is modified to be completely unsupervised (Adriaans, 1999). Both the ABL and EMILE techniques look for *minimal pairs*; a specific form of distributional learning, where the contexts are the rest of the sentence. Klein and Manning (Klein & Manning 2001a; Klein and Manning 2001b, Klein & Manning 2002) present a distributional method for inducing the bracketed tree structure of the sentence, with a dependency model to induce the word-to-word dependency structure. They received the best result in unsupervised inference with combining these two models together. We will compare our work with these approaches too. Other dependency models with less noticeable results are demonstrated by (Carroll & Charniak, 1992; Yuret, 1998 ; Paskin 2002).

On the other hand, applying probabilistic model to natural languages has been investigated in several works where the independence of the input sentence and its context is assumed in parsing (Charniak, 1997; Magerman, 1995; Johnson, 1998). In fact, most works have used even stronger independence assumptions. For instance, the PCFG model assumes the independence of the probability of each constituent and its neighboring constituents (Charniak, 1997a; Charniak97b). On the other hand, there are some richer models of context that incorporate some additional information with the probability of each constituent and present a way of calculating the probability model more accurately (Black et al., 1992b; Jelinek et al., 1994; Johnson, 1998).

In this paper, we introduce a new approach to incorporate a history-based notion into the IO algorithm. The main idea is adding more conditions to the probability model to be performed in the inside-outside iterative algorithm for estimating the new probability model.

2 Previous Works on History-Based Models

There have been some promising works adopted the history based grammar induction methods. For instance, *Pearl* is a probabilistic parser that is more sensitive to the model of context (Magerman & Marcus, 1991; Magerman & Weir, 1992). Using supervised learning methods; *Pearl* acquired 88% of bracketing accuracy.

Another important work, which increases the dependencies on the context, is the *history-based parser* that was originally developed by the researchers at IBM (Black et al., 1992a; Black et al., 1992b; Jelinek et al., 1994). In these models, the parse-tree representation was enriched in a couple of ways: non-terminal labels were augmented by some extra information such as lexical items and head word. An improvement from 59.8% to 74.6% in parsing accuracy was reported by using this model (Black et al., 1992b).

The idea of adding the parent of each non-terminal as the conditioning information to the grammar rules was also mentioned in (Johnson, 1998). Replacing $P(\alpha \rightarrow \beta \mid \alpha)$ by $P(\alpha \rightarrow \beta \mid \alpha, \text{Parent}(\alpha))$, where $\text{Parent}(\alpha)$ is the non-terminal dominating α, leads to an improvement from 69.6% / 73.5% to 79.3% / 80.1% of the precision/recall metrics.

3 Proposed Approach

All history-based method mentioned in the previous section are supervised. In this section, we propose an extension of the inside-outside algorithm, which infers the history-based models from an unsupervised data set. The new algorithm induces a richer model as its output grammar.

The new model also differs from other unsupervised grammar induction methods with respect to the form of the output grammar (Johnson, 1998; Lari & Young, 1990; Pereira & Schabes, 1992). In previous models, the final output grammar is often a probabilistic context free grammar (in the general form, or in more specific forms such as the probabilistic Chomsky normal form). Here, a general rule is defined as a couple <R, C>, where *R* is a rule in the Chomsky normal form, and *C* is a *parent* non-terminal. The parent non-terminal is defined as an immediately dominating non-terminal of rule *R* in the derivation process. Unlike PCFG, in which the sum of rule probabilities with the same left non-terminal is equal to one, here the sum of all rule probabilities with the same left hand side and the same *parent* is equal to one.

We have used an extension of the probabilistic CYK (PCYK) (Jurafsky & Martin, 2000; Kasami, 1965; Younger, 1967) as the parsing algorithm. In the extended PCYK, the parent non-terminal dominating CNF rules are also considered in parsing.

We evaluated our estimation model on English sentences. The evaluation results show that our approach outperforms previous unsupervised methods. In the following sections, at first the traditional inside-outside algorithm is briefly described then the new estimation model named *modified inside-outside* (MIO) is presented, and our experimental results are discussed. The final section includes our concluding points.

3.1 Inside-Outside Algorithm

The basic idea of the inside-outside algorithm is to use the current rule probabilities and the training set W to estimate the expected frequencies of certain types of the derivation step, and then compute new rule probability estimates as appropriate ratios of those expected frequency estimates. Since these are most conveniently expressed as relative frequencies, they are a bit loosely referred to as inside and outside probabilities. More precisely, for each $O \in W$, the inside probability e(s, t, i) estimates the likelihood that non-

terminal *i* derives O(s)…O(t), and the outside probability f(s, t, i) estimates the likelihood of deriving sentential form O(1)…O(s-1) *i* O(t+1)…O(T) from the start symbol S (Lari & Young, 1990). By defining inner and outer probabilities in an iterative manner, the grammar parameters are estimated and gradually converged to some plausible values.

3.2 Modified Inside-Outside Algorithm (MIO)

In this section, a new estimation model is introduced, which is referred to as a **M**odified version of the basic **I**nside-**O**utside algorithm (**MIO**). In MIO, the condition on parent non-terminal is assumed to be held and the values of the extended model parameters are converged in an iterative algorithm.

By considering history based approach and adding parent non-terminal to predefined Chomsky normal form rules, the probability of using rule $i \rightarrow jk$ can be stated as follows[1]:

$$A[C,i,j,k] = P(\, i \rightarrow jk \mid i \text{ used in derivation}, C = Parent(i), C \text{ used in derivation}) \quad (1)$$

In a similar way, the probability of using rule $i \rightarrow m$ is:

$$B[C,i,m] = P(\, i \rightarrow m \mid i \text{ used in derivation}, C = Parent(i), C \text{ used in derivation}) \quad (2)$$

These matrices have the following constraint:

$$\forall i, C: \ \sum_{j,k} A[C,i,j,k] + \sum_{m} B[C,i,m] = 1 \quad (3)$$

As in IO, we define *history-based inner* and *history based outer* probabilities to store partial probabilities of the observed data. *History based inner* probability denoted by *he(s,t, i,C)* is equal to :

$$he(s,t,i,C) = P(\, i \Rightarrow^{*} O(s),\ldots,O(t) \mid G, C=parent(i)), \quad (4)$$

where *O* is the current observation, and *O(i)* is the *i-th* element (word) of the observation. The formula *he(s,t,i,C)* is determined in a recursive manner. In the case that (*s=t*), and *O* is the current observation, we have:

$$he(s,s,i,C) = B[C, i, O(s)] \quad (5)$$

where *O(s)* is the *s-th* element of observation *O* (i.e. the *s-th* word of the input sentence *O*). In the case ($s \neq t$), similar to traditional IO algorithm, it can be calculated by using the following equation:

$$he(s,t,i,C) = \sum_{j,k}[\sum_{r=s}^{t-1} A[C,i,j,k].he[s,r,j,i].he(r+1,t,k,i)] \ \forall i \quad (6)$$

Similar to the outer probability defined in (Lari & Young, 1990), history based outer probability is defined as follows:

$$hf(s,t,i,C) = P(S \Rightarrow^{*} O(1)\ldots O(s-1), i, O(t+1)\ldots O(T) \mid C=Parent(i)) \quad (7)$$

Like inner *probability, this quantity can be computed by the following recursive formula:*

$$hf(s,t,i,C) = \sum_{j,k}[\sum_{r=1}^{s-1} hf(r,t,C,j).A[j,C,k,i].he(r,s-1,k,C) + \sum_{r=t+1}^{T} hf(s,r,C,j).A[j,C,i,k].e(t+1,r,k,C)] \quad (8)$$

where

$$hf(1,T,i,NULL) = \begin{Bmatrix} 1, & if\ i = S \\ 0, & otherwise \end{Bmatrix} \quad (9)$$

Figure 1, shows the history based outer probability definition, and Figure 2 shows two different cases during the process of computation of this probability.

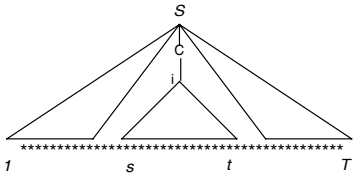


Figure 1: History based outer probability

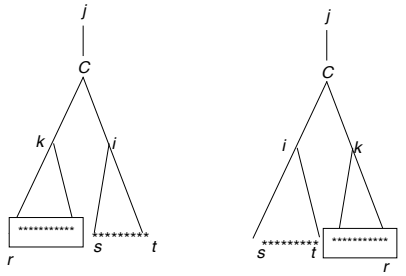


Figure 2: Different cases during calculation of the history based outer probability

Defining history-based *inner* and *outer* probabilities, leads to the following equation:

$$P(S \Rightarrow^{*} O \mid G) = \sum_{i,C} he(s,t,i,C).\ hf(s,t,i,C), \text{ for any } s \leq t. \quad (10)$$

Setting s=1, t=T, makes

$$P(S \Rightarrow^{*} O \mid G) = \sum_{i,C} he(1,T,I,C).\ hf(1,T,i,C) = he(1, T, S, NULL) \quad (11)$$

The last quantity is denoted as *P*. The product of inner and outer probabilities, implies a new result:

$$he(s,t,i,C)\ .\ hf(s,t,i,C) = P(\, S \Rightarrow^{*} O, i \Rightarrow O(s)\ldots O(t) \mid G, C=Parent(i)) = P(S \Rightarrow^{*} O \mid G)\ .\ P(i \Rightarrow^{*} O(s)\ldots O(t) \mid S \Rightarrow^{*} O(s)\ldots O(t) \mid S \Rightarrow^{*} O, G) \quad (12)$$

Here, we assume the independence between *C=Parent(i)* and $S \Rightarrow^{*} O$. Therefore:

$$P(i \Rightarrow^{*} O(s)\ldots O(t) \mid S \Rightarrow^{*} O\ , C=Parent(i), G) = he(s,t,i,C)\ hf(s,t,i,C) / P \quad (13)$$

Thus:

$$P(i \text{ used in derivation} \mid C = Parent(i)) = \sum_{s=1}^{T}\sum_{t=1}^{T} \frac{1}{P} he(s,t,i,C).hf(s,t,i,C) \quad (14)$$

[1] For the sake of simplicity, we ignore the assumption of using Grammar G in all probabilities.

Applying rule "$i \rightarrow j\ k$" in the derivation of observation O, we obtain:

$$P(i \overset{*}{\Rightarrow} jk \overset{*}{\Rightarrow} O(s)...O(t) \mid S \overset{*}{\Rightarrow} O, C = Parent(i), G) = \frac{1}{P}\sum_{r=s}^{t-1} A[C,i,j,k].he(s,r,j,i).e(r+1,t,k,i).f(s,t,i,C) \quad \forall j,k,t > s \qquad (15)$$

and by using equations (14) and (15), we obtain:

$$P(i \rightarrow jk, i\ used\ in\ derivation \mid C = parent(i)) = \sum_{s=1}^{T-1}\sum_{t=s+1}^{T}\frac{1}{P}\sum_{r=s}^{t-1} A[C,i,j,k].he(s,r,j,i).e(r+1,t,k,i).f(s,t,i,C) \qquad (16)$$

but from equation (1), we have:

$$A[C,i,j,k] = P(i \rightarrow jk \mid i\ used\ in\ derivation, C = parent(i)) = \frac{P(i \rightarrow jk, i\ used \mid C = Parent(i))}{P(i\ used \mid C = Parent(i))} \qquad (17)$$

Thus, the required probability can be determined by dividing (16) to (14). In a similar manner, matrix *B* can be computed as follows:

$$B[C,i,m] = \frac{\frac{1}{P}\sum_{t \in Q(t)=m} he(t,t,i,C).hf(t,t,i,C)}{\frac{1}{P}\sum_{s=1}^{T}\sum_{t=s}^{T} he(s,t,i,C).hf(s,t,i,C)} \qquad (18)$$

Equations (17) and (18) can be computed for any observation *O*. In practice, there are a finite number of observations that should be considered in a sequence in above equations. These parameters are evaluated in an iterative manner until the changes in the observed probabilities are less than a pre-defined threshold. Therefore in summary, MIO runs as the following loop:

REPEAT
A = … {Equation 17}
B = … {Equation 18}
P = … {Equation 11}
UNTIL changes in P are less than a pre-defined threshold.

In the next section, a new parsing algorithm for using in the evaluation of the grammar produced by MIO is described.

3.3 Extended PCYK Algorithm

In order to evaluate the quality of the grammar induced by MIO, we need to employ parent non-terminals in parsing. For this goal, the PCYK algorithm has been extended to focus on the parent non-terminal during filling the Table of CYK parsing.

Probabilistic CYK itself, is an extended version of the traditional CYK (Kasami, 1965; Grune & Jacobs, 1990), and has been tailored for calculating probabilities of different generated parses. Traditional CYK receives a CNF grammar and by using a dynamic programming notion, all possible structures of any given sentence are determined. Like any other dynamic programming algorithm, a Table is used to store partial results of parsing.

Similar to PCYK, in the extended PCYK, a 4-dimentional Table W of size N*N*M*M is used, where N is the number of non-terminals, and M is the number of words of the input sentence. Any entry W_{ijAC} of this Table is defined as:

$$W_{ijAC} = P(A \overset{*}{\Rightarrow} O(i)...O(j) \mid C = Parent(A)), \qquad (19)$$

where *O(i)* refers to the *i-th* word of the input sentence *O*.

The entries of this matrix can be calculated in a recursive manner. If rule $A \rightarrow X\ Y$ is used in the first step of deriving *"*$A \overset{*}{\Rightarrow} O(i)...O(j)$*"*, we can infer:

$$W_{ijAC} = MAX_k\,(P(A \rightarrow XY \mid C = Parent(A)) \,.\, W_{ikXA} \,.\, W_{k+1\,jYA}) \qquad (20)$$

The final state of this recursion is reached by using the second type of CNF rules (i.e., *"*$A \rightarrow m$*"*), and the following equation:

$$W_{iiAC} = P(A \rightarrow O(i) \mid C = Parent(A)) \qquad (21)$$

4 Experimental Results

In our experiments, spoken-language transcription of the Texas Instruments subset of Air Travel Information System (ATIS) corpus was used (Hemphill et al.; 1990). This corpus, which is included in Penn Treebank II, has been automatically labeled, analyzed and manually checked (Mitchell & Marcus, 1993). There are two different labeling information in this Treebank: a part of speech tag and a syntactic labeling. We used 577 sentences of the corpus with 12232 words. The main characteristics of this corpus are summarized in Table 1.

No. of sent.	Max. len.	Min. len.	Ave. len.	No. of words
577	35	2	8	4645

Table 1: the main characteristics of used corpus

Similar to other works, the corpus was divided into two distinct sets: the *training* set with approximately 90% of the data and the *test* set (i.e., the remaining 10%). We used the ten fold cross validation method in order to validate our results: the corpus was divided into 10 parts (with equal size), and in each run, one section was used for testing and the rest for training. In the training phase, we added one dummy non-terminal *NULL* as the parent of the starting symbol *S*. Initial grammar for training phase was a full grammar that contained all possible CNF rules with random probabilities. We ran every experiment 100 times in order to decrease the possibly bias of the initial rules probabilities.

Starting from an initial grammar and using the IO algorithm, more accurate grammar is induced iteratively. By re-executing the same process, this time using the MIO model, another grammar (in the extended PCFG form) was obtained. The process of each experiment was repeated until the increase in the estimated probability of the training sentences became negligible, or the decrease in the cross-entropy estimation (negative log probability), became ig-

norable. Cross-entropy estimation was performed by using the following equation (Pereira & Schabes, 1992):

$$H(C,G) = -\frac{\sum_{c \in C} \log P(c)}{\sum_{c \in C} |c|}, \qquad (22)$$

where, the probability *P(c)* of the sentence *c* is the sum of probabilities of all derivations compatible with the bracketing of the parsed sentence. Figure 3, shows the cross entropy of the training data set with respect to the inferred grammar, after each iteration of training sentences for both IO and MIO.

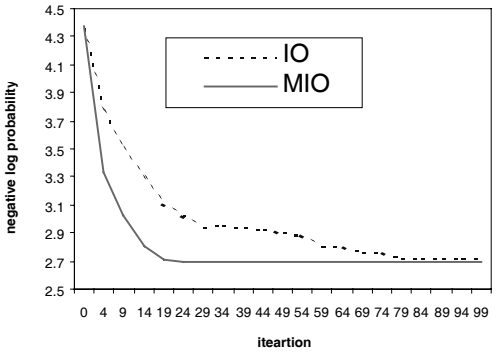


Figure 3: negative log probability of the observed data

As it is shown in Figure 3, MIO needs less iteration to reach to its optimal point. MIO needs only about 20 iterations to reach to its optimal state, while IO took over 80 iterations for reaching to the same state. But the time for each iteration in MIO is more than that of IO. This is due to the time complexity for the new estimation model. The time complexity of MIO at each iteration is $O(n^4|w|^3)$, while IO runs on $O(n^3|w|^3)$, where *n* is the number of non-terminal and $|w|$ is the size of the observed sentence. As mentioned in the previous sections, MIO uses Tables with one dimension more during the estimation process.

For evaluating the output grammars, we can use different metrics. The most popular metrics that are used for this purpose were introduced in (Black et al., 1991). The measure is called PARSEVAL to compare grammar's performance. Table 2 summarizes the evaluation results of these experiments compared with some of important related works: EMILE (Adriaans et al., 2000), ABL (van Zaanen, 2000), CDC with 40 iterations (Clark, 2001b) and CCM (Klein & Manning, 2005). LEFT and RIGHT are left- and right-branching baselines, which applied on ATIS data set. The results of the left and right baselines are borrowed from (Klein & Manning 2005).

As it is shown in Table 2, MIO shows a noticeable improvement over the baselines and all other works except CCM, which is a distributional induction approach.

CCM and other distributional methods use the idea that similar words normally occur in similar contexts. In other words, there are general patterns in the context where the words appear. However, these approaches don't work for the languages that do not obey such patterns (e.g., free-word-order languages). On the other hand, EM-based approaches (like MIO), which do not assume the existence of any particular pattern, are expected to have a better performance than that of distributional methods in handling such languages.

Induction method	UP	UR	F1
EMILE	51.588	16.814	25.351
ABL	43.640	35.564	39.189
CDC-40	53.4	34.6	42.0
CCM	55.4	47.6	51.2
LEFT	19.89	16.74	18.18
RIGHT	39.9	46.4	42.9
IO	42.19	35.51	38.56
MIO	49.75	46.43	48.03

Table 2: Evaluation results for different approaches on ATIS data set

5 Discussion

One of the weaknesses of a PCFG is its insensitivity to the non-local relationships between constituents. Where these relationships are crucial, a PCFG would be a weak modeling tool. Indeed, the sense in which the set of trees generated by a CFG is "context free" is precisely that the label on a node completely characterizes the relationships between the sub-tree dominated by the node and the set of nodes that properly dominate this sub-tree (Johnson, 1998). Therefore, the results of any experiment with a PCFG would be less accurate in terms of precision and recall metrics. Our new idea for relaxing the independence assumptions implicit in a PCFG model is systematically encoding more information in each node of the generated parse tree (i.e., enhancing the node information with the parent non-terminal label). The Johnson's experiments show similar results, where copying the label of the parent node onto labels of its children, dramatically improves the performance of a PCFG model (Johnson, 1998).

We used the main source code of IO algorithm, which has been published by Dr. Mark Johnson as the base for our method. This code is written in ANSI C, compiled by g++ in Linux operating system, run on a 2.4GHz PC with 256MB of RAM. During running the system, any rules with probability less than 10^{-9} is dropped and the training phase of the traditional IO ends when the reduction of negative log-probabilities of observations gets to less than 10^{-7}.

6 Conclusion

We described MIO, a new approach based on the well-known IO algorithm for inferring stochastic context-free grammars. We relaxed the independence assumptions often used with PCFG rules in the parsing process and extra information about the context was added to the context free

rules. Here, the parent non-terminal dominating PCFG rules are chosen as the context information.

Also a novel unsupervised estimation model based on the inside-outside algorithm was introduced and the experimental results on the ATIS data set were shown. The results show a considerable improvement over baseline methods and show a comparable result with the state of the art approaches.

We are going to test the new idea on languages other than English. However, since the new method is unsupervised and doesn't need rich corpora, we think that the idea can be easily applied to such languages. Also, working on defining semi-supervised methods like the one described in (Pereira & Schabes, 1992) is another possible future work. Using bracketed data set for inducing extended PCFG, can improve the speed and accuracy of the algorithm. Choosing more information beside the parent non-terminal as the context information could also be useful for enhancing the accuracy of parsing methods. However, estimating the required parameters of the extra information is the main obstacle here.

References

(Adriaans, 1999) Adriaans, P., Learning shallow context-free languages under simple distributions. Tech. rep. ILLC Report PP-1999-13, Institute for Logic, Language and Computation, Amsterdam, 1999.

(Adriaans et al., 2000) Adriaans, P., Trautwein, M., & Vervoort, M. Towards high speed grammar induction on large text corpora. In Hlavac, V., Jeffery, K. G., & Wiedermann, J. (Eds.), *SOFSEM 2000: Theory and Practice of Informatics*, pp. 173–186. Springer Verlag, 2000.

(Amaya et al., 1999) Amaya, F., Benedf, J.M. and Sanchez J.A. Learning Of Stochastic Context-Free Grammars From Bracketed Corpora By Means Of Reestimation Algorithms, in the VIII Symposium on Pattern Recognition and Image Analysis, Vol. 1, pp. 19-126, Bilbao, 1999.

(Black et al., 1991) Black, E., Abnery, S., Flickinger, D. and et al. A procedure for quantitatively comparing the syntactic coverage of English grammars, DARPA Speech and Natural Language Workshop, pp. 306-311, 1991.

(Black et al., 1992a) Black, E., Lafferty, J. and Roukos, S., Development and Evaluation of a Broad-Coverage Probabilistic Grammar of English-Language Computer Manuals, In the Proceedings of the 30th Annual Meeting of the Association for computational Linguistics, pp. 185-192, 1992.

(Black et al., 1992b) Black, E., Jelinek, F., Lafferty, J., Magerman, D., Mercer, R. and Roukos, S., Towards History-based Gramars: Using Richer Models for Probabilistic Parsing, In the Proceedings of the 5th DARPA Speech and Natural languages Workshop, Harriman, NY, 1992.

(Brill, 1993), Brill, E., A Corpus-Based Approach to Language Learning, PhD thesis, Department of the computer and Information Science, University of Pennsylvania, 1993.

(Briscoe & Waegner, 1992) Briscoe, T. and Waegner, N., "Robust Stochastic Parsing Using the InsideOutside Algorithm" in Workshop notes for the Statistically-Based NLP Techniques Workshop, pp. 39--53, 1992.

(Carroll & Charniak, 1992) Carroll, G. and Charniak, E., Two experiments on learning probabilistic dependency grammars from corpora, Technical reports CS-92-16, Department of Computer Science, Brown University, March, 1992.

(Clark, 2001a) Clark, A., Unsupervised induction of stochastic context-free grammars using distributional clustering. In The Fifth Conference on Natural Language Learning, 2001.

(Clark, 2001b.) Clark, A., Unsupervised Language Acquisition: Theory and Practice. PhD thesis, University of Sussex, 2001.

(Charniak, 1993) Charniak, E., Statistical Language Learning, MIT press, Cambridge, MA, 1993.

(Charniak, 1996) Charniak, E., Statistical Language Learning, Cambridge, London, UK: MIT Press, 1996.

(Charniak, 1997a) Charniak, E., Statistical parsing with a context-free grammar and word statistics. In the Proceedings of the 14th National Conference on Artificial Intelligence, pp. 598-603, Menlo Park. AAAI Press/MIT Press, 1997.

(Charniak, 1997b) Charniak, E., Statistical techniques for natural language parsing, AI Magazine, Vo. 18, No. 4, pp. 33-44, Winter 1997.

(Charniak, 2000) Charniak, E., A maximum-entropy-inspired parser, In NAACL 1, pp. 132-139, 2000.

(Chen, 1995), Chen, S. F., Bayesian grammar induction for language modeling. In Proceedings of the association for Computational Linguistics, pp. 228-235, 1995.

(Church, 1988) Church, K., A stochastic parts program and noun phrase parser for unrestricted text, In the Proceedings of the Second Conference on Applied Natural Language Processing, pp. 136-143, 1988.

(Collins, 1996) Collins, M., A new statistical parser based on bigram lexical dependencies, In the Proceedings of the 34th Annual Meeting of the ACL, Santa Cruz, 1996.

(Feili & Sani, 2004) Feili, H. and Ghassem-Sani, G., An Application of Lexicalized Grammars in English-Persian Translation, Proceedings of the 16th European Conference on Artificial Intelligence (ECAI 2004), Universidad Politecnica de Valencia, Spain, pp. 596-600, 2004.

(Hemphill et al., 1990) Hemphill, C.T., Godfrey, J., and Doddington, G., The ATIS spoken language systems pilot corpus, In DARPA Speech and Natural language Workshop, Hidden Valey, Pennsylvania, June 1990.

(Holmes, 1988) Homes, J. N., Speech Synthesis and Recognition, Van Nostrand Reinhold, 1988.

(Grune & Jacobs, 1990) Grune, D. and Jacobs C. J. H., Parsing techniques, a practical guide, Ellis Horwood, Chichester, England, 1990.

(Jelinek et al., 1994) Jelinek, F., Laferty, J. D., Magerman, D., Mercer, R., Ratnaparakhi, A. and Roukos, S., Decision-Tree Parsing using Hidden Derivation Model, In the Proceedings of the 1994 Human Language Technology Workshop, pp. 272-277, 1994.

(Johnson, 1998) Johnson, M., The Effect of Alternative Tree Representations on Tree Bank Grammars, In D.M.W. Powers (ed.) NeM-

LaP3/CoNLL98: New Methods in Language Processing and Computational Natural Language Learning, ACL, pp. 39-48, 1998.

(Jurafsky & Martin, 2000) Jurafsky, D., and Martin, J. H., Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition, Prentice-Hall Inc., New Jersey.

(Kasami, 1965) Kasami, T., An efficient recognition and syntax algorithm for context-free languages, Scientific Report AFCRL-65-758, Air Force Cambridge Research Laboratory, Bedford MA, 1965.

(Klein & Manning, 2001a) Klein, D., and Manning, C. D., Distributional phrase structure induction. In Proceedings of the Fifth Conference on Natural Language Learning (CoNLL 2001), 113–120, 2001.

(Klein & Manning, 2001b) Klein, D., and Manning, C. D., Natural language grammar induction using a constituent-context model. In T. G. Dietterich, S. Becker, and Z. Ghahramani (Eds.), Advances in Neural Information Processing Systems 14 (NIPS 2001), Vol. 1, 35–42, MIT Press, 2001.

(Klein & Manning, 2002) Klein, D., and Manning, C. D., A generative constituent-context model for improved grammar induction. In ACL 40, 128–135, 2002.

(Klein & Manning, 2005) Klein, D., and Manning, C. D., The Unsupervised Learning of Natural Language Structure, PhD Thesis, Department of Computer Science, Stanford University, 2005.

(Lari & Young, 1990) Lari, K. and Young, S.J., The estimation of stochastic context-free grammar using the inside-outside algorithm, Computer Speech and Language, Vol. 4, pp. 35-56, 1990.

(Magerman & Marcus, 1990) Magerman, D. M. and Marcus, M. P., Parsing a natural language using mutual information statistics, In Proceedings of the Eighth National conference on Artificial Intelligence, August, 1990.

(Magerman & Marcus, 1991) Magerman, D. and Marcus, M., Pearl: A Probabilistic Chart Parser, In the Proceedings of the 1991 European ACL conference, Berlin, Germany, 1991.

(Magerman & Weir, 1992) Magerman, D. and Weir, D. Efficiency, Robustness and Accuracy in Picky Chart Parsing, In the Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics, pp. 40-47, 1992.

(Magerman, 1995) Magerman, D. M., Statistical decision-tree models for parsing, In the Proceedings of ACL Conference, June1995.

(Marcken, 1995) Marcken, C., On the unsupervised induction of phrase-structure grammars, In the Proceedings of the 3rd Workshop on Very Large Corpora, 1995.

(Mitchell et al., 1993) Mitchell P., Marcus, B., Santorini, ce., and Marcinkiewicz, M. A., Building a large annotated corpus of English: the Penn Treebank, Computational Linguistics, Vol. 19, pp. 313-330, 1993.

(Paskin, 2002) Paskin, M. A., Grammatical bigrams. In T. G. Dietterich, S. Becker, and Z. Ghahramani (Eds.), Advances in Neural Information Processing Systems 14, Cambridge, MA. MIT Press, 2002.

(Pereira & Schabes, 1992) Pereira, F. and Schabes, Y., Inside-Outside reestimation from partially bracketed corpora, In the Proceeding of 30th annual Meeting of the ACL, pp. 128-135, 1992.

(Schabes et al., 1993) Y. Schabes, M. Roth and R. Obsorne, "Parsing the Wall Street Journal with the inside-outside Algorithm", In Proceedings of the Sixth Conference of the European Chapter of the ACL. Pages 341-347, 1993.

(Stolcke & Omohundro, 1994) Stolcke, A., and S. M. Omohundro, Inducing probabilistic grammars by Bayesian model merging, In Grammatical Inference and Applications: Proceedings of the Second International Colloquium on Grammatical Inference. Springer Verlag, 1994.

(Thanaruk & Omkumaru, 1995) Thanaruk, T. and Okumaru, M., Grammar Acquisition and Statistical Parsing, Journal of Natural language Processing, Vol 2, No. 3, July 1995.

(Van Zaanen, 2000) Van Zaanen, M., ABL: Alignment-Based Learning. In COLING 2000, pages 961-967, 2000.

(Van Zaanen & Adriaans, 2001) Van Zaanen, M. and Adriaans, P. W., Comparing Two unsupervised Grammar Induction Systems: Alignment-Based Learning vs. EMILE. Technical Report: TR2001.05, School of Computing, University of Leeds, 2001.

(Van Zaanen, 2002) Van Zaanen M., Bootstrapping stucture into Language: Alignment-based Learning. PhD Thesis, School of Computing University of Leeds, 2002.

(Younger, 1967) Younger, D., Recognition and parsing of context-free languages in time $O(n^3)$, Information and Control, Vol.10, pp. 189-208, 1967.

(Yuret, 1998) Yuret, D., Discovery of Linguistic Relations Using Lexical Attraction. PhD thesis, MIT, 1998.

Document Classification Using Semantic Networks with An Adaptive Similarity Measure

Filip Ginter, Sampo Pyysalo, and **Tapio Salakoski**
Turku Centre for Computer Science (TUCS) and
Department of IT, University of Turku
Lemminkäisenkatu 14 A
Turku 20520, Finland
firstname.lastname@it.utu.fi

Abstract

We consider supervised document classification where a semantic network is used to augment document features with their hypernyms. A novel document representation is introduced in which the contribution of the hypernyms to document similarity is determined by semantic network edge weights. We argue that the optimal edge weights are not a static property of the semantic network, but should rather be adapted to the given classification task. To determine the optimal weights, we introduce an efficient gradient descent method driven by the misclassifications of the k-nearest neighbor (kNN) classifier. The method iteratively adjusts the weights, increasing or decreasing the similarity of documents depending on their classes.

We thoroughly evaluate the method using ten randomly chosen datasets and seven training set sizes on the problem of classifying PubMed documents indexed with the MeSH biomedical ontology. Using the kNN classifier, the method is shown to statistically significantly outperform the commonly used bag-of-words representation as well as the more advanced hypernym density representation (Scott & Matwin 98).

1 Introduction

Semantic networks have been shown to offer opportunities for improving the performance of both supervised and unsupervised machine learning methods in a variety of classification tasks. Several semantic similarity measures have been proposed and applied in particular to word sense disambiguation-type problems, where the similarity between each ambiguous word candidate and the context words can be used to choose between the candidates (see e.g. (Budanitsky & Hirst 01; Patwardhan *et al.* 03) for recent evaluations). Methods applying semantic networks to document classification, where the class labels are not themselves part of the semantic network have also been proposed, although they are not as widely studied. For instance, semantic networks have been used to augment the terms occurring in documents with their synonyms (Gomez-Hidalgo & deBuenaga Rodriguez 97) and hypernyms (Scott & Matwin 98; Bloehdorn & Hotho 04), thus incorporating the information encoded in semantic networks on the level of features. Semantic networks have also been applied in modeling document similarity for a kernel-based document classification method (Basili *et al.* 05).

The similarity of terms is typically presented as a static property that can be directly measured either from the semantic network (Leacock & Chodorow 98; Agirre & Rigau 96), from external (unlabeled) data (Resnik 95), or using a combination of the two (Jiang & Conrath 97). In this paper, we consider the special case of similarity through the hyponymy/hypernymy relation, which is the focus of most proposed measures of semantic relatedness.

We have previously argued that in supervised classification tasks the similarity of terms should be considered dependent on the task and data (Ginter *et al.* 04). Simply put, terms commonly related to documents of the same class should be considered similar, while terms related to documents of different classes should be considered dissimilar to aid the classification method in distinguishing between the classes.

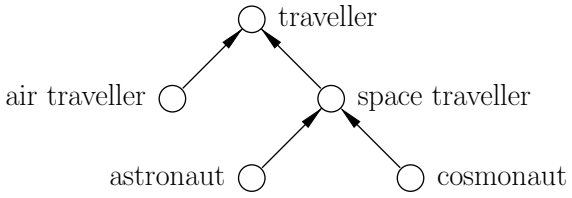
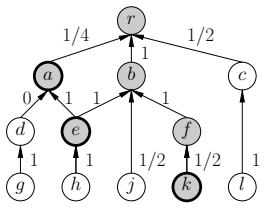


Figure 1: *Hyponymy.* The arrows represent hyponymy relationships between terms in a fragment of a semantic network.

To illustrate this idea, consider the fragment of a semantic network shown in Figure 1. Common measures of semantic similarity would assign high relatedness to the terms *astronaut* and *cosmonaut* as they are immediate hyponyms of the same term, *space traveller*. In most classification tasks, considering *astronaut* and *cosmonaut* essentially synonymous terms would be appropriate. In a document representation, this can be naturally realized by considering the term *space traveller* to be highly relevant to documents containing either of its two hyponyms. However, we suggest that in a hypothetical document classification task where the goal is to distinguish between documents about American and Russian space efforts, *space traveller* should not be considered relevant to documents containing either *astronaut* or *cosmonaut* to avoid increasing the similarity between documents of different classes.

We now discuss some desirable properties for a data-dependent semantic document representation and



$$Aff(d) = \{e,\, a,\, k,\, b,\, r,\, f\}$$
$$Base_r(d) = \{a,\, b\}\,,\ Base_a(d) = \{e\}\,,\ Base_b(d) = \{e,\, f\}\,,\ Base_f(d) = \{k\}$$
$$Base_g(d) = Base_h(d) = Base_j(d) = Base_k(d) = Base_l(d) = Base_d(d) = Base_e(d) = Base_c(d) = \emptyset$$
$$a_e(d) = a_a(d) = a_k(d) = 1,\ a_d(d) = a_g(d) = a_h(d) = a_j(d) = a_l(d) = a_c(d) = 0$$
$$a_f(d) = w_{kf}a_k(d) = \frac{1}{2},\ a_b(d) = \frac{w_{eb}a_e(d) + w_{fb}a_f(d)}{2} = \frac{3}{4},\ a_r(d) = \frac{w_{ar}a_a(d) + w_{br}a_b(d)}{2} = \frac{1}{2}$$

Figure 2: *Document representation.* An example illustrating the representation of a document d with direct terms $T(d) = \{e,\, a,\, k\}$. Direct terms of d are denoted by bold circles and terms affected by d are depicted in gray. The semantic network weights are shown by each edge.

means of realizing them. We assume that each document has been assigned a set of *direct* terms from a semantic network (e.g. terms that are mentioned in the document, or relevant keywords that have been assigned to the document). The representation should then determine the relevance of each semantic network term for each document. It is natural to limit this measure of relevance between 0 and 1, and to assign the value 1 to each direct term. As illustrated by the *astronaut/cosmonaut* example, hypernyms of direct document terms are typically relevant and their relevance values should be allowed to vary in a data-dependent fashion. We suggest that terms that are neither direct terms nor hypernyms of direct terms in a document are not relevant to that document and can be assigned the relevance value 0: for example, if *astronaut* is the only direct term, there is no reason to assume that either *cosmonaut* or *air traveller* are relevant. Finally, relevance should not increase with distance from the direct term: if, for example, *astronaut* is the only direct term, *traveller* should be considered at most as relevant as *space traveller*. This implies a representation where relevance propagates from direct terms to more general terms, decreasing according to the data-dependent strengths of connections between hyponyms and hypernyms.

We have previously introduced a data-driven method for determining hypernym relevance for document classification (Ginter *et al.* 04), where relevance was limited to the two cases "fully relevant" and "irrelevant", achieving a modest yet statistically significant 0.9 percentage unit average performance increase from a 81.7% bag-of-words baseline (average precision measure). In this paper, we present a method that applies a finer-grained concept of relevance and shows a more substantial performance advantage.

2 Document representation

Let $\mathcal{T}$ be a finite set of possible terms that are organized in a semantic network according to the semantic relation of hyponymy. Let $t,\, t' \in \mathcal{T}$ be terms. We denote by $t' \prec^\star t$ the relation when t' is a hyponym of t. Further, $t' \prec t$ denotes the relation when t' is an *immediate hyponym* of t, that is, the relation encoded by the semantic network. Hyponymy ($\prec^\star$) is the transitive closure of immediate hyponymy ($\prec$). For example, we have *astronaut* $\prec$ *space traveller*, *astronaut* $\prec^\star$ *traveller*, but *astronaut* $\not\prec$ *traveller*. The immediate hyponymy relation between the terms in $\mathcal{T}$ is commonly represented as a directed graph, such as the graph in Figure 1, with an edge from t' to t whenever $t' \prec t$. Hyponymy ($\prec^\star$) is by definition an asymmetric relation, and the corresponding directed graph is thus acyclic.

We define a document representation that implements the intuitions discussed in Section 1. Let $\mathcal{D}$ be a set of documents and let $d \in \mathcal{D}$ be a document with the set of direct terms $T(d) \subseteq \mathcal{T}$. As discussed previously, the document d is represented not only by the direct terms in $T(d)$ but also by their hypernyms. The proposed document representation implements this property through the notion of *activation* $a_t(d) \in [0, 1]$ of a term $t \in \mathcal{T}$ with respect to the document d that represents the relevance of t to d. For any term t that belongs to $T(d)$, $a_t(d)$ is by definition set to 1, the maximum possible activation value. The activation of any other term recursively depends on the activations of its immediate hyponyms so that the activation of hypernyms of direct terms typically results in a non-zero value. The activation of the remaining terms is zero by definition.

We say that a term $t \in \mathcal{T}$ is *affected* by a document d if $t \in T(d)$ or $\exists\, t' \in T(d) : t' \prec^\star t$. That is, t is affected by d if t is either a direct term of d or a hypernym of a direct term. The set of all terms affected by a document d is denoted $Aff(d)$. Let us further define the *base* of a term $t \in \mathcal{T}$ with respect to a document d as the set of immediate hyponyms of t that are affected by d. Formally,

$$Base_t(d) = \{t' \,|\, t' \prec t,\, t' \in Aff(d)\}\ .$$

Unless t is a direct term, its activation is based on the activations of the terms in $Base_t(d)$. For each term t' in the base of t, the contribution of t' to the activation of t is controlled by a *weight* $w_{t't}$ that is associated with the relationship $t' \prec t$. By definition, $0 \le w_{t't} \le 1$ for all weights in the semantic network. The activation $a_t(d)$ is computed as the weighted sum

of the activations of the terms in $Base_t(d)$. Thus,

$$a_t(d) = \begin{cases} 1 & \text{if } t \in T(d), \\ \frac{\sum_{t' \in Base_t(d)} w_{t't} a_{t'}(d)}{|Base_t(d)|} & \text{if } t \in Aff(d) \setminus T(d), \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Each document d is then represented in classification by its *activation vector* $a(d)$,

$$a(d) = (a_{t_1}(d), \ldots, a_{t_m}(d)) \ ,$$

where $t_k \in \mathcal{T}$, $1 \leq k \leq m$, and $m = |\mathcal{T}|$. Figure 2 illustrates the concepts introduced so far.

Note the following special cases of the document representation. If all weights in the network are set to 0, the document is represented by the set of its direct terms and the representation is thus equivalent to the common bag-of-words (BoW) representation—here we do not consider the case where duplicate terms occur. If all weights are set to 1, the document is represented by the set of its direct terms together with all their hypernyms, an intuitively plausible representation as well.

3 Weight update algorithm

In this section, we describe an algorithm that optimizes the semantic network weights in order to maximize the classification performance on a given document classification task. The algorithm thus implements the adaptive component of the proposed method. In short, the algorithm initializes all weights to 1 and then iteratively adjusts the weights until no more improvement in classification performance can be achieved. The algorithm implements the gradient-descent search strategy.

3.1 Document similarity and classification

Let $\hat{a}(d)$ be the *normalized activation vector* of d, that is,

$$\hat{a}(d) = \frac{a(d)}{\|a(d)\|} \ . \quad (2)$$

We calculate the similarity between any two documents $d_i, d_j \in \mathcal{D}$ from their normalized activation vectors with the commonly used dot-product measure

$$\mathrm{sim}(d_i, d_j) = \hat{a}(d_i) \cdot \hat{a}(d_j) = \sum_{t \in \mathcal{T}} \hat{a}_t(d_i) \hat{a}_t(d_j) \ . \quad (3)$$

The weight update algorithm is based on the k-nearest neighbor (kNN) classifier. Given a training set of documents $\mathcal{D}$, a document similarity measure, and a document d to be classified, the kNN classifier computes a set $N(d, k, \mathcal{D}) \subseteq \mathcal{D}$ of k documents most similar to d, also termed as the *k-neighborhood*. The document d does not itself belong to its k-neighborhood. The class assigned to d is the majority class among the documents in its k-neighborhood.

3.2 Weight update

The weight update algorithm implements the following intuition. As the documents are classified using the kNN classifier, a misclassification of a document d means that the majority of the documents in $N(d, k, \mathcal{D})$ are of a different class than d. The misclassification could therefore be corrected by modifying the k-neighborhood so that it would contain a majority of documents with the same class as that of d. This can be achieved by adjusting the semantic network weights, and thus the document representation, so that the similarity between d and its k-neighbors with a different class decreases and the similarity between d and its k-neighbors with the same class increases. As there is only one, global set of weights, any change affects all the documents and therefore directly optimizing the similarity of d with its k-neighbors also indirectly affects the similarity of d with all other documents. Generally, documents with the same class are "pulled" towards d while documents with another class are "pushed" away from d. Naturally, this effect is strongest for the k-neighbors of d, whose similarity with d is optimized directly. As the other class k-neighbors are "pushed" away from d, they are replaced in the k-neighborhood by same class documents that are "pulled" towards d. Other variations of the general scheme are possible as well. For example, the k-neighborhoods could be optimized for all documents rather than only for those that were misclassified.

Let us consider two documents $d_i, d_j \in \mathcal{D}$. The objective is to either increase or decrease $\mathrm{sim}(d_i, d_j)$ by modifying the semantic network weights. Let us define the vector w of all weights in the semantic network in an arbitrary but fixed order

$$w = (w_1, \ldots, w_n) \ ,$$

where n is the total number of weights. We then define the weight gradient $\nabla w(d_i, d_j)$ with respect to $\mathrm{sim}(d_i, d_j)$ as

$$\nabla w(d_i, d_j) = \left(\frac{\partial \, \mathrm{sim}(d_i, d_j)}{\partial w_1}, \ldots, \frac{\partial \, \mathrm{sim}(d_i, d_j)}{\partial w_n} \right) \ .$$

Adding the gradient $\nabla w(d_i, d_j)$ to the weight vector w leads to an increase of $\mathrm{sim}(d_i, d_j)$, while subtracting $\nabla w(d_i, d_j)$ from w leads to a decrease of $\mathrm{sim}(d_i, d_j)$. The formula to compute the partial derivative $\frac{\partial \, \mathrm{sim}(d_i, d_j)}{\partial \, w_{rs}}$ of $\mathrm{sim}(d_i, d_j)$ with respect to a weight w_{rs} is fully specified jointly by Equations 5, 11, and 12 in Appendix A which also details the derivation leading to the formula.

A *learning rate* constant $\eta \in \mathbb{R}$, $\eta > 0$, is introduced to control the magnitude of the weight adjustment by the gradient. The weight vector w is then updated according to the rule

$$w \leftarrow w + \delta \eta \nabla w(d_i, d_j) \ ,$$

where $\delta = +1$ (resp. $\delta = -1$) if $\mathrm{sim}(d_i, d_j)$ is to be increased (resp. decreased).

The complete weight update algorithm is introduced in Algorithm 1. In each iteration, the weight adjustments $\delta \nabla w(d_i, d_j)$ are summed into w' over all the document pairs (d_i, d_j) where d_i was misclassified and d_j belongs to its k-neighborhood. Subsequently, w', scaled by the learning rate η, is added to the weight vector w. Finally, each weight w_k in w is clipped such that the constraint $0 \leq w_k \leq 1$ holds. The iteration is finished using some stopping criterion, for example the classification performance failing to increase, which signals that the algorithm has reached a local optimum.

```
w ← 1̄
while not done:
    w' ← 0̄
    for each document d_i ∈ D:
        classify d_i using D \ {d_i} as training set
        if misclassified d_i then:
            for each d_j ∈ N(d_i, k, D \ {d_i}):
                if class(d_i) = class(d_j) then:
                    δ ← +1
                else
                    δ ← −1
                w' ← w' + δ∇w(d_i, d_j)
    w ← w + η · w'
    for each weight w_k in w:
        w_k ← max{0, min{1, w_k}}
```

Algorithm 1: Pseudocode of the weight update algorithm.

3.3 Implementation issues

An efficient implementation of the algorithm can be achieved through the following observation. Let us consider a weight w_{rs} and a term t such that t is not a hypernym of s. From the definition of activation, it is clear that $a_t(d)$ is constant with respect to w_{rs} and thus $\frac{\partial a_t(d)}{\partial w_{rs}} = 0$. Consequently, when computing $\frac{\partial \mathrm{sim}(d_i, d_j)}{\partial w_{rs}}$, it is only necessary to evaluate Equation 12 for s and its hypernyms instead of all terms in $\mathcal{T}$. The computation time of a single partial derivative $\frac{\partial \mathrm{sim}(d_i, d_j)}{\partial w_{rs}}$ is thus constant with respect to $|\mathcal{T}|$. It depends on the number of terms affected by d_i and d_j, which is typically several orders of magnitude smaller than $|\mathcal{T}|$.

Combining this observation and an efficient computation of the partial derivatives based on a linear walk through the semantic network in topological order, we were able to implement the computation of w' with the complexity $O(cM)$ with respect to the training set size M. Roughly, the constant c quadratically depends on the number of terms affected by the documents d_i and d_j and linearly depends on the k-neighborhood size.

4 Evaluation

In this section, we discuss the evaluation datasets, the experimental setup and the baseline methods.

4.1 Datasets

To evaluate the methods, a set of document classification tasks was required where the direct terms of documents belong to a semantic network. We consider datasets consisting of articles from the PubMed biomedical literature database[1], where each article has been manually assigned a set of relevant terms from the MeSH ontology[2]. This approach allows us to evaluate the method using large datasets and the use of manually assigned direct terms to represent the documents avoids the potential sources of error related to automatic mapping to a semantic network.

The datasets were formed as follows: for each dataset, a journal was selected that contains at least 2000 MeSH-indexed articles with abstracts; here we use the 10 journals we selected randomly in (Ginter *et al.* 04). Then, for each of the 10 journals, we randomly selected 2000 articles that have appeared in the journal (as positives) and 2000 that have appeared elsewhere (as negatives). Each task is then a binary classification problem where the documents must be classified either as originating from the journal or not. Since the journals are usually focused on a subdomain, these classification problems model document classification by topic.

To determine the performance of the methods with respect to different training set sizes, we formed for each dataset seven different training sets, the largest consisting of 1000 positive and 1000 negative examples (the other 2000 being used for testing). Smaller training sets were formed by downsampling so that the size is repeatedly halved.

4.2 Methods and performance measurement

We evaluate the proposed document representation with and without the adaptive component. In the *fixed* representation, the semantic network weights are all set to one constant value w_{fix}, $0 \leq w_{fix} \leq 1$, determined from the data. In the *adaptive* representation, the weights are computed using the algorithm introduced in Section 3, using a stopping criterion where iteration ends when the average performance increase on the training set over the last three rounds drops below 0.05%.

We compare the performance of the fixed and adaptive representations against two baselines, the commonly used bag-of-words (BoW) representation and a modification of the hypernym density (HD) representation (Scott & Matwin 98). In the BoW representation, each document is represented by its direct terms.

[1] `http://www.pubmed.com`

[2] We use the 2005 version of MeSH, available at `http://www.nlm.nih.gov/mesh/`

	1				2				3				4				5			
	BoW	HD	Fix.	Ad.	BoW	HD	Fix.	Ad.	BoW	HD	Fix.	Ad.	BoW	HD	Fix.	Ad.	BoW	HD	Fix.	Ad.
31	69.3	74.4	74.4	79.4	81.4	84.0	84.7	86.0	71.6	76.6	71.7	79.2	73.8	74.8	75.9	76.2	97.0	95.5	96.4	95.5
62	71.1	79.5	79.4	85.0	83.0	86.2	87.1	87.8	75.5	77.4	77.3	82.2	75.9	78.1	79.4	82.6	94.8	96.3	96.4	96.8
125	71.2	81.2	81.1	86.3	86.6	88.7	89.0	90.3	77.4	80.8	79.6	86.8	79.8	81.1	81.7	83.7	96.3	96.3	96.7	97.8
250	72.7	83.6	83.2	88.3	88.7	90.3	90.3	90.8	80.4	83.9	83.4	88.4	80.6	83.6	83.8	87.3	97.1	96.6	97.2	98.0
500	74.9	85.2	85.0	89.1	89.1	90.6	90.5	91.7	82.3	85.8	85.1	90.5	83.6	85.8	86.2	88.9	98.0	97.4	98.3	98.2
1000	76.5	86.5	86.3	89.8	90.3	91.8	91.6	92.5	84.0	87.7	87.3	91.0	85.8	87.7	87.2	90.1	97.8	97.5	98.1	98.3
2000	78.7	87.8	88.0	91.1	91.2	92.2	92.1	92.7	86.8	89.3	88.8	91.9	87.1	88.4	88.7	90.5	97.7	97.9	97.9	98.6

	6				7				8				9				10			
	BoW	HD	Fix.	Ad.	BoW	HD	Fix.	Ad.	BoW	HD	Fix.	Ad.	BoW	HD	Fix.	Ad.	BoW	HD	Fix.	Ad.
31	64.6	72.9	70.9	71.0	65.1	64.3	66.2	64.7	65.0	66.0	65.4	66.9	65.8	67.5	67.4	69.1	71.8	71.4	71.2	70.9
62	67.3	74.8	74.2	76.5	64.5	68.0	67.1	67.9	66.3	67.4	66.9	68.9	68.1	71.8	71.1	73.1	72.4	73.3	73.1	75.5
125	70.3	77.6	76.3	79.1	65.8	69.9	70.4	70.8	67.8	69.5	68.3	70.9	69.9	71.6	71.4	74.8	73.9	76.8	76.8	79.6
250	75.2	80.2	80.1	82.8	66.5	75.0	73.5	74.7	69.2	71.5	70.8	71.9	72.1	74.6	74.5	76.2	75.1	78.3	77.7	81.7
500	77.3	81.9	82.0	83.8	68.7	76.9	76.1	77.8	70.1	73.3	72.5	73.5	74.5	75.9	75.3	77.4	77.5	81.1	80.4	83.1
1000	80.9	84.5	84.2	85.5	68.3	78.1	76.9	79.4	71.2	74.3	73.6	75.3	76.2	77.5	77.1	78.7	79.2	83.0	82.6	84.9
2000	82.8	85.7	85.5	86.4	69.3	79.8	77.9	80.9	72.9	75.8	75.4	76.2	77.4	78.3	78.7	79.7	80.7	83.8	83.9	85.7

Table 1: *Classification performance of kNN.* Cross-validated accuracy measurements for each of the ten datasets and each of the seven training set sizes. The MEDLINE abbreviations of the corresponding journal names are, in order, *Acta Anat (Basel), Appl Environ Microbiol, Biol Psychiatry, Eur J Obstet Gynecol Reprod Biol, Fed Regist, J Pathol, Nippon Rinsho, Presse Med, Schweiz Rundsch Med Prax,* and *Toxicol Lett.*

In the HD representation, each document d_i is represented by a multiset consisting of all direct terms of d_i, together with their hypernyms up to a distance h from any of the direct terms. We modified the HD representation as follows. We found that in our case coercing the multiset into a set results in an improvement of performance, and thus we apply this step in our evaluation. Further, infrequent terms are not discarded. The HD normalization step is performed by the classifiers. Note that for $h = 0$ the HD representation is equivalent to the BoW representation, and for $h = \infty$ it is equivalent to the fixed representation with $w_{fix} = 1$.

The main evaluation was performed using the kNN classifier. In this evaluation, the parameters of the various methods—k for BoW, k, h for HD, k, w_{fix} for fixed, and k, η for adaptive—were selected separately in each fold by cross-validated grid search on the training set. To assess the applicability of the representations to other classification methods, we also performed a limited evaluation using Support Vector Machines (SVM), a state-of-the-art machine learning method (Vapnik 98). For this evaluation, only the SVM regularization parameter C was separately selected, while other parameters were set to their kNN optimum values.

We measure the performance of the various methods using average 5×2 cross-validated accuracy, reporting differences in accuracy as well as relative decreases in error rate to better estimate the performance of the methods with respect to different baselines. To assess the statistical significance of results for individual datasets, we use the robust 5×2 cross-validation test (Alpaydin 99). To assess the overall significance across all datasets, we use the standard two-tailed paired t-test.

5 Results and discussion

Results with kNN are given in Table 1, and average differences are plotted in Figure 3. Averages are also given in Table 2a.

As can be seen in Figure 3, the adaptive method statistically significantly outperforms all others for all except the smallest training set size. For training set sizes 62 and larger, the adaptive method outperforms BoW by 5–6 percentage units, reflecting a relative decrease in error rate systematically over 20% and approaching 30% for large dataset sizes. The fixed and HD representations also perform well against BoW, both achieving a statistically significant increase in accuracy of 3–4 percentage units (12–20% relative decrease in error rate) for all but the smallest training set size. The differences between these two representations suggest a small (0.1–0.4 percentage unit) advantage to the HD representation, but this difference is largely not statistically significant. Against the fixed and HD representations, the adaptive method offers an accuracy increase between 1 and 3 percentage units, that is, a systematic relative decrease in error rate of 10–14% for all but the smallest training set size.

Further, the average absolute performance advantage of the adaptive method over the BoW baseline grows with increasing training set size from 31 to 250 examples, and falls thereafter. In contrast, in terms of relative decrease in error rate this performance advantage grows almost monotonically, indicating that the adaptive method works better given more data. As the documents were assigned on average only 10

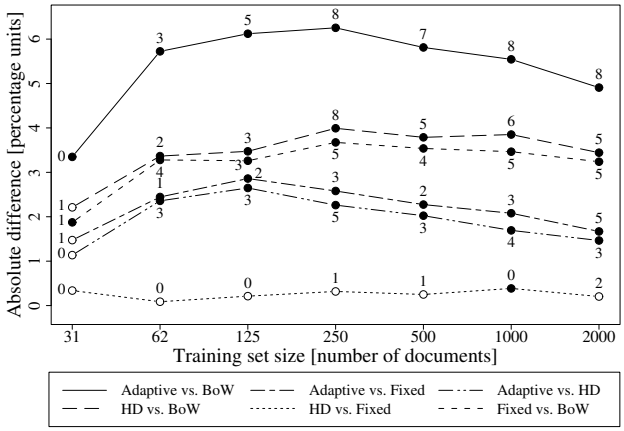


Figure 3: *Pairwise method differences and their per-dataset and overall statistical significances for kNN.* Results averaged over all datasets. The number displayed by each difference denotes the number of individual datasets for which the difference was statistically significant ($p < 0.05$, 5×2cv test). Full circle, as opposed to empty circle, denotes the average difference over all ten datasets being statistically significant ($p < 0.05$, t-test).

MeSH terms and the MeSH ontology contains almost 23000 nodes, reliable optimization of the edge weights is expected to be difficult with very small training sets. Nevertheless, the adaptive method works remarkably well with as few as 62 training examples.

	BoW	HD	Fix.	Ad.
31	72.5	74.8	74.4	75.9
62	73.9	77.3	77.2	79.6
125	75.9	79.4	79.1	82.0
250	77.8	81.8	81.4	84.0
500	79.6	83.4	83.1	85.4
1000	81.0	84.9	84.5	86.6
2000	82.4	85.9	85.7	87.4

(a)

	BoW	HD	Fix.	Ad.
31	75.6	79.1	78.6	78.1
62	78.2	81.6	81.5	80.9
125	80.9	84.0	84.3	83.0
250	83.2	85.8	85.9	84.7
500	85.4	87.5	87.8	86.7
1000	87.5	88.9	89.2	88.4
2000	88.9	90.0	90.3	89.7

(b)

Table 2: *Classification performance.* Accuracy measurements averaged over all ten datasets for each of the seven training set sizes: (a) kNN results, (b) SVM results.

We now present the results of the evaluation with SVM. The average SVM results over the ten datasets are given in Table 2b. The BoW baseline is again outperformed by the other three representations for all training set sizes, with relative decrease in error rate ranging between 12–18% for the fixed representation, 10–17% for the HD representation, and 6–13% for the adaptive method. These differences are statistically significant for all training set sizes for the fixed and HD representations and for training set sizes of 500 and larger for the adaptive method.

We observe that when applied to SVMs, the fixed and HD representations outperform the adaptive method. The difference is statistically significant for most training set sizes larger than 62, where the relative decrease in error rate over the adaptive method ranges between 2–9%. The SVM classification principle substantially differs from that of kNN. Clearly, the adaptive method does not optimize a criterion beneficial for SVM classification, and hence modification of the adaptive strategy is required to increase applicability to SVM classification. Nevertheless, as the fixed representation outperforms both the BoW and HD representations for larger training set sizes (the latter difference is mostly not statistically significant), the general strategy appears to apply well also to SVM.

6 Conclusions and future work

In this paper, we have developed the idea that semantic networks can be used to develop an adaptive document similarity measure. We have discussed desirable properties for such a measure and presented a document representation that implements these properties. Further, we have introduced a gradient descent-based algorithm driven by misclassifications that adapts the representation to data. We have evaluated the representation and the algorithm against the BoW, fixed and HD representations with ten randomly selected datasets from the PubMed biomedical literature database using MeSH 2005 terms as features.

Our results indicate that the proposed adaptive method can statistically significantly outperform the commonly used BoW representation and the more advanced HD representation as well as our new representation with fixed weights over a range of training set sizes from 62 to 2000, for which the relative decrease in error rate ranged between 20–30% against BoW and 10–14% against the fixed and HD representations.

A separate evaluation with Support Vector Machines indicated that while the semantic network-based document representations give a statistically significant improvement over the BoW baseline and the proposed representation performs as well as the HD representation, the gradient descent component of the adaptive method, driven by kNN misclassifications, requires modification to apply beneficially to SVMs. A possible future direction would thus be to introduce the gradient descent algorithm into the SVM training phase, potentially leading to further performance improvements for the classifier.

We conclude that the proposed adaptive similarity measure can successfully determine term-document relevance in a data-dependent manner, increasing performance in supervised document classification tasks. As future work, several aspects of the proposed method can be studied, such as the setting of the initial weights, the learning rate, and the stopping criterion. An additional natural extension of the method is to consider relationships other than hyponymy as activation paths. Careful analysis of these and other properties may offer further opportunities for the use of semantic networks in document classification.

Acknowledgments

This work has been supported by Tekes, the Finnish National Technology Agency.

References

(Agirre & Rigau 96) E. Agirre and G. Rigau. Word sense disambiguation using conceptual density. In *Proceedings of the 16th conference on Computational linguistics COLING '96, Copenhagen, Denmark*, pages 16–22, 1996.

(Alpaydin 99) E. Alpaydin. Combined 5×2 cv F-test for comparing supervised classification learning algorithms. *Neural Computation*, 11(8):1885–1892, 1999.

(Basili *et al.* 05) R. Basili, M. Cammisa, and A. Moschitti. Effective use of WordNet semantics via kernel-based learning. In I. Dagan and D. Gildea, editors, *Proceedings of the Ninth Conference on Computational Natural Language Learning CoNLL 2005, Ann Arbor, Michigan*, pages 1–8. Association for Computational Linguistics, 2005.

(Bloehdorn & Hotho 04) S. Bloehdorn and A. Hotho. Boosting for text classification with semantic features. In *Proceedings of the MSW 2004 workshop, 10th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Seattle*, pages 70–87, 2004.

(Budanitsky & Hirst 01) A. Budanitsky and G. Hirst. Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures. In *Workshop on WordNet and Other Lexical Resources, Second meeting of the North American Chapter of the Association for Computational Linguistics*, pages 29–34, Pittsburgh, 2001.

(Ginter *et al.* 04) F. Ginter, S. Pyysalo, J. Boberg, J. Järvinen, and T. Salakoski. Ontology-based feature transformations: A data-driven approach. In J. L. Vicedo, P. Martínez-Barco, R. Muñoz, and M. Saiz Noeda, editors, *Proceedings of the 4th International Conference EsTAL 2004*, pages 279–290. Springer, Heidelberg, 2004.

(Gomez-Hidalgo & deBuenaga Rodriguez 97) J. M. Gomez-Hidalgo and M. de Buenaga Rodriguez. Integrating a lexical database and a training collection for text categorization. In *Proceedings of the ACL/EACL 97 Workshop on Automatic Information Extraction and Building of Lexical Semantic Resources, Madrid, Spain*, pages 39–44, 1997.

(Jiang & Conrath 97) J. Jiang and D. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the International Conference on Research in Computational Linguistics*, pages 19–33. Academica Sinica, 1997.

(Leacock & Chodorow 98) C. Leacock and M. Chodorow. Combining local context and wordnet similarity for word sense identification. In C. Fellbaum, editor, *WordNet: An Electronic Lexical Database*, pages 265–283. MIT Press, Cambridge, MA, 1998.

(Patwardhan *et al.* 03) S. Patwardhan, S. Banerjee, and T. Pedersen. Using measures of semantic relatedness for word sense disambiguation. In A. Gelbukh, editor, *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*, pages 241–257, Mexico City, Mexico, 2003. Springer-Verlag, Heidelberg.

(Resnik 95) P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In C. Mellish, editor, *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 448–453. Morgan Kaufmann, San Francisco, 1995.

(Scott & Matwin 98) S. Scott and S. Matwin. Text classification using WordNet hypernyms. In S. Harabagiu, editor, *Proceedings of Use of WordNet in Natural Language Processing Systems*, pages 38–44, Somerset, New Jersey, 1998. Association for Computational Linguistics.

(Vapnik 98) V. Vapnik. *Statistical Learning Theory.* Wiley, New York, 1998.

A Derivation of the formula for $\frac{\partial \operatorname{sim}(d_i, d_j)}{\partial w_{rs}}$

This appendix details the derivation of the formula to compute the value of $\frac{\partial \operatorname{sim}(d_i, d_j)}{\partial w_{rs}}$. Starting from (4), the partial derivative is solved and the final formula is obtained jointly from Equations 5, 11, and 12.

$$\frac{\partial \operatorname{sim}(d_i,\, d_j)}{\partial w_{rs}} = \frac{\partial\, \hat{a}(d_i) \cdot \hat{a}(d_j)}{\partial w_{rs}} = \sum_{t \in \mathcal{T}} \frac{\partial\, \hat{a}_t(d_i) \hat{a}_t(d_j)}{\partial w_{rs}} = \sum_{t \in \mathcal{T}} \frac{\partial\, \hat{a}_t(d_i)}{\partial w_{rs}} \hat{a}_t(d_j) + \sum_{t \in \mathcal{T}} \frac{\partial\, \hat{a}_t(d_j)}{\partial w_{rs}} \hat{a}_t(d_i) \quad (4)$$

Let

$$\frac{\partial \operatorname{sim}(d_i,\, d_j)}{\partial w_{rs}} \stackrel{def}{=} Q(d_i,\, d_j) + Q(d_j,\, d_i)\ , \quad (5)$$

where

$$Q(d_i,\, d_j) \stackrel{def}{=} \sum_{t \in \mathcal{T}} \frac{\partial\, \hat{a}_t(d_i)}{\partial w_{rs}} \hat{a}_t(d_j) \quad (6)$$

In the following, we solve $Q(d_i,\, d_j)$; the formula for $Q(d_j,\, d_i)$ follows by symmetry.

$$\frac{\partial\, \hat{a}_t(d_i)}{\partial w_{rs}} \stackrel{(2)}{=} \frac{\partial\, \frac{a_t(d_i)}{\|a(d_i)\|}}{\partial w_{rs}} = \frac{\frac{\partial\, a_t(d_i)}{\partial w_{rs}} \|a(d_i)\| - a_t(d_i) \frac{\partial \|a(d_i)\|}{\partial w_{rs}}}{\|a(d_i)\|^2} \quad (7)$$

$$\frac{\partial \|a(d_i)\|}{\partial w_{rs}} = \frac{\partial \sqrt{\sum_{u \in \mathcal{T}} [a_u(d_i)]^2}}{\partial w_{rs}} = \frac{1}{2\|a(d_i)\|} \frac{\partial \sum_{u \in \mathcal{T}} [a_u(d_i)]^2}{\partial w_{rs}} = \frac{1}{\|a(d_i)\|} \sum_{u \in \mathcal{T}} \left(a_u(d_i) \frac{\partial\, a_u(d_i)}{\partial w_{rs}} \right) \quad (8)$$

Combining (7) and (8) yields

$$\frac{\partial\, \hat{a}_t(d_i)}{\partial w_{rs}} = \frac{\frac{\partial\, a_t(d_i)}{\partial w_{rs}} \|a(d_i)\| - \hat{a}_t(d_i) \sum_{u \in \mathcal{T}} \left(a_u(d_i) \frac{\partial\, a_u(d_i)}{\partial w_{rs}} \right)}{\|a(d_i)\|^2} \quad (9)$$

Substituting from (9) into (6) gives

$$\begin{aligned} Q(d_i,\, d_j) &= \frac{\sum_{t \in \mathcal{T}} \frac{\partial\, a_t(d_i)}{\partial w_{rs}} \|a(d_i)\| \hat{a}_t(d_j) - \sum_{t \in \mathcal{T}} \left(\hat{a}_t(d_j) \hat{a}_t(d_i) \sum_{u \in \mathcal{T}} a_u(d_i) \frac{\partial\, a_u(d_i)}{\partial w_{rs}} \right)}{\|a(d_i)\|^2} \\ &= \frac{\sum_{t \in \mathcal{T}} \frac{\partial\, a_t(d_i)}{\partial w_{rs}} \|a(d_i)\| \hat{a}_t(d_j) - \left(\sum_{u \in \mathcal{T}} a_u(d_i) \frac{\partial\, a_u(d_i)}{\partial w_{rs}} \right) \left(\sum_{t \in \mathcal{T}} \hat{a}_t(d_j) \hat{a}_t(d_i) \right)}{\|a(d_i)\|^2} \\ &\stackrel{(3)}{=} \frac{\sum_{t \in \mathcal{T}} \frac{\partial\, a_t(d_i)}{\partial w_{rs}} \|a(d_i)\| \hat{a}_t(d_j) - \sum_{u \in \mathcal{T}} a_u(d_i) \frac{\partial\, a_u(d_i)}{\partial w_{rs}} \operatorname{sim}(d_i,\, d_j)}{\|a(d_i)\|^2} \end{aligned} \quad (10)$$

Substituting t for u in the second term of (10) gives

$$\begin{aligned} Q(d_i,\, d_j) &= \frac{\sum_{t \in \mathcal{T}} \frac{\partial\, a_t(d_i)}{\partial w_{rs}} \left(\|a(d_i)\| \hat{a}_t(d_j) - a_t(d_i) \operatorname{sim}(d_i,\, d_j) \right)}{\|a(d_i)\|^2} \\ &= \frac{\sum_{t \in \mathcal{T}} \frac{\partial\, a_t(d_i)}{\partial w_{rs}} \left(\hat{a}_t(d_j) - \hat{a}_t(d_i) \operatorname{sim}(d_i,\, d_j) \right)}{\|a(d_i)\|} \end{aligned} \quad (11)$$

If $t \notin \mathit{Aff}(d_i) \setminus T(d_i)$ then $a_t(d_i)$ is by (1) constant and consequently $\frac{\partial\, a_t(d_i)}{\partial w_{rs}} = 0$. For $t \in \mathit{Aff}(d_i) \setminus T(d_i)$,

$$\begin{aligned} \frac{\partial\, a_t(d_i)}{\partial w_{rs}} &\stackrel{(1)}{=} \frac{1}{|\mathit{Base}_t(d_i)|} \sum_{t' \in \mathit{Base}_t(d_i)} \frac{\partial\, w_{t't} a_{t'}(d_i)}{\partial w_{rs}} \\ &= \frac{1}{|\mathit{Base}_t(d_i)|} \sum_{t' \in \mathit{Base}_t(d_i)} \begin{cases} a_r(d_i) & \text{if } (t',\, t) = (r,\, s) \\ w_{t't} \frac{\partial\, a_{t'}(d_i)}{\partial w_{rs}} & \text{otherwise.} \end{cases} \end{aligned} \quad (12)$$

The value of $\frac{\partial\, a_t(d_i)}{\partial w_{rs}}$ is computed recursively by (12). The recursion ends when $(t',\, t) = (r,\, s)$. Substituting from (12) into (11) and subsequently from (11) into (5) completes the formula.

Adapting Lexical Chaining to Summarize Conversational Dialogues

Iryna Gurevych
EML Research gGmbH
Schloss-Wolfsbrunnenweg 33
D-69118 Heidelberg, Germany
www.eml-research.de/∼gurevych

Thade Nahnsen
International University in Germany
School of Information Technology
D-76646 Bruchsal, Germany
thade.nahnsen@gmx.de

Abstract

We present a system for summarizing transcripts of conversational dialogues based on lexical chaining. The experiments were carried out with twenty Switchboard dialogues (LDC, 1993). We designed and implemented four summarization methods employing lexical chains as their source representation. The summarization task is defined as extracting the most relevant utterances conveying the meaning of the dialogue. We evaluate the methods against *lead* and *random* baseline systems and show that lexical chaining outperforms them in terms of precision and recall.

Keywords: summarization, dialogue, lexical chains.

1 Introduction

The paper addresses the challenge of summarizing transciprts of spoken dialogues in unrestricted domains. Previous work on summarization focused on such genres as news articles (McKeown et al., 1995), web pages (Berger and Mittal, 2000), scientific texts (Teufel and Moens, 2002). In dialogue summarization, the motivation is the automatic transcription and summarization of multi-party dialogues, e.g. meetings (Alexandersson and Poller, 1998; Reithinger et al., 2000; Zechner, 2002). Therefore, it needs to deal with the whole range of dialogue and speech phenomena. Alexandersson and Poller (1998) present a system for generating meeting minutes in multiple languages. The approach is domain-sensitive as it relies on a database of handcrafted knowledge. The summary is produced using natural language generation techniques. The employment of this methodology in unrestricted domains is not feasible, as deep understanding of unrestricted spoken discourse is still an unsolved problem. Going beyond restricted domains requires domain-independent processing. The system presented by Zechner (2002) is designed for summarizing conversational dialogues in unrestricted domains. He uses pre-processing techniques to "normalize" the dialogue input, i.e. remove speech disfluencies, false-starts, detect question-answer pairs, etc. Statistical techniques are used to create the summaries. The output of the system is based on words in the input.

Gurevych and Strube (2004) employ a set of WordNet-based semantic similarity metrics to perform dialogue summarization. The methods evaluate the noun portion of WordNet in order to determine semantic similarity between utterances and a whole dialogue. The approach operates on manually disambiguated nouns. Bellare et al. (2004) determine subgraphs of WordNet, which are most relevant with respect to the semantics of the document. The sentence selection is performed based on the synsets that are most relevant to the text. Erkan and Radev (2004) approach text summarization from a graph-theoretical point of view. Their approach assigns weights to connections based on the number of occurrence and on the type of elements a specific element is connected to.

Our approach attempts to perform dialogue summarization with the help of lexical semantics, thus bridging the gap between domain-dependent deep analysis and domain-independent statistical processing. The system is based on the intuition that if lexical chains are used as intermediate representation in dialogue summarization, then "strong" lexical chains will be represented by the most relevant utterances. We designed and implemented four different methods to summarize dialogues based on representations constituted by lexical chains.

2 Research on Lexical Chains

Lexical chains are defined as sets of lexical items, which are either identical or related to each other by conceptual similarity. Conceptual similarity is determined on the basis of a certain lexical-semantic resource, e.g. WordNet (Fellbaum, 1998) and lexical-semantic relations between individual lexemes. Work on lexical cohesion dates back to Halliday and Hasan (1976) and even earlier. Morris and Hirst (1991) suggest lexical chains to determine the discourse structure of the text. The criterion for the inclusion of the word in a chain is a cohesive relation, which is figured out with the help of a thesaurus. Hirst and St-Onge (1998) propose to

employ WordNet as a knowledge source for building lexical chains. Their definition of semantic relatedness is, hence, based on WordNet and synsets. Three kinds of relations can be distinguished: 1) extra-strong (holds between a word and its repetition); 2) strong (a synset is common to two lexemes, or there is a horizontal link, such as ANTONYMY, SIMILARITY, SEE-ALSO, or there is any kind of link between a synset associated with each word if one word is a compound phrase that includes the other); 3) medium-strong (there is a legal path connecting the synsets associated with each word).

Barzilay and Elhadad (1999) describe an algorithm for text summarization employing lexical chains as its intermediate representation. The algorithm includes three steps: 1) constructing lexical chains; 2) identifying strong chains; 3) extracting significant sentences from the text. The authors evaluate their algorithm on 30 texts. However, their evaluation is informal and does not provide an empirical proof whether the lexical chains model outperforms alternative summarization techniques. Also, there is no intrinsic evaluation, i.e. whether lexical chains constitute an appropriate representation of the discourse to be summarized. Silber and McCoy (2002) extend the work by Barzilay and Elhadad (1999). Two main contributions of their work are the following:

- an algorithm for computing lexical chains that is linear in time and space, thus eliminating one of the disadvantages in the earlier work, i.e. an exponential inefficiency for computing the chains. This makes it computationally feasible to compute lexical chains for large documents in real time;
- a new method for the evaluation of lexical chains as an intermediate representation in the summarization process. Their evaluation is based on a corpus of 10 scientific articles and 14 chapters from university textbooks.

Galley and McKeown (2003) focus on the lexical chaining algorithm in the context of work on word sense disambiguation (WSD). Along with the computational inefficiency mentioned earlier, a lack of accuracy in WSD is known to be a drawback of lexical chaining based algorithms. Galley and McKeown employ a different algorithm for computing lexical chains based on the *"one sense per discourse"* assumption. Their algorithm: 1) builds a representation of all possible interpretations of the text; 2) disambiguates all words; 3) finally constructs the lexical chains. The authors evaluate their algorithm with respect to the task of word sense disambiguation on the SEMCOR corpus. Their algorithm outperforms both Barzilay and Elhadad's and Silber and McCoy's algorithm (accuracies of 62.09%, 56.56% and 54.48% WSD respectively). No attempt is made to evaluate any further aspects of lexical chains.

The discourse type underlying our research, i.e. conversational dialogues, does not conform with the *one sense per discourse* constraint. In our corpus, topical changes occur rather frequently. Thus, one word may have different meanings within a single discourse. Therefore, our algorithm for building lexical chains follows other previous work (cf. e.g. Silber and McCoy (2002). Though slightly inferior in terms of WSD, it is both computationally efficient and imposes no constraints on the number of meanings that a single lexeme may have within a discourse.

The goals of this paper are the following: design summarization techniques based on lexical chaining for a new genre, i.e. conversational dialogue and carry out an extrinsic evaluation of lexical chains in dialogue summarization.

3 Experiments on Dialogue Summarization

3.1 Corpus

The experiments were carried out with twenty Switchboard dialogues on various topics, e.g. child care, dressing code. Data on our corpus is given in Table 1. The dialogue transcripts were manually annotated by three humans by selecting about 10% of utterances as being *relevant*, s. Table 7 for an excerpt from one of the dialogues. The reconciled version of the annotations, i.e. the *gold standard* was produced by selecting utterances labeled *relevant* by at least two annotators. It includes 9.47% of all utterances. When calculated for the whole corpus, the Kappa coefficient yielded .43. While this is not a high agreement rate on a general scale, it is comparable to what has been reported concerning the task of summarization in general (cf. (Mitra et al., 1997; Radev et al., 2003)).

3.2 Computing Lexical Chains

Lexical chains are computed on the basis of the noun portion of WordNet1.7. In the first step, the dialogue is processed and noun instances are selected. Thus, the dialogue D is represented as a set of nouns $D = \{N_1, ..., N_n\}$, each of them having a set of possible interpretations (synsets) $I_N = \{s_1, ..., s_m\}$ in WordNet. Then, the algorithm by Silber and McCoy (2002)

dialogue	words	utt./ markables	*relevant* utterances	lex. chains	*strong* lex. chains
1	2350	267	24	80	3
2	1069	79	15	50	2
3	1180	110	15	52	3
4	969	60	12	37	2
5	1428	133	15	55	1
6	1417	160	17	34	3
7	1159	131	15	28	2
8	2092	254	20	56	1
9	1284	162	12	43	2
10	1316	149	14	43	3
11	1521	138	16	37	3
12	1225	110	18	41	2
13	4046	416	22	83	2
14	2604	229	16	62	2
15	1542	53	9	49	3
16	1576	144	14	38	1
17	1966	159	11	54	3
18	1799	157	14	55	2
19	2751	210	15	66	2
20	1536	154	16	42	2
Total:	**34830**	**3275**	**310**		

Table 1: Descriptive corpus statistics

	1 utt.	3 utt.	5 utt.	Default
Identical word	1	1	1	1
Synonym	1	1	1	1
Hypernym	1	0.5	0.5	0.5
Sibling	1	0.3	0.2	0

Table 2: Computing word contributions to chains

is employed to automatically perform word sense disambiguation of the nouns.

We adapted the scheme for computing the contribution of a word to the chain as compared to that employed by Silber and McCoy due to a different discourse type, i.e. dialogues. Table 2 summarizes the values which are used to compute contributions of words to lexical chains in our system. It is similar to the original scheme in that it is based on two essential parameters: the nature of semantic relations between synsets and the distance between noun instances in the discourse. However, due to a different genre, i.e. dialogue versus text, the distance is defined in terms of utterances rather than paragraphs. Following Silber and McCoy, we allow different types of relations existing within the chain to contribute differently to that chain. The disambiguated sense of the noun is related to other synsets, see Table 3.

We store the corresponding interpretation s (synset) for each N (noun), resulting in the dialogue D being interpreted as a set of synsets $D = \{s_1, ..., s_m\}$. In Table 4, the "head" synsets of lexical chains that a given noun is related to are presented. On the other side, for each synset a corresponding lexical chain is stored, see Table 5. When the chains have been computed, they are ranked according to the scoring function defined by Barzilay and Elhadad (1999): $Score(Chain) = Length * Homogeneity$, where $Length$ is the total number of synset occurrences in the chain, while $Homogeneity$ is (1 – the number of distinct synset occurrences divided by $Length$). Strong chains are then defined as follows: $Score(Chain) > Average(Scores) + 2 * StandardDeviation(Scores)$. Table 1 gives an overview over the distribution of strong lexical chains in our data, and Table 6 gives examples of some initial synsets of chains ranked according to their strength.

3.3 Creating summaries

We designed and implemented four dialogue summarization methods operating on lexical chains. The set of lexical chains in D is represented as a two-dimensional matrix LC with the dimensions $(\#c \times \#s)$, where $\#c$ and $\#s$ denote the overall numbers of lexical chains and synsets in the dialogue, respectively. This can be formalized as: $LC = (b_{cs})_{1,...,\#c,s=1,...,\#s}$, where the matrix elements b_{cs} are the boolean values denoting whether the chain contains the corresponding synset or not. The chains are sorted numerically in a descending order according to their strength, i.e. the dialogue is also represented by the vector of lexical chains $(c_1, ..., c_{\#c})$. The knowledge represented by the lexical chains can be utilized in two ways by the summarization algorithm: from chains to utterances and from utterances to chains.

3.3.1 From chains to utterances

Utterances in the dialogue are ranked according to the strength of the strongest chain crossing them and their discourse position. The heuristics presented by Barzilay and Elhadad (1999) extract one sentence for each *strong* lexical chain. Method 1, called *one utterance per chain* method is similar to this heuristic, as we extract exactly one utterance per chain. However, it is also different from the original heuristics – we consider all lexical chains instead of only the strong ones, as the number of strong chains in our dialogues is small. The rest of the utterances are appended at the end in the order of their occurrence in the dialogue. This is done in order to fit a given compression rate when a summary is generated.

```
Step 1

  For each chain beginning with the strongest one
    Find the 1st utterance containing at least 1
    element belonging to the chain
    Insert the utterance into summary
```

noun	synset offset	gloss
child care	922884	a service involving care for other people's children
	922515	an act of help or assistance; "he did them a service"
	923360	childcare during the day while parents work

Table 3: Synsets related to a given sense of the noun

noun	synset offset	gloss
subject	5303	a human being; "there was too much for one person to do"
child care	922884	a service involving care for other people's children
children	5303	a human being; "there was too much for one person to do"
facility	15787	a man-made object taken as a whole
opinions	5079811	any cognitive content held as true
thoughts	5079811	any cognitive content held as true

Table 4: Disambiguated nouns

```
Step 2
 For each utterance
  If the utterance is not in the summary
   Append the utterance to the summary
```

Method 3, called *many utterances per chain* is similar to the previously introduced one. However, instead of extracting exactly one utterance per chain, we extract all utterances per chain (in the order of their dialogue occurrence), and process all chains in a descending order. At the end, we attach the utterances which are not represented by any chains in the order of their dialogue occurrence.

```
Step 1
  For each chain beginning with the strongest one
   Find all utterances containing at least 1
   element belonging to the chain
   Insert the utterances into summary
Step 2
 For each utterance
  If the utterance is not in the summary
   Append the utterance to the summary
```

3.3.2 From utterances to chains

The overall utterance score is a function of the number and type of chains crossing a particular utterance. In Methods 2 & 4, we find all noun instances in the utterance represented by synsets and assign a score to the noun based on the synset's chain membership. For Method 2, if a particular synset belongs to a *strong* chain, the contribution of the noun to the overall utterance score is 2, otherwise the contribution is 1. The utterance score is defined as a sum of all noun contributions. Then, the utterances are sorted numerically in a descending order according to their ranks.

```
For each utterance
  For each synset
  If synset belongs to a strong chain
   Add 2 to the utterance score
  Else
   Add 1 to the utterance score
Sort utterances numerically by overall score
```

For Method 4, the only difference is the scoring heuristic: instead of using binary weights (2 corresponding to a "strong" chain and 1 to any other chain), we employ the absolute weights of the respective lexical chains as scores for the synsets belonging to them.

```
For each utterance
  For each synset
   Add the strength score of the chain to the
   utterance score
Sort utterances numerically by overall score
```

4 Evaluation

Evaluating summaries produced on the basis of lexical chains is not straight-forward. We define dialogue summarization as the extraction of *relevant* utterances from the dialogue transcript. *Relevant* utterances are defined as those carrying the essential content of the dialogue. As it is desirable to support varying lengths of the resulting summaries, the *compression rate* is adjustable. Therefore, the summarization method supports ranking of all utterances in the dialogue, rather than a selection of individual utterances. We reformulate the problem in terms of standard information retrieval evaluation metrics: Precision, Recall and F-measure.

synset offset	gloss	nouns in the chain
5303	a human being; "there was too much for one person to do"	subject, children, child, children, child, children, person children, child, child, case, child, child, person
922884	a service involving care for other people's children	child care, child care, child care, child care, day care child care, child care
15787	a man-made object taken as a whole	facility, facilities, stuff, facility
5079811	any cognitive content held as true	opinions, thoughts, thought

Table 5: Synsets and lexical chains they belong to

synset offset	words and gloss	strength
5303	person, individual, someone, somebody, mortal, human, soul – (a human being; "there was too much for one person to do")	11.0
22634	group, grouping – (any number of entities (members) considered as a unit)	9.0
11745254	condition, status – (a state at a particular time; "a condition (or state) of disrepair"; "the current status of the arms negotiations")	6.0
12814143	time-of-life – (a period of time during which a person is normally in a particular life state)	6.0
922884	childcare, child-care – (a service involving care for other people's children)	5.0
8522773	parent – (a father or mother; one who begets or one who gives birth to or nurtures and raises a child; a relative who plays the role of guardian)	3.0

Table 6: Chains represented by initial synsets and their strengths

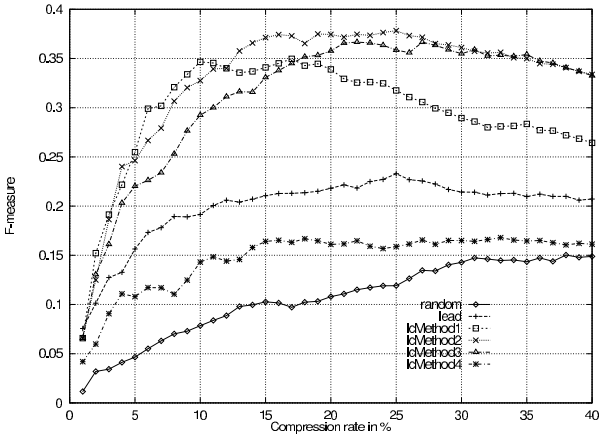


Figure 1: F-measure versus compression rate [1;40]

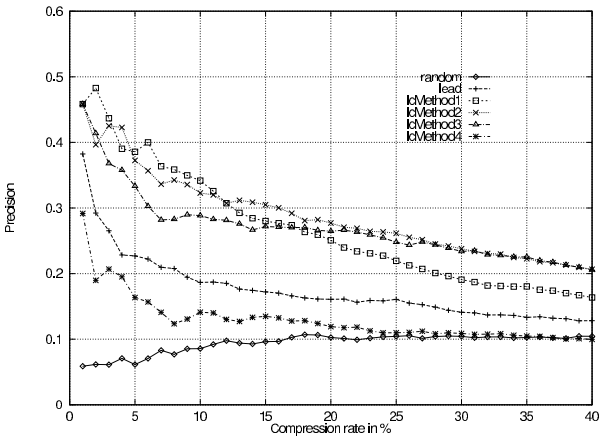


Figure 2: Precision versus compression rate [1;40]

Two baseline systems are employed in the evaluation. The first system is a *random* baseline, where relevant utterances (depending on the compression rate) were selected by chance. The second baseline, *lead*, is based on the intuition that the most important utterances tend to occur at the beginning of the discourse.

Figures 1 and 2 show that all lexical chaining based summarization methods, except for Method 4, outperform the baselines. Method 4 computes a score for each utterance by summing up the weights of nouns defined as the strength values of their respective chains. This strongly favours the utterances containing nouns belonging to the strongest chains, while the importance of other chains is minimized. Apparently this assumption is not true. Method 2 performs better than Methods 1 and 3, but this difference is not significant. The precision of all methods is rather low, e.g. about 23% for the compression rate 20%. Nevertheless the utterances selected by them differ (see Table 7), which suggests that an algorithm integrating multiple knowledge types is needed.

Our results are comparable to the results reported by Gurevych and Strube (2004) for the same dataset, e.g. at *compression rate* 25%, F-measure improves from .35 to .37. Both approaches employ WordNet as a knowledge source to determine the most relevant utterances. However, our algorithm disambiguates word senses automatically, whereas the results by Gurevych and Strube (2004) are based on manually disambiguated word senses. A comparison to the work by Zechner (2002) which is also based on Switchboard, i.e. domain-independent conversa-

utterance	gold standard	Method1	Method2	Method3	Method4
Go ahead.	none	21	39	39	39
oh, okay.	none	22	56	40	56
Yeah	none	23	52	41	52
the, uh, subject is child care and how to determine child care,	relevant	1	17	1	29
and that's, uh, an interesting one for me to talk about since I have no children,	none	24	33	2	11
but I did run a child care facility for a while.	none	9	19	27	1
Um .	none	25	46	42	46
And, uh, have some,	none	26	49	43	49
Well, you should, you should have some opinions on that, then.	none	5	36	29	20
I do have some thoughts on that,	none	27	35	30	15
yeah.	none	28	40	44	40
Uh, it's, uh, an interesting experience to be a surrogate parent for, or parent for a lot of people there,	none	2	3	12	6
and, uh, it's also very interesting in terms of how people choose the child care facilities	relevant	29	16	13	28
Well, I guess if I were going to choose, I mean, my first consideration would be safety.	relevant	3	23	19	25
My second consideration would be, uh, uh, health.	relevant	31	29	20	38
And, uh, I guess my third consideration would be, uh, warm environment, warm personal environment.	relevant	10	15	28	26
Well, right.	none	33	54	47	54
Uh, in Texas, we have to meet certain state standards in order to operate on a, at an institutional level and at a, like a small home level	none	8	4	21	10
so you meet the standards,	none	34	31	36	4
but then after that there's, there's a lot more.	none	35	34	34	14
I think it's important as the safety and health and that kind of stuff, is qualification of people who work there,	relevant	7	6	14	16
...					

Table 7: Utterances, their ranks and *gold standard*

tional dialogues, is not directly possible. He adopts a different view of the task, where summarization is performed by summarizing topical segments of dialogues (determined manually in his evaluation). In our approach, topic segmentation is performed implicitly through lexical chains. Additionally, his evaluation scheme is broken down to the word level. We redefine dialogue summarization as selecting higher-level *relevant* units, i.e. utterances, yielding much better inter-annotator agreement as originally reported by Zechner (.126), see Section 3.1.

5 Conclusions

We presented a system which adapts lexical chaining to summarize a new discourse type, i.e. conversational dialogues. Our research extends previous work on dialogue summarization by incorporating a broad coverage domain independent knowledge source and automatic word sense disambiguation. It is domain independent as opposed to approaches which aim at the deep semantic analysis and summary generation. Nevertheless, it is based on the semantic meaning of a dialogue as opposed to statistical approaches.

Additionally, we extend previous work on lexical chains by providing an extrinsic evaluation of the method against the human *gold standard* for the task of extracting the most relevant utterances. This relates the performance of the summarization model based on lexical chains to alternative models, e.g. *lead* and *random* baselines. Currently, our approach has been confined to the noun portion of WordNet, no predicates are considered and no anaphora resolution (about 10% of relevant utterances do not contain any nouns due to e.g. referential expressions) is performed.

Future research will, thus, aim at evaluating an extension to capture synsets of verbs and adjectives, as well. To achieve this goal, these will need to be conceptually integrated into the lexical chains algorithm,

which currently is optimized to consider noun relationships. Furthermore, the impact of using anaphora resolution, which frequently occurs in dialogues, on selection performance should be evaluated. Using the above mentioned additional computational steps, it will be possible to evaluate utterances such as "He lived there", which were annotated as relevant, but could not be captured by lexical chains because the utterance does not contain any noun.

Some other interesting points concern the definition of the summarization task used in this study as summarizing dialogue transcripts by selecting relevant utterances. So far, we did not address the issues of speech recognition errors and automatic utterance boundary detection. Those will entail imperfect input to the lexical chains algorithm, with which respect its robustness to errors has to be further investigated. Also, the unit of analysis has been defined as *utterance*. Replacing *utterance* with *adjacency pairs* (Galley et al., 2003) capturing information about the speaker interaction, such as question – answer, offer – acceptance can be considered in a new annotation study.

Topical changes and the dialogue structure represent further interesting challenges. While topics of the dialogue are reflected in *strong* lexical chains, the interplay with the resulting summary has to be analysed. Finally, this will provide important implications for summary presentation. E.g., the summary can be generated by selecting adjacency pairs referring to specific topics and converting those to reported speech complemented by a high-level description of the original dialogue.

Acknowledgments

This work has been funded by the Klaus Tschira Foundation. We thank the reviewers for their valuable comments concerning this work.

References

Jan Alexandersson and Peter Poller. 1998. Towards multilingual protocol generation. In *Proceedings of the International Workshop on Natural Language Generation 1998,* Niagara-On-The-Lake, Canada, August 1998, pages 198–207.

Regina Barzilay and Michael Elhadad. 1999. Using lexical chains for text summarization. In Inderjeet Mani and Mark T. Maybury, editors, *Advances in Automatic Text Summarization*, pages 111–121. Cambridge/MA, London/England: MIT Press.

Kedar Bellare, Anish Das Sharma, Atish Das Sharma, Navneet Loiwal, and Pushpak Bhattacharyya. 2004. Generic text summarization using WordNet. In *Language Resources Engineering Conference (LREC'2004),* Barcelona, May.

Adam L. Berger and Vibhu O. Mittal. 2000. OCELOT: a system for summarizing web pages. In *Research and Development in Information Retrieval*, pages 144–151.

Günes Erkan and Dragomir R. Radev. 2004. LexRank: Graph-based centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, Mass.

Michel Galley and Kathleen McKeown. 2003. Improving word sense disambiguation in lexical chaining. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-03)*, Mexico.

Michel Galley, Kathleen R. McKeown, Eric Fosler-Lussier, and Hongyan Jing. 2003. Discourse segmentation of multi-party conversation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics,* Sapporo, Japan, 7–12 July 2003, pages 562–569.

Iryna Gurevych and Michael Strube. 2004. Semantic similarity applied to spoken dialogue summarization. In *Proceedings of the 20th International Conference on Computational Linguistics,* Geneva, Switzerland, 23 – 27 August 2004, pages 764–770.

M. A. K. Halliday and Ruqaiya Hasan. 1976. *Cohesion in English*. London: Longman.

Graeme Hirst and David St-Onge. 1998. Lexical chains as representations of context for the detection and correction of malapropisms. In Christiane Fellbaum, editor, *WordNet: An electronic lexical database and some of its applications*, pages 305–332. Cambridge, MA: The MIT Press.

LDC. 1993. Switchboard. Linguistic Data Consortium. University of Pennsylvania, Philadelphia, Penn.

Kathleen R. McKeown, Jaques Robin, and Karen Kuckich. 1995. Generating concise natural language summaries. *Information Processing and Management*, 31(5):703–733.

Mandar Mitra, Amit Singhal, and Chris Buckley. 1997. Automatic text summarization by paragraph extraction. In *Proceedings of ACL Workshop on Intelligent Scalable Text Summarization,* Madrid, Spain, July 1997, pages 39–46.

Jane Morris and Graeme Hirst. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17(1):21–48.

Dragomir R. Radev, Simone Teufel, Horacio Saggion, Wai Lam, John Blitzer, Hong Qi, Arda Celebi, Danyu Liu, and Elliott Drabek. 2003. Evaluation challenges in large-scale document summarization. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics,* Sapporo, Japan, 7–12 July 2003, pages 375–382.

Norbert Reithinger, Michael Kipp, Ralf Engel, and Jan Alexandersson. 2000. Summarizing multilingual spoken negotiation dialogues. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics,* Hong Kong, 1–8 August 2000, pages 310–317.

H. Gregory Silber and Kathleen F. McCoy. 2002. Efficiently computed lexical chains as an intermediate representation for automatic text summarization. *Computational Linguistics*, 28(4):487–496.

Simone Teufel and Marc Moens. 2002. Summarizing scientific articles: Experiments with relevance and rhetorical status. *Computational Linguistics*, 28(4):409–445.

Klaus Zechner. 2002. Automatic summarization of open-domain multiparty dialogues in diverse genres. *Computational Linguistics*, 28(4):447–485.

Concept-centred summarisation: producing glossary entries for terms using summarisation methods

Le An Ha and **Constantin Orasan**
Research Group in Computational Linguistics
University of Wolverhampton
Stafford Street, Wolverhampton, WV1 1SB, UK
{L.A.Ha, C.Orasan}@wlv.ac.uk

Abstract

This paper describes a novel application of automatic summarisation methods for producing glossary entries. The proposed methodology is motivated by two observations: 1) glossary entries are increasingly used, especially on the Internet; and 2) information contained in a glossary entry is, in fact, a summary of information about the concept. From these two observations, we develop a method to automatically summarise contexts of a term[1] into a short text which can serve as a glossary entry. The method uses term-based and indicating phrase-based scoring methods to rank and select important contexts. A comparison of the similarity among the concept-centred summaries produced by our method, random baselines, and glossary entries produced by human shows that our summaries are better than the baselines used in this experiment.

1 Introduction

Terminology plays a vital role in scientific activities. Without proper terminology, the scientific community can no longer communicate efficiently and accurately. The current information era has created a problem: the number of new terms constantly introduced has already gone beyond our capacity to handle them. Sometimes, this also creates confusing situations because quite often it is difficult to know what the difference between two terms is. Searching for information on the Internet can be a solution, but often the amount of information about a specific term returned by search engines is far too large to be efficiently dealt with. In many cases this situation is made worse by the fact that many of the top retrieved documents do not offer the definition of the investigated concept.

In light of the above problems, two questions arise: 1) can NLP techniques be employed to extract most of the important information about a term/concept and then present it to the users, saving their time and effort? and 2) if yes, which techniques can be efficiently used?

One solution to these questions can be found in information extraction (IE), which manually or automatically builds templates for concepts, and then uses automatic processes to fill in the templates' slots. The problem is that in many cases it is difficult to know what a template should look like. Furthermore, current IE approaches heavily rely on named entities (person names, locations, times, organisation name etc.), thus it is very difficult to reuse those techniques for technical terms without major adjustments.

Recently, as a result of the introduction of definition questions into the question answering (QA) evaluation in TREC (trec.nist.gov), researchers have begun to build systems which can answer questions about concepts. These answers can also be considered glossary entries. Most of the existing systems rely on traditional definitional patterns (*such as*, *like*, *is a*) and/or resources such as Wordnet or a general encyclopaedia (TREC, 2003). While these patterns and resources are valuable, information that can be extracted using such approaches is often limited. For example, using the above question answering techniques, the answer for the question "*What is doxorubicin?*" could be very simple (e.g. "*doxorubicin is a chemotherapy drug*"). Other important facts related to *doxorubicin* such as: "*also known as adriamycin*", "*used to treat cancer*", "*orange-red*", "*given intravenously*" which should be included in an information-rich answer, often cannot be extracted by these systems.

Given the shortcomings of IE and QA approaches, this paper investigates alternative methods to extract glossary-like information about terms. Given that (multi-) document summarisation methods can already summarise information without relying on templates, this paper employs such methods for extracting information about a concept. The

[1] In this paper, terms are defined as linguistic labels of domain-specific concepts, regardless whether they are single-word or multi-word units.

summaries are then compared with glossary entries returned by Google.

The structure of the paper is as follows: Section 2 discusses glossaries and glossary entries. Section 3 explains why automatic summarisation is suitable for producing glossary entries. Section 4 presents in detail how the current summarisation system will be adjusted to produce the required output. Section 5 describes evaluation issues, settings, and results. Conclusions and future directions can be found in Section 6.

2 Glossaries and glossary entries

2.1 Glossaries

According to Collins Cobuild English Dictionary, (Cobuild, 1998), a glossary is "*an alphabetical list of special, usual or technical words or expressions, giving their meanings*". It is compiled by expert(s) in the field, and it can either serve as an additional resource for her/his books and lectures, or it can be used as a short reference point for readers. In the past, it was usually in the form of an appendix, and therefore it was often brief and considered informal. The Internet and hypertext era has changed the nature of glossaries, converting them into information-rich resources, featuring hyperlinks and multimedia explanation. This made glossaries become more and more widely used as sources of useful information about important concepts in a domain. The popularity of searching this type of information has led Google (www.google.com) to develop a search feature that takes advantage of existing human produced glossaries on the Internet to find definitions for certain technical terms.[2]

2.2 Glossary entries

A glossary usually is a list of entries from a domain. Each entry, in turn, contains a technical term and its meaning, definition and/or other additional information which is widely used to define and explain terms in that domain. For example, a glossary entry for *antioxidant* retrieved from Google is:

antioxidant:
Antioxidants are compounds that slow oxidation processes that degrade foods, fuels, rubber, plastic, and other materials. Antioxidants like butylated hydroxyanisole (BHA) are added to food to prevent fats from becoming rancid and to minimize decomposition of vitamins and essential fatty acids; they work by scavenging destructive free radicals from the food. (source:http://antoine.frostburg.edu/chem/senese/101/consumer/glossary.shtml)

In order to be useful a glossary entry should contain most important information about a concept. In this paper we argue that a glossary entry is, in fact, a summary of information about the concept (concept-centred summary). This fact is illustrated by the above example where the most important information about "antioxidant" is described, i.e. what it is (*a compound*), its function (*slowing the oxidation process*), its usage (*added to food to prevent....*), how it works (*by scavenging destructive free radicals from the food*). If we look at another glossary entry from a different domain (i.e. **B Lymphocytes (B cells)**, from www.cancerhelp.co.uk), we will notice that, while the specific information and the style are different, the overall structure stays the same: a summary about the concept. It still contains most important information about **B cells**, like what B cells are, what their functions are, how they work, etc.

B Lymphocytes (B cells)
Type of white blood cell. Lymphocytes make up a quarter to a third of the white blood cells. Then there are two types of lymphocytes, B and T cells. The B lymphocytes make antibodies in response to disease or anything the body recognises as foreign. The antibody response is part of the immune response. A cancer of the B lymphocytes is called a B cell Lymphoma. (source: cancerhelp.co.uk)

In this section, we identified two important facts about glossary entries, which are often overlooked, and provided arguments as to why we should (and could) be able to produce glossary entries automatically. Firstly, they are important for people to quickly understand a concept they come across in textual material, and secondly, they are actually summaries of information, and thus can be produced by specially developed automatic summarisation methods. Given their importance, one could argue that they should be manually produced, but this is very time-consuming and labour-intensive.

2.3 Indicating phrases in glossaries

Statistical analysis of glossaries can reveal which kinds of information are preferred to be included into a glossary entry. Methods, such as the one suggested by (Ha, 2003), can produce a list of words which are important in glossaries and which signal information which should be included in the glossary entries. These words are similar to indicating phrases used in automatic summarisation. For example, the following extract from a chemistry glossary: "acid: a compound **containing** detachable hydrogen

[2] To search for glossary entry of *carbon dioxide*, for example the user inputs *define: carbon dioxide* into Google, and a list of glossary entries for "*carbon dioxide*" will be returned.

ions; alloy: A mixture **containing** mostly metals; saturated fat: A lipid that **contains** no carbon-carbon double bonds. …" suggests that information signalled by the word "**contain**" is important in the domain of chemistry, and "**contain**" can be used as an indicating phrase. If we look at another glossary from a different domain, the appropriate indicating phrases are very different. Take the domain of cancer as an example, where the indicating phrases would be "cause", "affect", or "stop", instead of "contain", "produce", or "dissolve" as in the domain of chemistry. Details of extracted and used indicating phrases can be found in section 4.2.

3 From document-centred summarisation to concept-centred summarisation

In the previous section we mentioned the fact that one of various ways to obtain glossary entries is to use methods from automatic text summarisation. In this section, we introduce the notion of concept-centred summarisation as a way to produce glosses.[3]

We define the notion of *concept-centred summarisation* (CCS) as a summarisation method which produces a summary from sentences about a particular concept (e.g. concordance lines about a concept). Concept-centred summarisation should not be confused with *user-focused summarisation.* In user-focused summarisation sentences are selected on the basis of their appropriateness to the users' interests; some of the sentences from the source do not have anything to do with these interests making it easier to eliminate them. In contrast, in CCS all the sentences from the source are related to the concept of interest because they are concordance lines containing this concept. In light of this, methods from user-focused summarisation cannot be directly used here.

Concept-centred summarisation has also some similarities with multi-document summarisation because the concordance lines are extracted from several documents and issues such as redundant information in the source need to be tackled. However, as in the case of user-focused summarisation, a large number of the sentences in the documents to be summarised can be easily identified as not important by using different clues. This is not easy in concept-centred summarisation.

Because the input of the summariser is a single document, concept-centred summarisation can be considered an instance of single document summarisation. As aforementioned, in CCS all the sentences are linked to the concept and therefore sentence elimination methods cannot be easily employed. However, among the different summarisation methods mentioned above we believe that single document summarisation is the closest to the concept-centred summarisation and therefore the methods used to produce concept centred summaries are borrowed from single document summarisation.

In light of this, the methods which can be used in concept-centred summarisation are statistical methods where the importance of a sentence is determined by the statistical scores of the words constituting that sentence (Luhn, 1958; Zechner, 1996). As mentioned in section 2.3, indicating phrases are quite common in glossaries which suggests that they should be used in the summarisation process in a similar manner to the one proposed in (Paice, 1981). Discourse-based methods proved very effective in single document summarisation (Marcu, 1997; Azzam, Humphrey, & Gazauskas, 1999), but because the input of the concept-centred summariser is not a coherent piece of discourse, but a list of contexts for terms they cannot be used here. Because the same piece of information can be presented in several concordance lines, methods from multi-document summarisation which tackle this problem need to be employed. Clustering approaches and Maximal Marginal Relevance measures (Carbonell, Geng & Goldstein, 1997) can be used in order to minimise the quantity of redundant information present in a summary.

The concept-centred summarisation method employed here is explained in Section 4.2.

4 Settings for concept-centred summarisation

In this section we explain the data used to extract the documents to be summarised, the method employed to extract the indicating phrases and the settings used for the summarisation method.

4.1 Terms and concordance lines

Initially, the set of concepts used in this research consisted of 50 terms from the domain of chemistry and 50 terms from that of cancer extracted from relevant corpora. The terms are chosen randomly from the most frequent terms in each domain. Sentences which contain the selected terms are also extracted from these corpora. From the initial set of terms, we performed searches on Google for glossary entries, and found entries for 17 terms in the domain of cancer and 30 in chemistry. For this reason only 47 terms from a total of 100 were

[3] In this paper, we use "glosses" and "glossary entries" interchangeably.

included in the evaluation process. The corpus used for cancer contains texts collected from the cancerhelp.co.uk website, approximately 600000 words, whereas the chemistry one contains various elementary chemistry texts collected from the Internet (approximately 400000 words).

4.2 Extracting indicating phrases from resources

In order to extract indicating phrases from glossaries, we used a procedure similar to the one suggested by (Ha, 2003), where the usage of a specific verb in the glossary is compared to its usage in the BNC (Burnard, 1995). If statistical hypothesis testing (t-test in this case) indicates that the difference is significant, and the verb is heavily used in the glossary, it will be considered as an indicating phrase. Take the verb *contain* as an example. In the chemistry glossary, its normalised frequency is 0.02, compared to 0.00095 in BNC, and the t-test score of the difference is 10.23, showing that we can be over 99% confident that the difference is significant. On the basis of this the verb *contain* is considered an indicating phrase in chemistry domain.

Using this procedure, indicating phrases from a glossary of chemistry collected from (http://antoine.frostburg.edu/chem/senese/101/consumer/glossary.shtml) and from Cancerhelp's glossary have been extracted and used in the summarisation method described below. It should be noted that these glossaries, as training materials, are independent from the ones used in the evaluation (retrieved by Google).

4.3 Summarisation settings

In Section 3, we already indicated several possible ways to produce glosses from concordances. In this section we present our method to produce the summaries. As we already hinted in order to produce high quality summaries two problems need to be tackled. First, it is necessary to identify those sentences which contain the important information, and, secondly, it is necessary to minimise the amount of redundant information in the summaries.

The important sentences are identified by a combination of a term weighting method and one based on indicating phrases. The term weighting method relies on TF*IDF (Salton & McGill, 1983) to compute the importance of a word, the score of a sentence being obtained by adding the score of words constituting the sentence. In order to improve the results, before the score is calculated, all the words are reduced to their lemma. Indicating phrases are the second way to determine the importance of a sentence, the score of a sentence being the number of indicating phrases present in the sentence. The final score of the sentence is obtained by using a linear combination of the two normalised scores. The optimal weights were obtained through experiments (section 5.4).

In order to minimise the redundancy of the extract, every time a sentence is added to the extract, the score of each remaining sentences is penalised by the similarity between the remaining sentence and the extract. The motivation for this is the fact that if a sentence contains information which is already present in the extract it should not be included in the extract.

5 Evaluations

5.1 Evaluation schemes

Given that the glossary entries are concept-centred summaries, their evaluation is a difficult task. This difficulty was noticed in the case of the definition questions in TREC, a task similar to the one attempted here. When definition questions were first introduced in 2001, TREC accepted NIL answers when an answer to a question of this type could not be determined. In 2002, this type of question was removed. In 2003, a more complicated evaluation scheme was introduced. This scheme required the manual compilation of a set of target nuggets that were considered *vital* and *ok* for each question. After that, human judges would be asked to determine how many of those nuggets were retrieved in the automatic answers. The different evaluation methods used by TREC in such a short time reiterates how difficult is to evaluate this type of information.

The current evaluation method proposed in TREC requires a pre-compiled nugget set as well as humans to read the text and classify which of the answers has a correspondent in the set of nuggets. This type of evaluation is very time-consuming, and also very difficult to repeat many times. For this reason, in this paper we do not employ this methodology as the main evaluation. Instead, we decided to use a target based evaluation method where the output of our system is compared with a gold standard. Only after we identify the best parameters for the method, a manual, nugget-style evaluation on the output using the best parameters is performed. In the next section, we present how our gold standard was created, the target based evaluation method used and the evaluation results. We then perform a small scale nugget-style evaluation (section 5.5), to confirm the results of the automatic evaluation.

5.2 The gold standard

The gold standard used here was created using a semi-automatic method. For the terms chosen for the experiment, we used the *google:define* search feature to collect their definitions from well established glossaries.

When such a search was performed on Google, multiple glossary entries were returned. The use of multiple sources like this ensures that our gold standard is more objective. As mentioned before, searches on Google only returned glossary entries for 30 terms in chemistry and 17 terms for cancer, emphasising the domain glossaries and the necessity to find reliable methods to produce them. Some of the results from Google were too short, containing only one entry, as in the case of "*prostate cancer: cancer of the prostate gland*", and had to be reinforced by manually searching for glossary entries of these terms and manually inserting them into the gold standard.[4]

As a result of this process for each term we had a set of human produced definitions that contains the information which our summaries should contain and which constitutes our target summary.

5.3 The evaluation metric

In order to establish the amount of information present in a concept-centred summary, we compute the similarity between the summary and the gold standard produced for a term. Because quite often the gold standard contains redundant information, it is not possible to compute the simple cosine distance between the two texts. A solution to this problem could have been to identify those sentences in the gold standard which contain redundant information by using a clustering algorithm. We decided not to use this approach because clustering could have introduced errors reducing the quality of our gold standard. Instead we decided to make the evaluation an iterative process where we compare each sentence from the extract with each sentence in the gold standard, trying to identify a pair of sentences which has maximum cosine similarity. Once the pair is identified, the sentence from the gold standard is removed from it so it is not used in future comparisons. The reason for removing the sentence from the gold standard is that once the information in that sentence is identified in the extract another sentence from the extract which contains the same information should be penalised. Figure 1 presents the pseudocode for the algorithm.

[4] These data are collected in June 2004. Since then, Google has been indexing more glossaries.

```
T: set of sentences in the gold standard.
S: set of sentences in the [random,automatic]
summary.

Sim=0;
Foreach s in S
    Sim+=max sim (s,t) (t in T);
    Remove arg max sim (s,t) from T;
Endfor
return Sim/no sentences in the summary
```

Figure 1: Pseudocode for the calculation of similarity between summaries and the gold standard

5.4 Results

Using the evaluation method described in the previous section we compared the results of our concept-centred summarisation method with two different baselines. The first baseline randomly selected a specified number of concordance lines for a term. The justification for this baseline is that a human who wants to find information about a concept and who uses Google to obtain this information will receive more or less a similar summary. The second baseline uses clustering to determine a specified number of clusters which contain the most central information for the set of concordances. The distance between the concordances is computed using cosine distance.

From each term, we produced glossary entries of 5 sentences if the number of concordance lines for a term was below the average number of lines in the collection or 10 sentences otherwise. We decided to produce such short summaries because the human produced glossary entries are very short.

As mentioned in section 4.3, the decision to select a sentence is based on the weighted scores of three modules: TF*IDF scoring, indicating phrases and similarity between sentences and the extract. The weighting of each module was determined by testing different combinations for the parameters. The best set of parameters proved to be 1.5 for TF*IDF, 0.5 indicating phrases and 0 for the similarity between sentences. This result is rather surprising as research in automatic summarisation showed that usually indicating phrases have a great beneficial influence on the quality of the summary. The fact that the introduction of similarity does not help is another surprise which needs to be investigated further.

In addition to producing summaries using the described method, we determined a set of sentences from the input which has maximum similarity with the gold standard. The purpose of this exercise was to determine the best extract which can be produced from the concordances and therefore determine the upper limit of our summarisation method.

Using the best set of parameters, we compared our method with the two baselines and to the upper limit. According to our evaluation metric, the summaries produced by the proposed methodology are more similar to the target summaries than the baseline ones (see Table 1). For space reasons Table 1 presents only the results for the random baseline because the clustering one performed very poorly even in comparison with the random one. Our results also show that there is still room for improvement, as our automatic summaries are still significantly less similar to the target one, compared to the best case summaries, which consist of (5 or 10) sentences from term concordances most similar to target summaries.

	min			**max**			**avg**		
	r	**a**	**b**	**r**	**a**	**b**	**r**	**a**	**b**
chem.	0	0.02	0.21	0.27	0.37	0.63	0.14	0.19	0.41
cancer	0	0.05	0.08	0.36	0.44	0.73	0.17	0.23	0.44

Table 1: [**min**, **max**, and average (**avg**)] similarity between [random (**r**), automatic (**a**), best case (**b**)] and the target summaries.

We also performed hypothesis testing (t-test) to find out whether or not these differences between the random baselines and the automatic summaries, in term of similarity with the target summary, are significant. The results of this calculation also confirm the hypothesis that summaries produced by our methods are significantly more similar to the target summaries than the random ones (the calculated levels of confidence of this hypothesis are 99% and 95% in the case of chemistry and cancer information respectively).

5.5 Manual evaluation

The above evaluation is a fully automatic one, fast and inexpensive, but some critics may say that it does not evaluate the content of a glossary entry accurately. To confirm our proposed evaluation, we designed an additional nugget-style evaluation, in which, we first identify important facts (nuggets) from the target definitions used in the above experiment based on their frequency, then score our automatic glossary entries according to these nuggets to see how good our system is on the task of retrieving them.

For example, for the term: *doxorubixin*, google returns 6 definitions, among which, the fact that *doxorubixin* is "*used in the treatment of cancer*" is repeated 4 times, that it is "*known as adriamycin*" 3 times, "*antibiotic*" 2, "*chemotherapy drug*" 2; and other less frequent facts such as it is orange-red, or is given intravenously appear only once. From this, we consider "*used in the treatment of cancer*", "*known as adriamycin*", "*chemotherapy drug*", "*antibiotic*", "*orange-red*", "*given intravenously*" target nuggets for the glossary entry of "*doxorubixin*". Scores for each of these nuggets are assigned according to their frequency rank (which indicates their importance), the most frequent one has a score of 4, the next one 3, 2, 1, and the rest 0.5. In the "*doxorubixin*" case, the scores are assigned as: "*used in the treatment of cancer*": 4 (nugget a); "*known as adriamycin*": 3 (b); "*antibiotic/chemotherapy drug/drug*": 2 (c); and "*orange-red/given intravenously*" 0.5 (d)

	AS	RS	FS
adenocarcinoma	1	0	7
B-cell	4	7	8
biological therapy	6	3	10
blood clot	0	0	0
doxorubicin	7.5	4	9.5
growth factor	0	4	7.5
Leukaemia	4	0	7
liver cancer	4	3	6
lymph land	6	0	6
Melanoma	0	0	10
monoclonal antibody	0	0	2
prostate cancer	1	0	4
red blood cell	4	0	5.5
rhabdomyosarcoma	6.5	8	8
testicular cancer	0	0	0

Table 2: Nugget scores of automatic glossary entries and full concordances (AS: automatic glossary entry score, the best set of parameters used; RS: randomly selected sentence score; FS: FullCon score)

This process of identifying important nuggets is performed on 15[5] terms in the domain of cancers. Then for each automatic glossary entry produced, we score them according to the appearance of those nuggets. For example, an automatic glossary entry for *doxorubixin* containing nugget a, b and one of the ds will have a score of 7.5 (4+3+0.5). We also assigned scores to the full concordances of the term in the same manner, to see what the maximum score we can get is. In the case of *doxorubixin*, its full concordances contain all (a), (b), (c) and "given intravenously" nuggets, thus having the score of 9.5. To have a baseline figure, we randomly extract a number of sentences from the full concordance (the same number as the automatic glossary entry, which is either 5 or 10, see 5.4), and score those "random entries" using the same procedure.

[5] For "blood test" and "lung", we were unable to identify the important nuggets because of none of the nuggets appears more than twice in the target definitions.

It should be said that this is a lengthy process, as it involves manually looking through hundreds of concordance lines. The results are given in Table 2. The total score for the automatic glossary entries is 44; random entries 29; and for full concordances 90.5. This indicates that the method perform better than the baseline, and successfully identifies about half of the available (weighted) target nuggets. When the total number of sentences in these full concordances (1423) and in these automatic glossary entries (95) is taken into account, it becomes clear that the proposed method of producing glossary entries is attractive, as it reduces the number of sentences nearly 15 times, while retaining about half of the (weighted) important nuggets. These results are also compatible with the results produced by the automatic evaluation, indicating that our proposed metrics are valid.

6 Conclusion and future directions

This paper presents a novel method to produce glossary entries on the basis of terms' concordance lines. Glossary entries, which can be interpreted as concept-centred summaries, are very useful resources. In contrast to other methods which perform a similar task (i.e. TREC 2003), the method proposed here produces more informative entries and can be easily adapted to any domain.

Even though the results show that our method performs significantly better than the baselines, further investigation is necessary in order to improve the results more. The first problem which needs to be addressed is the coherence of the extracts. At present, the sentences in the extract are presented in the order of their scores. In the future, we are planning to investigate ways to reorganise these sentences in order to produce a more coherent text.

The scoring function can also be improved. Currently, the context for the terms is restricted to only one sentence. We intend to extend this context to several sentences, in order to be able to experiment with other methods and produce higher quality summaries. Indicating phrases are extracted in a simplified procedure, and will need more attention. This experiment also does not fully explore different combination among term weight, indicating weight and similarity (see 4.3), leaving room for further improvement.

The evaluation procedure we propose in the paper is an automatic, inexpensive and objective one, but we are yet to deal with the quality of the produced summaries in term of human judgement. In future we plan to experiment with other evaluation methods inspired by the ones used in text summarisation (e.g. ROUGE (Lin, 2004) or pyramid method (Nenkova & Passonneau, 2004)

References

(Azzam, Humphrey, & Gazauskas, 1999) Azzam, S., K. Humphrey, and R. Gazauskas. "Using coreference chains for text summarisation". In Amit Bagga, Brek Baldwin, and Sara Shelton, editors, *Coreference and Its Applications.* Pages 77 - - 84, University of Maryland, College Park, Maryland, USA, June 1999.

(Burnard, 1995) Burnard, L. *Users Reference Guide: British National Corpus Version 1.0*. Oxford: Oxford University Computing Services. 1995

(Carbonell ,Geng, & Goldstein. 1997) Carbonell, J., Y. Geng, and J. Goldstein. "Automated Query-Relevant Summarization and Diversity-Based Reranking". *In Proceedings of the IJCAI-97 Workshop on AI in Digital Libraries*, pages 12—19, 1997

(Ha, 2003) Ha, L. A. 2003. "Extracting important domain-specific concepts and relations from a glossary". In *Proceedings of the 6th CLUK Colloquium*, 6 - 7 January, pp. 49 – 56, Edinburgh, UK 2003

(Lin, 2004) Lin, C. Y. ROUGE: a Package for Automatic Evaluation of Summaries. In *Proceedings of the Workshop on Text Summarization Branches Out*, Barcelona, Spain, July 25 - 26, 2004

(Luhn, 1958) Luhn, H. P. "The automatic creation of literature abstracts". *IBM Journal of research and development*, 2(2):159 -- 165. 1958

(Marcu, 1997) Marcu, D "From discourse structures to text summaries". In Inderjeet Mani and Mark Maybury, editors, *Proceedings of the ACL/EACL '97 Workshop on Intelligent Scalable Text Summarization*, pages 82 -- 88, Madrid, Spain. 1997

(Nenkova & Passonneau, 2004) Nenkova A and R Passonneau. Evaluating Content Selection in Summarization: the Pyramid Method In *Proceedings of NAACL-HLT 2004*, May 2-7, pp. 145-152, Boston, Massachusetts, 2004

(Orasan, Mitkov & Hasler, 2003) Orasan, C., R. Mitkov, and L. Hasler.. "CAST: a Computer-Aided Summarisation Tool". In *Proceedings of Research Notes Sessions of EACL2003*, Budapest, Hungary. 2003

(Paice 1981) Paice, C. D. "The automatic generation of literature abstracts: an approach based on the identification of self-indicating phrases". In R. N. Oddy, C. J. Rijsbergen, and P. W. Williams, editors, *Information Retrieval Research*, pages 172 -- 191. London: Butterworths, 1981

(Salton & McGill 1983) Salton, G and M. J. McGill *Introduction to Modern Information Retrieval.* New York: McGraw-Hill, 1983

(Sinclair, 1998) Sinclair, J. ed.. *Collins Cobuild English Dictionary.* HarperCollins Publishers. 1998

(TREC 2003). *Text REtrieval Conference TREC 2003 Proceedings.* http://trec.nist.gov/pubs/trec12/-t12_proceedings.html

(Zechner, 1996) Zechner, K. 1996. "Fast generation of abstracts from general domain text corpora by extracting relevant sentences." In *COLING–96, The International Conference on Computational Linguistics.*, pp 986-989, Copenhagen, Denmark, 1996

Automation of Summary Evaluation by the Pyramid Method

Aaron Harnly* **Ani Nenkova*** **Rebecca Passonneau*** **Owen Rambow†**
∗ Department of Computer Science, † Center for Computational Learning Systems
Columbia University
New York, NY, USA
{aaron, ani, becky, rambow}@cs.columbia.edu

Abstract

The manual Pyramid method for summary evaluation, which focuses on the task of determining if a summary expresses the same content as a set of manual models, has shown sufficient promise that the Document Understanding Conference 2005 effort will make use of it. However, an automated approach would make the method far more useful for developers and evaluators of automated summarization systems. We present an experimental environment for testing automated evaluation of summaries, pre-annotated for shared information. We reduce the problem to a combination of similarity measure computation and clustering. The best results are achieved with a unigram overlap similarity measure and single-link clustering, which yields high correlation to manual pyramid scores (r=0.942, p=0.01), and shows better correlation than the n-gram overlap automatic approaches of the ROUGE system.

1 Introduction

Automatic summarization is usually evaluated through comparison to human summarization choices for the same texts.[1] Traditionally, the comparison is done through eliciting human judgments on content. When humans write short, abstractive summaries based on their reading of multiple documents, they select content they think belongs in a summary, and put it in their own words. While many words and phrases may be similar to those another human summarizer would employ, people can use different forms of the same words (inflectional or derivational variants), different word order, syntactic structure, and paraphrases. See for example the spans of words in bold below, coming from five different summaries of the same set of documents[2] about a Swissair crash off of Nova Scotia in 1998, all expressing the fact that the cause of the crash has not been determined.

S1 The cause of the Sept. 2, 1998 crash **has not been determined**.

S2 Investigators of a Swissair crash that killed 229 people off the coast of Nova Scotia searched for clues as to a cause but **but refrained from naming one**.

S3 **The cause has not been determined**, but there was extreme heat damage to the front of the aircraft and it is suspected that an in-flight entertainment system had electrical problems.

S4 **The specific cause of the tragedy was never determined**, but suspicions are that an electrical short caused a fire.

S5 Wreckage showed evidence of high heat and heat damaged wiring above the cockpit area but **investigators remain unsure of its cause**.

Note that while this example illustrates some overlap of 4-grams (*has not been determined*), much of the semantic similarity is obscured by alternate phrasings (*was never determined*, *remain unsure*) or by various forms of explicit anaphora (*the tragedy* instead of *the crash*, *its cause* instead of *the cause of the crash*, *naming one* instead of *naming a cause*).

A set of word spans which express similar meaning (such as those in bold in the example above) is referred to as a *Summary Content Unit* (SCU). After similar manual annotation of a complete set of reference summaries, the resulting set of SCUs is called a *pyramid*. A pyramid can be used to evaluate new summaries, following a method proposed by Nenkova & Passonneau (04). Each span of words in an SCU or in a summary to be evaluated is referred to as a *contributor* (and may have discontinuities). A new summary that is to be evaluated against the pyramid (or *peer summary*) will have some contributors that express content already represented in a pyramid, and perhaps some spans that do not. The Pyramid evaluation consists in identifying relevant contributors in the peer summary and matching them against SCUs in the pyramid. This match is used to assign a score, with SCUs that have more contributors providing a higher score. But the Pyramid method goes beyond telling us a score: because of the matching process, we also know which key ideas from the source documents the summary has chosen to include.

In this paper, we explore the automation of this evaluation approach. Since the number of possible

[1] We would like to thank Chin-Yew Lin for helpful comments on an earlier version of this paper. This work was supported by the National Science Foundation under the KDD program. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

[2] These sentences are from summaries written by university students for DUC 2003 set *D30016*.

candidate contributor sets is exponential in the number of words in the sentence, we use dynamic programming to find an optimal candidate contributor set of a summary based on different clustering methods and similarity metrics. Our results indicate that using automatic Pyramid scoring leads to better correlation with human Pyramid scoring over the use of the n-gram overlap automatic evaluation metric ROUGE.

2 Related Work

The development of automated or semi-automated methods for evaluating content selection in summarization has recently been an area of active research. A completely manual evaluation method was used in the Document Understanding Conferences (DUC) in 2001–2003. The method involved human judgments about how much of the content of a *single* model summary is expressed in a new peer summary. Analysis of the DUC evaluations results revealed some weaknesses— the stability of human judgments of "information overlap" (Lin & Hovy 02), the coarse-grained and subjective nature of the judgments required (Halteren & Teufel 03; Nenkova & Passonneau 04) , and the use of single reference summaries, despite the observation that summaries with different content can be equally good (Nenkova & Passonneau 04). The "factoid" (Halteren & Teufel 03) and manual Pyramid annotation methods have been proposed to address these limitations.

At the same time, several automated methods have been proposed to address the cost/time issues imposed by manual annotation, most notably the ROUGE family of ngram-overlap measures (Saggion *et al.* 02; Lin & Hovy 03; Pastra & Saggion 03). All of these methods rely on the comparison of peer summaries to one or more human-written reference summaries. The summarization task, by definition, demands high compactness relative to its source documents. Paraphrase and synonymy are expected to be used to achieve the desired compactness, and indeed we find mostly 1- or 2-grams matching between source text and abstractive multi-document summaries (Banko & Vanderwende 04).

3 The Pyramid Method

The pyramid method addresses the following characteristics of abstractive summaries that present a challenge for evaluation: that summaries written by equally skilled writers are highly likely to have some overlap in content, and highly likely to have some content that is unique to each summary; and that when different summaries express the same content, the wording can vary in unpredictable ways. The pyramid method adopts the following strategies:

- We explicitly assume that multiple reference summaries are required to evaluate a peer summary.
- A pyramid is created by identifying SCUs, i.e., sets of contributors (text fragments) in the reference summaries that express approximately the same meaning.
- The number of contributors in an SCU is the frequency with which an SCU was expressed in the pool of model summaries. This frequency is used to weight the importance of the SCU.

A pyramid, or set of SCUs, tends to have very few SCUs with high weights, increasing numbers of SCUs as the weights decrease, and finally, a very large number of SCUs with weights of one or two. It is this fact that gives the method its name.

When a peer summary is evaluated against the pyramid, its content is matched against SCUS to identify *candidate contributors*, which are fragments of text that express roughly the same meaning as an SCU in the pyramid, and there will typically be remaining fragments that have no match. A candidate contributor which has the same meaning as the contributors in an SCU in the pyramid are rewarded with the score n, where n is the weight of the matching SCU in the pyramid. Candidate contributors with no match are assigned weight zero. The score of the peer summary is the ratio between the sum of weights of its candidate contributors and the sum of weights of a optimal summary of the same size. The optimal summary is defined as the informationally ideal summary, that expresses the most highly weighted pyramid SCUs.

4 Automation: Motivation and Algorithms

There are two tasks involved in pyramid evaluation: creating a pyramid by annotating model summaries, and evaluating a new summary (peer) against a pyramid. Ideally, an automated evaluation component would address both tasks. However, the task of creating a pyramid is far more complex than the task of scoring a new summary against existing (hand-created) pyramid, and the automated scoring component is useful when doing a large amount of evaluation (of multiple summarizers, or different versions of the same summarizer). Therefore, we decided to explore first the automation of scoring a new summary

against an extant, human-produced pyramid. We anticipate that what we learn in this process will apply when we turn to automating pyramid construction.

Our algorithm consists of four steps.

Enumerate Enumerate all candidate contributors (contiguous phrases) in each sentence of the peer summary.

Match For each candidate contributor, find the most similar SCU in the pyramid. In the process, the similarity between the candidate contributor and all pyramid SCUs is computed.

Select From the set of candidate contributors, find a covering, disjoint set of contributors that have maximum overall similarity with the pyramid.

Score Calculate the pyramid score for the summary, using the chosen contributors and their SCU weights.

For example, the enumeration of all candidate contributors for a peer summary sentence ABC might be $\{A, B, C, AB, AC, BC, ABC\}$, where A, B, and C are words. In the **Match** step, each member of this set will be assigned a score, based on its similarity with pyramid SCUs. In the **Select** step, the overall optimal subset of candidates will be chosen, for example $\{A, BC\}$ and A and BC will also be mapped to SCUs in the pyramid. In the **Score** step, the pyramid summary score for the peer based on the SCU assignment from the previous step will be computed. We next discuss the four steps in detail.

4.1 Enumeration of candidate contributors

What set of text fragments could be contributors in an SCU? We have chosen to consider all contiguous spans of words that do not cross sentence boundaries. Without the restriction that the candidate contributor spans be contiguous spans of words, an n-word sentence would yield 2^n possible candidate contributors consisting of all possible subsets of words from the original sentence. But imposing the contiguity requirement on candidate contributors, the size of the set of all candidate contributors is reduced to $\frac{n(n-1)}{2} = 1+2+...+n$ since there are $(n+1-k)$ contributors of length k. Note that this restriction to contiguous spans of words is a departure from the manual pyramid method, which permits, in limited circumstances, noncontiguous words to comprise a contributor.

4.2 Matching of contributors to SCUs

Next, we match each candidate contributor $\hat{c}$ to the SCU $C = \{c_1, c_2, ..., c_n\}$ with which it shares the most meaning (c_i are the contributors of C and express the same meaning, possibly with a different wording). The degree of shared meaning is measured using a similarity metric `set_sim` between the candidate contributor and a pyramid SCU:

$$\texttt{set\_sim}(\hat{c}, C) = \texttt{combine}_{c_i \in C}(\texttt{span\_sim}(\hat{c}, c_i))$$

`set_sim` is defined in terms of a function `span_sim` which expresses the similarity between two text spans, and the function `combine`, which, given scores for the similarity between the candidate contributor and the contributors from the pyramid, returns a single score. Thus, we must choose the two functions `span_sim` and `combine`, and these choices represent an important part of our research. Note that the **Matching** step can be seen as a clustering problem. The SCUs in the gold-standard pyramid can be viewed as clusters of contributors. The task is to merge the candidate contributor (viewed as a cluster with a single element) to the most appropriate SCU cluster in the pyramid.

We explore several choices for `combine`. In the single-link method, the overall similarity between the candidate contributor and an SCU is the maximum of the pairwise span similarity between their contributors, i.e., $\texttt{combine} = \texttt{max}$. In the average-link method, the overall similarity is the mean of pairwise similarity, and $\texttt{combine} = \texttt{mean}$. In the complete-link method, the overall similarity is the minimum of the pairwise similarity, and $\texttt{combine} = \texttt{min}$.

Many alternatives for the pairwise similarity metric `span_sim` between contributors are possible. We experimented with simple cosine similarity, cosine similarity with TF*IDF weighting, unigram overlap, bigram overlap, and word-wise edit distance.

Currently, we assign each contributor to its "best fit" SCU. It may be that retaining an n-best list would allow the next step (**Select**) to choose a disjoint set of contributors.

4.3 Selecting a covering, disjoint set of possible contributors

Once all candidate contributors have been matched to their most-similar SCUs, the similarity scores can be used to find an optimal subset of the candidate SCUs. As in the manual pyramid method, we have chosen to require a covering, disjoint set of contributors, i.e. each word of a peer summary should belong to one

of the final contributors, and no word can belong to more than one contributor. There are $O(2^n)$ possible such sets for sentences of n words; to avoid exponential runtime, we use a two-dimensional dynamic programing algorithm, which selects the best contributor set for each span of words between the ith and jth words of a sentence, eventually producing a preferred covering for the entire sentence. The scoring method chooses the contributor set that produces the highest total overall similarity score between the chosen contributors and their SCUs. The score for the best covering for a span (i, j) in a sentence is the maximum of the sums of the scores of the subsequences (i, k) and $(k+1, j)$ for $k = i, \ldots, j-1$, and of the direct score for the span (i, j) itself.

Consider a brief example with a sentence beginning *In 1998 two Libyans* .... Initially the span (1, 1) is considered, and hence the optimal contributor set is simply the word *In*. The overall score for this span is simply the similarity score between *In* and its best-match SCU. Next, the spans (1,2) and (2,2) are considered. The optimal contributor set for the span (2,2) is simply the word *1998*. The dynamic programming comes into play in the next span, (1,2). The optimal set of contributors for the span (1,2) can be either the contributor *In 1998* (i.e., the span (1,2)), or the union of each of the optimal sets for the spans (1,1) and (2,2), i.e. *In* and *1998*. Suppose that the single-contributor set *In 1998* produces a better score. We record this fact and need not examine the span (1,2) again, even as this span participates in larger spans. Then we consider the spans (1,3), (2,3), and (3,3). The process continues in typical dynamic programming fashion until an optimal set of contributors for the span (1,n) is chosen.

4.4 Score

Finally, the selected set of contributors are scored as in the manual pyramid method. The sum of the weights of all SCUs in the peer summary (assigned in the preceding step) is normalized by the maximum sum possible for an "ideal" summary which contains as many high-weight SCUs as possible in a summary of the same size (see section 3). This gives a normalized score between 0 and 1.

5 Evaluation

5.1 Comparing Human Pyramid Score to Automated Pyramid Score

The goal of this evaluation is to determine the correlation between human Pyramid scores and our automatically obtained Pyramid scores. It is not the object of this paper to show that the human Pyramid scores correlate with other measures of summary quality; see (Nenkova & Passonneau 04) for details. Because of methodological issues in averaging correlations, we use for our correlation study not the scores for individual summaries, but instead for human summarizers. This evaluation mimics the standard case where we wish to evaluate (or rank) several summarization systems which have produced summaries for the same document sets.

For our evaluation, we used the three sets of data from (Nenkova & Passonneau 04). The three document sets are from the DUC'03 test set. For each document set, we have 10 summaries, each manually annotated for content units. We chose to evaluate six human summarizers from whom we had summaries for each of the three sets (the other summarizers did not summarize all three sets). These summarizers are Columbia University graduate students in the School of Journalism, who were compensated for their work, and who followed the guidelines for summary creation used in DUC.

We evaluated each summary by one of the six Columbia summarizers against a pyramid consisting of the remaining nine summaries for that document set. This gives us 18 manual and 18 automated scores. To obtain an overall summarizer performance score, we calculated the mean human Pyramid score and mean automated Pyramid score for each summarizer across the three sets, giving us six scores for each scoring method (human or automated). Then we computed the correlation between the automatic scores and the original Pyramid scores. Both Pearson's correlation (a measure of the linear association between the two types of score), and Spearman's rank correlation (a correlation based only on the rank of the scores, not their value) were computed. The Pearson correlation is a useful measure of whether the automatic scores could be used as drop-in replacements for human scores. Since the usual ultimate goal of summary evaluation is to compare summarization systems, and hence relative rank rather than raw score is more important, the Spearman rank correlation is arguably a better measure of whether the automated evaluation system can produce similar judgments as human scorers.

Figure 1 shows the main results. The upper table is the Pearson correlation, the lower table the Spearman rank correlation. The rows are labeled with the `span_sim` metric used to compute the sim-

	Min	Mean	Max
Unigram Overlap	**0.942***	0.866*	0.026
Simple Cosine	0.890*	0.751*	−0.052
Edit Distance	0.941*	0.551	−0.1478
Bigram Overlap	−0.119	−0.085	0.529
Cosine-TF*IDF	0.268	0.717	−0.074

	Min	Mean	Max
Unigram Overlap	**0.886***	0.714	−0.029
Simple Cosine	0.886*	0.257	−0.200
Edit Distance	0.886*	0.371	−0.143
Bigram Overlap	−0.200	−0.086	0.428
Cosine-TF*IDF	0.200	0.771	0.086

Figure 1: Pearson (above) and Spearman (below) correlation between automatically scored summary and fully manual scores, for different `scan_sim` functions (rows) and `combine` functions (columns). Starred cells (*) have a p-value ≤ 0.05, single-tailed.

Stop words list	Yes	No
Words unchanged	0.843*	0.726*
Lowercased	0.903*	0.594
Lemmatized	**0.942***	0.819*

Stop words list	Yes	No
Words unchanged	**0.943***	0.714
Lowercased	0.829*	0.371
Lemmatized	0.886*	0.600

Figure 2: Pearson (above) and Spearman (below) correlation for different ways of preparing data. All results in Figure 1 are for Lemmatized, Using Stop Words List. All Results here are for Min, Unigram Overlap. Starred cells (*) have a p-value ≤ 0.05, single-tailed.

ilarity between a candidate span is to a contributor. The columns are labeled with the different `combine` functions, which, as discussed above, correspond to choosing a method in clustering. All figures assume the use of a stop list and a lemmatizer; we return to these parameters below. We have boldfaced the best results, which for both types of correlation is a unigram overlap `span_sim` metric, with the `combine` function being the minimum.

We make the following interpretative observations about the results in Figure 1. We find that for different `combine` methods, different `span_sim` metrics are better. The unigram overlap metric counts the number of shared words between two spans, but abstracts completely from word order. By using the minimum `combine` function (i.e., the single-link clustering method), we require that *all* contributors for a particular SCU in the pyramid show some word overlap with the candidate. Thus, we want a similarity metric which imposes as few constraints as possible, which is the unigram metric. (In fact, we fail to identify the correct SCU if there is a contributor which is a radical paraphrase, to the point of having no overlapping words at all.) On the other hand, for the maximum `combine` function, we require only one contributor to match, so we expect this match to be more constrained. Indeed, for the maximum `combine` function, the best overlap metric is the cosine-TF*IDF metric. In contrast, for minimum and mean, the cosine-TF*IDF is the worst performing.

The lower table in Figure 1 shows the Spearman rank correlation. We see that the results are similar to the Pearson correlation, but with some exceptions, especially for the maximum and mean `combine` functions.

5.2 Preprocessing the Data

Further, we examine how the different ways to prepare the data impacts results. We consider two questions:

- Should we use a list of stop words, which we exclude from both SCU contributors and candidate sentences before we apply the similarity metrics?
- Should we normalize words by either lemmatizing them, or lowercasing them, or should we leave them unchanged?

To investigate these issues, we used the best performing combination `span_sim` metric and `combine` function, namely unigram overlap and minimum. We then varied the two new parameters. The results are shown in Figure 2. As expected, the use of a stop word list helps, since it eliminates noise caused by matches on function words and other content-free or common words. At the same time, we find that we get a slight improvement by lemmatizing words, but only for the Pearson correlation. For the Spearman (rank) correlation, keeping the words unchanged results in a higher correlation, a difference for which we have no explanation at present. Overall, our best results are 0.942 for the Pearson and 0.943 for the Spearman correlations (both significant with $p < 0.05$.

5.3 Comparison with ROUGE

We compare our results with those achieved by the ROUGE system. We report recall and precision scores for ROUGE-1 (the most used metric until 2005), ROUGE-2 and ROUGE-SU4 (which are used for the

	Recall	Precision
ROUGE-1	**0.805**	0.242*
ROUGE-2	0.552*	0.212*
ROUGE-SU4	0.572*	0.176*
Automatic Pyramid	0.942	

	Recall	Precision
ROUGE-1	0.600*	0.543*
ROUGE-2	0.543*	0.371*
ROUGE-SU4	0.314*	0.118*
Automatic Pyramid	0.943	

Figure 3: Summary of results: Pearson (above) and Spearman (below) correlations between manual pyramid scores and six different versions of ROUGE. Starred cells (*) are significantly different from corresponding correlations between the manual and automated Pyramid methods at a p-value ≤ 0.05, single-tailed.

DUC'05 evaluation). ROUGE was originally developed as a recall metric — in fact, its name is an acronym for Recall-Oriented Understudy for Gisting Evaluation. The precision version of ROUGE was added in 2005. The Pyramid evaluation has characteristics of both a precision measure (as the score is a function of the size of the summary) and of a recall measure (as the score is also a function of the weights of the optimal SCUs). The settings we used for all ROUGE experiments were exactly the ones used in DUC.[3]

Figure 3 compares our performance to ROUGE. We use three ROUGE variants: unigram overlap (ROUGE-1), bigram overlap (ROUGE-2), and skip bigram and unigram combination(ROUGE-SU4), where a skip bigram is any pair of words in their sentence order, with up to four intervening words in between. We report both recall and precision scores for the ROUGEs. We see that the automatic Pyramid evaluation has higher Pearson and Spearman correlation than all three ROUGE scores. The difference in correlation between the automatic Pyramid and the ROUGE scores is statistically significant ($p \leq 0.05$) for all cases except the Pearson correlation between the automatic Pyramid (0.942) and ROUGE-1 recall score (0.805), which is not statistically significant ($p = 0.129$). We expect that more data will allow us to establish statistical significance for the remaining comparison as well.[4]

[3]ROUGE-1.5.5.pl -n 2 -x -m -2 4 -u -c 95 -r 1000 -f A -p 0.5 -t 0 -d

[4]We also performed experiments with ROUGE with stopwords removed, which did not lead to a consistent improvement in correlations.

Note that for ROUGE, as for our automatic evaluation, unigrams performs best, followed by the skip bigrams/unigrams combination, followed by the bigrams. The differences among the ROUGE scores are considerable. Experiments on the correlation between ROUGE and the DUC manual evaluation showed that for both DUC'02 and DUC'03 hundred words summaries, the best correlation was achieved for bigram matches, with stopwords removed (Lin 04). We have no immediate explanation for our different result (favoring unigrams), other than to point out that the human evaluations (to which correlation is being measured) differ.

6 Discussion and Future Work

We consider the work reported in this paper to be a foundation for future work. In this section, we discuss some possible extensions of this approach.

6.1 Tree-Based Approaches

We initially explored a more linguistically motivated order of operations, in which the peer summary was first broken into text fragments corresponding to subtrees in a dependency parse of the sentence, using a machine learning approach with human-annotated summaries as training data. The use of dependency tree representations was motivated by the observation that the overwhelming majority of SCU contributors chosen by humans are in a single subtree of a dependency tree, in particular, including constituents that are discontinuous in surface structure. For example, in *The report, later published by the Times, cost the government half a million*, the *later published by the Times* may be a separate contributor, making *The report . . . cost the government half a million* discontinuous, but only in the linear order, not in the tree. In addition, we hoped to develop a feature set that would take advantage of dependency relations to express more of the semantics of a contributor than is given by the actual word sequence; e.g., that a temporal locative PP like *on November 9* gives the date of the event described in the governing phrase.

The approach uses a set of features extracted from a dependency tree of each sentence to machine-learn the binary classifier of whether to "clip" each subtree into a separate contributor. However, this method does not yield contributors that are very similar to those chosen by human annotators The likely reason for the poor performance is that this purely local and syntactic selection of contributors does not capture the key decision in SCU contributor selection, which is

whether a possible contributor expresses roughly the same meaning as other contributors from reference summaries. Therefore we rejected the purely syntactic method of contributor selection in favor of the above set of steps, which performs an optimization over the whole summary.

A natural consideration is extending the dynamic programming approach proposed here to trees. We would enumerate all subtrees of a dependency parse as possible contributors, and compare them to trees derived from the contributors in the pyramid. Unfortunately, this approach would also produce exponentially many candidate contributors. A solution may be to use dynamic programming in the matching itself (and not just in the selection of a covering, as we do now), so that when we match a larger tree, we base the results on the matches of its constituent trees.

6.2 Improving the Matching

For the span distance function `span_sim`, we can consider variants such as word-wise edit distance weighted by TF*IDF scores, centroid measures, and so on. Even more sophisticated possibilities include a tree edit distance of a dependency parse of the contributors, or incorporating syntactic features in other ways, for example favoring contributors that are bounded on either side by a mother and child in the dependency tree. (In this proposal, the contributors are still defined as word sequences but are then parsed, unlike the tree-based approaches proposed in Section 6.1, where contributors are defined in terms of tree structure.)

Another possible strategy is to measure similarity of the target contributor to a derived template contributor in the pyramid that incorporates elements of each member contributor. Or, borrowing from computational biology, one can do a multiple sequence alignment of the peer candidate contributor to the entire set of member contributors.

For the score combination function `combine`, we found that the single-link method produces SCU assignments with highest accuracy compared to human judgments; but this choice can be revisited as we choose different similarity metrics (`span_sim`) in that there is likely to be a trade-off between the features and weightings associated with a specific metric, and the way pairwise similarity scores of a candidate with each SCU contributor are combined.

6.3 Score Stability

The manual pyramid method has been found to elicit stable rankings of individual summaries when five or more reference summaries are used (Nenkova & Passonneau 04). It would be interesting to discover whether the automatic Pyramid scoring method shows similar behavior, and to investigate system rankings from the automatic Pyramid method across more document sets, to explore whether stable single-summary scores yield stable system ranking across many document sets, and to determine whether even unstable single-summary scores could yield stable rankings over a sufficient number of document sets.

7 Conclusion

We have presented a method for automation of summary evaluation that incorporates the insights of the manual Pyramid method. We believe the method, in addition to correlating better with human Pyramid scores on our test set, offers some advantages over the automated ROUGE methods, as it is a more general framework that takes human insight into meaning into account, and that can incorporate different ways of measuring similarity, not simply n-grams.

References

(Banko & Vanderwende 04) Michele Banko and Lucy Vanderwende. Using n-grams to understand the nature of summaries. In *Proceedings of HLT/NAACL'04*, 2004.

(Halteren & Teufel 03) Hans Halteren and Simone Teufel. Examining the consensus between human summaries: initial experiments with factoid analysis. In *HLT-NAACL DUC Workshop*, 2003.

(Lin & Hovy 02) Chin-Yew Lin and Eduard Hovy. Manual and automatic evaluation of summaries. In *Proceedings of the Workshop on Automatic Summarization, post conference workshop of ACL 2002*, 2002.

(Lin & Hovy 03) Chin-Yew Lin and Eduard Hovy. Automatic evaluation of summaries using n-gram co-occurance statistics. In *Proceedings of HLT-NAACL 2003*, 2003.

(Lin 04) Chin-Yew Lin. Rouge: a package for automatic evaluation of summaries. In *Proceedings of the Workshop in Text Summarization, ACL'04*, 2004.

(Nenkova & Passonneau 04) Ani Nenkova and Rebecca Passonneau. Evaluating content selection in summarization: The pyramid method. In *Proceedings of HLT/NAACL 2004*, 2004.

(Pastra & Saggion 03) Katerina Pastra and Horacio Saggion. Colouring summaries bleu. In *EACL 2003*, 2003.

(Saggion *et al.* 02) H. Saggion, D. Radev, S. Teufel, and W. Lam. Meta-evaluation of summaries in a cross-lingual environment using content-based metrics. In *International Conference on Computational Linguistics (COLING'02)*, 2002.

Word Sense Disambiguation Using Co-Occurrence Statistics on Random Labels

Martin Hassel
KTH KOD
Royal Institute of Technology
SE-100 44 Stockholm
Sweden
xmartin@kth.se

Abstract

In this paper we present experiments using Random Indexing for "query expansion" in Word Sense Disambiguation. Random Indexing is an efficient, scalable and incremental latent semantic indexing method somewhat akin to LSA, and has in these experiments shown promising results on a small test set for Swedish with an accuracy up to 80% with relatively little training data. We also compare it to results obtained when applying a Naïve Bayes classifier to the same training and data sets, retrieving a maximum accuracy of 56%.

1 Introduction

A given word can have several *senses*. For example, the word "hot" can mean a high temperature, fiery, excited, eager, spicy or simply incredibly good-looking. A word sense is thus a given *meaning* of a word. While humans display an uncanny ability to select the appropriate meaning when hearing such words in context, natural language applications do seldom fare as well.

The automatic disambiguation of word senses has been an interest and concern since the earliest days of computer treatment of language in the 1950s. Word sense disambiguation (WSD) is an "intermediate task" (Wilks & Stevenson 96), which is not an end in itself, but rather is necessary at one level or another to accomplish many natural language processing tasks. It is obviously essential for language understanding applications, such as message understanding and man-machine communication; and is at least helpful for applications whose aim is not language understanding, e.g. machine translation, information retrieval and hypertext navigation, content and thematic analysis, grammatical analysis, speech processing, and text processing.

Since the senses being discriminated between all are realized with the same lexical sequence, disambiguation work traditionally involves matching the context of the instance of the word to be disambiguated with either information from an external knowledge source (knowledgedriven WSD), or information about the contexts of previously disambiguated instances of the word derived from corpora (data-driven or corpus-based WSD). Any of a variety of association methods is used to determine the best match between the current context and one of these sources of information, in order to assign a sense to each word occurrence (Ide & Véronis 98).

The context is often divided into microcontext and topical context. The microcontext generally means a context of a few words up to an entire sentence. Early findings (Kaplan 50) suggest that ± 2 word contexts are highly reliable, and that even ± 1 contexts are reliable in as much as 8 out of 10 cases. In the microcontext it is also recognized that the distance to the keyword, the collocations as well as the syntactic relations are significant for local word sense disambiguation. Topical context usually means a window of several sentences or more. While local context can account for most of the ambiguities, topical context often can improve the result (Lindén 05).

2 Word Spaces and Random Indexing

Word space models, most notably Latent Semantic Analysis/Indexing, enjoy considerable attention in current research on computational semantics. Since its introduction in 1990 it has more or less spawned an entire research field with a wide range of word space models as a result, and numerous publications reporting exceptional results in many different tasks, such as information retrieval, various semantic knowledge tests (for example TOEFL[1]), text categorization and also word sense disambiguation.

The general idea behind word space models is to use statistics on word distributions in order to generate a high-dimensional vector space.

[1] Test of English as a Foreign Language

In this vector space the words are represented by context vectors whose relative directions are assumed to indicate semantic similarity. The basis of this assumption is the *distributional hypothesis* (Harris 85), according to which words that occur in similar contexts also tend to have similar properties (meanings/functions). From this follows that if we repeatedly observe two words in the same (or very similar) contexts, then it's not too far fetched to assume that they also mean similar things.

In these experiments with word sense disambiguation we have used the Random Indexing (Kanerva *et al.* 00; Sahlgren 05) word space approach, which presents an efficient, scalable and inherently incremental alternative to standard word space methods. As an alternative to LSA-like models that first construct a huge co-occurrence matrix and then use a separate dimension reduction phase, Random Indexing instead accumulates context vectors on-the-fly based on the occurrence of words (tokens) in contexts, without a specific need of a separate dimension reduction phase. This technique can readily be used with any type of linguistic context and can be used to index using a more traditional bag-of-tokens approach as well as using a sliding context window capturing sequential relations between tokens. These tokens can be the word simply represented by its lexical string as well as its lemma, or more elaborate approaches utilizing tagging, chunking, parsing or other linguistic units can be employed.

The construction of context vectors using Random Indexing is perhaps easiest described as a two-step operation (Sahlgren 05). First, each context (e.g. each document, paragraph, word etc) in the data is assigned a unique and randomly generated label. These labels can be viewed as sparse, high-dimensional, and ternary vectors. This means that their dimensionality (d) usually is chosen to be in the range of a couple of hundred up to several thousands, depending of the size and redundancy of the data, and that they consist of a very small number (usually about 1-2%) of randomly distributed +1s and -1s, with the rest of the elements of the vectors set to 0.

Next, the actual context vectors are produced by scanning through the text and each time a token w occurs in a context (e.g. in a document or paragraph, or within a sliding context window), that context's d-dimensional random label is added to the context vector for the token w. Thus, when using a sliding context window, all tokens that appear within the context window contribute (to some degree) with its random label to w's context vector. Words are in this way effectively represented by d-dimensional context vectors that are the sum of the random labels of the co-occurring words.

In practice the random labels are usually represented in more efficient ways than extremely sparse vectors and are generated on-the-fly during the context vector indexing whenever a never before seen token is detected in the context. When using a sliding context window it is also common to use some kind of distance weighting in order to give more weight to tokens closer in context.

3 The Task at Hand

The task chosen for these experiments concerns word sense disambiguation, in our case the construction of a computer program capable of discriminating three different senses of the Swedish word form "resa", one *noun* sense and two *verb* senses, exemplified in the following sentence:

1 Hon vill göra en *resa*. [noun]
She wants to make a *journey*.
2 Hon vill *resa* till USA. [verb1]
She wants to *travel* to USA.
3 Hon vill *resa* en staty. [verb2]
She wants to *raise* a statue.

Reflexive uses of the verb "resa" meaning "rise, stand up" are considered instances of the third sense.

Extending the principles behind the distributional hypothesis and Random Indexing to the field of word sense disambiguation we can, as well as assuming that different words in similar contexts mean similar things, also assume that the same word in different contexts likewise means different things. The hypothesis here is therefore that if we model the different senses by the co-occurrence of "concepts", here represented by context vectors produced by means of Random Indexing, then we should not only be able distinguish the different senses, but also to some extent overcome the problem of sparse data that

here would hamper a traditional Naïve Bayesian approach.

4 Data and Baselines

As "training data" for the Random Indexing step approximately 900.000 words from the Swedish Parole corpus (Gellerstam *et al.* 00) were used together with the approximately 90.000 words from the WSD training set and 20.000 words from the test set; both latter slices taken from the Stockholm-Umeå Corpus, SUC (Ejerhed *et al.* 92), of Swedish texts. The WSD training and test sets where sense tagged, each with one of the three different senses for the word in question, by hand by two persons and then compared, showing basically no conflicting tags to resolve. The tagging resulted in 108 training examples with the following distribution:

sense 1 : 45 instances
sense 2 : 43 instances
sense 3 : 20 instances

The testing data set was similarly annotated in order to be able to automate the scoring of the results. This resulted in 25 test instances with the following distribution:

sense 1 : 7 instances
sense 2 : 7 instances
sense 3 : 11 instances

At this point we can easily deduce two simple baselines to compare our results to. One oft-used baseline is to randomly choose one of the possible alternatives in each instance. Since we in each instance have exactly three senses to choose from this gives us a baseline of 33% correct.

Having tagged the WSD training data we can also inspect the frequency of each of the three senses, and note that sense 1 is by a margin the most common. A promising baseline would thus be to always assign sense 1 to each instance in the WSD test data. However, after tagging of the test data we can establish that this only gives us a baseline of 28%, which is less than random. We are of course aware of that this discrepancy in sense frequency probably is due to the relative smallness of our training and test sets, which increases the risk of unbalanced as well as sparse data.

5 Naïve Bayes

In order to be able to judge how well the Random Indexing approach fares we opted to apply a Naïve Bayes classifier (Mitchell 97) to the same training and data sets for comparison.

We experimented with context windows for the classifier of zero and up to ten words before and/or after the target word "resa", in several different permutations. These permutations were run on lemmatized only as well as lemmatized and PoS-tagged data. We also tried not letting context windows cross sentence boundaries. Furthermore, all combinations of window sizes and data were run with neither normalization nor smoothing, with Lidstone smoothing (Lidstone 20) using several different lambda values, as well as giving less weight to words further away in context.

The best results were obtained with lemmatization only on a context window of ten words before and three words after the target word using distance weighting. Distance weighting was performed by applying a linearly decreasing weight, and the lambda value giving the best result was the Jeffreys-Perks value of 0.5 in added frequency. Also, sentence boundaries were not crossed. Using these settings we achieved a maximum accuracy of 56%.

6 Experimental Set-Up

The three data sets used in the experiments where first transformed into running text by stripping all tag data, and then lemmatized with the Granska tagger (Domeij *et al.* 00) in order to guarantee uniform lemmatization. These sequences of lemmas were then fed into the JavaSDM package (Hassel 04), which is a Java class package for working with Random Indexing that produces a context vector for each word (token) by adding up the random labels of the words in a distance weighted context window of desired size. Apart from a four-fold variation on the seed used, the relevant settings used in JavaSDM throughout these experiments where:

dimensionality = 1000
random_degree = 8
left_window_size = 4
right_window_size = 4
weighting_scheme = moj.ri.weighting.MangesWS
unary_labels = false
document_labels = false
granska = lemmatize

The seeds used for these particular experiments were 710225, 751128, 666 and 777. These seeds are internally in JavaSDM combined with the lexical string of the indexed token (here in the form of a words lemma) in order to guarantee reproducibility.

Two basic approaches where tried in these experiments. The first approach creates a context vector for each training example in a way similar to the way JavaSDM constructs context vectors for words. This means that we here have a distance weighted context window spanning backwards as well as in front of the target word adding up the context vectors for each word found in the context window. The second approach simply adds up all the context vectors, for all training examples, for each sense (no weighting) giving us a context vector per sense - a sense model.

Having these context vectors, and by identically constructing context vectors for each instance of "resa" in the test data, we can now compare each test instance against best matching sense model as well as best matching training example (which's sense we can assign to the test instance). This comparison can be done using any of a wide range of possible vector similarity measures, in our experiments we have used the *cosine* of the angles between the vectors. Using this measure the closest match for each test instance was chosen as the correct sense corresponding to each of the two approaches.

7 Results

The two approaches showcased a wide range in accuracy depending mainly on the size of the context window, spanning from 80% down to 40%, still beating the better baseline by an inch. When taking the mean accuracy over the four tested seeds for each variant, at each tested size of the context window, we can plot a graph to visualize different traits in the four variants.

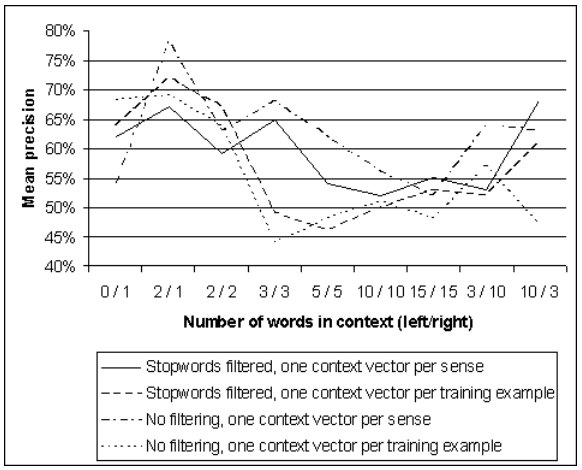


Figure 1: Mean precision over four different seeds for each variant.

In figure 1 we can clearly see a distinct difference between using one context vector per sense or one per training instance in narrow to mid-sized context windows as well as a difference between using stopwords in wide context windows mainly spanning before or after the word being classified. All four variants display a varying demand on a backward-looking context and peak at a context of two words before and one word after. The interesting part is that while using one context vector per training example proves to be particularly unfavourable in mid-sized contexts, this is not the case when using one context vector per sense. This pattern repeats itself regardless if we use stopword filtering or not. On the other hand, while stopword filtering seems to be the way to go when using a context window mainly spanning before the instance being disambiguated, this is clearly not the case in a mainly forward-looking context. However, in doing these observations we must be aware that the three context vectors that are per sense also include, or rather represent, the same amount of information as all the combined training example context vectors per sense. Taking this into account, it is also not so surprising that the main contender is one of the sense model approaches. As we can see in figure 1 the best results were obtained using a context window spanning two words before and one word after the instance of "resa" being classified.

Because of an inherent property of the d-dimensional vectors representing the random labels making them nearly orthogonal, we can

approximate orthogonality simply by choosing random directions in the high-dimensional space. This means that if we collect the context vectors we produce with Random Indexing in a matrix, this matrix will be an approximation of the standard co-occurrence matrix in the sense that their corresponding rows are similar or dissimilar to the same degree. In this way, we can achieve the same dimensional reduction as is done in LSA by the use of SVD: transforming the original co-occurrence counts into a much smaller and denser representation (Sahlgren 05). A key factor in proving the theory to hold in practice is thus the stability in the results over different random projections, here represented by the four different seeds.

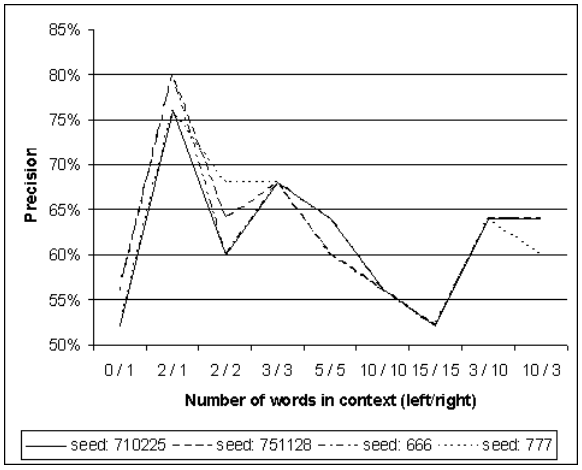


Figure 2: Variation in precision over the four seeds for the best combination, *No stopword filtering, one context vector per sense.*

As we can see in figure 2 the four seeds to a great extent plot against the same lines, with two seeds reaching a maximum of 80% followed closely by the other two at 76%. This can be compared to the above mentioned WSD experiments using a Naïve Bayes classifier on the same training and test sets that reached a top performance of 56%, a result the Random Indexing approach beats hands down. As in the case with the Naïve Bayes approach, words closer in context tend to weigh more in discriminating the different senses of "resa". Other words and their respective senses can of course, depending on differing syntactic and lexical "constraints", display other patterns. This also, naturally, applies to stopword filtering. Using one context vector per sense, rather than one per training example, seems to be a generally good idea since this generates more information dense context vectors.

8 Conclusions and Future Work

We have applied a word co-occurrence based method called Random Indexing to word sense disambiguation for Swedish, modeling the different senses by the co-occurrence of "concepts". The Random Indexing method faired well compared to a standard Naïve Bayes package reaching a maximum accuracy of 80%. As in the case with Naïve Bayes approach, words closer in context tend to weigh more in discriminating the different senses of "resa". The most favourable context window size proved to be two words before and one word after the word being disambiguated, indicating a local ambiguity. Also, stopword filtering proved to remove important syntactic clues in such narrow contexts. One possible explanation for preferring a short context window in this case could be that (mainly) sense 1 and 2 share the same domain, travelling. A wider context window will result in a likewise higher degree of shared co-occurring words. For other ambiguous words, different distance relations may however be more efficient.

Apart from the obvious studies on more words and their corresponding senses, there is also a need for studying how different parameter settings affect the quality of the sense models. One obvious example is of course the dimensionality d, another such property is the size of the context window during the Random Indexing phase, i.e. when building the initial context vector for each word/token. Throughout these experiments we used for the Random Indexing phase a sliding context window spanning four words (lemmas) before and four after the current token. An interesting thought is how it would affect the results in figure 1 if we also vary the size of the context window in this phase.

9 Acknowledgement

We would like to express gratitude to Botond Pakucs of the Centre for Speech Technology at KTH who participated in the experiments involving the Naïve Bayes classifier, and who also took part in the initial sense annotation.

References

(Domeij *et al.* 00) Rickard Domeij, Ola Knutsson, Johan Carlberger, and Viggo Kann. Granska - An efficient hybrid system for Swedish grammar checking. In *Proceedings of NoDaLiDa'99 - 12th Nordic Conference on Computational Linguistics*, 2000.

(Ejerhed *et al.* 92) Eva Ejerhed, Gunnel Källgren, Ola Wennstedt, and Magnus Åström. *SUC - The Stockholm-Umeå Corpus*, version 1.0 (suc 1.0). CD-ROM produced by the Dept of Linguistics, University of Stockholm and the Dept of Linguistics, University of Umeå. ISBN 91-7191-348-3, 1992.

(Gellerstam *et al.* 00) Martin Gellerstam, Yvonne Cederholm, and Torgny Rasmark. The bank of swedish. In *In the proceedings of Second International Conference on Language Resources and Evaluation. LREC-2000*, pages 329–333, Athens, Greece, 2000.

(Harris 85) Zelig Harris. *Distributional Structure*. Oxford University Press, New York, 1985.

(Hassel 04) Martin Hassel. JavaSDM - A Java package for working with Random Indexing and Granska, 2004. http://www.nada.kth.se/~xmartin/java/JavaSDM/.

(Ide & Véronis 98) Nancy Ide and Jean Véronis. Word Sense Disambiguation: The State of the Art. *Computational Linguistics*, 24(1), 1998.

(Kanerva *et al.* 00) Pentti Kanerva, Jan Kristoferson, and Anders Holst. Random Indexing of text samples for Latent Semantic Analysis. In L.R. Gleitman and A.K. Josh, editors, *Proceedings 22nd Annual Conference of the Cognitive Science Society*, Pennsylvania, August 2000.

(Kaplan 50) Abraham Kaplan. An experimental study of ambiguity and context. *Santa Monica: The RAND Corporation. Repr. in Mechanical Translation 2 (1955)*, pages 39–46, 1950.

(Lidstone 20) George James Lidstone. *Note on the general case of the Bayes-Laplace formula for inductive or a posteriori probabilities. Transactions of the Faculty of Actuaries*, volume 8. Brown University Press, Providence R.I., 1920.

(Lindén 05) Krister Lindén. *Word Sense Discovery and Disambiguation*. PhD dissertation, University of Helsinki, Department of General Linguistics, June 2005.

(Mitchell 97) Tom Mitchell. *Machine Learning*. McGraw-Hill, 1997.

(Sahlgren 05) Magnus Sahlgren. An Introduction to Random Indexing. In *Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering, TKE 2005)*, Copenhagen, Denmark, August 16 2005.

(Wilks & Stevenson 96) Yorick Wilks and Mark Stevenson. The grammar of sense: Is word sense tagging much more than part-of-speech tagging? Technical report, University of Sheffield, Sheffield, UK, 1996.

A Data-driven Approach to Pronominal Anaphora Resolution for German

Erhard W. Hinrichs, Katja Filippova, Holger Wunsch
SfS-CL, University of Tübingen
Wilhelmstr. 19
72074 Tübingen, Germany
{eh,ephilpva,wunsch}@sfs.uni-tuebingen.de

Abstract

This paper reports on a hybrid architecture for computational anaphora resolution (CAR) of German that combines a rule-based pre-filtering component with a memory-based resolution module (using the Tilburg Memory Based Learner – TiMBL). The data source is provided by the TüBa-D/Z treebank of German newspaper text (Telljohann *et al.* 04) that is annotated with anaphoric relations. The CAR experiments performed on these treebank data corroborate the importance of modelling aspects of discourse structure for robust, data-driven anaphora resolution. The best result with an F-measure of 0.734 achieved by these experiments outperforms the results reported by (Schiehlen 04), the only other study of German CAR that is based on newspaper treebank data.

1 Introduction

The present study focuses exclusively on the resolution of pronominal anaphora with NP antecedents for German, where the term *pronoun* is used as a cover term for 3rd person reflexive, possessive, and personal pronouns. The purpose of this paper is threefold:

(i) to apply the machine learning paradigm of memory-based learning to the task of CAR for German,

(ii) to provide a series of experiments that corroborate the importance of modelling aspects of discourse structure for robust, data-driven anaphora resolution and that induce more fine-grained information from the data than previous approaches,

(iii) to apply CAR to a corpus of German newspaper texts, yielding competitive results for a genre that is known to be considerably more difficult than the Heidelberg corpus of tourist information texts (see (Kouchnir 03) for more discussion on this issue.)

2 Previous Research on CAR

Computational anaphora resolution has been a very active research area in computational linguistics for more than three decades. While early work on CAR was carried out almost exclusively in a rule-based paradigm, there have been numerous studies during the last ten years that have demonstrated that machine-learning and statistical approaches to CAR can offer competitive results to rule-based approaches. In particular, this more recent work has shown that the hand-tuned weights for anaphora resolution introduced by (Lappin & Leass 94), by (Kennedy & Boguraev 96), and (Mitkov 02) can be successfully simulated by data-driven methods (Preiss 02b).

While there is a rich diversity of methods that have been applied to CAR, there is also a striking convergence of grammatical features that are used as linguistic knowledge across different algorithms.[1] Most approaches base their resolution algorithm on some combination of distance between pronouns and potential antecedents, grammatical agreement between pronouns and antecedents, constituent structure information, grammatical function assignment for potential antecedents, and the type of NP involved (e.g. whether it is definite or indefinite). The combined effect of these features is to establish a notion of *discourse salience* that can help rank potential antecedents. An important aspect of discourse salience is its dynamic character since there seems to be a strong correlation between salience and discourse recency. This aspect of salience was first captured by (Lappin & Leass 94) and by (Kennedy & Boguraev 96) through the use of a decay function that decreases the score of a potential antecedent each time a new sentence is processed. In data-driven approaches this decay function is simulated by the distance measure between

[1] See (Tetreault 05) for a comprehensive survey.

pronoun and antecedent.

With the exception of the Bayesian model of (Ge *et al.* 98) and the maximum-entropy system of (Kehler 97), most data-driven approaches to CAR are based on machine learning techniques, with decision trees as the widely used paradigm (McCarthy & Lehnert 95; Soon *et al.* 01; Ng & Cardie 02; Strube & Müller 03).

Previous studies of CAR have focused on English and have been based on text corpora of fairly modest size, however see (Ge *et al.* 98) for an exception. The only previous studies for German have been presented by (Strube & Hahn 99), based on centering theory, (Müller *et al.* 02), using co-training, and by (Kouchnir 03), who applies boosting. (Schiehlen 04) provides an overview of adapting CAR algorithms to German that were originally developed for English.

While memory-based learning (MBL) has been successfully applied to a wide variety of NLP tasks, there has been only one previous study of CAR using MBL (Preiss 02a). In contrast to decision trees that have been applied to CAR by a variety of authors, memory-based learning suffers less from problems of overfitting due to its lack of data abstraction. It is also known to be more sensitive to pockets of exceptions in the data – a feature characteristic of natural language data.

3 Data

The present research focuses on German and utilizes the TüBa-D/Z (Telljohann *et al.* 04), a large treebank of German newspaper text that has been manually annotated with constituent structure and grammatical relations such as *subject*, *direct object*, *indirect object* and *modifier*. These types of syntactic information have proven crucial in previous CAR algorithms. More recently, the TüBa-D/Z annotations have been further enriched to also include anaphoric relations (Hinrichs *et al.* 04), thereby making the treebank suitable for research on CAR. German constitutes an interesting point of comparison to English since German exhibits a much richer inflectional morphology and a relatively free word order at the phrase level.

The sample sentences in (1) illustrate the annotation of referentially dependent relations in the TüBa-D/Z anaphora corpus.

(1) [1 Der neue Vorsitzende der Gewerkschaft
The new chairman of the union
Erziehung und Wissenschaft] heißt [2 Ulli
Education and Science is called Ulli
Thöne]. [3 Er] wurde gestern mit 217 von
Thöne. He was yesterday with 217 out of
355 Stimmen gewählt.
355 votes elected.

'The new chairman of the union of educators and scholars is called Ulli Thöne. He was elected yesterday with 217 of 355 votes.'

In (1) a coreference relation exists between the noun phrases [1] and [2], and an *anaphoric relation* between the noun phrase [2] and the personal pronoun [3].[2] Since noun phrases [1] and [2] are coreferential, there exists an implicit anaphoric relation between NP [1] and NP [3], with all three NPs belonging to the same coreference chain. In keeping with the MUC-6 annotation standard[3], the anaphoric relation of a pronoun is established only to its most recently mentioned antecedent. (1) also illustrates the longest-match principle for identifying markables. In case of complex NPs, the entire NP counts as a markable, but so do its subconstituents.[4] Thus, part of the CAR task consists in determining that in the case at hand the complex NP as a whole is the correct antecedent for the pronoun *er*, and not only the sub-NP *der neue Vorsitzende*.

The TüBa-D/Z currently consists of 766 newspaper texts with a total of 15260 sentences and an average number of 19.46 sentences per text. The TüBa-D/Z contains 7606 reflexive and personal pronouns, 2195 possessive pronouns, and 99585 markables (i.e. potential antecedent NPs). The number of pronouns in the TüBa-D/Z corpus is considerably larger than in the hand-annotated portion of the German NEGRA newspaper corpus (2198 possessive pronouns, 3115 personal pronouns) utilized in (Schiehlen 04) and substantially larger than the German Heidelberg tourism information corpus (36924 tokens, 2179 anaphoric NPs) used by (Müller *et al.* 02) and by (Kouchnir 03).

[2] Even though the referent of the personal pronoun [3] is the same as the referent of the noun phrases [1] and [2], the relation between a pronoun and its antecedent is taken to be anaphoric, rather than coreferent. See (vanDeemter & Kibble 00) for a detailed discussion of principled reasons not to conflate the terms *coreferent* and *anaphoric*.

[3] See `www.cs.nyu.edu/cs/faculty/grishman/COtask21.book_1.html`.

[4] This means that in example (1) the NP *Der neue Vorsitzende* and the NP *der Gewerkschaft Erziehung und Wissenschaft* are separate markables. However, the latter will be filtered out by the XIP-module (described in section 4) since its gender (feminine) does not match the gender of the pronoun (masculine).

pronoun/antecedent	cataphoric	parallel	clause-mate	distance
discourse history	ON	OD	OA	PRED
	MOD	OPP	FOPP	APP
	TITLE	CONJ	HD	OTHER
pronoun	reflexive	possessive		

Table 1: Feature Set

4 Experiments

The experiments are based on a hybrid architecture that combines a rule-based pre-filtering module with a memory-based resolution algorithm. In the memory-based encoding used in the experiments, anaphora resolution is turned into a binary classification problem. If an anaphoric relation holds between an anaphor and an antecedent, then this is encoded as a positive instance. If no anaphoric relation holds between a pronoun and an NP, then this encoded as a negative instance.

The purpose of the pre-filtering module, which has been implemented in the Xerox Incremental Deep Parsing System (XIP) (Aït-Mokhtar *et al.* 02), is to retain only those NPs as potential antecedents that match a given pronoun in number and gender. Due to the richness of inflectional endings in German, this pre-processing step is crucial for cutting down the size of the search space of possible antecedents. Without XIP pre-filtering, the TüBa-D/Z corpus yields a total of 1,412,784 of anaphor/candidate-antecedent pairs. This number represents all possible ways of pairing a pronoun with an antecedent NP in each of the 766 texts of the TüBa-D/Z corpus. After pre-filtering this number is reduced to appr. 190,000 pairs.

The memory-based resolution module utilizes the Tilburg Memory Based Learner (TiMBL), version 5.1 (Daelemans *et al.* 05). Unless otherwise specified, the experiments use the default settings of TiMBL.

4.1 Feature Set

In the experiments, the TiMBL learner was presented with the set of features summarized in table 1. The features on line 1 all refer to relational properties of the pronoun and potential antecedents. The feature *parallel* encodes whether the anaphor and the potential antecedent have the same grammatical function. The features on line 3 refer to the pronoun alone and encode whether it is possessive or reflexive. The features on line 2 are designed to model the discourse history in terms of the grammatical functions of NPs that are in the same coreference class as the candidate antecedent. The grammatical functions are those provided by the syntactic annotation of the TüBa-D/Z treebank: ON (for: *subject*), OA (for: *direct object*), OD (for: *dative object*), PRED (for: *predicative complement*), MOD (for: *modifier*), etc.

The main purpose of the experiments reported here was to systematically study the impact that information about discourse context has on the performance of data-driven approaches to CAR. To this end, we designed two experiments that differ from each other in the amount of information about the coreference chains that are encoded in the training data.

4.2 Knowledge-Rich Encoding of Instances – Experiment I

In Experiment I, complete information about coreference chains is used for training. In example (1) the three bracketed NPs form one coreference chain since the first two NPs are coreferent and the pronoun is anaphoric to both. Accordingly, for example (1), two positive instances are created as shown in table 2. The sequence of features in each vector follows the description of features shown in table 1. Binary features are encoded as *yes*/*no*. Numeric features are given values from 1 to 30, with a special value of 31 reserved for the value *undefined*. Inspection of the data showed that a context window of this size contains the antecedent in more than 99% of all cases. For technical reasons, the numeric values are prefixed by a dash in order for TiMBL to treat them as discrete rather than continuous values. In the case at hand, the closest member of the same coreference class is in the previous sentence. Thus, the distance feature has value -1.

The first vector in table 2 displays the pairing of the pronoun with the NP *der neue Vorsitzende der Gewerkschaft Erziehung und Wissenschaft*, the first NP in the text. This NP is the subject (ON) of its clause. The value for this grammatical function is -1 since the NP occurs in the clause immediately preceding the pronoun. The second vector pairs the two preceding NPs with the pronoun *er*. Since the NP *Ulli Thöne* is in predicative position (PRED) and occurs in the same clause as the subject NP *der neue Vorsitzende der Gewerkschaft Erziehung und Wissenschaft*, the value

cat,par,cl-mate,dist,ON,OD,OA,PRED,MOD,OPP,FOPP,APP,TITLE,CONJ,HD,OTHER,refl,poss;class																		
< no,	no,	no,	-1,	-1,	-31,	-31,	-31,	-31,	-31,	-31,	-31,	-31,	-31,	-31,	-31,	no,	no;	yes >
< no,	no,	no,	-1,	-1,	-31,	-31,	-1,	-31,	-31,	-31,	-31,	-31,	-31,	-31,	-31,	no,	no;	yes >

Table 2: Sample Instances

for these two grammatical functions ON and PRED is -1. Thus, the intended semantics of the features for each grammatical function is to encode the distance of the last occurrence of a member of the same coreference class with that particular grammatical function.[5] One aspect of the discourse history that the current encoding does not model is the frequency with which a given grammatical function occurs in the text, since the encoding only registers the most recent occurrence of a given grammatical function. To control for this, a variant of the experiments reported here was conducted where for each grammatical function a pair of values was introduced consisting of the distance of the closest antecedent NP and the number of times that grammatical function appeared in the same coreference class. However, such additional mention counts did not significantly change the results of the expermiments and were therefore omitted from the feature vectors.

The sample vectors in table 2 illustrate the incremental encoding of instances. The initial vector encodes only the relation between the pronoun and the antecedent first mentioned in the text. Each subsequent instance adds one more member of the same coreference class. This incremental encoding follows the strategy of (Kennedy & Boguraev 96) and reflects a dynamic modelling of the discourse history. The last item in the vector, which is separated from the other entries by a semicolon, indicates class membership. In the memory-based encoding used in the experiments, anaphora resolution is turned into a binary classification problem. If an anaphoric relation holds between an anaphor and an antecedent, then this is encoded as a positive instance, i.e., as a vector ending in *yes*. If no anaphoric relation holds between a pronoun and an NP, then this encoded as a negative instance, i.e., as a vector ending in *no*.

4.3 Knowledge-poor Encoding of Instances – Experiment II

Experiment II uses a more knowledge-poor encoding of the data and pairs each pronoun only with the most recent antecedent in the same coreference class, thereby losing both information inherent in the entire coreference class and at the same time truncating the discourse history. Using example (1) once more as an illustration, two positive instances are created. The first vector is the same as in Experiment I. The second vector retains value -1 only for PRED, the grammatical function of the candidate itself. The value of ON is now undefined (-31).

4.4 Two Variants

For each of the two experiments described above, two variants were conducted. In one version, the evaluation focused on the closest antecedent to calculate the result for recall, precision and F-measure.[6] In a second variant, the most confident antecedent was chosen. The confidence measure was calculated by the function $\mathrm{conf}(t, c_k)$ defined as follows:

Definition Given classes $c_1 \ldots c_n$, and class distributions $d_1 \ldots d_n$ (where d_i is the number of neighbors that classified the test instance t as belonging to class c_i), the confidence $\mathrm{conf}(t, c_k)$ in the final classification c_k is

$$\mathrm{conf}(t, c_k) = \frac{d_k}{\sum_{i=1}^{n} d_i}$$

5 Evaluation

To assess the difficulty of the pronoun resolution task for the TüBa-D/Z corpus, we established as a baseline a simple heuristic that picks the closest preceding subject as the antecedent. This baseline is summarized in table 3 together with results of the experiments described in the previous section. For each experiment ten-fold cross-validation was performed, using 90% of the corpus for training and 10% for testing.

5.1 Results of Experiments I and II

Both experiments significantly outperform the baseline approach in F-measure. The findings summarized in table 3 corroborate the importance of modelling the discourse history for pronoun resolution since the results of Experiment I are consistently better than those of Experiment II. An explicit modelling of the

[5] A similar encoding is also used by (Preiss 02a).

[6] Throughout this paper the term *F-measure* implies the parameter setting of $\beta = 1$.

	av. precision	av. recall	av. F-measure
Baseline	0.500	0.647	0.564
Experiment I			
closest antecedent	0.826	0.640	0.721
most conf. antecedent	0.801	0.621	0.700
Experiment II			
closest antecedent	0.779	0.600	0.678
most conf. antecedent	0.786	0.606	0.684

Table 3: Summary of Results

6 most informative features:	clause-mate,parallel,possessive,FOPP,ON,OD
3 least informative features:	TITLE, distance,CONJ

Table 4: Summary of Feature Weights Based on GainRatioValues

discourse history with a hand-coded decay function was first proposed by (Lappin & Leass 94) and by (Kennedy & Boguraev 96). The present paper does not have to rely on the hand-coding of such a decay function. Rather, it induces the relevant aspects of the discourse history directly from the instance base used by the memory-based learner.

It is also noteworthy that in Experiment I the strategy of picking the closest antecedent outperforms the strategy of picking the most confident antecedent chosen by TiMBL.

5.2 Benchmarking Feature Impact

It is instructive to benchmark the importance of the features used in the experiments. This can be ascertained from the weights that the gain ratio measure (as the default feature weighting used by TiMBL) assigns to each feature. Gain ratio is an entropy-based measure that assigns higher weights to more informative features. Table 4 displays the top six most informative features and the three least informative features in decreasing order of informativeness. The fact that the features *clause-mate*, *parallel*, and *possessive* are the three most informative features concurs with the importance given to such features in hand-crafted algorithms for CAR. However, the ranking of some of the features included in table 4 is rather unexpected. The fact that the grammatical function FOPP (for: *optional PP complement*) outranks the grammatical function *subject* (ON) runs counter to hand-coded salience rankings found in the literature which give the feature *subject* the highest weights among all grammatical functions. That the FOPP feature outranks the function *subject* is due to the fact that the presence of an optional PP-complement is almost exclusively paired with negative instances. This finding points to an important advantage of data-driven approaches over hand-crafted models. While the latter only take into account positive evidence, data-driven models can profit from considering positive and negative evidence alike. Perhaps the most surprising result is the fact that *distance* between anaphor and antecedent is given the second lowest weight among all eighteen features. This sharply contrasts with the intuition often cited in hand-crafted approaches that the distance between anaphor and antecedent is a very important feature for an adequate resolution algorithm. The reason why distance receives such a low weight might well have to do with the fact that this feature becomes almost redundant when used together with the other distance-based features for grammatical functions.

The empirical findings concerning feature weights summarized in table 4 underscore the limitation of hand-crafted approaches that are based on the analysts' intuitions about the task domain. In many cases, the relative weights of features assigned by data-driven approaches will coincide with the weights assigned by human analysts and fine-tuned by trial and error. However, in some cases, feature weightings obtained automatically by data-driven methods will be more objective and diverge considerably from manual methods, as the weight assigned by TiMBL to the feature *distance* illustrates.

5.3 Optimization by Fine-tuning of TiMBL Parameters

It has been frequently observed (e.g. by (Hoste *et*

	av. precision	av. recall	av. F-measure
Baseline	0.500	0.647	0.564
Experiment I closest antecedent	**0.827**	**0.661**	**0.734**

Table 5: Summary of Best Results

al. 02)) that the default settings provided by a classifier often do not yield the optimal results for a given task. The CAR task for German is no exception in this regard. TiMBL offers a rich suite of parameter settings that can be explored for optimizing the results obtained by its default settings. Some key parameters concern the choice of feature distance metrics, the value of k for the number of nearest neighbors that are considered during classification as well as the choice of voting method among the k-nearest neighbors used in classification. TiMBL's default settings provide the feature distance metric of weighted overlap (with the gain ratio measure for feature weighting), $k = 1$ as the number of k-nearest neighbors, and majority class voting.

To assess the possibilities of optimizing the results of Experiments I and II, the best result (Experiment I with closest antecedent) was chosen as a starting point. The best results, shown in table 5, were obtained by using TiMBL with the following parameters: modified value distance metric (MVDM), no feature weighting, $k = 3$, and inverse distance weighting for class voting.

The optimizing effect of the parameters is not entirely surprising.[7] The MVDM metric determines the similarities of feature values by computing the difference of the conditional distribution of the target classes for these values.[8] For informative features, $\delta(v_1, v_2)$ will on average be large, while for less informative features will tend to be small. (Daelemans *et al.* 05) report that for NLP tasks MVDM should be combined with values of k larger than one. The present task confirms this result by achieving optimal results for a value of $k = 3$.

[7] See (Hoste *et al.* 02) for the optimizing effect of MVDM in the word sense disambiguation task.

[8] More specifically, the distance $\delta(v_1, v_2)$ between two feature values v_1 and v_2 is defined as

$$\delta(v_1, v_2) = \sum_{i=1}^{n} | P(C_i|v_1) - P(C_i|v_2) |$$

6 Comparison with Related Work

The only previous study of German CAR that is based on newspaper treebank data is that of (Schiehlen 04).[9] Schiehlen compares an impressive collection of published algorithms, ranging from reimplementations of rule-based algorithms to reimplementations of machine-learning and statistical approaches. The best results of testing on the NEGRA corpus were achieved with an F-measure of 0.711 by a decision-tree classifier, using C4.5 and a pre-filtering module similar to the one used here. The best result with an F-measure of 0.734 achieved by the memory-based classifier and the XIP-based pre-filtering component outperforms Schiehlen's results, although a direct comparison is not possible due to the different data sets.

7 Summary and Future Work

The current paper presents a hybrid architecture for computational anaphora resolution (CAR) of German that combines a rule-based pre-filtering component with a memory-based resolution module (using the Tilburg Memory Based Learner – TiMBL). The data source is provided by the TüBa-D/Z treebank of German newspaper text that is annotated with anaphoric relations. The CAR experiments performed on these treebank data corroborate the importance of modelling aspects of discourse structure for robust, data-driven anaphora resolution. The best result with an F-measure of 0.734 achieved by the memory-based classifier and the XIP-based pre-filtering component outperforms Schiehlen's results, although a direct comparison is not possible due to the different data sets.

The experiments reported here are all based on treebank data. In future work it is planned to use the output of a robust parser for German as input to the hybrid model presented here. Several parsers are good candidates for such an extension. The parsers for German developed by (Trushkina 04), (Müller 05) and by (Foth *et al.* 04) all produce the relevant grammatical

[9] (Kouchnir 03) briefly discusses results of applying her ensemble learning classifier to a hand-annotated corpus of the German weekly newspaper *Der Spiegel*. However, compared to her results on the Heidelberg tourism corpus, the best results for the *Spiegel* data are rather low with an F-measure of 34.4 %.

information needed for the features employed by the memory-based module.

References

(Aït-Mokhtar *et al.* 02) Salah Aït-Mokhtar, Jean-Pierre Chanod, and Claude Roux. Robustness beyond shallowness: incremental deep parsing. *Natural Language Engineering*, 8(2–3):121–144, 2002.

(Daelemans *et al.* 05) Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. TiMBL: Tilburg memory based learner– version 5.1–reference guide. Technical Report ILK 01-04, Induction of Linguistic Knowledge, Computational Linguistics, Tilburg University, 2005.

(Foth *et al.* 04) Kilian Foth, Michael Daum, and Wolfgang Menzel. A broad-coverage parser for german based on defeasible constraint. In *KONVENS 2004, Beiträge zur 7. Konferenz zur Verarbeitung natürlicher Sprache*, pages 45–52, Vienna, 2004.

(Ge *et al.* 98) Niyu Ge, John Hale, and Eugene Charniak. A statistical approach to anaphora resolution. In *Proceedings of the Sixth Workshop on Very Large Corpora*, pages 161–170, Montreal, Canada, 1998.

(Hinrichs *et al.* 04) Erhard Hinrichs, Sandra Kübler, Karin Naumann, Heike Telljohann, and Julia Trushkina. Recent developments in linguistic annotations of the TüBa-D/Z treebank. In Sandra Kübler, Joakim Nivre, Erhard Hinrichs, and Holger Wunsch, editors, *Proceedings of the Third Workshop on Treebanks and Linguistic Theories*, Tübingen, Germany, 2004.

(Hoste *et al.* 02) Veronique Hoste, Iris Hendrickx, Walter Daelemans, and Antal van den Bosch. Parameter optimization for machine-learning of word sense disambiguation. *Natural Language Engineering*, 8(4):311–325, 2002.

(Kehler 97) Andrew Kehler. Probabilistic coreference in information extraction. In *Proceedings of the Second Conference on Empirical Methods in NLP (EMNLP-2)*, Providence, RI, 1997.

(Kennedy & Boguraev 96) Christopher Kennedy and Branimir Boguraev. Anaphora for everyone: Pronominal anaphora resolution without a parser. In *The Proceedings of the 16th International Conference on Computational Linguistics*, 1996.

(Kouchnir 03) Beata Kouchnir. A machine learning approach to German pronoun resolution. Unpublished M.Sc. thesis, School of Informatics, University of Edinburgh, 2003.

(Lappin & Leass 94) Shalom Lappin and Herbert Leass. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561, 1994.

(McCarthy & Lehnert 95) Joseph F. McCarthy and Wendy G. Lehnert. Using decision trees for coreference resolution. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 1050–1055, Montreal, Canada, 1995.

(Mitkov 02) Ruslan Mitkov. *Anaphora Resolution*. John Benjamins, Amsterdam, 2002.

(Müller 05) Frank H. Müller. *'A Finite-State Approach to Shallow Parsing and Grammatical Functions Annotation of German*. Unpublished PhD thesis, University of Tübingen, 2005.

(Müller *et al.* 02) Christoph Müller, Stefan Rapp, and Michael Strube. Applying co-training to reference resolution. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, pages 352–359, Philadelphia, PA, USA, 2002.

(Ng & Cardie 02) Vincent Ng and Claire Cardie. Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, pages 104–111, Philadelphia, PA, USA, 2002.

(Preiss 02a) Judita Preiss. Anaphora resolution with memory based learning. In *Proceedings of the 5th UK Special Interest Group for Computational Linguistics (CLUK5)*, pages 1–8, 2002.

(Preiss 02b) Judita Preiss. A comparison of probabilistic and non-probabilistic anaphora resolution algorithms. In *Proceedings of the Student Workshop at the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, pages 42–47, Philadelphia, PA, USA, 2002.

(Schiehlen 04) Michael Schiehlen. Optimizing algorithms for pronoun resolution. In *Proceedings of the 20th International Conference on Computational Linguistics*, Geneva, August 2004.

(Soon *et al.* 01) Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544, 2001.

(Strube & Hahn 99) Michael Strube and Udo Hahn. Functional centering - grounding referential coherence in information structure. *Computational Linguistics*, 25(3):309–344, 1999.

(Strube & Müller 03) Michael Strube and Christoph Müller. A machine learning approach to pronoun resolution in spoken dialogue. In *Proceedings of ACL-03*, pages 168–175, Sapporo, July 2003.

(Telljohann *et al.* 04) Heike Telljohann, Erhard W. Hinrichs, and Sandra Kübler. The TüBa-D/Z Treebank – Annotating German with a Context-Free Backbone. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004)*, Lisbon, Portugal, 2004.

(Tetreault 05) Joel Tetreault. *Empirical Evaluations of Pronoun Resolution*. Unpublished PhD thesis, University of Rochester, Department of Computer Science, 2005.

(Trushkina 04) Julia S. Trushkina. *Morpho-syntactic Annotation and Dependency Parsing of German*. Unpublished PhD thesis, University of Tübingen, 2004.

(vanDeemter & Kibble 00) Kees van Deemter and Roger Kibble. On coreferring: Coreference annotation in muc and related schemes. *Computational Linguistics*, 26(4):615–623, 2000.

Towards the Automatic Derivation of Lexical Prototypes

Aleem Hossain and **Mark G. Lee**
School of Computer Science
University of Birmingham
Birmingham B15 2TT, United Kingdom
{A.Hossain,M.G.Lee}@cs.bham.ac.uk

Abstract

In order to model human understanding of natural language, it is necessary to be able to represent the knowledge and processes that humans use. As such, Natural Language Processing (NLP) research needs to take into consideration cognitive theories of language processing; one candidate theory is Prototype Theory. In this paper, we argue for the adoption of a conceptual approach to word meaning in order to achieve a view of lexical semantics consistent with cognitive representation and processes. We detail ongoing research concerned with extending an existing semi-automatic method for deriving patterns of verb usage from corpora. Finally, we argue that this approach has connections with Prototype Theory and detail initial steps to fully-automate this process.

1 Introduction

Choosing appropriate representations and modelling the processes associated with word meaning is an important challenge to designing systems which can understand and respond intelligently to natural language. However, most approaches in NLP do not take into consideration appropriate cognitive theories of categorisation which can be considered to be central to the problem of lexical semantics. WordNet (Miller *et al.* 90), for example, classifies sets of synonyms using links such as hypernym and meronym chains. All the features of parent synsets are inherited by their children and further distinguishing features are added. As such, features are treated, as under the classical theory of categorisation to be *necessary* and *sufficient* for defining a word.

An alternative approach to lexical semantics is found in cognitive linguistics. Rather than seeking to satisfy a set of *necessary* and *sufficient* conditions, meanings should be represented by prototypes of word usage. These prototypes consist of syntactic and semantic contextual patterns which can then be used in tasks such as word sense disambiguation in terms of a similarity measure between the prototype and the context under consideration.

The rest of this paper proceeds as follows. First, cognitive foundations to categorisation will be outlined with respect to human language comprehension. Secondly, we describe and critique a semi-automatic approach to lexical semantics and draw connections between this approach and Prototype Theory before detailing our ongoing work to automate the derivation of lexical prototypes. Finally, we describe some related work before drawing conclusions.

2 Cognitive Foundations

For the purpose of this paper, *concepts* and *categories* will be defined as in (Murphy 04). *Concepts* are "mental representations of classes of things" whereas a *category* is "the class of objects in the world." As we interact with the world, we form concepts relating to particular categories. These concepts can be changed as we discover something new about a member of the corresponding class of objects.

The cognitive task of categorisation is central to the understanding of meaning. For example, (Labov 73) considers linguistics to be "the study of categories" and views man as a "categorizing animal." If we did not perform a categorisation function but treated each object that we encounter as unique, we would be overwhelmed by the task of trying to define everything we see. Instead, we classify entities using our mental representation as members of particular categories; we exploit the fact that objects we encounter are alike in some way so that we do not need to identify each uniquely. Consequently, when presented with some entity, we can use its perceptual properties to categorise it. Once an entity has been categorised, knowledge of the conceptual class allows inferences to be made about the perceptual, and even the non-perceptual properties, of that entity.

Therefore, it is clear that concepts have two functions: first, to categorise, and secondly to al-

low inferences about properties of the members of the corresponding category to be made. What is unclear, is how such concepts are structured.

2.1 Prototype Theory

Under the classical theory of categorisation, a concept is considered to be a mental 'store' of the defining properties of objects (Murphy 04). Given a definition, in order for an object to be classified as a member of a category, the object under consideration must possess all the features specified by that definition. That is, the features contained within the concept are *necessary* in order for a given object to be considered a member of a category. Moreover, these features are *sufficient* to specify each member of that particular category. The necessary features are not found together in any other category and any other features of a given object do not affect how it is categorised just as long as it has all the necessary features.

In this way, the concepts that we form define features of objects which are both necessary and sufficient for categorical membership. Therefore, an object either is a member of a category, or it is not. Furthermore, all members are equal; there are no 'lesser' members of a category.

This approach to categorisation has its origins in the writings of Aristotle and through Western philosophy until the later work of Wittgenstein (referenced edition published in 1972) who argued that categorical members show *family resemblances* to one another, but that there are not necessarily common features shared by all members. In other words, it is difficult to specify the necessary and sufficient properties which adjudicate categorical membership. Indeed, "failure to define specifying features" (Smith & Medin 81) is a frequently cited argument against the classical approach to concepts. Consider the concept of *cup*: the feature 'has handle' cannot be suggested as a necessary condition since, for example, a coffee house's 'to go' cup does not have a handle. If, though, the necessary conditions are relaxed so that a handle is not required, then bowls may also be categorised as *cups*. Similar observations by Labov led him to suggest the less-rigid feature "usually with a handle, sometimes without." (Labov 73)

On the basis of this work and the empirical evidence detailed primarily in (Rosch 73) (Rosch 75), Prototype Theory was proposed as an alternative approach to the classical view of categorisation. Prototype Theory says that rather than seeking to satisfy necessary and sufficient conditions, category membership is assigned with reference to a prototype or 'typical' representation of that category.

Rosch herself made no claims about the way that typicality effects might be realised in conceptual structure other than to say that the effects existed. (Smith & Medin 81) describes a number of different models which have been proposed on the basis of her empirical evidence. For instance, the *summary representation* of concepts says that there is a unitary description of an entire category consisting of weighted features; the more typical a feature is, the higher its weight is. Categorisation, then, is performed on the basis of a weight summation with some threshold defining categorical membership. Alternatively, the *exemplar view* of categories holds that rather than having a unitary description, concepts are represented by a number of typical examples in each category. Membership to a category, under this theory, is dependent on how close a given entity is to these exemplars.

2.2 Meaning and Concepts

Referential semantics considers that the orthographic form of a word is given semantic significance by virtue of being related to real objects. It can be argued that this is a psychologically inadequate explanation since it would require us to have access to every member in a category (Murphy 04). Whilst this is impossible, we can call upon our mental concepts to provide a definition of categories which can then be used to assign a meaning to a word. Therefore, the meaning of words is not provided by mapping a word to a real world object. Rather, the mapping is performed between words and concepts. In turn, these concepts can be mapped to categories of entities in the real world. This makes logical sense since concepts are an abstract psychological description of the world around us.

It follows that linguistic information must allow us to form and alter our concepts so that knowledge that is communicated to us can be re-used when we interact with the world. If, though, as is common in NLP, meanings are represented by features and not with reference to our concepts, then linguistic information would not allow us to develop or modify our categorical representations (Murphy 04).

Approaches to lexical semantics should not,

therefore, define meaning in terms of features, but rather with reference to a conceptual prototype which can be used to judge categorical membership.

3 Towards Prototypes of Word Meaning

We performed an initial experiment concerning the unsupervised clustering of nouns in the British National Corpus (BNC) using prototypicality as an organzine principle. Groups of co-occurring nouns were grouped-together and then organised with reference to a prototype using a statistical notion of Mutual Information (MI) to identify the prototypical noun. The results showed that MI was not a good measure of 'family resemblance' in this case. This suggests that word frequencies (represented using a Mutual Information measure) are not sufficient to capture prototypicality. We suggest that *syntagmatic* (construction) patterns are required. This will be explained further in the next section. For full implementation details and analysis see (Hossain & Lee 05).

3.1 Semi-Automatic Corpus Pattern Analysis

(Pustejovsky *et al.* 04) describes initial work concerned with semi-automatically deriving syntagmatic patterns of verb usage. They claim that word senses are associated with patterns of syntax and semantics and that these are the typical patterns of word usage. The goal of their research, then, is:

> "to construct a dictionary of normal *selection contexts* for natural language; that is, a computational lexical database of rich selectional contexts, associated with procedures for assigning interpretations on a probabilistic basis to less normal contexts."

There are three stages to CPA (taken from (Pustejovsky *et al.* 04)):

1. The manual discovery of selectional context patterns for specific verbs.
2. The automatic recognition of instances of the identified patterns.
3. The automatic acquisition of patterns for unanalyzed cases.

The first stage involves the generation of an initial sample of concordance lines for a target verb and from these, the (manual) identification of significant collocates. Every concordance line is classified based on context; for instance, 'normal' usage, alternations, exploitations and errors. Further concordance lines can be generated to establish the conventionality of a particular phraseology if necessary. Identifying the relevant contextual features which give a word its meaning in context is an important task. The processes required in order to identify pattern elements considered to be significant to a verb's meaning, as identified in (Pustejovsky *et al.* 04) are:

1. Shallow (phrase level) syntactic parsing.
2. Shallow semantic typing.
3. Minor syntactic category parsing (e.g. Adverbial phrases, temporal adjuncts).
4. Subphrasal Syntactic Cue Recognition (e.g. or Genitives, negatives).

The above elements are identified in the BNC by using the Robust Accurate Statistical Parsing (RASP) system (Briscoe & Carroll 02) and then (manually) tagging the corpus semantically (using a method similar to that described in (Pustejovsky *et al.* 02)) according to top types of the Brandeis Shallow Ontology (BSO). These parse trees can then be applied to enrich the patterns that are derived initially with the appropriate semantic and syntactic information.

The patterns derived are then tested by automatically extracting instances of those patterns from the corpus. This also allows refinement of the features associated with each pattern.

The aim for the automatic acquisition of patterns for unanalysed predicates is ultimately to be able to extract pattern-defining features for predicates that the lexicographer has not analysed. This is done by calculating the (semantic) similarity of clusters of nouns which appear in a particular argument position to the lexical sets populated during the lexical discovery stage. If the cluster under consideration is not 'close enough' to the existing lexical sets, then a new set is established.

3.2 Towards Automatic Pattern Extraction

The approach to lexical semantics described above is consistent with the exemplar approach

to categorisation. Namely, that concepts are represented by a subset of 'typical' members of that category. According to this approach, categorical membership is judged in terms of some similarity measure with reference to the exemplar members.

The methodology outlined in the previous section is partially dependent on the intuition of the lexicographer. Consequently, the patterns generated can be regarded as subjective. Rather than base patterns of usage on subjectivity, corpus evidence and statistical techniques should be used to give a more objective view of word meaning. More specifically, how much of the lexical discovery stage described in the previous section can automated? The initial generation of syntagmatic patterns of usage and population of lexical sets associated with these patterns was performed manually.

The analysis of 'outliers' (the instances that do not fit manually identified patterns), is another interesting area. If an instance does not match any of the patterns initially derived, then a new pattern needs to be created. If the identification of patterns can be automated, then this would provide a solution to this problem.

We have begun the initial tasks required in order to extract these patterns automatically from corpora, assigning appropriate semantic tags to participating complements and populating lexical sets. Currently, a concordance line generator capable of building concordances for all forms of a given verb has been implemented. These concordances are then processed by RASP (Briscoe & Carroll 02) in order to enrich them with appropriate syntactic information.

The next task is to perform shallow semantic typing on these concordance lines. Once the concordance lines have been annotated with syntactic and semantic information, the next stage is to automatically identify significant contextual patterns and in doing so generate prototypes of verb usage.

3.3 Related Work

(Mason & Hunston 01) describes a study which confirmed the feasibility of automatically recognising verb patterns taken from Cobuild in running text. They proceed by performing part of speech tagging on the input text and then performing shallow parsing to identify noun groups, verb groups and potential clauses (for instance, a *that*-clause). Any ambiguous parses are stored in parallel and it is left up to the recogniser to choose the best pattern for its task. The parsing performed here is more coarse-grained than the parsing performed by RASP (which we use in our work). We use more detailed syntactic information in order that the significant pattern elements identified in (Pustejovsky *et al.* 04) can be extracted.

Once this analysis has been done, the next task is to recognise these patterns. Each verb in the input stream is lemmatised and compared with the corresponding Cobuild patterns which are retrieved for matching against. Matching proceeds by first eliminating those patterns whose components do not appear in the textual context under consideration. The remaining Cobuild patterns are evaluated according to a weighting scheme and the highest-ranked pattern is chosen as the match.

This work usefully highlights some difficulties associated with this kind of work. For instance, identifying the *that*-clause in sentences such as the following where the word 'that' is not explicit:

> "He decided the project was unacceptable."

(Mason & Hunston 01) also describes the problem of *non-canonical* patterns which exhibit unusual word order deviating from the 'prototypical' pattern. (Mason & Hunston 01) cites the example

> "The question a director has to decide is how..."

where the direct object is the *theme* of the sentence. The 'prototypical' pattern for this sentence is *V that* followed by a direct object. In terms of Prototype Theory, there needs to be some sort of similarity measure mediated by a threshold function to decide whether the above sentence can be classified as a given pattern.

Whilst the task here is different to automatically generating patterns of verb usage, it is interesting to note how matching can be performed with a minimal amount of syntactic information and no semantic annotation. It will be interesting to discover how fine-grained (or otherwise) patterns need to be in order to maximise their discriminatory effect without being too computationally expensive.

4 Conclusion

We have argued that any representative theory of lexical semantics needs to account for the cognitive aspects found in language-use such as categorisation by prototype. We then reviewed recent work on the semi-automatic derivation of lexical information regarding verb usage and drew connections from this work to Prototype Theory. Finally, we described some initial steps to extend this approach by fully-automating the process to identify word meaning prototypes based purely on corpus evidence.

References

(Briscoe & Carroll 02) Ted Briscoe and John Carroll. Robust Accurate Statistical Annotation of General Text. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC'02), 29-31 May*, pages 1499–1504, Las Palmas, 2002.

(Hossain & Lee 05) Aleem Hossain and Mark G. Lee. A Cognitive Approach to Conceptual Ontologies. In *Proceedings of the 8th Research Colloquium of the UK Special-Interest Group in Computational Linguistics (CLUK'05), 11 January*, pages 32–37, Manchester, 2005.

(Labov 73) William Labov. The Boundaries of Words and their Meanings. In Charles-James N. Bailey and Roger W. Shuy, editors, *New Ways of Analyzing Variation in English*, pages 340–373. Georgetown University Press, Washington, D.C, 1973.

(Mason & Hunston 01) Oliver Mason and Susan Hunston. The Automatic Recognition of Verb Patterns: A Feasiblity Study. In *Proceedings of the 6th Conference on Computational Lexicography and Corpus Research (COMPLEX'01), 28 June-1 July*, pages 103–118, Birmingham, 2001.

(Miller *et al.* 90) George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. Introduction to WordNet: An On-line Lexical Database. *International Journal of Lexicography*, 3(4):235–244, 1990.

(Murphy 04) Gregory L. Murphy. *The Big Book of Concepts*. MIT Press, London, 2004.

(Pustejovsky *et al.* 02) James Pustejovsky, Anna Rumshisky, and José Castaño. Rerendering Semantic Ontologies: Automatic Extensions to UMLS through Corpus Analysis. In *Proceedings of the 3rd International Language Resources and Evaluation Conference Workshop on Ontologies and Lexical Knowledge Bases (OntoLex'02), 27 May*, pages 60–67, Las Palmas, 2002.

(Pustejovsky *et al.* 04) James Pustejovsky, Patrick Hanks, and Anna Rumshisky. Automated Induction of Sense in Context. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING'04), 23-27 August*, pages 924–931, Geneva, 2004.

(Rosch 73) Eleanor H. Rosch. On the Internal Structure of Perceptual and Semantic Categories. In Timothy E. Moore, editor, *Cognitive Development and the Acquisition of Language*, pages 111–144. Academic Press, London, 1973.

(Rosch 75) Eleanor Rosch. Cognitive Representations of Semantic Categories. *Journal of Experimental Psychology: General*, 104(3):192–233, 1975.

(Smith & Medin 81) Edward E. Smith and Douglas L. Medin. *Categories and Concepts*. Number 4 in Cognitive Science Series. Harvard University Press, Cambridge, MA, 1981.

(Wittgenstein 72) Ludwig Wittgenstein. *Philosophical Investigations*. Blackwell, Oxford, third edition, 1972. Translated by G. E. M. Anscombe.

Automatic Identification of Cognates and False Friends in French and English

Diana Inkpen and **Oana Frunza**
School of Information Technology and Eng.
University of Ottawa
Ottawa, ON, K1N 6N5, Canada
{diana,ofrunza}@site.uottawa.ca

Grzegorz Kondrak
Department of Computing Science
University of Alberta
Edmonton, AB, T6G 2E8, Canada
kondrak@cs.ualberta.ca

Abstract

Cognates are words in different languages that have similar spelling and meaning. They can help a second-language learner on the tasks of vocabulary expansion and reading comprehension. The learner also needs to pay attention to pairs of words that appear similar but are in fact *false friends*: they have different meaning in some contexts or in all contexts. In this paper we propose a method to automatically classify a pair of words as cognates or false friends. We focus on French and English, but the methods are applicable to other language pairs. We use several measures of orthographic similarity as features for classification. We study the impact of selecting different features, averaging them, and combining them through machine learning techniques.

1 Introduction

When learning a second language, a student can benefit from knowledge in his/her first language (Gass 87) (Ringbom 87). Cognates – words that have similar spelling and meaning in the two languages – help with vocabulary expansion and with reading comprehension. On the other hand, there are also pairs of words that appear similar, but have different meaning in some or all contexts: false friends. Dictionaries often include information about false friends; and there are dictionaries devoted exclusively to them, such as (Prado 96).

Cognates have also been employed in natural language processing. The applications include sentence alignment (Simard *et al.* 92; Melamed 99), inducing translation lexicons (Mann & Yarowsky 01; Tufis 02), improving statistical machine translation models (Al-Onaizan *et al.* 99), and identification of confusable drug names (Kondrak & Dorr 04). All those applications depend on an effective method of identifying cognates by computing a numerical score that reflects the likelihood that the two words are cognates.

In this paper we focus on the automatic identification of cognates and false friends for the purpose of preparing lists of them for inclusions in dictionaries and other learning aids. Special cognate lists exist only for very few language pairs. Moreover, it takes a lot of time for people to prepare such lists manually, while a program can do this very quickly.

We propose a method to automatically classify pairs of words as cognates or false friends. Our approach is based on several orthographic similarity measures that we use as features for classification. We test each feature separately; we also test, for each pair of words, the average value of all the features. Then we explore various ways to combine the features, by applying several machine learning techniques from the Weka package (Witten & Frank 00). The two classes for the automatic classification are: Cognates/False-Friends and Unrelated. Cognates and False-Friends can be distinguished on the basis of an additional "translation" feature: if the two words are translations of each other in a bilingual dictionary, they are classified as Cognates; otherwise, they are assumed to be False-Friends.

Although French and English belong to different branches of the Indoeuropean family of languages, they share an extraordinary high number of cognates. The cognates derive from several distinct sources. The majority are words of Latin and Greek origin that permeate the vocabularies of European languages, e. g., *éducation - education* and *théorie - theory*. A small number of very old, "genetic" cognates go back all the way to Proto-Indoeuropean, e. g., *mère - mother* and *pied - foot*. Other cognates can be traced to the conquest of Gaul by Germanic tribes after the collapse of the Roman Empire, and by the period of French domination of England after the Norman conquest.

While our focus is on French and English, the methods that we describe are also applicable to other language pairs. Nowadays, new terms related to modern technology are often adopted in similar form across completely unrelated lan-

guages. Even if languages are written in distinct scripts, approximate phonetic transcription of orthographic data is relatively straightforward in most cases. For example, after transcribing the Japanese word for *sprint* from the Katakana script into semi-phonetic *supurinto*, it is possible to detect its similarity to a French word *sprinter*, which has the same meaning.

2 Related Work

Previous work on automatic cognate identification is mostly related to bilingual corpora and translation lexicons. Simard et al. (Simard *et al.* 92) use cognates to align sentences in bitexts. They employ a very simple test: French-English word pairs are assumed to be cognates if their first four characters are identical. Brew and McKelvie (Brew & McKelvie 96) extract French-English cognates and false friends from bitexts using a variety of orthographic similarity measures. Mann and Yarowsky (Mann & Yarowsky 01) automatically induce translation lexicons on the basis of cognate pairs. They found that edit distance with variable weights outperformed both hidden Markov models and stochastic transducers. Kondrak (Kondrak 04) identifies genetic cognates directly in the vocabularies of related languages by combining the phonetic similarity of lexemes with the semantic similarity of glosses. Kondrak & Dorr 04) report that a simple average of several orthographic similarity measures outperforms all individual measures on the task of the identification of drug names.

For French and English, substantial work on cognate detection was done manually. LeBlanc and Seguin (LeBlanc & Séguin 96) collected 23,160 French-English cognate pairs from two general-purpose dictionaries: Robert-Collins (Robert-Collins 87) and Larousse-Saturne (Dubois 81). 6,447 of the cognates had identical spelling, disregarding diacritics. Since the two dictionaries contain approximately 70,000 entries, cognates appear to make up over 30% of the vocabulary.

The use of cognates in second language teaching was shown to accelerate vocabulary acquisition and to facilitate reading comprehension tasks (LeBlanc *et al.* 89). Morphological rules for conversion from English to French were also proved to help. Tréville (Tréville 90) proposed 25 such rules. An example is: *cal* → *que* in pairs such as *logical* - *logique*, *political* - *politique*.

3 Background

3.1 Definitions

We adopt the following definitions. The definitions are language-independent, but the examples are pairs of French and English words, respectively.

Cognates, or True Friends (Vrais Amis), are pairs of words that are perceived as similar and are mutual translations. The spelling can be identical or not, e. g., *nature* - *nature*, *recognition* - *reconnaissance*.

False Friends (Faux Amis) are pairs of words in two languages that are perceived as similar but have different meanings, e. g., *main* "hand" - *main*, *blesser* "to injure" - *bless*.

Partial Cognates are pairs of words that have the same meaning in both languages in some but not all contexts. They behave as cognates or as false friends, depending on the sense that is used in each context. For example, in French, *facteur* means not only "factor", but also "mailman", while *étiquette* can also mean "label".

Genetic Cognates are word pairs in related languages that derive directly from the same word in the ancestor (proto-) language. Because of gradual phonetic and semantic changes over long periods of time, genetic cognates often differ in form and/or meaning, e. g., *père* - *father*, *chef* - *head*. This category excludes lexical borrowings, i. e., words transferred from one language to another at some point of time, such as *concierge*.

Unrelated pairs are words that exhibit no orthographic similarity. They can be translations of each other, e. g., *glace* - *ice*, but not necessarily, e. g., *glace* - *chair*.

3.2 Orthographic Similarity Measures

Many different orthographic similarity measures have been proposed. Their goal is to quantify human perception of similarity, which is often quite subjective. In this section, we briefly describe the measures that we use as features for the cognate classification task.

- IDENT is a baseline measure that returns 1 if the words are identical, and 0 otherwise.
- PREFIX is a simple measure that returns the length of the common prefix divided by the

length of the longer string.[1] E. g., the common prefix for *factory* and *fabrique* has length 2 (the first two letters) which, divided by the length of 8, yields 0.25.

- DICE (Adamson & Boreham 74) is calculated by dividing twice the number of shared letter bigrams by the total number of bigrams in both words:

$$\text{DICE}(x,y) = \frac{2|bigrams(x) \cap bigrams(y)|}{|bigrams(x)| + |bigrams(y)|}$$

where *bigrams(x)* is a multi-set of character bigrams in word x. E. g., DICE(*colour*, *couleur*) = 6/11 = 0.55 (the shared bigrams are *co*, *ou*, *ur*).

- TRIGRAM is defined in the same way as DICE, but employs trigrams instead of bigrams.

- XDICE (Brew & McKelvie 96) is also defined in the same way as DICE, but employs "extended bigrams", which are trigrams without the middle letter.

- XXDICE (Brew & McKelvie 96) is an extension of the XDICE measure that takes into account the positions of bigrams. Each pair of shared bigrams is weighted by the factor:

$$\frac{1}{1+(pos(a)-pos(b))^2}$$

where $pos(a)$ is the string position of the bigram a.[2]

- LCSR (Melamed 99) stands for the Longest Common Subsequence Ratio, and is computed by dividing the length of the longest common subsequence by the length of the longer string. E. g., LCSR(*colour*, *couleur*) = 5/7 = 0.71

- NED is a normalized edit distance. The edit distance (Wagner & Fischer 74) is calculated by counts up the minimum number of edit operations necessary to transform one word into another. In the standard definition, the edit operations are substitutions, insertions, and deletions, all with the cost of 1. A normalized edit distance is obtained by dividing the total edit cost by the length of the longer string.

- SOUNDEX (Hall & Dowling 80) is an approximation to phonetic name matching. SOUNDEX transforms all but the first letter to numeric codes and after removing zeroes truncates the resulting string to 4 characters. For the purposes of comparison, our implementation of SOUNDEX returns the edit distance between the corresponding codes.

- BI-SIM, TRI-SIM, BI-DIST, and TRI-DIST belong to a family of n-gram measures (Kondrak & Dorr 04) that generalize LCSR and NED measures. The difference lies in considering letter bigrams or trigrams instead of single letter (i. e., unigrams). For example, BI-SIM finds the longest common subsequence of bigrams, while TRI-DIST calculates the edit distance between sequences of trigrams. n-gram similarity is calculated by the formula:

$$s(x_1 \ldots x_n, y_1 \ldots y_n) = \frac{1}{n} \sum_{i=1}^{n} id(x_i, y_i)$$

where $id(a,b)$ returns 1 if a and b are identical, and 0 otherwise.

4 The Data

The training dataset that we used consists of 1454 pairs of French and English words (see Table 1). They were extracted from the following sources:

1. An on-line[3] bilingual list of 1047 basic words and expressions. (After excluding multi-word expressions, we manually classified 203 pairs as Cognates and 527 pairs as Unrelated.)
2. A manually word-aligned bitext (Melamed 98). (We manually identified 258 Cognate pairs among the aligned word pairs.)
3. A set of exercises for Anglophone learners of French (Tréville 90) (152 Cognate pairs).
4. An on-line[4] list of "French-English False Cognates" (314 False-Friends).

A separate test set is composed of 1040 pairs (see Table 1), extracted from the following sources:

[1] The PREFIX measure can be seen as a generalization of Simard et al. (Simard *et al.* 92) approach.

[2] The original definition of XXDICE does not specify which bigrams should be matched if they are not unique within a word. In our implementation, we match non-unique bigrams in the order of decreasing positions, starting from the end of the word.

[3] http://mypage.bluewin.ch/a-z/cusipage/basicfrench.html

[4] http://french.about.com/library/fauxamis/blfauxam.htm

	Training set	Test set
Cognates	613 (73)	603 (178)
False-Friends	314 (135)	94 (46)
Unrelated	527 (0)	343 (0)
Total	1454	1040

Table 1: The composition of data sets. The numbers in brackets are counts of word pairs that are identical (ignoring accents).

1. A random sample of 1000 word pairs from an automatically generated translation lexicon. (We manually classified 603 pairs as Cognates and 343 pairs as Unrelated.)
2. The above-mentioned on-line list of "French-English False Cognates" (94 additional False-Friends).

In order to avoid any overlap between the two sets, we removed from the test set all pairs that happened to be already included in the training set. The dataset has a 2:1 imbalance in favour of the class Cognates/False-Friends; this is not a problem for the classification algorithms (the precision and recall values are similar for both classes in the experiments presented in Section 5). All the Unrelated pairs in our datasets are translation pairs. It would have been easy to add more pairs that are not translations, but we wanted to preserve the natural proportion of cognates in the sample translation lexicons.

5 Evaluation

We present evaluation experiments using the two datasets described in Section 4: a training/development set, and a test set. We classify the word pairs on the basis of similarity into two classes: Cognates/False-Friends and Unrelated. Cognates are distinguished from False-Friends by virtue of being mutual translations. We report the accuracy values for the classification task (the precision and recall values for the two classes are similar to the accuracy values). We test various feature combinations for our classification task. We test each orthographic similarity measure individually, and we also average the values returned by all the 13 measures. Then, in order to combine the measures, we run several machine learning classifiers from the Weka package.

5.1 Results on the Training Data Set

Table 2 presents the results of testing each of the 13 orthographic measures individually. For each measure, we need to choose a specific similarity threshold for separating Cognates/False-Friends from the Unrelated pairs. For the IDENT measure, the threshold was set to 1 (identical spelling ignoring accents). For the rest of the measures, we determined the best thresholds by running Decision Stump classifiers with a single feature. Decision Stumps are Decision Trees that have a single node containing the feature value that produces the best split. The values of the thresholds obtained in this way are also included in Table 2.

Orthographic similarity measure	Threshold	Accuracy
IDENT	1	43.90%
PREFIX	0.03845	92.70%
DICE	0.29669	89.40%
LCSR	0.45800	92.91%
NED	0.34845	93.39%
SOUNDEX	0.62500	85.28%
TRI	0.0476	88.30%
XDICE	0.21825	92.84%
XXDICE	0.12915	91.74%
BI-SIM	0.37980	94.84%
BI-DIST	0.34165	94.84%
TRI-SIM	0.34845	95.66%
TRI-DIST	0.34845	95.11%
Average measure	0.14770	93.83%

Table 2: Results of each orthographic similarity measure individually, on the training dataset. The last line presents a new measure which is the average of all measures for each pair of words.

The training dataset for machine learning experiments consists of 13 features for each pair of words: the values of the 13 orthographic similarity measures. We trained several machine learning classifiers from the Weka package: OneRule (a shallow Decision Rule that considers only the best feature and several values for it), Naive Bayes, Decision Trees, Instance-based Learning (IBK), Ada Boost, Multi-layered Perceptron, and a light version of Support Vector Machine.

The Decision Tree classifier has the advantage of being relatively transparent. Some of the nodes in the decision tree contain counter-intuitive decisions. For example, one of the leaves classifies an instance as Unrelated if the BI-SIM value is *greater* than 0.3. Since all measures attempt to assign high values to similar pairs and low values to dissimilar pairs, the presence of such a node suggest overtraining. One possible remedy to this problem is more aggressive pruning. We kept lowering the *confidence level* threshold from the default $CF = 0.25$ until we obtained a tree without

Classifier	Accuracy on training set	Accuracy cross-val
Baseline	63.75%	63.75%
OneRule	95.94%	95.66%
Naive Bayes	94.91%	94.84%
Decision Trees	97.45%	95.66%
DecTree (pruned)	96.28%	95.66%
IBK	99.10%	93.81%
Ada Boost	95.66%	95.66%
Perceptron	95.73%	95.11%
SVM (SMO)	95.66%	95.46%

Table 3: Results of several classifiers for the task of detecting Cognates/False-Friends versus Unrelated pairs on the training data (cross-validation).

```
TRI-SIM <= 0.3333
|   TRI-SIM <= 0.2083: UNREL (447.0/17.0)
|   TRI-SIM > 0.2083
|   |   XDICE <= 0.2: UNREL (97.0/20.0)
|   |   XDICE > 0.2
|   |   |   BI-SIM <= 0.3: UNREL (3.0)
|   |   |   BI-SIM > 0.3: CG_FF (9.0)
TRI-SIM > 0.3333: CG_FF (898.0/17.0)
```

Figure 1: Example of Decision Tree classifier, heavily pruned (confidence threshold for pruning CF=16%).

counter-intuitive decisions, at $CF = 0.16$ (Figure 1). Our hypothesis was that the latter tree would perform better on a test set.

The results presented in the rightmost column of Table 3 are obtained by 10-fold cross-validation on training dataset (the data is randomly split in 10 parts, a classifier is trained on 9 parts and tested on the tenth part; the process is repeated for all the possible splits). We also report, in the middle column, the results of testing on the training set: they are artificially high, due to overtraining. The baseline algorithm in the Table 3 always chooses the most frequent class in the dataset, which happened to be Cognates/False-Friends. The best classification accuracy (for cross-validation) is achieved by Decision Trees, OneRule, and Ada Boost (95.66%). The performance equals the one achieved by the TRI-SIM measure alone in Table 2.

Error analysis: We examined the misclassified pairs for the classifiers built on the training data. There were many shared pairs among the 60–70 pairs misclassified by several of the best classifiers. Most of the false negatives were genetic cognates that have different orthographic form due to changes of language over time. False positives, on the other hand, were mostly caused by accidental similarity. Several of the measures are particularly sensitive to the initial letter of the word, which is a strong clue of cognation. Also, the presence of an identical prefix made some pairs look similar, but they are not cognates unless the word roots are related.

5.2 Results on the Test Set

The rightmost column of Table 4 shows the results obtained on the test set described in Section 4. The accuracy values are given for all orthographic similarity measures and for the machine learning classifiers that use all the orthographic measures as features. The classifiers are the ones built on the training set.

The ranking of measures on the test set differs from the ranking obtained on the training set, which may be caused by the absence of genetic cognates in the test set. Surprisingly, only the Naive Bayes classifier outperforms the simple average of orthographic measures. The pruned Decision Tree shown in Figure 1 achieves higher accuracy than the overtrained Decision Tree, but still below the simple average. Among the individual orthographic measures, XXDICE performs the best, supporting the results on French-English cognates reported in (Brew & McKelvie 96). Overall, the measures that performed best on the training set achieve more than 93% on the test set. We conclude that our classifiers are generic enough: they perform very well on the test set.

5.3 The Genetic Cognates Dataset

Greenberg (Greenberg 87) gives a list of "most of the cognates from French and English". The list serves as an illustration how difficult it would to demonstrate that French and English are genetically related by examining only the genetic cognates between those two languages. We transcribed the list of 82 cognate pairs from IPA to standard orthography. We augmented the list with 14 pairs from the Comparative Indoeuropean Data Corpus[5]and 17 pairs that we identified ourselves. The final list contains 113 true genetic cognates that go back to Proto-Indoeuropean[6].

We decided to also test the classifier trained in Section 5.1 on this genetic cognates set. The results are shown in the middle column of Table 4. Among the individual measures, the best accuracy is achieved by SOUNDEX, because it is

[5]http://www.ntu.edu.au/education/langs/ielex/

[6]http://www.cs.ualberta.ca/~kondrak/cognatesEF.html

Classifier (measure or combination)	Accuracy on genetic cognates set	Accuracy on test set
IDENT	1.76%	55.00%
PREFIX	36.28%	90.97%
DICE	13.27%	93.37%
LCSR	24.77%	94.24%
NED	23.89%	93.57%
SOUNDEX	39.82%	84.54%
TRI	4.42%	92.13%
XDICE	15.92%	94.52%
XXDICE	13.27%	95.39%
BI-SIM	29.20%	93.95%
BI-DIST	29.20%	94.04%
TRI-SIM	35.39%	93.28%
TRI-DIST	34.51%	93.85%
Average measure	36.28%	94.14%
Baseline	—	66.98%
OneRule	35.39%	92.89%
Naive Bayes	29.20%	94.62%
Decision Trees	35.39%	92.08%
DecTree (pruned)	38.05%	93.18%
IBK	43.36%	92.80%
Ada Boost	35.39%	93.47%
Perceptron	42.47%	91.55%
SVM (SMO)	35.39%	93.76%

Table 4: Results of testing the classifiers built on the training set (individual measures and machine learning combinations). The middle column tests on the set of 113 genetic cognate pairs. The rightmost column tests on the test set of 1040 pairs.

designed for semi-phonetic comparison. Most of the simple orthographic measures perform poorly. The misclassifications are due to radical changes in spelling, such as: *frère* - *brother*, *chaud* - *hot*, *chien* - *hound*, *faire* - *do*, *fendre* - *bite*. One exception is PREFIX, which can be attributed to the fact that the initial segments are the most stable diachronically. TRI-SIM and TRI-DIST also did relatively well, thanks to their robust design based on approximate matching of trigrams. The IDENT measure is almost useless here because there are only two identical pairs (*long* - *long*, *six* - *six*) among the 113 pairs. Since the set contains only cognates, our baseline algorithm would achieve 100% accuracy by always choosing the Cognates/False Friends class.

The results on genetic cognates suggest that a different approach may be more appropriate when dealing with closely related languages (e.g., Dutch and German), which share a large number of genetic cognates. For such languages, recurrent sound and/or letter correspondences should also be considered. Methods for detecting recurrent exist (Tiedemann 99; Kondrak 04) and could be used to improve the accuracy on genetic cognates. However, for languages that are unrelated or only remotely related, the identification of genetic cognates is of little importance. For example, in our lexicon sample of 1000 words, only 4 out of 603 French-English cognate pairs were genetic cognates.

5.4 Three-way Classification

We also experimented with a three-way classification into Cognates, False-Friends and Unrelated. We used an extra feature in our machine learning experiments, which is set to 1 if the two words are translations of each other, and to 0 otherwise. Since all the examples of pairs of class Unrelated in our training set were mutual translations, we had to add Unrelated pairs that are not translations. (Otherwise all pairs with the translation feature equal to 0 would have been classified as False-Friends by the machine learning algorithms.) We generated these extra pairs automatically, by taking French and English words from the existing Unrelated pairs, and pairing them with words other then their pairs. We manually checked to insure that all these generated pairs were not translations of each other by chance.

As expected, this experiment achieved slightly lower results than the ones from Table 2 when running on the same dataset (cross-validation). Most of the machine learning algorithms (except the Decision Tree) did not perfectly separate the Cognate/False-Friends class. We conclude that it is better to do the two-way classification that we presented above (into Cognates/False-Friends and Unrelated), and then split the first class into Cognates and False-Friends on the basis on the value of the translation feature. Nevertheless, the three-way classification could still be useful provided that the translation feature is assigned a meaningful score, such as the probability that the two words occur as mutual translations in a bitext.

6 Conclusion and Future Work

We presented several methods to automatically identify cognates and false friends. We tested a number of orthographic similarity measures individually, and then combined them using several different machine learning classifiers. We evaluated the methods on a training set, on a test set, and on a list of genetic cognates. The re-

sults show that, for French and English, it is possible to achieve very good accuracy even without the training data by employing orthographic measures of word similarity.

In future work we plan to automatically identify partial cognates, which have senses that behave as cognates and senses that behave as false friends. Word sense disambiguation would make it possible to place the partial cognates in their context. We plan to use translation probabilities from a word-aligned parallel corpus. Another direction of future work is to produce complete lists of cognates and false friends, given two vocabulary lists for the two languages. We would also like to apply our method to other pairs of languages (since the orthographic similarity measures are not language-dependent) and to include our lists of cognates and false friends into language learning tools.

Acknowledgments

We thank to Lise Duquette for giving us the idea to work on this project. Our research is supported by the Natural Sciences and Engineering Research Council of Canada, Social Sciences and Humanities Research Council of Canada, the University of Ottawa, and the University of Alberta.

References

(Adamson & Boreham 74) George W. Adamson and Jillian Boreham. The use of an association measure based on character structure to identify semantically related pairs of words and document titles. *Information Storage and Retrieval*, 10:253–260, 1974.

(Al-Onaizan *et al.* 99) Y. Al-Onaizan, J. Curin, M. Jahr, K. Knight, J. Lafferty, D. Melamed, F. Och, D. Purdy, N. Smith, and D. Yarowsky. Statistical machine translation. Technical report, Johns Hopkins University, 1999.

(Brew & McKelvie 96) Chris Brew and David McKelvie. Word-pair extraction for lexicography. In *Proceedings of the 2nd International Conference on New Methods in Language Processing*, pages 45–55, Ankara, Turkey, 1996.

(Dubois 81) Marguerite M. Dubois. *Saturn Larousse French-English, English-French Dictionary: Dictionnaire Larousse Saturne Français-Anglais-Français*. French & European Publications, 1981.

(Gass 87) S.M. Gass, editor. *The use and acquisition of the second language lexicon (Special issue)*. Studies in Second Language Acquisition 9(2), 1987.

(Greenberg 87) Joseph H. Greenberg. *Language in the Americas*. Stanford University Press, Stanford, CA, USA, 1987.

(Hall & Dowling 80) Patrick A. V. Hall and Geoff R. Dowling. Approximate string matching. *Computing Surveys*, 12(4):381–402, 1980.

(Kondrak & Dorr 04) Grzegorz Kondrak and Bonnie Dorr. Identification of confusable drug names: A new approach and evaluation methodology. In *Proceedings of COLING 2004: 20th International Conference on Computational Linguistics*, pages 952–958, 2004.

(Kondrak 04) Grzegorz Kondrak. Combining evidence in cognate identification. In *Proceedings of Canadian AI 2004: 17th Conference of the Canadian Society for Computational Studies of Intelligence*, pages 44–59, 2004.

(LeBlanc & Séguin 96) Raymond LeBlanc and Hubert Séguin. Les congénères homographes et parographes anglais-français. In *Twenty-Five Years of Second Language Teaching at the University of Ottawa*, pages 69–91. University of Ottawa Press, 1996.

(LeBlanc *et al.* 89) Raymond LeBlanc, Jean Compain, Lise Duquette, and Hubert Séguin, editors. *L'enseignement des langues secondes aux adultes: recherches et pratiques*. Les Presses de l'Université d'Ottawa, 1989.

(Mann & Yarowsky 01) Gideon S. Mann and David Yarowsky. Multipath translation lexicon induction via bridge languages. In *Proceedings of NAACL 2001: 2nd Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 151–158, 2001.

(Melamed 98) I. Dan Melamed. Manual annotation of translational equivalence: The Blinker project. Technical Report IRCS #98-07, University of Pennsylvania, 1998.

(Melamed 99) I. Dan Melamed. Bitext maps and alignment via pattern recognition. *Computational Linguistics*, 25(1):107–130, 1999.

(Prado 96) Marcial Prado. *NTC's Dictionary of Spanish False Cognates*. McGraw-Hill, 1996.

(Ringbom 87) H. Ringbom. *The Role of the First Language in Foreign Language Learning*. Multilingual Matters Ltd., Clevedon, England, 1987.

(Robert-Collins 87) Robert-Collins. *Robert-Collins French-English English-French Dictionary*. Collins, London, 1987.

(Simard *et al.* 92) Michel Simard, George F. Foster, and Pierre Isabelle. Using cognates to align sentences in bilingual corpora. In *Proceedings of the 4th International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 67–81, Montreal, Canada, 1992.

(Tiedemann 99) Jörg Tiedemann. Automatic construction of weighted string similarity measures. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, College Park, Maryland, USA, 1999.

(Tréville 90) Marie-Claude Tréville. *Rôle des congénères interlinguaux dans le dévelopment du vocabulair réceptif*. Unpublished PhD thesis, Université de Montreal, 1990.

(Tufis 02) Dan Tufis. A cheap and fast way to build useful translation lexicons. In *Proceedings of COLING 2002: 19th International Conference on Computational Linguistics*, pages 1030–1036, 2002.

(Wagner & Fischer 74) Robert A. Wagner and Michael J. Fischer. The string-to-string correction problem. *Journal of the ACM*, 21(1):168–173, 1974.

(Witten & Frank 00) Ian H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools with Java implementations*. Morgan Kaufmann, San Francisco, USA, 2000.

Robust stochastic parsing using optimal maximum coverage

Vladimír Kadlec[1] **and Marita Ailomaa**[2]
and Jean-Cédric Chappelier[2] **and Martin Rajman**[2]
[1] Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic;
[2] I&C - IIF, EPFL, 1015 Lausanne, Switzerland
E-mail: xkadlec@fi.muni.cz,
{marita.ailomaa,jean-cedric.chappelier,martin.rajman}@epfl.ch

Abstract

This paper presents a robust syntactic parser that is able to return a "correct" derivation tree even if the grammar cannot generate the input sentence. The following two steps solution is proposed: the corresponding most probable optimal maximum coverage is generated first, then the trees from this coverage are glued into one resulting tree. The technique was tested on the ATIS and Susanne corpora and experimental results, as well as conclusions on performance, are provided.

There are many NLP applications (e.g. with speech recognition or dialog systems) where it is difficult to find a context-free grammar (CFG) that generates a sufficient subset of the processed language (under-generation problem). In addition, when the coverage of the grammar is improved, the accuracy usually decreases. Our goal is to develop a robust syntactic parser that is able to return a "correct" derivation tree even if the grammar cannot generate the input sentence.

In previous works, a variety of approaches have been proposed to robustly handle natural language (Carroll & Briscoe 96). Some techniques are based on modifying the input sentence, for example by removing words that disturb the fluency (Bear *et al.* 92; Heeman & Allen 94). More recent approaches are based on selecting the right sequence of partial analyses (Worm & Rupp 98; vanNoord *et al.* 99). Minimum Distance Parsing (Hipp 92) is a third approach based on relaxing the formal grammar, allowing rules to be modified by insertions, deletions and substitutions.

The definition of correctness is however strongly dependent on the target application and our framework allows to change the correctness criteria to fit various application needs. We propose the following two steps solution:

- for the sentence to analyze, the corresponding most probable optimal maximum coverage is generated first (see sections 1 and 2);
- then the possibly partial trees from this coverage are "glued" into one resulting tree (see section 3).

The implementation of the robust parser is discussed in section 4.

1 Coverage

For a given sentence a *coverage*, with respect to an input grammar G, is a sequence of non-overlapping, possibly partial, derivation trees, such that the concatenation of the leaves of these trees corresponds to the whole input sentence.

Because of restriction to derivation trees (i.e. trees fulfilling the left most non-terminal rewriting convention) cases as depicted in figure 1 are not considered as coverages.

For an arbitrary derivation tree T, its foliage $f(T)$ is defined as the sequence of its leaves. So for a coverage $C = (T_1, T_2, ..., T_k)$ of the input sentence $w_1 w_2 ..., w_n$ we have:

$$f(T_1)f(T_2)...f(T_k) = w_1 w_2 ... w_n.$$

In other words, if we define $f_i(T)$ as i-th leaf of T and $f_{last}(T)$ as the last leaf of T, then for coverage $C = (T_1, T_2, ..., T_k)$ of the input sentence $w_1 w_2 ... w_n$ we have:

$$f_1(T_1) = w_1,\ f_{last}(T_k) = w_n \text{ and}$$
$$\text{if } f_{last}(T_i) = w_j \text{ for some } 1 \leq i < k \text{ and}$$
$$1 \leq j < n \text{ then } f_1(T_{i+1}) = w_{j+1}.$$

See figure 2 for an example.

If there are no unknown words in the input sentence, then at least one trivial coverage is obtained, consisting of these trees that use only lexical rules (i.e. one rule per tree).

1.1 Maximum coverage

Consider the subsumed relation $\prec$ is a relation over coverages such that, for any coverages C and C':

$C' \prec C$ iff $\exists i, j, k,\ 1 \leq i \leq k, 1 \leq j$ and there exists rule r in the grammar such that

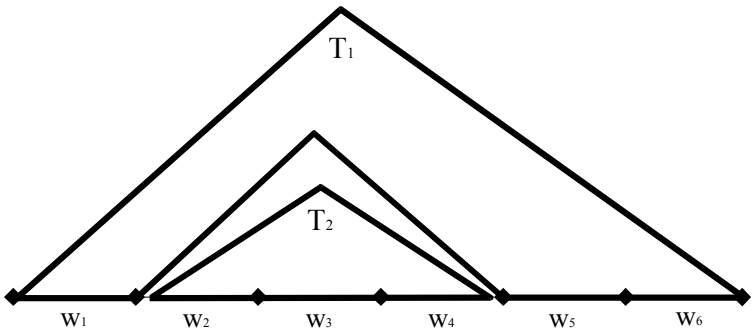


Figure 1: Partial trees, that can not be composed into a coverage: T_1 is actually not a derivation tree.

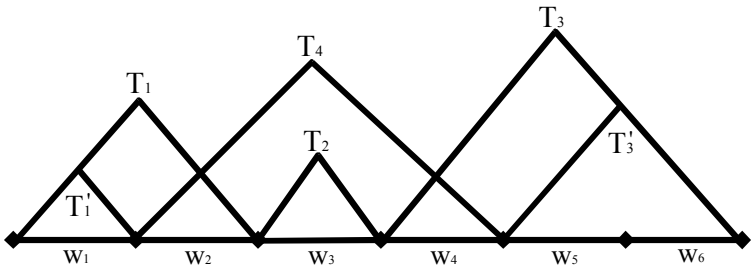


Figure 2: Coverage $C = (T_1, T_2, T_3)$ consisting of trees T_1, T_2 and T_3. If there are $T_1^{'}$ and $T_3^{'}$, $T_1^{'}$ is a subtree of tree T_1 and $T_3^{'}$ is a subtree of T_3, then we also have coverage $C^{'} = (T_1^{'}, T_4, T_3^{'})$. Conversely $(T_1, T_3^{'})$ and (T_1, T_4, T_3) are not coverages.

$$C = (T_1, ..., T_i, ..., T_k),$$
$$C^{'} = (T_1, ...T_{i-1}, T_1^{'}, T_2^{'}, ..., T_j^{'}, T_{i+1}, ..., T_k) \text{ and}$$
$$T_i = r \circ T_1^{'} \circ T_2^{'} ... \circ T_j^{'},$$

i.e. if there exists a sub-sequence of trees in $C^{'}$ that can be connected by rule r and the resulting tree is element of C, the other trees in $C^{'}$ being the same as in C. Notice that the rule r can be a unary rule.

The relation $\leq$ defined as the reflexive and transitive closure of the relation $\prec$, is also antisymmetric. Indeed, if $C^{'} \leq C$ and $C \leq C^{'}$ then:

- $|C^{'}| \leq |C|$ and $|C| \leq |C^{'}|$, so $|C^{'}| = |C|$, where $|C|$ denotes number of trees in the coverage $|C|$.

- If $C^{'} \prec C$ then $\exists T, T^{'}, T \in C, T^{'} \in C^{'}$ such that $T = r_1 \circ T^{'}$ for some unary rule r_1 from grammar G. If also $C \prec C^{'}$ then $T^{'} = r_2 \circ T$. But this is not possible, because $T = r_1 \circ T^{'}$. Notice that all the remaining corresponding trees in C and $C^{'}$ have to be the same. Thus $C^{'} \not\prec C$ and $C \not\prec C^{'}$. And also $C = C^{'}$, because the relation $\leq$ is reflexive closure of the relation $\prec$.

Thus the relation $\leq$ corresponds to a partial order on the set of all coverages of a given input sentence. A maximum coverage (m-coverage) is a coverage that is maximum with respect to the $\leq$ relation. See figure 3 for an example.

Notice if there is a successful parse (a single derivation tree that covers whole input sentence) then there are as many m-coverages as full parse trees for that sentence and every m-coverage contains only one tree.

1.2 Optimal m-coverage

In addition to maximality, we focus on *optimal* m-coverage (OMC), where optimality can be defined with respect to different measures. In contrast to maximality, which is defined for the coverages in general, the choice of a optimality measure depends on the target application.

Here we propose the following two measures:

- the first optimality measure S_1 relates to the average width (number of leaves) of the derivation trees in the coverage. For an m-coverage $C = (T_1, T_2, ...T_k)$ of input sentence $w_1, w_2, ..., w_n$, $n > 1$, we define

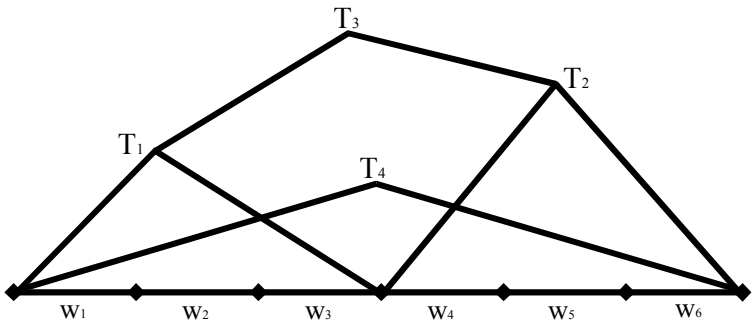


Figure 3: An example to illustrate a maximum coverage. The coverage $C_1 = (T_3)$ is m-coverage. The coverage $C_2 = (T_1, T_2)$ is not maximum, because $C_2 \leq C_1$. There is also another m-coverage $C_3 = (T_4)$. Notice that C_1 and C_3 are not comparable by $\leq$ relation.

$$S_1(C) = \frac{1}{n-1}(\frac{n}{k} - 1).$$

Notice that $0 \leq S_1(C) \leq 1$ and $\frac{n}{k}$ is the average width of the derivation trees in the coverage. With this measure, the value of a trivial coverage (i.e. exclusively made of lexical rules) is 0 and the value of a successful full parse is 1.

- The second measure favours coverages with the widest trees (trees with the largest number of leaves). We define

$$l_{max}(C) = \max_{T \in C} |f(T)|$$

and

$$S_2(C) = \frac{1}{n-1}(l_{max}(C) - 1)$$

for number of input words $n > 1$. Similarly to S_1, $0 \leq S_2(C) \leq 1$, and the value obtained for a trivial coverage is 0 and the value of a successful full parse is 1.

Several other optimality measures could be defined. For instance, an optimality measure might be sensitive to the internal structure of the trees in a coverage, e.g. count the number of nodes in trees. These additional criteria can be used in a combination with measures S_1 and S_2. See figure 4 for an example.

1.3 Probability of a coverage

The probability of a coverage is defined as the product of the probabilities of the trees it contains, i.e. for a coverage C we define

$$p(C) = \prod_{T \in C} p(T).$$

Notice that, by construction, the probability of any coverage is always less than or equal to the probability of the corresponding trivial coverage. The probability of a coverage can be viewed as another optimality measure. So the most probable coverages can be found in the same way as optimal m-coverages. But, usually we find all optimal m-coverages first (optimal with respect to some other measure than probability) and then the most probable one is chosen. Both OMC and most probable OMC are not necessarily unique.

2 Finding optimal m-coverage

We use a bottom-up parsing algorithm that produces all possible incomplete parses (i.e. whenever there exists a derivation tree that covers the part of the given input sentence, the algorithm stores that tree). Then, the incomplete parses can be combined to find the maximum coverage(s).

The described algorithm finds OMC with respect to the measure S_1 (the average width of the derivation trees in the coverage), but it can be easily adapted to different optimality measures.

All operations are applied to a set of Earley's items (Earley 70). In particular, no changes are made during the parsing phase (except some initialization of internal structures for better efficiency of the algorithm).

The Dijkstra's algorithm for shortest path problem in graphs is used to find OMCs with respect to the measure S_1. The input graph for the Dijkstra's algorithm consists of weighted edges and vertices. The edges are Earley's items and the weight of each edge is 1. The vertices are word positions, thus for n input words we have $n + 1$ vertices. Whenever the Dijkstra's algorithm finds

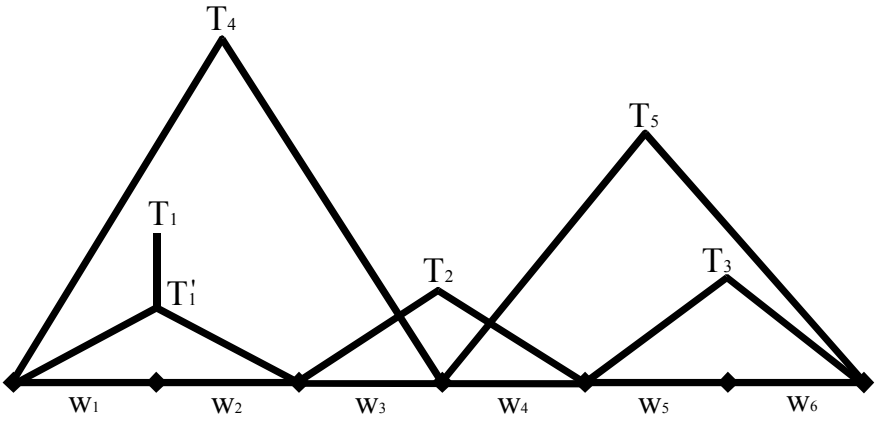


Figure 4: Figure illustrates m-coverages $C_1 = (T_1, T_2, T_3)$ and $C_2 = (T_4, T_5)$. The coverage $C_1^{'} = (T_1^{'}, T_2, T_3)$ is not m-coverage. The coverage C_2 is more optimal for the measure S_1: $S_1(C_1) < S_1(C_2)$, but it is less optimal for the measure S_2: $S_2(C_2) < S_2(C_1)$. Notice that the coverages C_1 and C_2 are not comparable with the $\leq$ relation.

paths with equal length (i.e. identical number of items), we use the probability to select the most probable ones. Notice that, if we assume that there are no unknown words, there exists at least one path from position 0 to n corresponding to the trivial coverage. The worst-case running time for the algorithm is $O(n^2)$ (Dijkstra 59). Figure 5 illustrates an example of the input graph for the Dijkstra's algorithm .

The output of the algorithm is a list of Earley's items. These items can represent several derivation trees and, to get the most probable OMC, the most probable tree from each item is selected. The resulting OMC is not unique because there could be several trees with the same probability.

3 Gluing

The intended result for our robust parser is a derivation tree covering the whole input sentence. For this reason our goal is to connect (glue) the trees present in the OMC to construct a single one.

The gluing can be realized by adding new rule(s) to the grammar. We impose the constraint that the new rules use new non-terminals and just connect the roots of the trees together. Notice that there might be several other ways of constructing a unique tree and therefore our choice mainly rely on technical reasons.

Figure 6 shows an example of gluing with new rules added to the grammar.

4 Experiments

The SLP toolkit (Chappelier & Rajman 98) is used to implement the above mentioned ideas. It provides fast and robust bottom-up chart parsing algorithm derived from Earley's chart parsing (Earley 70) and CYK (Kasami 65; Younger 67; Aho & Ullman 72; Graham *et al.* 80).

The robust parsing technique presented in the previous sections was tested on subsets of two treebanks, ATIS (Hemphill *et al.* 90) and Susanne (Sampson 94). From these treebanks two separate grammars were extracted having different characteristics. Concretely each treebank was divided into a learning set that was used for producing the probabilistic grammar and a test set that was then parsed with the extracted grammar. Around 10% of the sentences in the test set were not covered by the grammar. These sentences represented the real focus of our experiments, as the goal of a robust parser is to process the sentences that the initial grammar fails to describe.

For each sentence the 1-best derivation tree was categorized as good, acceptable or bad, depending on how closely it corresponded to the reference tree in the corpus and how useful the syntactic analysis was for extracting a correct semantic interpretation. The results are presented in table 1. It may be argued that the definition of a "useful" analysis might not be decidable only by observing the syntactic tree. Although we found this to be a quite usable hypothesis during our experi-

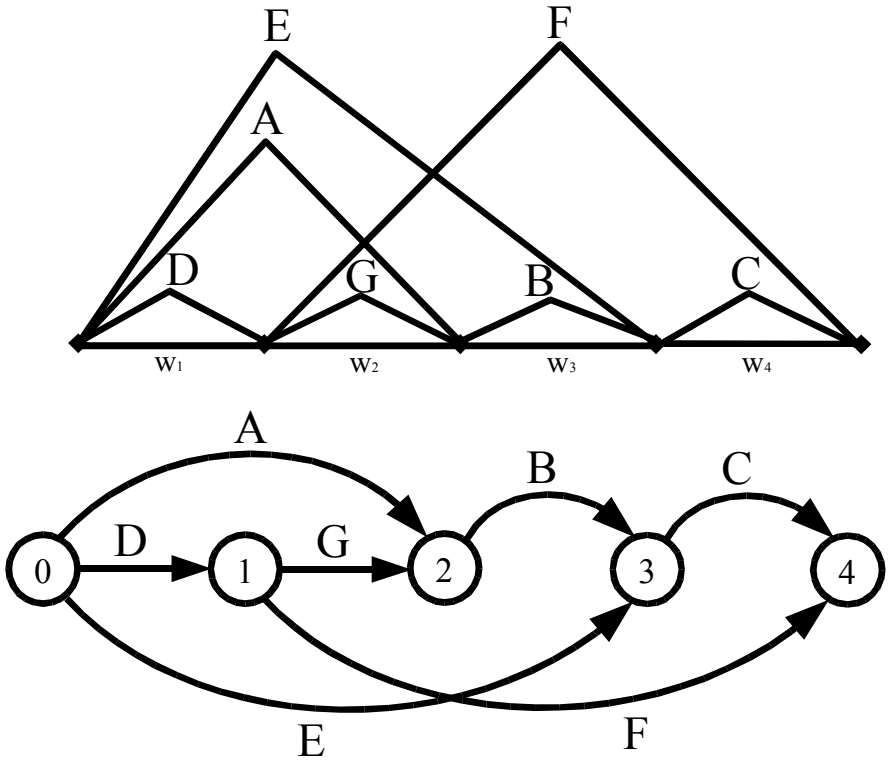


Figure 5: The input graph for the Dijkstra's algorithm and the corresponding derivation trees for Earley's items $[A, 0, 2]$, $[B, 2, 3]$, $[C, 3, 4]$, $[D, 0, 1]$, $[E, 0, 3]$, $[F, 1, 4]$ and $[G, 1, 2]$. The shortest paths are $[E, 0, 3]$, $[C, 3, 4]$ and $[D, 0, 1]$, $[F, 1, 4]$. The paths correspond to two optimal m-coverages with two trees in each coverage.

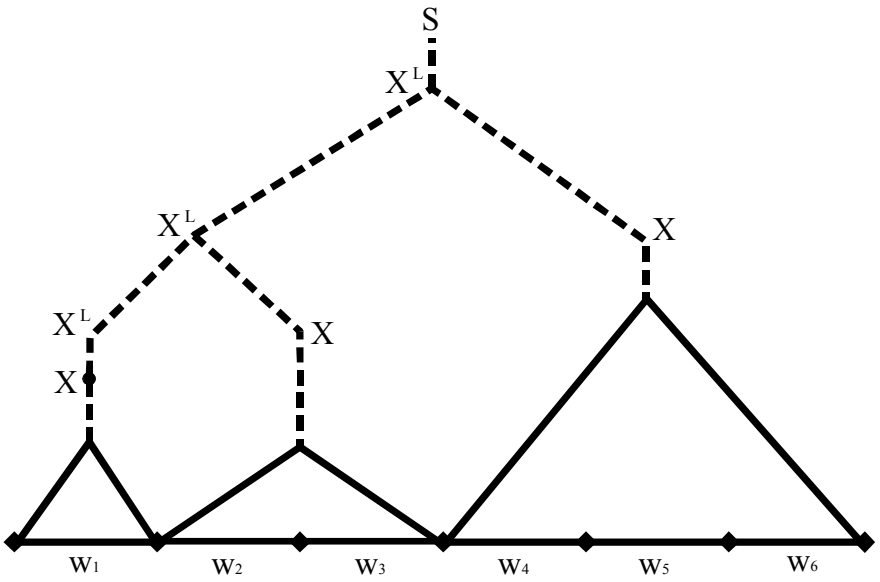


Figure 6: Gluing with new rules $S \rightarrow X^L$, $X^L \rightarrow X^L X$, $X^L \rightarrow X$, $X \rightarrow A_i$, where S is the root of the grammar, X and X^L are new non-terminals and A_i is the root of the i-th tree in the coverage (we have three trees in this example). The dotted lines represent newly added rules. Bottom bold trees correspond to the OMC.

	Good (%)	Acceptable (%)	Bad (%)
ATIS corpus	10	60	30
Susanne corpus	16	29	55

Table 1: Experimental results. Percentage of good, acceptable and bad analyses.

ments, some more objective procedure should be defined. In a concrete application, the usefulness might for example be determined by the actions that the system should perform based on the produced syntactic analysis.

From the experimental results one can see that, our technique behaves better with the ATIS grammar that has relatively few rules, than with Susanne, which is a considerably larger grammar describing a rich variety of syntactic structures.

The number of bad 1-best analyses that are produced can be explained by the fact that the probabilistically best analysis is not always the linguistically best one. This is a non-trivial problem related to all types of natural language parsing, not only to robust parsers.

5 Conclusion

In this paper we presented our approaches to the robust stochastic parsing. We introduced the optimal maximum coverage framework and several measures for the optimality of the parser. Our definition of the maximality is independent of the target application. On the other hand, the choice of an optimality measure is strongly application dependent. We proposed the algorithm that finds OMC (with respect to the measure average width of derivation trees) efficiently.

The evaluation of the robust parsing technique was based on manually checking the derivation trees. An important issue is to integrate the technique into some target application so that we have more realistic ways of measuring the usefulness of the produced robust analyses. In the near future we plan to repeat this experiment on a larger treebank and to use a more rigorous evaluation method like Parseval labeled precision and recall.

Acknowledgments

This work has been partly supported by Czech Science Foundation under the project 201/05/2781 and by the project T100300414 of the Information Society Program (the Thematic Program II of the National Research Program of the Czech Republic).

References

(Aho & Ullman 72) A.V. Aho and J.D. Ullman. *The Theory of Parsing, Translation and Compiling*, volume I: Parsing. Prentice-Hall, Englewood Cliffs, N.J., 1972.

(Bear *et al.* 92) John Bear, John Dowding, and Elizabeth Shriberg. Integrating multiple knowledge sources for the detection and correction of repairs in human-computer dialogue. In *Proceedings of the 30th ACL*, pages 56–63, Newark, Delaware, 1992.

(Carroll & Briscoe 96) John Carroll and Ted Briscoe. Robust parsing — a brief overview. In John Carroll, editor, *Proceedings of the Workshop on Robust Parsing at the 8th European Summer School in Logic, Language and Information (ESSLLI'96), Report CSRP 435*, pages 1–7, COGS, University of Sussex, 1996.

(Chappelier & Rajman 98) J.-C. Chappelier and M. Rajman. A generalized CYK algorithm for parsing stochastic CFG. In *TAPD'98 Workshop*, pages 133–137, Paris, France, 1998. (`http://slptk.sourceforge.net`).

(Dijkstra 59) E. W. Dijkstra. A note on two problems in connection with graphs. *Numerische Math*, (1):269–271, 1959.

(Earley 70) J. Earley. An efficient context-free parsing algorithm. In *Communications of the ACM*, volume 13, pages 94–102, 1970.

(Graham *et al.* 80) S.L. Graham, M.A. Harrison, and W.L. Ruzzo. An improved context-free recognizer. *ACM Transactions on Programming Languages and Systems*, 2(3):415–462, 1980.

(Heeman & Allen 94) Peter A. Heeman and James F. Allen. Detecting and correcting speech repairs. In *Proceedings of the 32th ACL*, pages 295–302, Las Cruces, New Mexico, 1994.

(Hemphill *et al.* 90) Charles T. Hemphill, John J. Godfrey, and George R. Doddington. The atis spoken language systems pilot corpus. In *Proc. of DARPA Speech and Natural Language Workshop*, pages 96–101, Hidden Valley, PA, 1990.

(Hipp 92) Dwayne R. Hipp. *Design and development of spoken natural language dialog parsing systems.* Unpublished PhD thesis, Duke University, 1992.

(Kasami 65) T. Kasami. An efficient recognition and syntax analysis algorithm for context-free languages. In *Technical report AF CRL-65-758*, Bedford, Massachusetts, 1965. Air Force Cambridge Research Laboratory.

(Sampson 94) G. Sampson. The Susanne corpus, release 3. In *School of Cognitive & Computing Sciences*, Brighton (England), 1994. University of Sussex, Falmer.

(vanNoord *et al.* 99) Gertjan van Noord, Gosse Bouma, Rob Koeling, and Mark-Jan Nederhof. Robust grammatical analysis for spoken dialogue systems. *Natural Language Engineering*, 5(1):45–93, 1999.

(Worm & Rupp 98) Karsten L. Worm and C. J. Rupp. Towards robust understanding of speech by combination of partial analyses. In *Proceedings of the 13th biennial European Conference on Artificial Intelligence (ECAI'98), August 23-28*, pages 190–194, Brighton, UK, 1998.

(Younger 67) D.H. Younger. Recognition of context-free languages in time n^3. *Inf. Control*, 10(2):189–208, 1967.

Extracting Predicate Structures from Parse Trees

Manfred Klenner
Institute of Computational Linguistics
University of Zurich
klenner@cl.unizh.ch

Abstract

We evaluate two different approaches for extracting predicate structures from parse trees. We compare the results of a rule-based algorithm incorporating decision tree learning to an integer linear programming (ILP) approach with an underlying statistical model. It turns out that the rule-based approach yields higher precision but lower recall than the ILP approach. Both approaches achieve precision rates of more than 90 %.

1 Introduction

Recently, much attention has been paid to semantic role labeling (e.g. the CoNLL-2004 and CoNLL-2005 shared task). The task is to identify the semantic roles of a verb, i.e. which phrase (e.g. NP, PP, S) realizes which semantic role (e.g. agent, patient). A related problem is grammatical relation finding (e.g. identifying the subject of a verb) which can be done as part of parsing or as a separate process on top of parse trees or chunks (Buchholz, 2002). While for some applications semantic role labeling goes too far, grammatical role finding is not sufficient. Especially in logic based approaches, predicate structures are more common than frames with semantic roles. An example of a predicate structure is: believe(peter,like(mary,books_of(max_frisch))). Of course, these structures can, in principle, be derived from semantic role labeled verb frames, but the question is whether there is a more direct way. Grammatical relations (GR), on the other hand, provide useful information to support predicate argument extraction, if a mapping from grammatical relations to argument positions of verb predicates is assumed. For example: the subject of an active verb is mapped to the first argument position of the underlying verb predicate.

Syntactic parsing and grammatical relation finding for unrestricted text requires robust, statistical approaches. The resulting structures (e.g parse trees) are noisy: tagging errors, attachment mistakes and wrong case assignments (i.e. grammatical relation identification is false) are to be expected. As a consequence, a robust method for the extraction of predicate structures from parse trees is needed as well.

2 Tools and Resources

We use the BitPar parser (Schmid , 2004) and a treebank grammar (Schiehlen , 2004) derived from the Negra corpus (Brants et al., 1999), a German tree bank of 20.000 sentences. Although the grammar does not specify GR, case is assigned to noun phrases. The case feature serves as an indicator of GR (e.g. nominative case indicates subject). Note that clauses (e.g. complement clauses) do not bear case, the decision whether they are verb complements (and thus arguments of predicates) or not, cannot be drawn from functional information, thus.

In our experiments, we used 1001 manually extracted predicate structures (the gold standard) derived from 870 sentences. There were 16 three-placed predicates, 512 two-placed and 454 one-placed. Because of the low frequency, we omitted the three-placed predicates from our experiments. That is, no rules are being learned for three-placed predicates.

Since the parser does not identify heads, we defined a head heuristic. Its precision is $> 99\%$. Given the 2543 heads in our corpus of 870 sentences, 25 head assignments are wrong (in the worst case). Note, that these mistakes propagate to the precision of the rule learner and the ILP approach.

3 The Problem of Argument Assignment

There are two problems to be solved: an identification problem (which heads of which phrases

are arguments) and an assignment problem (which argument position do they fill). Input is a parse tree, output is a predicate structure. For example, the sentence "Das Volk laesst die Scharia kalt (Sharia law leaves people cold)" is mapped onto "kalt_lassen(scharia,volk)" (leaves_cold(sharia, people)). As previously mentioned, the statistical parse is imperfect: sometimes there are more e.g. nominative heads than possible subjects. Sometimes case assignment is wrong (e.g accusative instead of nominative) or case is even missing. Especially, verbal heads never get case but they are potential verb arguments (complement clauses). There are tagging errors (e.g. a head is tagged as a non-head category) and also attachment mistakes (dislocated complements) are taking place.

In other words, there is noise in the data, and a statistical or machine learning approach could help to recover from those flaws.

4 Learning Interpretation Rules

Information in parse trees is structurally encoded. To extract it, the structural patterns need to be identified. This could be done either manually (writing semantic interpretation rules) or automatically. In both cases, reliable data (a gold standard) for evaluation purposes is needed. The second variant, however, has some obvious advantages. Extraction rules must be tailored to the tree format produced by the parser. If in the course of the lifetime of a system the parser is replaced by a better one (or a better version of the old) new interpretation rules must be either manually written or automatically derived. The later alternative is clearly preferable. But there is another reason why to prefer the second solution: the noise. Machine Learning approaches are better than humans to cope with noisy data (at least given mass data).

Given a set of syntax trees produced by a statistical parser and given a gold standard of manually extracted predicate structures that corresponds to these parse trees, interpretation rules can be learned by a simple procedure. Each predicate structure unambiguously identifies a verb (via the predicate name) and the complements of the verb (via the predicate arguments). The basic rule learning algorithm is as follows:

starting from the verb node in the syntax tree

- search for an *anchor* node, i.e., a predecessor of the verb node (often its mother) which dominates all verb complements
- save the paths from the anchor to the complement heads in a left to right order
- save the features of each node of the path (e.g. syntactic label, case)
- save the mapping (how is the linear order of the parse tree projected onto the order of predicate arguments)

Assume the (partial) syntax tree given in Fig. 1 and its gold standard predicate structure " bearbeite(er,Konzerte)" (adapt(he, concerts)).

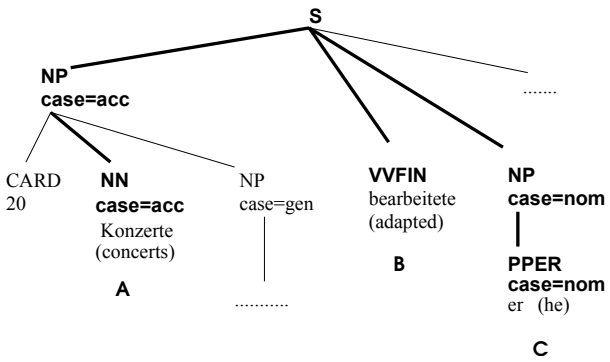


Figure 1: Fragment Indicating a Rule Pattern

The direct object ($concerts$) precedes the verb, the subject (he) follows the verb. 'S' is the anchor node and there are three paths connecting the anchor to the heads (including the verb). The underlying structural pattern, the extraction rule derived from that positive example, is highlighted (bold) - see Fig. 2 for a rule representation. The anchor node has category S. It is the root of three paths: A,B,C. NN, VVFIN and PPER

```
anchor: S
A=[NP, NN with case=acc]
B=[VVFIN]
C=[NP, PPER with case=nom]
Linear Precedence: A < B < C
Mapping: B(C,A)
```

Figure 2: Rule Representation

```
A=[NP, {PPER|NN|PRF} with case=acc]
B=[VVFIN]
C=[NP,{PPER|NN} with case=nom]
```

Figure 3: Generalized Rule

are the leave nodes. The words attached to these nodes form the arguments of the predicate. *Linear Precedence* fixes the order in which these paths are given in the parse tree and *Mapping* is used to construct the predicate structure from the words at the leave nodes.

5 Evaluation

We run the algorithm on the 870 sentences Input was the set of parse trees, output were the learned rules. 223 rules were generated in the training set at each run (on the average). There were 147 (idiosyncratic) rules stemming from exactly 1 positive examples. The rest of the rules covers 2 or more examples (up to 70 per rule). On the average, every rule thus covers four positive examples. That is a poor verbs per rule ratio. We found however that these rules often are just minor variants of each other. We implemented a rule generalization component that reduced the 223 rules to 81 (leaving 29 idiosyncratic rules). The generalized rules assemble structural patterns with identical paths, but different categorical realizations of leave nodes. Fig. 3 shows a generalized rule covering all the categorical variants of the rule from Fig. 2. E.g., the leave of path A might be a personal pronoun (PPER), a normal noun (NN) or a reflexive pronoun (PRF).

We evaluated precision and recall on the training set and test set for 1-ary and 2-ary predicates, respectively (see Fig. 4). Input was the set of parse trees together with the learned rules, output were the predicate argument structures found.

	Prec_{train}	Rec_{train}	Prec_{test}	Rec_{test}
2	84.2	98.4	79.1	70.9
1	99.8	99.7	99.3	86.6

Figure 4: Evaluation Results

First of all, precision on the 2-ary predicates in the training set is low (84.2 %). There are various reasons for this. As already mentioned, errors stemming from the head heuristic propagate to the rule learner. If the wrong head (of a np) is chosen, the predicate structure will have an incorrect argument. Moreover, parsing errors might result in erroneous extraction rules. To give an example consider the wrong (case) parse: "Das Volk$_{[nom]}$ laesst die Scharia$_{[acc]}$ kalt (Sharia law leaves people cold). "Das Volk (people)" is the direct object (accusative case). However, the parser attached the nominative case. Actually, this is a morphologically licensed assignment. "Das Volk" and "die Scharia" can be nominative or accusative, respectively. Since it is more likely to have the subject (nominative) preceding the verb, the parser did a reasonable but erroneous job by assigning nominative case to "Das Volk". Since the gold standard predicate structure, i.e. leave_cold(sharia,people), has *people* as the second argument, a rule is generated that maps the head of a nominative np to the second argument position of the verb predicate. This way contradicting rules are generated: one that maps nominative to the first argument position (correct decision) and one that maps nominative to the second. Both rules have the some triggering conditions (i.e. the same paths), they produce conflicting interpretations (and reduce precision). In the experimental setting reported in Fig. 4 and Fig. 5 every rule that matches is applied. We also implemented a version of the rule learner that does rule weighting and deletes contradicting rules keeping the rules with the higher weight (see section 7).

	Prec	Rec	F-meas
train	92.0	99.1	95.4
test	89.2	78.8	83.7

Figure 5: Summary of the Results

Fig. 5 gives a summary of the results and provides the values of the f-measure. Note that a recall < 100 on the training set stems from errors of the head heuristic. Rules are learned according to the gold standard, that is, with perfect head information. But in the evaluation the head heuristic is used. If it fails, the wrong argument is extracted and precision drops down. We also defined a simple procedure to fix a base line: every verb gets as its arguments the nominative, accusative and da-

tive heads (in that order) under the anchor node (the dominating S node). If an embedded S node is present, then its head verb fills the last argument position of the predicate. The results are given in Fig. 6.

	Prec	Rec	F-meas
train	71.1	75.3	73.14
test	75.2	72.1	73.61

Figure 6: Base Line

6 Rule Specialization with Decision Trees

Precision drops, if rules classify negative instances as positive (e.g. the case of contradicting rules from the previous discussion). One way to improve rule precision is to make rules more specific. This can be accomplished with a decision tree learner. We incorporated this along the following lines: If the paths of a rule match a syntax tree, the rule is applicable. To make it more specific, a decision tree is attached to each rule to further restrict its application. The decision tree learner is trained with vectors derived from positive and negative examples of the syntax trees accepted by the rule.

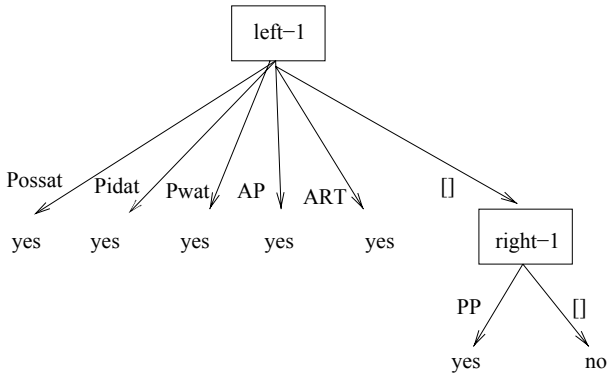


Figure 7: Decision Tree for Rule Application

Fig. 7 shows the very simple decision tree learned for the rule derived from the tree in Fig. 1. We used contextual features to specify the training vectors: the feature values (syntactic label and case) of the sister nodes (left and right neighbors) of the leave nodes of each path. In this learned decision rule, only the left and right sisters of the leave node of path A (cf. Fig. 2) come into play: *left-1* and *right-1*: if the left sister of the leave node is $Possat$ (an attributing possessive pronoun) then the rule triggers. The same is true with other pronouns ($Pidat$, $Pwat$), an adjective phrase (AP) and a determiner (ART). Only if the left (*left-1*) and the right (*right-1*) context are empty ([]), then the rule is not allowed to trigger. Such rules are not very instructive, linguistically. But they work very well (see Fig. 8). Precision goes up to >98 %, however recall drops (70.7 %). Recall drops since rules are getting more specific.

	Pred	Rec	F-meas
train	99.6	98.7	99.1
test	98.4	70.7	82.3

Figure 8: Results of the Decision Tree Version

7 Rule Weighting

The best results were achieved with a version of the rule learner based on a simple form of statistics, namely rule weighting. The measure is:

$$\frac{|\ positive\ examples\ |}{|\ positive\ and\ negative\ examples\ |}$$

If more than one rule applies to a parse tree, then only the rule with the highest score is applied. See Fig. 9 for the results.

	Prec	Rec	F-meas
train	98.9	98.6	98.7
test	98.5	78.4	87.3

Figure 9: Results of the Weighted Version

Precision and recall in the training set are excellent - the f-measure is 98.7%. Also the precision on the data of the test set is good. However, recall is still too low (78.4 %). A low recall means that the rules generated from the training set do not capture enough of the syntactic patterns needed to process the data in the test set. In other words, there is too much variance in the syntax trees - rules are missing.

One have to bear in mind that predicate extraction is simpler than semantic role labeling. Nethertheless, we have argued that a machine

learning approach is sensible, because the structures provided by current parsers for unrestricted text are noisy. The idea is to let the rule extractor learn how to cope with that noise and in the best case it should be able to correct systematic mistakes made by the parser. Our rule learner is straightforward. However, we found it interesting to compare it to the results of a more general machine learning approach. We started with TIMBL (Daelemans et al., 2004), but found it more convenient to use integer linear programming, because linguistic constraints (e.g. that a verb has, say, at most 3 arguments) can be expressed more naturally with ILP than with memory-based learners like TIMBL (where such global constraints are to be modelled as class decisions, which, at least in our experiments, results in a poor performance).

8 ILP for NLP

Integer Linear Programming (ILP) is the name of a class of constraint satisfaction algorithms which are restricted to a numerical representation of the problem to be solved. The objective is to optimize the numerical solution (the *objective function* below). Optimization means maximization or minimization of linear equations. The general form of an ILP specification is given in Fig. 10.

Objective Function:

$$maxf(X_1,\ldots,X_n) := c_1X_1+c_2X_2+\ldots+c_nX_n$$

Constraints:

$$a_{i1}X_1 + a_{i2}X_2 + \ldots + a_{in}X_n \left(\begin{array}{c} \leq \\ = \\ \geq \end{array} \right) b_i,$$

$i = 1,\ldots,m$

X_i are variables, c_i, b_i and a_{ij} are constants.

Figure 10: ILP Specification

The goal is to maximize a n-ary function f, which is defined (':=') as the sum of all c_iX_i. Argument assignment decisions can be modeled in the following way: X_i are binary variables that indicate the (non-)assignment of a head to an argument position of a verb. If the value of X_i is 1, the attachment was successful, otherwise ($X_i = 0$) it failed. c_i and a_{ij} are weights that represent the impact of an assigment; they provide an empirically based numerical justification of the assignment. Finally, the variables b_i are used to restrict the number of X_i that are to be chosen (in our model, a verb predicate can have at most 3 argument positions).

To our knowledge, (Punyakanok et al., 2004) were the first who applied ILP to NLP. Their treatment of semantic role labeling shares some similarities with our approach, however there are differences (see related work).

Given a sentence with a number of verbs v ($v \geq 1$) and a number of heads h (nominal categories or verbs) where $h \geq v$.

1. Determine for each verb the number of arguments it has.

2. Choose for each argument position of a (verb) predicate a head that fills it.

Since $v \geq 1$, variable names must have a verb index, an argument index and a head index. To satisfy (1), a variable v_x is introduced whose value is an integer indicating the number of arguments the verb has. The righthand side of such a v equation sums up variables that represent verb-argument-head assignments. These variables are binary (indicator functions), they realize the variables $X_1,\ldots,X_n$ of the general ILP specification given above. Their format is $v_xa_ih_j$ with

$1 \leq$ x $\leq$ | verbs |
$1 \leq$ i $\leq$ | argument positions |
$1 \leq$ j $\leq$ | heads |

For example, if the ILP algorithm assigns $v_2a_1h_3$ the value 1, the first argument of the second verb predicate is said to be filled by head h_3. We also have to specify variables that consume heads that are not consumed by any verb (i.e. non-arguments), and we have to determine the weights of an assignment decision (see section 9).

The full specification of the ILP formulation of the assignment problem is: [1]

[1]Please note that $v_xa_ih_j$ is one (!) variable name and not a multiplication of v_x, a_i, and h_j.

(C1a) an argument consumes at most one head

$$\sum_j v_x a_i h_j \leq 1, \quad \forall i, x$$

(C1b) a_1 of each verb consumes exactly one head

$$\sum_j v_x a_1 h_j = 1, \quad \forall x$$

(C2) a head is attached at most once (to an argument)

$$\sum_x \sum_i v_x a_i h_j \leq 1, \quad \forall j$$

(C3) the number of arguments is at least one and at most three: $1 \leq v_x \leq 3, \ \forall x$

(C4) the argument assignment of a predicate is:

$$v_x = \sum_i \sum_j v_x a_i h_j$$

(C5) all heads are consumed by predicate variables and non-argument variables (z_i)

$$v_1 + \ldots + v_{|verbs|} + z_1 + \ldots + z_{|heads|} = |heads|$$

(C6) a head is either an argument or a non-argument:

$$z_j + \sum_x v_x a_i h_j = 1, \quad \forall i, j$$

(C7) the impact of the argument assignment as specified in C4 is:

$$i_{v_x} = \sum_i \sum_j w_{v_x a_i h_j} * v_x a_i h_j$$

(C8) the impact of the non-argument assignment is:

$$i_z = \sum_j w_{z_j} * z_j$$

Given such a set of equations where all coefficients are instantiated, the objective function is:

$$max: \ i_z + \sum_x i_{v_x}$$

As a side effect of the maximization, all indicator variables are instantiated, either to 0 (*not chosen*) or to 1 (*chosen*).

cc_1 the case of the head - if there is none (e.g. verbal heads), we use the syntactic label instead

cc_2 the distance from the verb predicate v_x to a predecessor node (the anchor) which dominates the head h_i.

- if the mother of v_x is the anchor, then the distance is set to 1
- if the grandmother of v_x is the anchor (without crossing an 'S' node), the distance is set to 2
- otherwise, the distance is infinite

cc_3 a coordination flag that indicates whether h_i is part of a coordination or not.

Figure 11: Contextual Criteria

9 The Weighting Scheme

We use conditional probabilities to compute the weights of the indicator variables: $P(v_x a_j h_i | contextual\ criteria)$. The contextual criteria are given in Fig. 11.

As usual, independence is assumed:

$$w_{v_x a_j h_i} = P(v_x a_j h_i | cc_1, cc_2, cc_3) = \prod_{k=1}^{n} P(v_x a_j h_i | c_k)$$

These probabilities are estimated with maximum likelihood (we do some smoothing as well), e.g.

$$P(v_x a_1 h_i | case_{h_i} = nom) = \frac{freq(a_1 \wedge case = nom)}{freq(case = nom)}$$

That is, the conditional probability of being argument 1 of some verb v_x and some head h_i (given case=nom) is estimated by the frequency of argument 1 being nominative divided by the frequency of heads being nominative (whether they are argument heads or non-argument heads). The weights w_{z_i} of a non-argument head z_i (cf. C8) are estimated correspondingly.

These contextual features are simple, but we found them sufficient (see the next section for the evaluation). They are simple, but they rely on structural information of parse trees. Thus, their simplicity stems from the results of a complex machinery, namely the parser.

Criterion 1 from Fig. 11 reflects the reliability on the case feature (for nominal objects, not for verbs). Criterion 2 represents the sentence context: is there a head within the same sentence as the verb and at what distance. In rare cases, heads beyond a sentence border might be arguments as well (in e.g. elliptical constructions or given parsing errors). Finally the coordination criterion: if a head is part of a coordination, it is not a good candidate for an argument position (because the whole coordination is the argument). These criteria determine the weight of a decision. The assignment of heads to argument positions of a predicate that has got the highest weight is selected.

10 ILP Compared to the Rule Learner

Fig. 12 shows the results of the ILP approach compared to the rule learner (weighted version).

	Prec	Rec	F-meas
ILP	91.98	89.83	90.89
RL+weight	98.5	78.4	87.3

Figure 12: Comparison of ILP and Rule Learner

ILP is the winner. This is perfectly explainable, although it is a bit amazing that the simple statistical model underlying the ILP optimization task works so fine (given the small amount of data it is based on). The rule learner extracts tree structure fragments as rules. Every unseen structural encoding of semantic information lessens its recall - because there is no rule to apply. The rule learner is in a sense too fine grained: Precision rides on the back of recall. The ILP approach is coarse grained, but in balanced way: precision and recall are close together.

11 Related Work

Semantic role labeling is the topic of a number of articles, for example Gildea & Jurafsky (2002). Their algorithms are based on the FrameNet corpus, a lexical resource of more than 40.000 sentences. They use the output of the Collins Parser to train their statistical model(s). Gildea & Jurafsky (2002) rely (as we do) on structural information in the form of paths, but they do not utilize functional information (e.g. GR).

(Punyakanok et al., 2004) applied integer linear programming to semantic role labeling. They do not use a parser but a chunker and the scoring (statistical) model is provided by the SNoW learning architecture. Our model is inspired by the ILP formulation of (Punyakanok et al., 2004), but there are differences in the formalization; also the features used to train the model are different.

12 Conclusion and Outlook

We have focused on the problem of predicate argument extraction from parse trees. The performance of a rule-based learner is compared to those of an ILP approach. Both approaches have good results, the rule-based one yields a higher precision, but a lower recall than the ILP approach, which has a superior f-measure value. We found ILP a good method to succinctly express linguistic constraints. The underlying statistic model works fine, but our data base is small (1001 predicate argument structures). In order to find out whether our approaches scale up, are reliable and competitive, we have to enlarge our data base.

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet Project. *Proceedings of the COLING-ACL*.

Thorsten Brants, Wojciech Skut, and Hans Uszkoreit 1999. Syntactic Annotation of a German Newspaper Corpus. *Proceedings of the ATALA Treebank Workshop*.

Sabine Buchholz. 2002. *Memory-Based Grammatical Relation Finding*. PhD thesis, Tilburg University, The Netherlands.

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2004. TiMBL: Tilburg Memory Based Learner, version 5.1, Reference Guide *ILK pub: ILK-0402, Tilburg University, The Netherlands.*.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic Labeling of Semantic Roles. *Computational Linguistics 28:3, 245-288*.

Vasin Punyakanok, Dan Roth, Wen-tau Yih, and Dave Zimak. 2004. Role Labeling via Integer Linear Programming Inference. *Proceedings of the 20th International Conference on Computational Linguistics*.

Michael Schiehlen. 2004. Annotation Strategies for Probabilistic Parsing in German. *Proceedings of the 20th International Conference on Computational Linguistics*.

Helmut Schmid. 2004. Efficient Parsing of Highly Ambiguous Context-Free Grammars with Bit Vectors. *Proceedings of the 20th International Conference on Computational Linguistics*.

Tree Edit Distance for Textual Entailment

Milen Kouylekov[1,2] **and Bernardo Magnini**[1]
ITC-irst, Centro per la Ricerca Scientifica e Tecnologica [1]
University of Trento[2]
38050, Povo, Trento, Italy
milen@kouylekov.net,magnini@itc.it

Keywords: textual entailment, tree edit distance, PASCAL-RTE

Abstract

This paper addresses Textual Entailment (i.e. recognizing that the meaning of a text entails the meaning of another text) using a Tree Edit Distance algorithm between the syntactic trees of the two texts. A key aspect of the approach is the estimation of the cost for the editing operations (i.e. insertion, deletion, substitution) among words. Strongly related words are assumed to have high probability of entailment and their substitution has a low cost, while unrelated words have higher cost, making entailment less probable.

The paper compares the contribution of lexical resources for recognizing textual entailment in an experiment carried on over the PASCAL-RTE dataset.

1 Introduction

The problem of language variability (i.e. the fact that the same information can be expressed with different words and syntactic constructs) has been attracting a lot of interest during the years and it poses significant issues in front of systems aimed at natural language understanding. The example below shows that recognizing the equivalence of the statements *came in power*, *was prime-minister* and *stepped in as prime-minister* is a challenging problem.

- Ivan Kostov came in power in 1997.
- Ivan Kostov was prime-minister of Bulgaria from 1997 to 2001.
- Ivan Kostov stepped in as prime-minister 6 months after the December 1996 riots in Bulgaria.

While the language variability problem is well known in Computational Linguistics, a general unifying framework has been proposed only recently in (Dagan and Glickman 2004). In this approach, language variability is addressed by defining the notion of *entailment* as a relation that holds between two language expressions (i.e. a text T and an hypothesis H) if the meaning of H as interpreted in the context of T, can be inferred from the meaning of T. The entailment relation is directional as the meaning of one expression can entail the meaning of the other, while the opposite may not.

The Recognizing Textual Entailment (RTE) task takes as input a T/H pair and consists in automatically determining whether an entailment relation holds between T and H or not. The task, potentially, covers almost all the phenomena in language variability: entailment can be due to lexical variations, as it is shown in example (1), to syntactic variation (example 2), to semantic inferences (example 3) or to complex combinations of all such levels. As a consequence of the complexity of task, one of the crucial aspects for any RTE system is the amount of knowledge required for filling the gap between T and H. The following examples are taken from the RTE-PASCAL dataset:

1. *T* - Euro-Scandinavian media cheer Denmark v Sweden draw.

   H - Denmark and Sweden tie.

2. *T* - Jennifer Hawkins is the 21-year-old beauty queen from Australia. *H* - Jennifer Hawkins is Australia's 21-year-old beauty queen.

3. *T* - The nomadic Raiders moved to LA in 1982 and won their third Super Bowl a year later.

   H - The nomadic Raiders won the Super Bowl in 1983.

In example 1 we need to know that *T* lexically entails *H*; in example 2 we need to understand that the syntactic structures of the text and the hypothesis are

equivalent; finally, in 3 we need to reason about temporal entities.

This paper provides a clear and homogeneous framework for the evaluation of lexical resources for the RTE task. The framework is based on the intuition that the probability of an entailment relation between T and H is related to the ability to show that the whole content of H can be mapped into the content of T. The more straightforward the mapping can be established, the more probable is the entailment relation. Since a mapping can be described as the sequence of editing operations needed to transform T into H, where each edit operation has a cost associated with it, we assign an entailment relation if the overall cost of the transformation is below a certain threshold, empirically estimated on the training data.

Within the Tree Edit Distance (TED) framework, the complexity of RTE is put on the availability of entailment rules and on the definition of cost functions for the three editing operations. In this paper we investigate the role of different resources which provide entailment rules for the definition of cost functions. We have experimented the TED approach with a non annotated document collection and a similarity relation estimated over a corpus of dependency trees. Experiments, carried on the PASCAL-RTE dataset, provide significant insight for future research on RTE.

The paper is organized as follows. Section 2 describes in brief the PASCAL-RTE challenge. Section 3 presents the Tree Edit Distance algorithm we have adopted and its application to dependency trees. Section 4 describes the architecture of the system, the resources we have used and how we have estimated cost functions over them. Section 5 presents the results we have obtained while Section 6 contains a general discussion. Finally, Section 7 describes future work directions.

2 The PASCAL Recognizing Textual Entailment Challenge

The PASCAL-RTE challenge is a recent evaluation campaign which attracted considerably attention (16 different groups participated to the 2005 campaign). The view underlying the RTE challenge (Dagan and Glickman and Magnini 2005) is that different natural language processing applications, including Question Answering (QA), Information Extraction (IE), (multi-document) summarization, and Machine Translation (MT), have to address the language variability problem and would benefit from textual entailment in order to recognize that a particular target meaning can be inferred from different text variants. The different applications address the problem with application-oriented manners and methods and the impact of RTE is evaluated on the final application performance.

The PASCAL-RTE campaign was based on a human annotated dataset of T H pairs, collected from different text processing applications. Each pair corresponds to a success or failure case of an actual application. The collected examples represent a range of different levels of entailment reasoning based on lexical syntactic logical and word knowledge, at different levels of difficulty. The pairs are taken from seven different application scenario:

- Information Retrieval - queries selected by examining prominent sentences in news stories.
- Comparable Documents - comparable news articles that cover a common story.
- Reading Comprehension - exercises in human language teaching.
- Question Answering - Question from CLEF-QA (Cross Language evaluation Forum) and TREC (Text Retrieval Conference).
- Information Extraction - dataset of annotated relations *kill* and *birth place*
- Machine Translation - automatic translations.
- Paraphrase Acquisition.

The most basic inference technique used by participants at PASCAL-RTE is the degree of overlap between T and H. Such overlap is computed using a number of different approaches, ranging from statistic measures like *idf*, deep syntactic processing and semantic reasoning. The difficulty of the task explains the poor performance of the systems, which achieved an accuracy between 50-60%. In the rest of the Section we briefly mention some of the systems which are relevant for the approach we describe in this paper.

In (Bayer et al. 2005) the authors describe two systems for recognizing textual entailment. The first system is based on deep syntactic processing. Both T and H are parsed and converted into a logical form. An event-oriented statistical inference engine is used to separate the TRUE from FALSE pairs. The second system is based on statistical machine translation models.

A system based on frequency-based term weighting in combination with different similarity measures is

presented in (Jijkoun and de Rijke 2005). The weight of the words in the hypothesis is calculated with normalized inverse frequency:

$$ICF(w) = \frac{\#occurrences\_of\_w}{\#occurrences\_of\_all\_words} \quad (1)$$

$$weight(w) = 1 - \frac{ICF(W) - ICF_{min}}{ICF_{max} - ICF_{min}} \quad (2)$$

where ICF_{min} and ICF_{min} are the minimum and maximum inverse frequencies. The second measure is the dependency based word similarity described in (Lin 1998b).

A method for recognizing textual entailment based on graph matching is described in (Raina et al. 2005). To handle language variability problems the system uses a maximum entropy coreference classifier and calculates term similarities using WordNet (Fellbaum 1998) by means of a similarity module based on techniques described in (Pedersen et al. 2004).

An approach based on the BLEU (BiLingual Evaluation Understudy) algorithm (Papineni et al. 2001) was presented in (Perez and Alfonseca 2004). The algorithm looks for n-gram coincidences between T and H.

3 Tree Edit Distance on Dependency Trees

We adopted a tree edit distance algorithm applied to the syntactic representations (i.e. dependency trees) of both T and H. A similar use of tree edit distance has been presented by (Punyakanok et al. 2004) for a Question Answering system, showing that the technique outperforms a simple bag-of-word approach. While the cost function presented in (Punyakanok et al. 2004) is quite simple, for the RTE challenge we tried to elaborate more complex and task specific measures.

According to our approach, T entails H if there exists a sequence of transformations applied to T such that we can obtain H with an overall cost below a certain threshold. The underlying assumption is that pairs between which an entailment relation holds have a low cost of transformation. The kind of transformations we can apply (i.e. deletion, insertion and substitution) are determined by a set of predefined entailment rules, which also determine a cost for each editing operation.

We have implemented the tree edit distance algorithm described in (Zhang and Shasha 1990) and applied to the dependency trees derived from T and H. Edit operations are defined at the level of single nodes of the dependency tree (i.e. transformations on subtrees are not allowed in the current implementation). Since the (Zhang and Shasha 1990) algorithm does not consider labels on edges, while dependency trees provide them, each dependency relation R from a node A to a node B has been re-written as a complex label B-R concatenating the name of the destination node and the name of the relation. All nodes except the root of the tree are relabeled in such way. The algorithm is directional: we aim to find the better (i.e. less costly) sequence of edit operation that transform T (the source) into H (the target). According to the constraints described above, the following transformations are allowed:

- **Insertion**: insert a node from the dependency tree of H into the dependency tree of T. When a node is inserted it is attached with the dependency relation of the source label.
- **Deletion**: delete a node N from the dependency tree of T. When N is deleted all its children are attached to the parent of N. It is not required to explicitly delete the children of N as they are going to be either deleted or substituted on a following step.
- **Substitution**: change the label of a node N1 in the source tree into a label of a node N2 of the target tree. Substitution is allowed only if the two nodes share the same part-of-speech. In case of substitution the relation attached to the substituted node is changed with the relation of the new node.

4 System Architecture

The system is composed by the following modules, showed in Figure 1: (i) a text processing module, for the preprocessing of the input T/H pair; (ii) a matching module, which performs the mapping between T and H; (iii) a cost module, which computes the cost of the edit operations.

4.1 Text processing module

The *text processing module* creates a syntactic representation of a T/H pair and relies on a sentence splitter and a syntactic parser. For sentence splitting we used *MXTerm* (Ratnaparkhi 1996), a Maximum entropy sentence splitter. For parsing we used *Minipar*, a principle-based English parser (Lin 1998a) which has high processing speed and good precision.

A relevant problem we encountered, affecting about 30% of the pairs in the dataset we used, is that

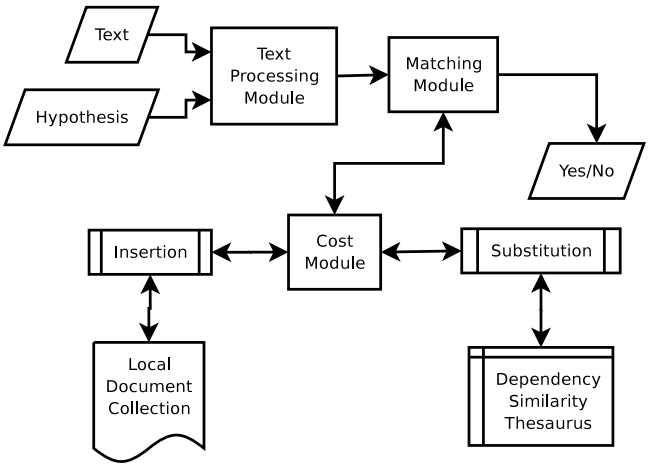


Figure 1: System architecture

the parser represents in a different way occurrences of similar expressions, making harder to apply edit transformations. For instance, "Wal-Mart" and "Wal-Mart Stores inc." have different trees, being "Mart" the governing node in the first case and the governed node in the second. The problem could be addressed by changing the order of the nodes in T which is however complex because it introduces changes in the tree edit-distance algorithm. Another solution, which we intend to explore in the future, is the integration of specialized tools and resources for handling named entities and acronyms. In addition, for about 20% of the pairs, the parser did not produce the right analysis either for T or for H.

4.2 Matching module

The *matching module* finds the best sequence of edit operations between the dependency trees obtained from T and H. It implements the edit distance algorithm described in Section 2.

The entailment score *score* of a given pair is calculated in the following way:

$$score(T,H) = \frac{ed(T,H)}{ed(,H)} \quad (3)$$

where $ed(T,H)$ is the function that calculates the edit distance cost and $ed(,H)$ is the cost of inserting the entire tree H. A similar approach is presented in (Monz and de Rijke 2001), where the entailment score of two document d and d' is calculated by comparing the sum of the weights (idf) of the terms that appear in both documents to the sum of the weights of all terms in d'.

We used a threshold *t* such that if $score(T,H) < t$ then T entails H, otherwise no entailment relation holds for the pair. To set the threshold we have used both the positive and negative examples of the training set provided by the PASCAL-RTE dataset (see Section 5.1 for details).

4.3 Cost Module

The matching module makes requests to the *cost module* in order to receive the cost of single edit operations needed to transform T into H. We have different cost strategies for the three edit operations.

Insertion. The intuition underlying insertion is that its cost is proportional to the relevance of the word w to be inserted (i.e. inserting an informative word has an higher cost than inserting a less informative word). More precisely:

$$Cost[Ins(w)] = Rel(w) \quad (4)$$

where $Rel(w)$, in the current version of the system, is computed on a document collection as the *inverse document frequency (idf)* of w, a measure commonly used in *Information Retrieval*. If N is the number of documents in a text collection and N_w is the number of documents of the collection that contain w then the *idf* of w is given by the formula:

$$idf(w) = \log \frac{N}{N_w} \quad (5)$$

The most frequent words (e.g. stop words) have a zero cost of insertion.

Substitution. The cost of substituting a word w_1 with a word w_2 can be estimated considering the semantic entailment between the words. The more the two words are entailed, the less the cost of substituting one word with the other.

We have used the following formula:

$$Cost[Subs(w_1,w_2)] = Ins(w_2) * (1 - Ent(w_1,w_2)) \quad (6)$$

where $Ins(w_2)$ is calculated using (3) and $Ent(w_1,w_2)$ can be approximated with a variety of relatedness functions between w_1 and w_2.

There are two crucial issues for the definition of an effective function for lexical entailment: first, it is necessary a database of entailment relations with enough coverage; second, we have to estimate a quantitative measure for such relations.

We experimented the use of a dependency based thesaurus available at *http://www.cs.ualberta.ca/lindek/downloads.htm*. For each word, the thesaurus lists up to 200 most similar words and their similarities. The similarities are calculated on a parsed corpus using frequency counts of the dependency triples. A complete review of the method including comparing with different approaches is presented in (Lin 1998b). Dependency triples consists of the head, a dependency type and a modifier. They can be viewed as features for the head and the modifiers in the triples when calculating similarity.

The cost of a *substitution* is calculated by the following formula:

$$Ent_{sim}(w_1, w_2) = sim_{th}(w_1, w_2) \quad (7)$$

where w_1 is the word from T that is being replaced by the word w_2 from H and $sim_{th}(w_1, w_2)$ is the similarity between w_1 and w_2 in the thesaurus multiplied by the similarity between the corresponding relations. The similarity between relations is stored in a database of relation similarities obtained by comparing dependency relations from a parsed local corpus. The similarities have values from 1 (very similar) to 0 (not similar). If there is no similarity, the cost of substitution is equal to the cost of inserting the word *w2*.

Deletion. In the PASCAL-RTE dataset T is typically shorter than H. As a consequence, we expect that much more deletions are necessary to transform T into H than insertions or substitutions. Given this bias toward deletion, in the current version of the system we set the cost of deletion to 0. This expectation has been empirically confirmed (see results of system 2 in Section 5.3).

An example of mapping between the dependency tree of T and H is depicted in Figure 2. The tree on the left is the dependency tree of the text: *Iran is said to give up al Qaeda members*. The tree on the right is the dependency tree corresponding to the hypothesis: *Iran hands over al Qaeda members*. The algorithm finds as the best mapping the subtree with root *give*. The verb *hands* is substituted by the verb *give* because it exists a similarity relation between them in the thesaurus. The blue lines connect nodes that are exactly matched. Nodes connected with the light brown line (give-hands) are substitutions for which the similarity database is used. Nodes in the text that do not participate in a mapping are removed. The lexical modifier *over* of the verb *hands* is inserted.

5 Experiments and Results

We carried out a number of experiments in order to estimate the contribution of different combinations of the available resources. In this section we report about the dataset, the experiments and the results we have obtained.

5.1 Dataset

For the experiments we have used the PASCAL-RTE dataset (Dagan and Glickman and Magnini 2005). The dataset [1] has been collected by human annotators and it is composed of 1367 text (*T*) - hypothesis (*H*) pairs split into positive and negative examples (a 50%-50% split).

Typically, *T* consists of one sentence while *H* was often made of a shorter sentence. The dataset has been split in a training (576 pairs) and a test (800 pairs) part.

5.2 Experiments

The following configurations of the system have been experimented.

System 1: Tree Edit Distance Baseline. In this configuration, considered as a baseline for the Tree Edit Distance approach, the cost of the three edit operations are set as follows:
Deletion: always 0
Insertion: the idf of the word to be inserted
Substitution: 0 if $w_1 = w_2$, infinite in all the other cases.

In this configuration the system just needs a non annotated corpus for estimating the idf of the word to be inserted. Deletion is 0 because we expect much more deletion that insertions, due to the fact that *T* is longer than *H*, implying that much more deletions than insertions are necessary.

System 2: Deletion as idf. In this configuration we wanted to check the impact of assigning a cost to the deletion operation.
Deletion: the idf of the word to be deleted
Insertion: the idf of the word to be inserted
Substitution: same as system 1.

System 3: Similarity Database. This is the same than system 1, but we estimate the cost of substitutions using the similarity database described in Section 4.3. We expect a broad coverage with respect to the previous system.

[1] The dataset is available for download at http://www.pascal-network.org/Challenges/RTE/Datasets

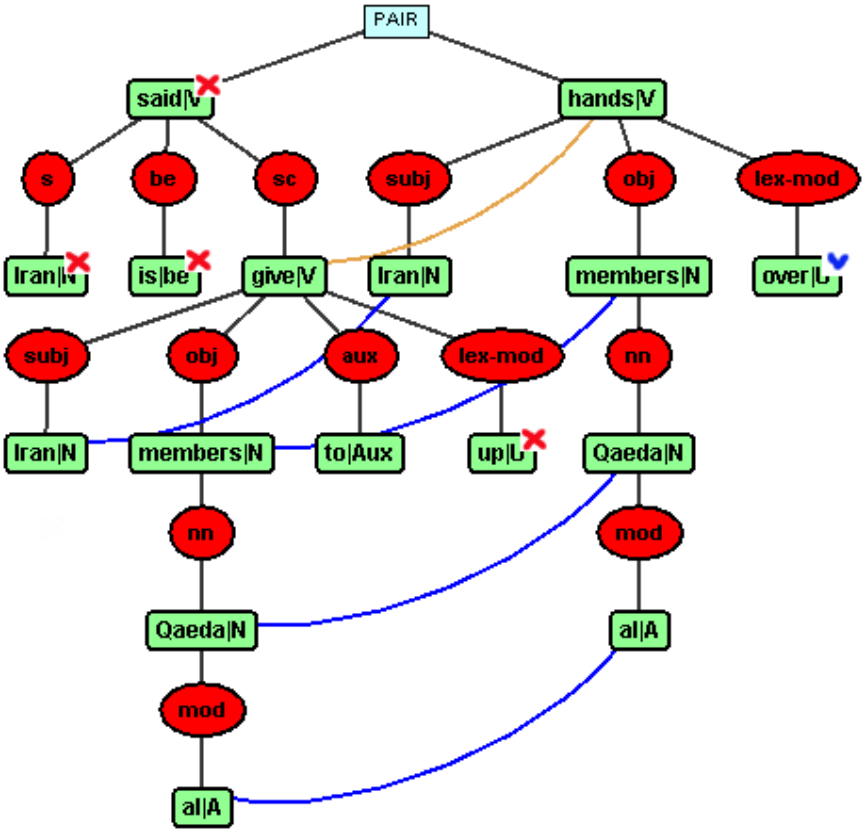


Figure 2: An example of a T H pair mapping.

Deletion: always 0
Insertion: the idf of the word to be inserted
Substitution: same as system 2, plus similarity rules.

5.3 Results

For each system we have tested we report a table with the following data:

- #Attempted: the number of deletions, insertions and substitutions that the algorithm successfully attempted (i.e. which are included in the best sequence of editing transformations).
- %Success: the proportion of deletions, insertions and substitutions that the algorithm successfully attempted over the total of attempted edit transformations.
- Accuracy: the proportion of *T-H* pairs correctly classified by the system over the total number of pairs.
- CWS: Confidence Weighted Score (also known as Average Precision), is given by the formula:

$$cws = \frac{1}{n}\sum_{i=1}^{n}\frac{\#correct - upto - i}{i} \quad (8)$$

where n is the number of the pairs in the test set, and i ranges over the sorted pairs. The Confidence-Weighted Score ranges between 0 (no correct judgments at all) and 1 (perfect classification), and rewards the systems' ability to assign a higher confidence score to the correct judgments than to the wrong ones.

Table 1 shows the results obtained by the three systems we experimented.

The hypothesis about the 0 cost of the deletion operation was confirmed by the results of System 2.

The similarity database used in System 3 increased the number of the successful substitutions made by the algorithm from 25% to 27%. It also increased the performance of the system for both accuracy and cws. The impact of the similarity database on the result is small because of the low similarity between the dependency trees of H and T.

6 Discussion

The approach we have presented can be considered as a framework for testing the contribution of different kinds of linguistic resources for the textual entailment

	System 1	System 2	System 3
#deletions	20325	19984	19148
#insertions	7927	7586	7703
#substitutions	2686	3027	2910
%deletions	0.88	0.87	0.83
%insertions	0.75	0.71	0.73
%substitutions	0.25	0.29	0.27
accuracy	0.560	0.481	0.566
cws	0.615	0.453	0.624

Table 1: Results

task. The intuition is that the performance of the system is correlated with the contribution, in terms of entailment rules, of the used resources. As an example, a wrong substitution with a low cost can significantly affect the optimal cost of the tree mapping. A lesson we learned is that, in order to obtain good results, we should consider for substitution only pairs with high entailment score (in our experiment similarity). The experiments we have carried out show that a word similarity databases coupled with the edit distance algorithm can be used for successfully recognizing textual entailment. However, in order to test the specific contribution of a certain resource, a set of pairs from the RTE dataset which require specific lexical entailment rules must be selected.

The tree edit distance algorithm is designed to work with substitution on the level of tree nodes while our analysis of the PASCAL-RTE dataset show that subtree substitutions are more suitable for the task. Other resources of entailment rules (e.g.paraphrases in (Lin and Pantel 2001), entailment patterns as acquired in (Szpektor et al. 2004)) could significantly widen the application of entailment rules and, consequently, improve performances. We estimated that for about 40% of the true positive pairs the system could have used entailment rules found in entailment and paraphrasing resources. As an example, the pair 565:

> T - Soprano's Square: Milan, Italy, home of the famed La Scala opera house, honored soprano Maria Callas on Wednesday when it renamed a new square after the diva.
>
> H - La Scala opera house is located in Milan, Italy.

could be successfully solved using a paraphrase pattern such as *Y home of X <=> X is located in Y*, which can be found in (Lin and Pantel 2001). However, in order to use this kind of entailment rules, it would be necessary to extend the "single node" implementation of tree edit distance to address editing operations among sub-trees. A system with an algorithm capable of calculating the cost of substitution on the level of subtrees can be used as a framework for testing paraphrase and entailment acquisition systems.

A drawback of the tree edit distance approach is that it is not able to observe the whole tree, but only the subtree of the processed node. For example, the cost of the insertion of a subtree in H could be smaller if the same subtree is deleted from T at a prior or later stage. A context sensitive extension of the insertion and deletion module will increase the performance of the system.

7 Conclusion and Future Work

We have presented an approach for recognizing textual entailment based on tree edit distance applied to the dependency trees of *T* and *H*. We have also demonstrated that using lexical similarity resources can increase the performance of a system based on such algorithm.

In the future we plan to incorporate more resources from which we can derive lexical entailment rules. In particular we would like to compare the performance of a similarity database to WordNet (Fellbaum 1998), a lexical database which includes lexical and semantic relations among word senses. The idea is to define a set of entailment rules over the WordNet relations (hypernym, synonym, entails, pertains, etc.) with their respective probabilities.

In addition, in order to use entailment and paraphrasing resources, we plan to extend the tree edit distance algorithm with sub-tree substitutions.

References

Samuel Bayer, John Burger, Lisa Ferro,John Henderson, Alexander Yeh MITRE's Submissions to the EU Pascal

RTE Challenge *In Proceedings of PASCAL Workshop on Recognizing Textual Entailment* Southampton, UK, 2005

Ido Dagan and Oren Glickman Generic applied modeling of language variability *In Proceedings of PASCAL Workshop on Learning Methods for Text Understanding and Mining* Grenoble, 2004

Ido Dagan, Oren Glickman, Bernardo Magnini The PASCAL Recognizing Textual Entailment Challenge *In Proceedings of PASCAL Workshop on Recognizing Textual Entailment* Southampton, UK 2005

Christiane Fellbaum WordNet, an electronic lexical database *MIT Press, 1998*

Valentin Jijkoun and Maarten de Rijke Recognizing Textual Entailment Using Lexical Similarity *In Proceedings of PASCAL Workshop on Recognizing Textual Entailment* Southampton, UK, 2005

Milen Kouleykov and Bernardo Magnini Recognizing Textual Entailment with Tree edit Distance Algorithms *In Proceedings of PASCAL Workshop on Recognizing Textual Entailment* Southampton, UK, 2005

Dekang Lin Dependency-based evaluation of MINIPAR. *In Proceedings of the Workshop on Evaluation of Parsing Systems at LREC-98.* Granada, Spain, 1998

Dekang Lin An Information-Theoretic Definition of Similarity. *Proceedings of International Conference on Machine Learning, Madison, Wisconsin*, July, 1998.

Dekang Lin and Patrick Pantel. Discovery of inference rules for Question Answering. *Natural Language Engineering, 7(4)*, pages 343-360, 2001

Christof Monz and Maarten de Rijke Light-Weight Entailment Checking for Computational Semantics. *The third workshop on inference in computational semantics (ICoS-3, 2001)*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. B L E U: a method for automatic evaluation of machine translation. *Research report, IBM 2001*

Tedd Pedersen , Siddharth Patwardhan and Jason Michelizzi WordNet::Similarity - Measuring the relatedness of concepts. *AAAI-2004*

Diana Perez and Enrique Alfonseca Application of the BLEU algorithm for recognizing textual entailments *In Proceedings of PASCAL Workshop on Recognizing Textual Entailment* Southampton, UK, 2005

Vasin Punyakanok, Dan Roth and Wen-tau Yih, Mapping Dependencies Trees: An Application to Question Answering *Proceedings of AI & Math*, 2004

Rajat Raina, Aria Haghighi, Christopher Cox, Jenny Finkel, Jeff Michels, Kristina Toutanova Bill MacCartney, Marie-Catherine de Marneffe, Christopher D. Manning, Andrew Y. Ng Robust Textual Inference using Diverse Knowledge Sources. *In Proceedings of PASCAL Workshop on Recognizing Textual Entailment* Southampton, UK, 2005

Adwait Ratnaparkhi A Maximum Entropy Part-Of-Speech Tagger. *In proceeding of the Empirical Methods in Natural Language Processing Conference, May 17-18*, 1996

Idan Szpektor, Hristo Tanev, Ido Dagan, and Bonaventura Coppola 2004 Scaling Web-based Acquisition of Entailment Relations *In Proceedings of EMNLP-04 - Empirical Methods in Natural Language Processing*, Barcelona, July 2004

Kaizhong Zhang ,Dennis Shasha. Fast algorithm for the unit cost editing distance between trees. *Journal of algorithms, vol. 11, p. 1245-1262*, December 1990.

Using Language Resource Independent Detection for Spanish Named Entity Recognition

Zornitsa Kozareva* **Oscar Ferrández** **Andres Montoyo** and **Rafael Muñoz**
Departamento de Lenguajes y Sistemas Informaticos
University of Alicante
Carretera San Vicente S/N
Alicante 03690, Spain
{zkozareva,ofe,montoyo,rafael}@dlsi.ua.es

Abstract

In this paper we propose a language resource independent Named Entity detection module, developed and tested over Spanish and Portuguese. The impact of various feature combinations was studied. We examined the differences in language models learned by three data-driven systems performing the same NLP tasks and how they can be exploited to yield a higher accuracy than the best individual system.

1 Introduction

The increasing flow of digital information requires the extraction, filtering and classification of pertinent information from large volumes of texts. For this task, Named Entity (NE) recognition and classification modules play important role. For English the available resources and the developed systems outnumber. However for Spanish, Portuguese or eastern European languages, where the resources as gazetteers[1], annotated corpora are not sufficient, or some tools such as POS taggers, syntactic analyzers even might not be developed, we should not forget that the need is still the same.

This fact motivated us to start the development of a language resource independent NER system during its detection phase and using less resources while classifying into LOC, PER and ORG classes.

In this paper, we present a NE system developed for Spanish, using three machine learning algorithms: Hidden Markov Model from ICOPOST[2] toolkit (Schröder 02); Maximum Entropy implemented by (Suárez & Palomar 02) and Memory-based learner from TiMBL's package (Daelemans *et al.* 03).

* This research has been partially funded by the Spanish Government under project CICyT number TIC2003-0664-C02-02 and PROFIT number FIT-340100-2004-14 and by the Valencia Government under project numbers GV04B-276 and GV04B-268.

[1] catalogues of names of people, locations, organizations etc.

[2] http://acopost.sourceforge.net/

For entity detection, a language resource independent and portable set was used. Initially this set was tested for Spanish, but when applied to Portuguese the experiments demonstrated how features valid for Spanish were directly adopted by Portuguese. For improving overall NE performance, feature selection and systems' combination were done. Aiming at minimal feature space, less processing time and gaining high performance while restraining from gazetteers, morphological or syntactic analyzers, the obtained results are quite encouraging. For Spanish 92.96% f-score was reached for entity detection using the language portable set and 78.59% f-score for entity classification. For Portuguese we gained 78.86% f-score, due to the insignificant amount of training data.

2 Feature description

For NE detection and classification task, the Memory-based learning and Maximum Entropy classifiers utilize the features described below. HMM takes only the three most informative attributes.

2.1 Features for NE detection

For NE detection, the well-known BIO model was employed. There a tag shows that a word is at the beginning of a NE (B), inside a NE (I) or outside a NE (O). For the sentence: *Paulo Suarez es mi amigo.* , the following tags have been associated, "B I O O O O ". *Paulo* starts the named entity; *Suarez* continues it, while the other words and the full stop are not part of a NE.

The original set for BIO is composed of the features described in Figure 1. We denote this set by *A*. For *aSubStr* attribute, we extracted substrings of the anchor word, knowing that some prefixes and suffixes are good indicators for certain classes of entities. Taking into account the morphological structure of a word and its paradigm, suffixes as *-er,-or,-ista* imply person's occupation *pianista,*

- **a**: anchor word (e.g. the word to be classified)
- ***c[1-6]***: word context at position ±1, ±2, ±3
- ***C[1-7]***: word capitalization at position 0, ±1, ±2, ±3
- ***d[1-3]***: word +1,+2,+3 in dictionary of entities
- ***p***: position of anchor word
- ***aC***: capitalization of the whole anchor word
- ***aD***: anchor word in any dictionary
- ***aT***: anchor word in dictionary of trigger words
- ***wT***: word at position ±1, ±2, ±3 in a dictionary of trigger words
- ***aL***: lema of the anchor word
- ***aS***: stem of the anchor word
- ***aSubStr[1-5]***: ±2, ±3 and half substring of the anchor word

Figure 1: Features for NE detection

futbolista, profesor, director, others as *-ez* meaning "son of", indicate Spanish surnames. This information helped us both for the detection and classification task.

2.2 Features for NE classification

For classification, the first seven features used by the BIO model (e.g. a, c[1-6], p) were incorporated together with the set described in Figure 2. The gazetteers for gP, gL and gO attributes, have been collected from the yellow pages.

3 Classifier combination and Data

3.1 Classifier combination

It is a well-known fact that if several classifiers are available, they can be combined in various ways to create a system that outperforms the best individual classifier. Since we had several classifiers, it was reasonable to investigate combining them in different ways. The simplest approach is through voting. The outputs of the various models are examined and the classification with weight exceeding some threshold is selected. It is possible to assign varying weights to the models, in effect giving one model more importance than the others. In our system, we assigned to each model the weight corresponding to the correct class it determines.

3.2 Data and its evaluation

The Spanish train and test data we used are part of the CoNLL-2002 (Sang 02) corpus. For training we had corpus containing 264715 tokens and

- ***eP***: entity is trigger PER
- ***eL***: entity is trigger LOC
- ***eO***: entity is trigger ORG
- ***eM***: entity is trigger MISC
- ***tP***: word ±1 is trigger PER
- ***tL***: word ±1 is trigger LOC
- ***tO***: word ±1 is trigger ORG
- ***gP***: part of NE in gazetteer for PER
- ***gL***: part of NE in gazetteer for LOC
- ***gO***: part of NE in gazetteer for ORG
- ***wP***: whole entity is PER
- ***wL***: whole entity is LOC
- ***wO***: whole entity is ORG
- ***NoE***: whole entity not in one of the defined three classes
- ***f***: first word of the entity
- ***s***: second word of the entity
- ***clx***: capitalization, lowercase, other symbol

Figure 2: Features for NE classification

18794 entities and for testing we used Test-B corpus with 51533 tokens and 3558 entities.

The Portuguese corpus is part of HAREM-2005[3] competition having 68597 tokens and 3094 entities for training, and 22624 tokens and 1013 entities for testing.

Scores were computed per NE class. *Conlleval*[4] evaluation script was used in order to obtain comparable results to the CoNLL-2002 systems.

4 NE recognition

Our NER system is composed of two passages

1. detection: identification of sequence of words that make up the name of an entity.

2. classification: deciding to which category our previously recognized entity should belong.

We started our experiments with set $C24 = A/\{aSubStr[1-5]\}$, which contained the attributes as lemma, dictionaries, trigger words etc. The obtained results have been satisfactory as can be seen in Table 1, but since we have been searching for an appropriate feature set F that maximizes the performance, minimizes the computational cost and being resource independent, we made a study of the features. According to the information gain measure, the most informative

[3] http://poloxldb.linguateca.pt/harem.php

[4] http://www.cnts.ua.ac.be/conll2002/ner/bin/conlleval.txt

Tags	B(%)			I(%)			BIO(%)		
Classifier	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$
TMB-C24	94.42	95.19	94.81	87.25	85.67	86.45	92.51	92.61	92.56
TMB-C17	94.47	95.11	94.79	87.28	85.37	86.31	92.56	92.47	92.51
TMB-C24r	94.63	94.01	94.32	87.99	85.07	86.50	92.86	91.58	92.22
HMM-CD	92.18	93.82	92.99	83.94	81.98	82.95	90.01	90.60	90.31
HMM-CW	92.40	93.99	93.19	83.71	81.00	82.33	90.13	90.46	90.29
Vote 1 ld	95.31	95.36	95.34	88.02	87.56	87.79	93.34	93.24	93.29
TMB-E12	94.33	94.91	94.62	87.00	85.29	86.14	92.38	92.30	92.34
TMB-E17	94.17	95.28	94.72	87.62	85.37	86.48	92.44	92.59	92.51
HMM-CW	92.40	93.99	93.19	83.71	81.00	82.33	90.13	90.46	90.29
Vote 2 li	94.43	95.73	95.07	88.31	86.05	87.17	92.81	93.10	92.96

Table 1: *BIO for Spanish*

ones were selected and four candidate sets were formed.

$C24r = C24/\{d[1-3], aT\}$;

$C17 = C24r/\{c[5-6], C[6-7]\}$;

considered as language dependent (they use dictionaries, tools as lemmatizers, stemmers) and

$E12 = \{a, c[1-4], C[1-5], p, aC\}$;

$E17 = E12 \cup \{aSubStr[1-5]\}$;

considered as language independent. The results of each individual set can be seen in Table 1.

Initially to HMM we passed the NE and the tag associated with it. The obtained performance of 88.63% is less than each one of TiMBL's individual sets, however this difference is compensated with the number of features. Compared to the other methods, HMM's advantage is its time performance, but fails in adding lots of features.

As studied by (Rössler 02) features can be passed to HMM through corpus or tag transformation. We studied both possibilities and saw that tag transformation gives higher results. The three most informative attributes from set A were passed to B and I tags. For *La Coruña*, we have B-XX and I-XX tags, where the XX takes the binary features word capitalization, whole word in capitals and word in gazetteer. Adding these three features increased HMM's performance with 1.68%.

Tag O has frequent appearance, however its importance is insignificant compared to B and I tags, who actually detect the named entities. For this reason, we demonstrate separately system's precision, recall and f-score for B and I tags in Tables 1 and 2. The best score for Spanish BIO was obtained by TiMBL considering the complete $C24$ set with f-score of 92.56%. Comparing this score with set $C17$ where he number of features is reduced, the word window diminished from ± 3 to ± 2, the difference of 0.05% is insignificant. Set $C24r$ was studied for reducing some noisy attributes from set $C24$ but still keeping the ± 3 window. Its total BIO performance decreased but gained 86.50% - the highest f-score per I tag.

The resource independent sets perform quite similar to the dependent ones. For tag B, set E12 with its 12 attributes performs better than C24r. The complete BIO for E12 is better than those of C24r. TMB-E17 improves slightly the overall results of E12 and has similar results to C17. For tag I it performs better than C24, C17 and has 0.02% less performance than C24r.

The classifiers used different feature sets and we noticed that one classifier detects an entity while the other doesn't. After obtaining the different results, voting was applied. The resource dependent sets were grouped by vote one and the independent ones were grouped by vote two. The difference of 0.33% between *Vote 1 language dependent* with 93.29% performance and *Vote 2 language independent* with 92.96% f-score shows how small feature set containing attributes independent from any tools, dictionaries or gazetteers can give good and similar results to the dependent sets.

Taking in mind that Spanish and Portuguese are languages having similar behavior, we studied and saw how attributes valid for Spanish were directly adopted by Portuguese. Table 2 shows the results for Portuguese using the same set of resource independent features as applied for Spanish. With voting 83.32% f-score for B tag and 78.86% for complete BIO were achieved. These results are acceptable, considering the insufficient amount of training data we had.

Tags	B(%)			I(%)			BIO(%)		
Classifier	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$
TMB-E12	82.50	83.32	82.91	72.77	64.77	68.53	79.59	77.26	78.41
TMB-E17	80.13	83.22	81.65	69.64	58.86	63.80	77.16	75.27	76.20
HMM-CW	77.83	68.61	72.93	61.02	58.66	59.81	72.01	65.36	68.53
Vote 3 li	82.35	84.30	83.32	72.75	65.78	69.09	79.47	78.26	78.86

Table 2: *BIO for Portuguese*

Tags	LOC(%)			MISC(%)			ORG(%)			PER(%)		
Classifier	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$
ME-F24	81.16	74.72	77.81	69.29	49.12	57.49	74.21	84.07	78.83	82.95	88.03	85.41
TMB-F24	75.70	75.28	75.49	55.03	51.47	53.19	75.22	79.79	77.44	84.53	83.27	83.89
ME-F24clx	81.94	74.91	78.27	69.67	50.00	58.22	73.92	84.00	78.64	83.18	88.16	85.60
TMB-F24clx	74.84	75.46	75.15	55.88	50.29	52.94	75.88	79.79	77.79	85.42	85.31	85.36
TMB-R24	80.08	75.65	77.80	57.95	48.24	52.65	77.01	81.36	79.12	79.24	88.30	83.53
TMB-R24clx	79.20	75.18	77.14	63.20	50.00	55.83	76.14	81.36	78.66	80.15	88.44	84.09
HMM	74.85	67.80	71.15	44.66	46.76	45.69	72.06	73.86	72.95	66.11	74.83	70.20
VM24T24fclxH	81.16	75.92	78.46	66.80	49.71	57.00	75.06	83.21	78.93	83.72	89.52	86.52

Table 3: *NE classification*

5 NE classification

After detection follows NE classification. For this task, we used the results obtained from the language resource independent detection.

For ME and TiMBL, we started the classification with a set composed of 24 features as described in subsection 2.2. Let us denote by $F24$ the set having features: a, c[1-6], p, eP, eL, eO, eM, tP, tL, tO, gP, gL, gO, wP, wL, wO, NoE, f and s. In Table 3 comparing the performance of ME and TiMBL with the same set can be seen how ME classifies better for each one of the classes.

Choosing the most informative attributes, $\{a, c[1], eP, gP, gL, gO, wP, wL, wO, NoE, f\}$, we create a set $R24 \subset F24$. In Table 3 we displayed only the results obtained by TiMBL, because ME needs a lot of time for training and testing. When both classifiers were compared on small random samples from the original set, we saw that TiMBL performs better with the reduced set. When $R24$ was tested with the complete data, TiMBL achieved the highest result for ORG class of 79.12%. Two additional sets $R24clx = R24 \cup \{clx\}$ and $F24clx = F24 \cup \{clx\}$, where clx is the attribute described in Figure 2, were constructed. $R24clx$ lowered the performance for LOC and ORG class compared to the $R24$ set but performed better dealing with MISC and PER class. By adding clx attribute to $F24$, ME improved its performance with 0.46% for LOC and 0.19% for PER class and gained the maximum score of 58.22% for MISC class. Among all classifiers, HMM has the lowest score per class.

6 Comparison with CoNLL-2002 systems

The performance of NER considering various machine learning methods, where the advantages and disadvantages of each one of them being in time performance or feature maintenance was shown. Apart from this will be interesting to expose a comparative study with some systems participating in CoNLL-2002 NER shared task. Our system has been developed using the same data as the others, but we should take in mind that our classification is based on the language resource independent and portable detection set.

Table 4 represents the results per class for our system and the first four best performing CoNLL-2002 systems - WNC(Dekai Wu & Yang 02), CY(Cucerzan & Yarowsky 02), Flo(Florian 02), CMP(Carreras *et al.* 02). When classifying into LOC class our system performed with 0.2% and 2.09% better than the one of Wu and Cucerzan and less with 2.22% and 3.97% from the systems of Florian and Carreras. Our classification into MISC class was better with 7.74% and 8.84% compared to the one of Wu and Cucerzan and less with 3.58% and 1.73% from Florian and Carreras. For ORG and PER classes we outperformed all systems except the one of Carreras. With Wu's system we have 2.02% and 2.04% better score per ORG and PER class, from Cucerzan's 0.06% and 1.18% and from the system of Florian 0.53% and

Tags	LOC(%)			MISC(%)			ORG(%)			PER(%)		
Classifier	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$	Prec.	Rec.	$F_{\beta=1}$
ourNE	81.16	75.92	78.46	66.80	49.71	57.00	75.06	83.21	78.93	83.72	89.52	86.52
WNC	79.15	77.40	78.26	55.76	44.12	49.26	74.73	79.21	76.91	80.20	89.25	84.48
CY	79.66	73.34	76.37	64.22	38.53	48.16	76.79	81.07	78.87	82.57	88.30	85.34
Flo	82.06	79.34	80.68	59.71	61.47	60.58	78.51	78.29	78.40	82.94	89.93	86.29
CMP	85.76	79.43	82.43	60.19	57.35	58.73	81.21	82.43	81.81	84.71	93.47	88.87

Table 4: *CoNLL-2002 NE classification*

Classifier	Prec. %	Rec. %	$F_{\beta=1}$ %
CMP	81.36	81.40	81.39
Flo	78.70	79.40	79.05
ourNE	78.09	79.10	78.59
CY	78.19	76.14	77.15

Table 5: *Complete system performance*

0.23%.

We separated the overall performance of the first three best performing systems in Table 5. Comparing the f-score our system performs with 1.44% better than the third one, with 0.46% less than the second and with 2.8% less than the first system.

7 Conclusions and future work

We presented a combination of three machine learning methods, for performing NE detection and classification task for Spanish. Aiming at minimal feature space and restraining from dictionaries or other language dependent tools, we found resource independent detection set for Spanish, which was later easily ported to Portuguese. At present we didn't study the achievement of language resource independent classification, but in future we intend to work on this task. Comparing our results to CoNLL-2002 participants, the f-score of 78.46% for LOC, 57.00% for MISC, 78.93% for ORG and 86.52% for PER are quite encouraging, placing our system among the second and third position.

In future, we are interested in dividing the original categories into more detailed ones, for example: ORG class into administration, institution, company classes. A Word Sense Disambiguation module is going to be included for resolving name ambiguity. A rule based system which is separately developed and deals with weak entities such as *el presidente del Gobierno de La Rioja* is going to be merged with the machine learning module we have developed.

References

(Carreras *et al.* 02) Xavier Carreras, Lluís Màrques, and Lluís Padró. Named entity extraction using adaboost. In *Proceedings of CoNLL-2002*, pages 167–170. Taipei, Taiwan, 2002.

(Cucerzan & Yarowsky 02) Silviu Cucerzan and David Yarowsky. Language independent ner using a unified model of internal and contextual evidence. In *Proceedings of CoNLL-2002*, pages 171–174. Taipei, Taiwan, 2002.

(Daelemans *et al.* 03) Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. TiMBL: Tilburg Memory-Based Learner. Technical Report ILK 03-10, Tilburg University, November 2003.

(Dekai Wu & Yang 02) Marine Carpuat Jeppe Larsen Dekai Wu, Grace Ngai and Yongsheng Yang. Boosting for named entity recognition. In *Proceedings of CoNLL-2002*, pages 195–198. Taipei, Taiwan, 2002.

(Florian 02) Radu Florian. Named entity recognition as a house of cards: Classifier stacking. In *Proceedings of CoNLL-2002*, pages 175–178. Taipei, Taiwan, 2002.

(Rössler 02) M. Rössler. Using markov models for named entity recognition in german newspapers. In *Proceedings of the Workshop on Machine Learning Aproaches in Computational Linguistics*, pages 29–37. Trento, Italy, 2002.

(Sang 02) Tijong Kim Sang. Introduction to the conll-2002 shared task: Language independent named entity recognition. In *Proceedings of CoNLL-2002*, pages 155–158, 2002.

(Schröder 02) Ingo Schröder. A case study in part-of-speech tagging using the icopost toolkit. Technical Report FBI-HH-M-314/02, Department of Computer Science, University of Hamburg, 2002.

(Suárez & Palomar 02) Armando Suárez and Manuel Palomar. A maximum entropy-based word sense disambiguation system. In Hsin-Hsi Chen and Chin-Yew Lin, editors, *Proceedings of the 19th International Conference on Computational Linguistics, COLING 2002*, pages 960–966, August 2002.

Learning Spanish Named Entities using Unlabeled Data

Zornitsa Kozareva* and **Andres Montoyo**
Departamento de Lenguajes y Sistemas Informaticos
University of Alicante
Carretera San Vicente S/N
Alicante 03690, Spain
{zkozareva,montoyo}@dlsi.ua.es

Abstract

The paper gives a brief overview of the effect of tagging Spanish Named Entities leaning upon unlabeled data. By the help of two semi-supervised algorithms this task was accomplished.

1 Introduction

Nowadays, parallel with the amount of unstructured data, the need of Information Extraction, Information Retrieval and Document classification systems grows rapidly. At present Named Entity Recognition (NER) places important role for these applications, by recognizing the words containing the core information in a text. Living in the ages of multilinguality, it doesn't make sense to maintain English NER systems with impressive performance, rather than to focus on the development of such systems for languages as Spanish, Portuguese, Chinese etc. We should not forget that the available resources as annotated corpora or gazetteer lists[1] may be undeveloped or non-existent, which makes this task even more difficult and challenging.

We decided to carry out the construction of Spanish Named Entity recognizer relying on unlabeled data. The experiments gave us the notion of the complexity of the task and the extent to which its realization was possible. For our experimental set up, two semi-supervised approaches were used. They boost a small initial set of hand-labeled data, that during the learning process turns the unlabeled set into labeled. Self-training was responsible for entity delimitation, while co-training for entity classification.

* This research has been partially funded by the Spanish Government under project CICyT number TIC2003-0664-C02-02 and PROFIT number FIT-340100-2004-14 and by the Valencia Government under project numbers GV04B-276 and GV04B-268.

[1] catalogues of names of people, locations, organizations etc.

2 Related work

2.1 Named Entity Recognition

Recently lots of NER systems encompassing the rule based or the machine learning approaches have been developed. Among the common choices for learning methods are Hidden Markov Models (Bikel *et al.* 97), Maximum Entropy Models (Borthwick *et al.* 98), Support Vector Machine (Takeuchi & Collier 02) etc. All these systems function exploiting labeled data, whose preparation is very expensive and time-consuming. Once constructed, they are tuned and perform significantly well for the training data they have, but when applied to other corpora or domains, their performance becomes significantly low. For maintaining the same best performance, large human-annotated corpus is needed, but it might not be available for some languages. In this case it is reasonable to exploit the effect of unannotated data. (Màrquez *et al.*) developed a Catalan NE system using Spanish resources. (De Meulder & Daelemans 03) used unlabeled data to construct gazetteer lists. We built the complete NER using unlabeled data.

2.2 Co-training and Self-training

The general idea behind self-training and co-training algorithms is that they start with a small set of pre-labeled data and large set of unlabeled one. A bootstrapping algorithm aims to improve the classification performance by integrating examples from the unlabeled data into the labeled data set. To avoid introducing imbalance in the training data set, a constant ratio of the labeled classes is maintained for each iteration.

(Blum & Mitchell 98) introduced the co-training process by assuming that there exists two independent and compatible feature sets or views of data. A classifier learns with each one of those redundant feature subsets and afterwards labels the data for the other. According to (Nigam &

Ghani 00) in a real-world application, finding independent and redundant feature splits can be unrealistic and this can lead to deterioration in performance. (Collins & Singer 99) introduced the CoBoost algorithm for performing Named Entity classification. There the classifiers boosted either the spelling of the named entity or the context in which that entity occurred.

(Goldman & Zhou 00), proposed a co-training strategy that doesn't assume feature independence and redundancy. Two different classifiers having the same feature set learn the unlabeled data. The idea behind this strategy is that the two algorithms use diverse representations for their hypotheses and thus learn two various models that complement each other by labeling some unlabeled data and enlarge the training set of the other. In order to decide which unlabeled examples a classifier should label, they derive confidence intervals. We followed the same strategy, with the difference in the inclusion of the correctly labeled instances.

Self-training is a variant of co-training. Regarding (Nigam & Ghani 00), self-training initially builds a single classifier using the labeled training data with all features. Only one classifier is needed, with no split of features. For several iterations the classifier labels the unlabeled data and converts the most confidently predicted examples of each class into a labeled training example.

The co-training scheme we employed doesn't need any split of features as mentioned by (Goldman & Zhou 00). The scheme of the algorithm is represented in Figure 1. Describing it in brief, it takes two diverse machine learning classifiers C_1 and C_2, which learn the same pool of unlabeled examples U. At the time the unlabeled data set is turned into labeled, the classifiers compare the predicted classes for each example. Mutual class agreement indicates the addition of the instance into a temporal set T, and classifier disagreement leads to its rejection. We refer to P as the pool size (e.g. number of examples selected from the unlabeled set U for annotation at each iteration), to G as the growing size (e.g. the number of most confidently labeled examples added at each iteration to the set of labeled data L).

Given:

- C_1 and C_2 two different classifiers
- L a set of labeled training examples
- U a set of unlabeled examples
- T a temporal set of instances

Loop for I iterations:

1. do a pool pU of P randomly selected examples e_j from U
2. use L to individually train classifiers C_1, C_2 and label examples in pU
3. $\forall e_j \in pU$ whose classes agree by C_1 and C_2, do $T = T \cup \{e_j\}$
4. take randomly G examples from T and add them to L, while maintaining the class distribution in L
5. empty T

Figure 1: The Co-training scheme

3 The process of Entity Recognition

The task of Named Entity Recognition consists of delimiting the boundaries for each entity and deciding to which category (location, person, organization, etc.) it should belong. The sets of needed features are described below.

3.1 Named Entity Detection (NED)

In order to detect the entities we used the BIO model proposed by the CoNLL 2002 shared task (Sang 02). There are three tags: **B** indicates a word at the beginning of a NE, **I** states that a word is inside a NE and all words outside a NE are tagged as **O**.

Example: *El_O jefe_O de_O policía_O de_O Itarema_B ,_O Antonio_B Honorato_I dos_I Santos_I ,_O declaró_O que_O encontró_O fotografías_O ._O*

For recognizing tags **B, I, O**, a set of 12 features has been passed to the instance-based model. It contained only lexical and orthographic features, of the anchor word w_0(e.g. the word to be classified) and the words in a window ± 2 of the anchor word.

lexical features: represent the word forms[2] of w_0 and $w_{-2}, .., w_{+2}$; and position of w_0 in the sentence.

orthographic features: are binary and not mutually exclusive testing whether w_0 is all in capitals and if $w_{-2}, w_{-1}, w_0, w_{+1}, w_{+2}$ initiate in capitals.

This feature set was previously studied by

[2]In our example the word form at position +2 is *de*

(Kozareva *et al.* 05) and proven to be portable to Spanish and Portuguese. For detecting the possible entities in the corpus, the self-training method was applied, using the memory based learning toolkit TiMBL (Daelemans *et al.* 04).

3.2 Named Entity Classification (NEC)

Once detected, the named entities should be classified into PER, LOC, ORG and MISC classes (e.g. as defined by CoNLL-2002 shared task). The features used are:

lexical: representing the word forms of ± 3 window, the entity to be classified, the first word making up the entity and the second one if present.

orthographic: the same as in subsection 3.1 but in a ± 3 window.

trigger word[3] **and gazetteer**: check if the entity belongs to some of the gazetteer lists[4] for person names, locations or organizations (e.g. Antonio belongs to the list of person names); looks if the words ± 1 around the entity are trigger words for people, location or organization.

3.3 The data set and its evaluation

The Spanish data we worked with, has been a part of the EFE corpus used in the competitions of Clef[5]. The corpus contains *sgml* tags, which we removed by simple preprocessing. The text among the tags was first extracted, then tokenized and finally divided into test and train data sets.

From the train file we hand-labeled the first sentence and used the rest as unlabeled data. In the test file we had around 21300 tokens of which 2000 were annotated by human as NEs.

System's evaluation was made through *conlleval* script[6]. *Precision* considers from the number of tags allocated by the system, how many were right; and *Recall* measures from the tags the system should have found, how many did it spot.

4 Entity Detection through Self-training

The possible Named Entities were detected following the scheme and the features described in subsection 3.1. They were passed to the self-training algorithm which utilized the K-nearest neighbors algorithm.

The boosting process was initiated with 20 word hand-tagged sentence. On each iteration the unlabeled instances were turned into labeled, but only the most confident ones were later added into the training set. We conducted several experiments with growing size of $G = \{10, 50, 200, 500\}$, pool size of $P = \{30, 80, 500, 1000\}$ for 40 iterations. In order to avoid introducing imbalance into the training set, a constant ratio of $5 : 3 : 2$ for **O**, **B** and **I** tags was maintained.

Discussion: The achieved performances with these settings can be seen in Figure 2. The best performance for growing size of G=10 was obtained at 32 iteration. The score of 81.71% was reached using 320 unlabeled examples. The best performance for growing sizes G=50, G=200 and G=500 is around 78.88%, 84.41% and 84.39%.

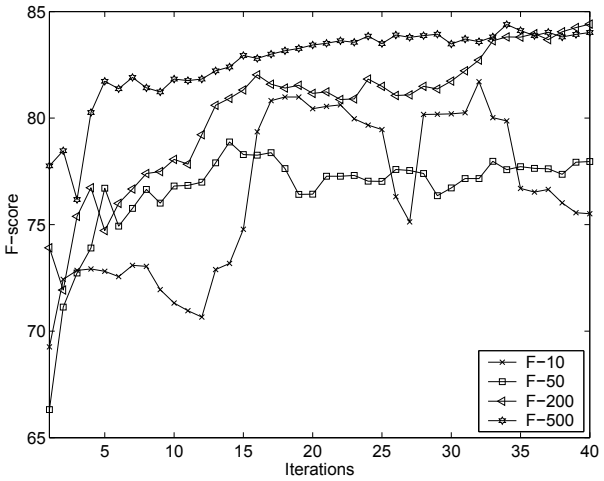


Figure 2: BIO results from self-training

As can be seen from Figure 2, test's accuracy doesn't continue as self-training progresses. There are peaks followed by declines, due to the degradation in the quality of labeled examples and their informativeness to the K-nn classifier.

In conclusion, we can say that learning entity detection with unlabeled data is not so difficult and good performance can be reached. In our case, high-score detection has been possible due to the attributes we worked with (they were previously studied by (Kozareva *et al.* 05) and proven to be robust); and the K-nn algorithm which stores every training instance into the memory and compares the test instance with the training ones when taking the decision of class association.

[3] semantically significant word pointing to some of the categories person, location, organization; e.g. city is a trigger word for locations

[4] the lists were created using the Spanish yellow pages, the number of the entries is around 900

[5] http://clef.isti.cnr.it/

[6] http://www.cnts.ua.ac.be/conll2002/ner/conlleval.txt

5 Entity Classification through Co-training

Once detected by the self-training process, the instances are classified with the algorithm described in Figure 1. The learning process started with 10 hand-labeled examples in the following ratio 3 : 3 : 3 : 1, respectively for ORG, PER, LOC and MISC classes.

For co-training, the two classifiers have been K-nn and decision trees, implemented in TiMBL's package (Daelemans *et al.* 04). To them, the same amount of pooled unlabled data has been passed. In our experiments, the pool P has been three times the size of G.

Discussion: We made three runs of forty iterations with growing size G={10,20,30}. The maximally obtained f-scores are:

- LOC: 37.45% for G=10; 48.15% for G=20; 56.56% for G=30;
- PER: 46.71% for G=10; 58.27% for G=20; 60.85% for G=30;
- ORG: 42.85% for G=10; 59.49% for G=20; 61.54% for G=30;
- MISC: 2.89% for G=10; 3.35% for G=20; 4.76% for G=30;

MISC class gave the worst results comparing it to the others. This is due to its heterogeneity, varying from names of book titles, movies to sport events. Other factors are the unfrequent presence of MISC class in the corpus we worked with and the class disagreement between the co-training classifiers. For the other classes LOC, PER and ORG, the performance grows as the training material increases. In future we'll conduct more experiments with growing size of 50, 100 and 200 to see maximum execution of NER.

6 Conclusions

The paper demonstrates the construction of Spanish Named Entity Recognition using unlabeled data. The experiments reveled how entity detection can be easily solved even when unlabeled data is used. However entity classification demands more training data and a better feature set for MISC class. In future we intend to make detailed and comparative study for Entity Recognition using other co-training algorithms, active-learning techniques and also try to obtain in an automatic way gazetteer lists extracted from unlabeled data.

References

(Bikel *et al.* 97) D. M. Bikel, S. Miller, R. Schwartz, and R. Weischedel. Nymble: a high-performance learning name-finder. In *Proceedings of ANLP-97.*, pages 194–201, 1997.

(Blum & Mitchell 98) A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Workshop on Computational Learning Theory.*, pages 92–100, 1998.

(Borthwick *et al.* 98) A. Borthwick, J. Sterling, E. Agichtein, and G. R. Nyu: Description of the mene named entity system as used in muc-7. In *Proceedings of the Seventh Message Understanding Conference.*, 1998.

(Collins & Singer 99) M. Collins and Y. Singer. Unsupervised models for named entity classification. In *Proceedings of the Joint SIGAT Conference on EMNLP and VLC*, pages 100–11, 1999.

(Daelemans *et al.* 04) W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch. Timbl: Tilburg memory-based learner. Technical Report ILK 04-02, Tilburg University, 2004.

(De Meulder & Daelemans 03) F. De Meulder and W. Daelemans. Memory-based named entity recognition using unannotated data. In *Proceedings of CoNLL-2003*, pages 208–211, 2003.

(Goldman & Zhou 00) S. Goldman and Y. Zhou. Enhancing supervised learning with unlabled data. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 327–334, 2000.

(Kozareva *et al.* 05) Z. Kozareva, O. Ferrandez, A. Montoyo, R. Muñoz, and A. Suárez. Combining data-driven systems for improving named entity recognition. In *Proceedings of Tenth International Conference on Applications of Natural Language to Information Systems*, pages 80–90, 2005.

(Màrquez *et al.*) L. Màrquez, A. de Gispert, X. Carreras, and L. Padró. Low-cost named entity classification for catalan: Exploiting multilingual resources and unlabeled data. In *Proceedings of the ACL 2003 Workshop on Multilingual and Mixed-language Named Entity Recognition.*

(Nigam & Ghani 00) K. Nigam and R. Ghani. Analyzing the effectiveness and applicability of co-training. In *Proceedings of Ninth International Conference on Information and Knowledge Management*, pages 86–93, 2000.

(Sang 02) T. K. Sang. Introduction to the conll-2002 shared task: Language independent named entity recognition. In *Proceedings of CoNLL-2002*, pages 155–158, 2002.

(Takeuchi & Collier 02) K. Takeuchi and N. Collier. Use of support vector machines in extended named entity recognition. In *Proceedings of CoNLL-2002, The Sixth Workshop on Computational Language Learning.*, 2002.

Recognition of Personal Names in Serbian Texts

Cvetana Krstev[1], **Duško Vitas**[2] and **Sandra Gucul**[1]
[1]Faculty of Philology, University of Belgrade, Studentski trg 3
[2]Faculty of Mathematics, University of Belgrade, Studentski trg 16
Belgrade, Serbia & Montenegro
cvetana@matf.bg.ac.yu, vitas@matf.bg.ac.yu, undra@EUnet.yu

Abstract

In this paper we present a method for accurate and precise recognition of personal names implemented for Serbian. It is based on development of comprehensive e-dictionaries of Serbian personal names, as well as foreign personal names transcribed to Serbian. In order to obtain high precision, the set of finite state automata (FSA) were developed to model various constraints. The same automata are also used to extract from a text personal names not yet covered by e-dictionaries.

1 Introduction

Recently, the importance of proper names in texts has been widely recognized since they can be successfully used in various NLP applications (Steinberger *et al.* 04). Thus, many attempts have been made to correctly recognize and tag them. These attempts are based on methods that vary from very simple ones (Mikheev *et al.* 99) to those that tend to produce the thorough inventory of proper names and their attributes. The advantage of simple methods is that they can be easily implemented and that the recognition accuracy is rather high. However, this method has serious disadvantages. First, it can not distinguish between various kinds of proper names, and second, it can associate neither morphosyntactic information to the recognized forms nor the appropriate lemma.

The method chosen for the recognition of proper names, such as geographic names, in Serbian texts is based on the approach described in (Grass *et al.* 02). In this paper we describe the method we develop for the recognition of personal names that is in accordance with the text processing based on lexical recognition using e-dictionaries and finite-state transducers (FST), method developed by LADL (Gross 88).

2 E-dictionaries of personal names

Electronic dictionaries of personal names are produced in the same format that is used for the general lexica. An entry in a dictionary of lemmas of DELAS type has a form `lemma,Cxxx[+SynSem]`. This means that to each lemma a Part-of-Speech (PoS) code (C) is attached as well as a code that determines its inflectional paradigm (xxx). Besides these obligatory elements, a various syntactic and semantic markers can be associated with each lemma (+SynSem). The DELAS type dictionary, in conjunction with the FSTs that model various inflectional paradigms, enables the production of a DELAF type dictionary of all inflected forms. The format of an entry in this dictionary is `form,lemma.Cxxx[+SynSem]`$\{$`:y`$^+\}^*$. The codes for grammatical information as well as syntactic and semantic markers can be used to retrieve information from the text.

The e-dictionary of Serbian personal names is based on an official list of Belgarade inhabitants dated from 1991 that can be considered representative for the whole Serbia and Montenegro. We have chosen for our dictionary the most frequent 3,300 first names and 17,000 surnames. The dictionary is being permanently expanded by adding unrecognized names that occur in texts being analyzed.

Since Serbian personal names inflect, it is necessary to assign the inflectional class codes to the chosen first names and surnames. All these names belong to the inflectional classes already determined for the common nouns. The first names belong to 25 different inflectional classes (21 classess for musculine names and 4 classes for feminine names), while surnames belong to 22 different inflectional classes (Table 1).

A note should be made on the gender of surnames. Surnames in Serbian behave like nouns, thus one of their features is the gender. On the other hand, surnames are equally used for men and women. Surnames never inflect if used as a part of a woman's name, while they do inflect if used individually for a man or as a part of a man's name that comes after his first name. For

that reason the masculine gender was assigned to all surnames. If a surname is individually used to refer to a woman, than certain derivative forms are used (see section 3).

I	Petrović,N28+NProp+Hum+Last+SR
	Sandra,N1637+NProp+Hum+First+SR
II	Petrovićem,Petrović.N28+...+SR:ms6v
	Sandrom,Sandra.N1637+...+SR:fs6v
	Sandrom,Sandro.N1068+...+SR:ms6v

Table 1: In the first part a few entries from DELAS dictionary of personal names are given. In the second part the entries from DELAF that represent the singular forms in the instrumental case for the same entries are given. It can be seen that this form of the chosen first name is ambiguous with some other first name.

Surnames can have plural forms, in which case they denote members of the family. The plural forms of the surnames that end in *-ić* are quite common, for instance *Petrovići* for *Petrović*, and can be used for a number of other surnames as well. For the others it is not clear what the plural forms would be or they look rather awkward, like for *Goati* or *Lisjak*. In order to reduce the unnecessary ambiguity all the surnames for which the plural forms are not straightforward are put into the inflectional classes for which the plural forms are not defined. If the occurences of plural forms for some particular surnames happen, their inflectional classes can be easily corrected.

The semantic markers `+First` and `+Last` were asigned to all first names and surnames, respectively. Also, all personal names in use in Serbia are given the markers `+NProp`, denoting that the entry is the proper name, `+Hum` denoting that it refers to a human being, and `+SR` denoting that the personal name is in use for the inhabitants of Serbia and Montenegro. In addition, nicknames have the marker `+Nick` associated to them. Many nicknames in Serbia are also used as first names so they have both markers associated to them (e.g. *Bane*). The usage of these markers will be described in the following sections.

Foreign names are in Serbian texts almost always used transcribed, rarely in its original form. For instance *George Bush* and *Tony Blair* would in Serbian text appear as *Džordž Buš* and *Toni Bler*. The foreign names inflect in the same way as the Serbian names; for instance, the instrumental forms of the mentioned names would be *Džordžom Bušem* and *Tonijem Blerom*.

We tackle foreign personal names in the same way as we do Serbian names, that is by producing the dictionaries of first names and surnames in LADL format. First, we have started to produce dictionaries for the English transcribed names, on the basis of (Prčić 92). At present, DELAS dictionaries of the English first names and surnames transcribed to Serbian have 330 and 1340 entries, respectively. All the first names are grouped in 13 inflectional classes, as well as the surnames, though the two sets of inflectional classes are not the same.

Klerk,N1002u+NProp+Hum+First+EN
+Val=Clark+Val=Clarke
+Val=Clerk+Val=Clerke+Norm=Klark
Olbrajt,N1002+NProp+Hum+Last+EN
+Val=Albright+Val=Allbright

Table 2: Excerpts from the DELAS dictionaries of English first names and surnames

The same markers are associated with the entries in DELAS dictionary of English transcribed personal names as for the entries in DELAS dictionary of Serbian names (except that the marker `+SR` is replaced by `+EN`), and two more markers are added: `+Val` and `+Norm`, both of which are actually attributes to which the values are assigned. The value of the `+Val` is the name as originally written, while the value of `+Norm` is the correct transcription of the name. Namely, many English names are often incorrectly transcribed and used, and this attribute connects all the transcriptions, both correct and incorrect, of one name. It can be seen in Table 2 that four English names *Clark*, *Clarke*, *Clerke* and *Clerke* have the same transcription, *Klark*.

The accurate recognition of personal names in Serbian texts is far from being straightforward due to their high homonymy. The examples are numerous. Some frequent surnames are also first names, and vice versa. Some first names are used both for men and women. Many surnames and first names are homonymous with other proper — mountains, rivers, and cities. Many surnames are also names of the inhabitants of cities, regions, and countries. Surnames and first names are often homonymous with other common names for animals, plants, proffesions, etc.

The other source of problems in personal name recognition is the ambiguity of the forms. For many masculine first names the corresponding female names exists: *Ivan* and *Ivana*, with many coinciding forms: genitive and accusative case forms of the masculine name are the same as the nom-

a)	trgova i crkava, potpuno kao kod nas.	Anđeli	, nekada ljudi, ispisuju svoje misli na listiće
b)	na košulja, cilinder, crn iberciger.	Ide	on tako i tetura se, i ja naletim na njega, ona
c)	esa kao što znam ulice u Kadiksu.	Divna	stvar, to njihovo namesništvo! To carstvo vec

Table 3: Concordance lines retrieved by the query <N+First>: a) Nominative plural form of the noun *anđeo* (Engl. angel) is recognized as a dative singular form of the first name *Anđela*; b) Third person present form of the verb *ići* (Engl. to go) is recognized as a genitive form of the first name *Ida*; c) Feminine nominative singular form of the adjective *divan* (Engl. wonderful) is recognized as the nominative form of the first name *Divna*.

okupacijskim. Umjesto toga,	Buš	je rekao kako je gruzijska ružičasta revol uci
dsednika SAD. Američki predsednik	Džordž Buš	, koji je juče boravio u poseti toj zemlj
demokraciju Američki predsjednik	George W. Bush	u ponedjeljak je iz Moskve doputovao u

Table 4: An excerpt from the concordances obtained by applying the regular expression for *George Bush* to a text containing news from one Belgrade and one Zagreb daily newspaper.

inative and vocative case forms of the feminine name, etc. Also, many masculine names have variant forms whose inflected forms also coincide, as for *Đura* and *Đuro*, where the nominative case of the first one is the genitive case of the second one, etc. Finally, many forms of personal names are ambiguous with the forms of other lemmas (Table 3).

3 The methods for personal name recognition

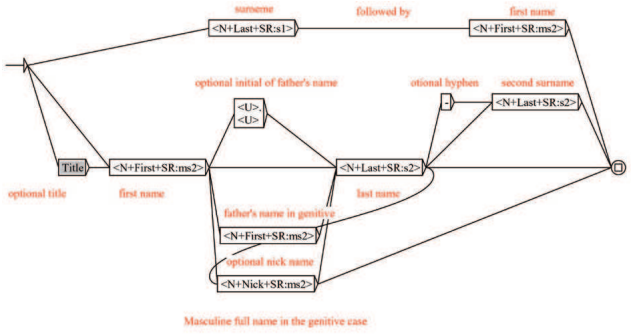


Figure 1: The subgraph IP_M_sr_2 recognizes Serbian masculine full name in genitive case

In Intex environment (Silberztein 04) personal names can be retrieved from a text using the described e-dictionaries. The queries can be formulatted either in a form of a regular expression or in form of a FSA. In a query, all the associated grammatical information, as well as syntactic and semantic markers can be used. For instance, in order to retrieve all musculine full personal names, consisting from both first name and surname, we could use the query (<N+First:m> <N+Last:s>) + (<N+Last:s1> <N+First:m>) that takes into account two possible orders of the first name and surname, and the rules of declination. This query is rather naïve since it does not take into consideration the agreement constraints. Thus, it retrieves many false occurrences.

When retrieving English names, the specific markers +Val and +Norm can be used. For instance, in order to retrieve all the occurrences of the name *Tony Blair*, no matter how it is written, in original or transcribed, the query (<N+Val=Tony> + Tony + <E>) (<N+Val=Blair> + Blair) can be used (Table 4). This query is naïve too, since names originally written also inflect (for instance, "Dio poslanika žali se da je dosta glasova izgubljeno upravo zbog Blaira..."). However, since originally written names are regularly used in Croatian, and rarely in Serbian, we are not dealing with that problem presently.

In order to recognize personal names properly it is necessary to model their usage more precisely. Since in the newspaper texts persons are usually referred to by a full name, our first goal is to model that type of usage. In this model, we take into account: (a) Two possible orders of a first name and a surname; (b) The rules of the agreement between the first name and the surname depending on the gender, as well as their agreement in case for the masculine names; (c) The optional usage of a title before the name, like *prof.dr*; (d) The optional usage of a second surname, separated from the first one by a hyphen or a space; (e) The optional usage of a nick name, between a first name and a surname, or after a surname; (f) The optional usage of a father's name between a first name and a surname, either as an initial, or as a first name in genitive case.

Our model of full personal names is developed modularly, so it is realised by numerous subgraphs (Figure 1). The subgraphs can thus be combined in various ways in order to satisfy specific demands, such as to retrieve the English transcribed names or to retrieve all the masculine names. (Table 5, Part I).

The application of these FSA shows that the

a)	pomenuta lična inicijativa,	Branka Otašević-Trbojević	ilustrovala je konkretnim
b)	dr Jelica Jokanović-Mihailov,	dr Ljiljana Subotić	, dr Mato Pižurica, dr Duško Vit
c)	Beogradski majstor fotografije	Dragan S. Tanasijević	, autor pomenutih "svetlopisa",
d)	za poslanike u Veću građana	Radoslav Raka Dimitrijević	, Svetislav Tanasković Ket
e)	januara direktor Poreske uprave	Marija Drča Ugren	. U prihod za oporezivanje raču
f)	objašnjava za naš list	Saša Gajin	, saradnik Instituta za Uporedno
g)	je pomoćnik direktora Zavoda	Dragi Stojiljković	na konferenciji za novinare
h)	Podgorički stomatolog	Đorđije Milić	, kandidat grupe građana, nastupa

Table 5: Part I: some correctly retrieved Serbian full names: a) Two surnames separated by a hyphen; b) Name preceded by a title; c) Father's name as an inital; d) A nick name between a first name and a surname; e) Two surnames separated by a space; Part II: Some masculine names falsly retrieved among feminine names.

a)	nacionalnom referendumu 15. februara.	Mićićka	je rekla da će zakazati izbore tek
b)	Zxivka D. Pavlović isto, Darinka	Stanarevićka	1.050, koliko i Tanasije Mitrović
c)	biti održan u petak (6. septembar).	Mićićeva	je ukazala da će komisija usvojiti
d)	na Terazijama je gostovalo sa	Nušićevom	komedijom "Dr", u kojoj je prvakinja
e)	čuju ni Klinton na samitu niti	Olbrajtova	u Generalnoj skupštini a danas je

Table 6: Some examples of references to female persons by s surname only (a) The expression of the second type always yields correct results; (b) This type of address can be used in combination with the first name; (c) The expression of the first type gives all instances of female persons addresed in this way; (d) False retrieval, also a possessive adjective is actually used; (e) The first type of derivation is used for the transcribed foreign names as well (*Olbrajtova* stands for *Madeleine Albright*).

problem of ambiguity between feminine and masculine names still persists, though in a much smaller degree (Table 5, Part II). There are still masculine names falsely retrieved among feminine names. In some cases, it is difficult to say whether it is an error at all (example 5 f), since *Saša Gajin* can be a name of a man or a woman, and even a wider context does not give a clue. The case of a syntactic ambiguity is exemplified by the example g), as the sentence has two possible interpretations: either "the depute director of the Institution, the man whose name is *Dragi Stojiljković*, has said something at the press-conference" or "the depute director of the Institution, whose name is not given, has said something at the press-conference to a woman with the name *Draga Stojiljković*." Only context wider than a sentence can resolve this problem. The example h) shows that sometimes the immediate context of a personal name can resolve the ambiguity. Since *Podgorički stomatolog* (Engl. a dentist from Podgorica) is in the nominative case, so should also be the name that follows, and that excludes the possibility that it is a feminine name.

In the newspaper texts persons are rarely referred to by a first name only. However, if a person is well-known or his/her identity has been previously established the surnames alone can be used. Since the surnames of feminine persons never inflect, they are rarely addressed by a surname only. Two derivative forms are rather used: one is derived from a possessive adjective of a surname, and an other is obtained by a gender motion. The first form, being obtained from a possessive adjective coincides with all feminine inflected forms of the adjective.

Not all derivational forms are incorporated in Serbian e-dictionaries (Krstev & Vitas 05). Those that are regularly produced and whose meaning can be deduced from the meaning of the basic word are rather recognized during the text processing by the so called transducers with lexical constraints (Silberztein 04). The recognized form is associated with an appropriate lemma and grammatical information, it inherits the syntactic and semantic markers from the basic lemma, with two more markers added: `+D`, which signifies that it is a derived form, `+Pos` or `+GM` that identify the type of a derivational process, possessive adjective and gender motion, respectively.

The use of this information enables the recognition of derived forms of surnames that are used to address female persons: the expression <`A+Last+SR+D+Pos:fs`> is used for the first type of the address, and <`N+Last+SR+D+GM`> is used for the second type (Table 6).

4 One application

The e-dictionaries and FSA described can serve various purposes. We show further how the constructed FSA can be used to extract from text a person's function or role. The person's role or function is often mentioned just before his/her personal name, or immediately after it in apposition. This function is often expressed in a form of a noun phrase of restricted structure whose head

vršilac dužnosti predsednika Srbije i predsednik parlamenta Nataša Mićić
Nebojša Čović, predsednik Koordinacionog centra za Kosovo i Metohiju i potpredsednik Vlade Srbije
predsednikom Sjedinjenih Država Džordžom V. Bušom
bivšem američkom državnom sekretaru Medlin Olbrajt

Table 7: Serbian and English personal names with their functions

Tamir Gadban, zvaničnik zadužen za iračku naftnu industriju
potpredsednikom banke za Evropu i centralnu Aziju Šigeom Katsuom
Redžep Tajip Erdogan, lider vladajuće Partije pravde i razvoja
Dojče Telekom, većinskog vlasnika Hrvatskog telekoma,

Table 8: Recognized foreign personal names adjacent to the syntactic structure representig the function of a person. A false retrieval is given in the last line (*Dojče Telekom* stands for *Deutsche Telekom*): the noun *vlasnik* (Engl. owner), marked as human, is used for an organization

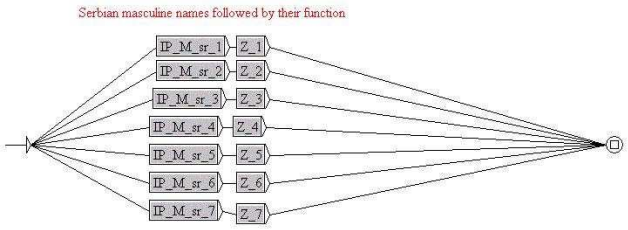


Figure 2: The subgraph `IP_M_sr_samo_zvanja` recognizes Serbian masculine full name followed by person's function; it takes into account that the full name and the noun phrase that follows have to agree in case.

is a common noun to which a semantic marker `+Hum` (for human) is added. The function is often accomapnied by the institution where it is performed, and which is also expressed as the noun phrase of its own structure (Figure 2). Some full names retrieved from the sample text with their accompanying functions are given in Table 7.

For the construction of this FSA personal names were used as the anchors to model the syntactic structure of their functions (Gross 98). Since our dictionaries presently contain only Serbian names and a small number of English transcribed names, a number of personal names in the text still remains unrecognized. The FSA that model the syntactic structure of the persons' functions or roles can be used as the anchors to retrieve personal names among vaguely recognized proper names — simple words that begin with an upper-case letter and that remain unrecognized after applying all dictionaries. To achieve this, in a graph from Figure 2 the subgraphs that recognize the masculine personal names `IP_M_sr_1`, `IP_M_sr_2`, etc. should be replaced by a simple query: `<N+NProp+Unk> <N+NProp+Unk>`. Here marker `+Unk` stands for a proper name of unknown type. In Table 8 some extracted names of various origin are given.

5 Conclusion

The method we have developed for personal name recognition is giving very promising results. Not only can we recognize personal names with high precision and recall, but the full grammatical information associated with them enables their usage for many advanced purposes, such as text disambiguation. Also, by transforming the developed FSA into FSTs it is possible to automatically tag personal names in a text with XML tags, in a manner of TEI tags `<persName>` and `<name>`.

References

(Grass *et al.* 02) T. Grass, Denis Maurel, and O. Piton. Description of a multilingual database of proper names. Number 2389 in Lecture Notes in Computer Science, pages 137–140, Berlin, 2002. Springer-Verlag.

(Gross 88) Maurice Gross. The use of finite automata in the lexical representation of natural languages. In *Electronic Dictionaries and Automata in Computational Linguistics*, number 337 in Lecture Notes in Computer Science, pages 34–50, Berlin, 1988. Springer-Verlag.

(Gross 98) Maurice Gross. A bootstrap method for constructing local grammars. In *Proceedings of the Symposium 'Contemporary Mathematics'*. University of Belgrade, Faculty of Mathematics, 1998.

(Krstev & Vitas 05) Cvetana Krstev and Duško Vitas. Extending Serbian E-dictionary by the Use of the Lexical Transducers. In *Proceedings of the 7th IntexWorkshop*. Presses Universitaires de Franche Compté, 2005. 7-9 June 2004, Tours, France.

(Mikheev *et al.* 99) A. Mikheev, M. Moens, and C. Grover. Named entity recognition without gaetteers. In *Proceedings of the EACL'99*, pages 1–8. ACL, 1999. June 1999, Bergen, Norway.

(Prčić 92) Tvrtko Prčić. *Transkripcioni rečnik engleskih ličnih imena*. Nolit, 1992.

(Silberztein 04) Max Silberztein. *INTEX Manual, v.4.33*. 2004. http://intex.univ-fcomte.fr/downloads/Manual.pdf.

(Steinberger *et al.* 04) Ralf Steinberger, Bruno Pouliquen, and Camelia Ignat. Providing Cross-Lingual Information Access with Knowledge-Poor Methods. *Informatica*, 28(4):415–423, 2004.

How Do Treebank Annotation Schemes Influence Parsing Results? Or How Not to Compare Apples And Oranges

Sandra Kübler
Universität Tübingen
Seminar für Sprachwissenschaft
Wilhelmstr. 19
D-72074 Tübingen, Germany
kuebler@sfs.uni-tuebingen.de

Abstract

In the last decade, the Penn treebank has become the standard data set for evaluating parsers. The fact that most parsers are solely evaluated on this specific data set leaves the question unanswered how much these results depend on the annotation scheme of the treebank. In this paper, we will investigate the influence which different decisions in the annotation schemes of treebanks have on parsing. The investigation uses the comparison of similar treebanks of German, NEGRA and TüBa-D/Z, which are subsequently modified to allow a comparison of the differences. The results show that deleted unary nodes and a flat phrase structure have a negative influence on parsing quality while a flat clause structure has a positive influence.

1 Introduction

In the last decade, the Penn treebank (Marcus *et al.* 94) has become the standard data set for evaluating parsers. The fact that most parsers are solely evaluated on this specific data set leaves the question unanswered how much these results depend on the annotation scheme of the treebank. This point becomes more urgent in the light of more recent publications on parsing the Penn treebank such as (Charniak 00; Charniak 01; Klein & Manning 03; Dubey & Keller 03), which show that parsing results can be improved if certain peculiarities of the Penn and the NEGRA treebank annotations are taken into consideration in the probability model. (Klein & Manning 03), e.g., gain approximately 1 point in F-score when they extend POS tag information by the mother node or the lemma. This directly reflects shortcomings in the annotation scheme, which groups prepositions, subordinating conjunctions, and complementizers under the same POS tag. (Charniak 01) reports a 10% reduction in grammar perplexity for his trihead model, which models deeper structure in flat NPs such as "Monday night football". These findings raise the question whether such shortcomings in the annotation can be avoided during the design of the annotation scheme of a treebank. The question, however, can only be answered if it is known which design decisions are more or less favorable for PCFG parsing.

In this paper, we will investigate how different decisions in the annotation scheme influence parsing results. In order to answer this question, however, a method needs to be developed which allows the comparison of different annotation decisions without comparing unequal categories.

For a comparison of different annotation schemes, one ideally needs one treebank with two different sets of (manual) annotations. An automatic conversion from one annotation scheme to the other is only possible from deeper structures to flatter ones. The other direction would have to be based on heuristics. In this case, there is a high probability that systematic errors are introduced so that only a corrupted annotation in the target annotation scheme will be reached. In the absence of more detailed methods of comparison, testing the effect of modifying individual annotation decisions gives insight into the factors that influence parsing results.

Section 2 gives an overview of treebank pairs for a single language. In section 3, we will describe the treebanks used in this investigation in more detail, section 4 describes the preparatory steps necessary for converting these treebanks into a format that can be treated by a PCFG parser. Section 5 describes the method of comparison, and section 6 discusses the results of the comparison.

2 Comparable Treebanks

For the comparison described above, we need different treebanks which are based on the same language and the same text genre and which are annotated with different annotation schemes. But the annotation schemes must be similar enough to enable a comparison. A comparison between a constituent-based and a dependency-based annotation scheme would be very difficult since, in their original form, they require two different parsing algorithms. A completely determined rule-based conversion between the two is only possible from constituents to dependencies. This is not an optimal solution since decisions in dependency annotations are made on a lexical level and can only

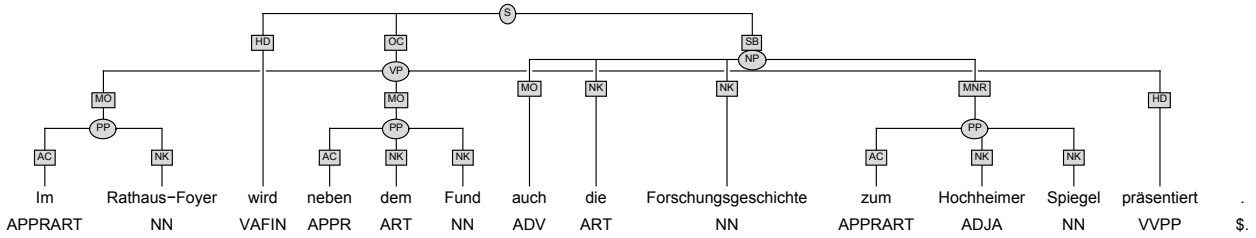


Figure 1: A sample tree from the NEGRA treebank.

be generalized to a certain extent.

One of the very few examples of two treebanks for one language are the Penn treebank (Marcus *et al.* 94) and the SUSANNE corpus (Sampson 93) for English. However, there are significant differences in the size of the treebanks and in the text genres, which make a comparison of the two annotation schemes unfeasible. Another example of such a pair are the two treebanks for Italian, ISST (Montegmagni *et al.* 00) and TUT (Bosco *et al.* 00). ISST uses a constituent-based annotating scheme augmented with grammatical functions; TUT, in contrast, is annotated with dependency relations. For the reason given above, this would not allow a comparison based on constituents. Additionally, both treebanks are of a very restricted size, which makes data sparseness problems very likely.

Only recently, a new pair of treebanks for German has become available, the NEGRA (Skut *et al.* 97) and the TüBa-D/Z (Telljohann *et al.* 04) treebanks. Both treebanks are based on newspaper text, both use the STTS POS tagset (Thielen & Schiller 94), and both use an annotation scheme based on constituent structure augmented with grammatical functions. However, they differ in other respects, which makes them ideally suited for an investigation on how decisions in the design of an annotation scheme influence parsing accuracy.

3 The NEGRA and the TüBa-D/Z Treebanks

Both treebanks use German newspapers as their data source: the Frankfurter Rundschau newspaper for NEGRA and the 'die tageszeitung' (taz) newspaper for TüBa-D/Z. NEGRA comprises 20 000 sentences, TüBa-D/Z 15 000 sentences. Both treebanks use an annotation framework that is based on phrase structure grammar and that is enhanced by a level of predicate-argument structure. Annotation for both was performed semi-automatically. Despite all these similarities, the treebank annotations differ in four important aspects: 1) NEGRA does not allow unary branching while TüBa-D/Z does; 2) in NEGRA, phrases receive a flat annotation while TüBa-D/Z uses phrase internal structure; 3) NEGRA uses crossing branches to represent long-distance relationships while TüBa-D/Z uses a pure tree structure combined with functional labels to encode this information; 4) NEGRA encodes grammatical functions in a combination of structural and functional labeling while TüBa-D/Z uses a combination of topological fields (Drach 37; Höhle 86) and functional labels, which results in a flatter structure on the clausal level. The two treebanks also use different notions of grammatical functions: TüBa-D/Z defines 36 grammatical functions covering head and non-head information, as well as subcategorization for complements and modifiers. NEGRA utilizes 48 grammatical functions. Apart from commonly accepted grammatical functions, such as *SB* (subject) or *OA* (accusative object), NEGRA grammatical functions also comprise a more extended notion, e.g. *RE* (repeated element) or *RC* (relative clause) [1]. The difference in grammatical functions, however, is difficult to compare since this can only be done in a task-based evaluation within an application that uses these grammatical functions as input.

Figure 1 shows a typical tree from the NEGRA treebank. The syntactic categories are shown in circular nodes, the grammatical functions as edge labels in square boxes. The prepositional phrase "Im Rathaus-Foyer" (in the foyer of the town hall) and the noun phrase "auch die Forschungsgeschichte zum Hochheimer Spiegel" (also the research history of the Hochheimer Spiegel) do not contain internal structure, the noun kernel elements are marked via the functional labels *NK*. The fronted PP is grouped under the verb phrase, resulting in crossing branches. Figure 2 shows a typical example from TüBa-D/Z. Here, the complex noun phrase "Der Autokonvoi mit den Probenbesuchern" (the car convoy with the visitors of the rehearsal) contains a noun phrase and the prepositional phrase with an internal noun phrase, with

[1] For a more detailed comparison of Tüba-D/Z and TIGER, the successor of NEGRA, cf. (Telljohann *et al.* 04).

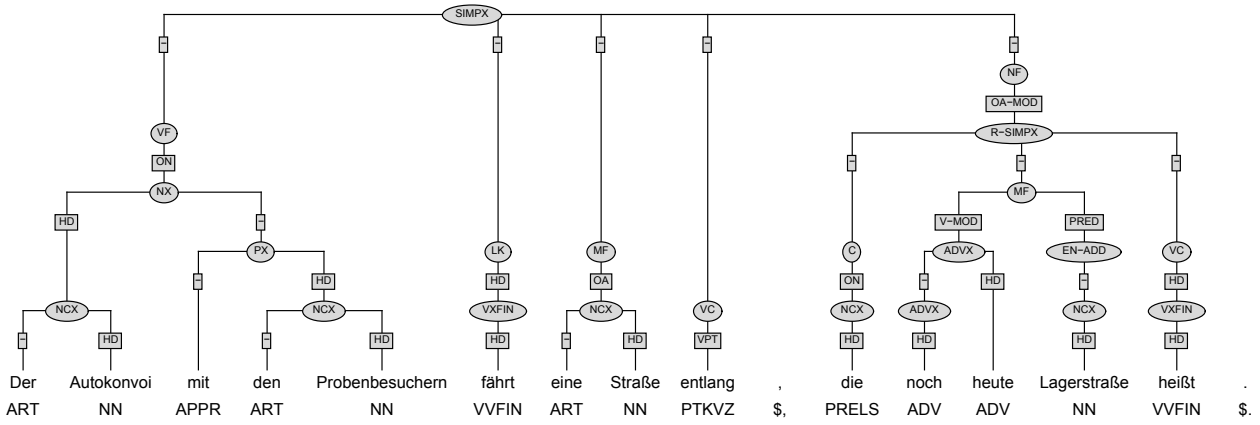


Figure 2: A sample tree from the TüBa-D/Z treebank.

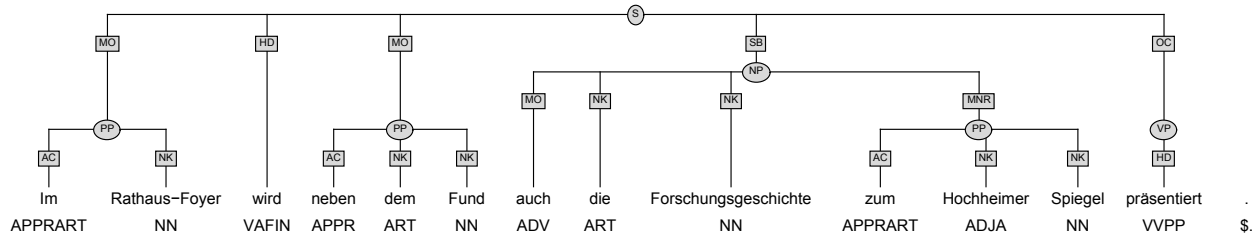


Figure 3: The NEGRA sentence from Figure 1 without crossing branches.

both noun phrases being explicitly annotated. The tree also contains several unary nodes, i.e. nodes with only one daughter, e.g. the verb phrases "fährt" (goes) and "heißt" (is called) or the street name "Lagerstraße". The main ordering principle on the clausal level are the topological fields, long-distance relationships such as the relation between the noun phrase "eine Straße" (a street) and the extraposed relative clause "die heute noch Lagerstraße heißt" (which is still called Lagerstraße) are marked via functional labeling; *OA-MOD* specifies that this noun phrase modifies the accusative object *OA*.

4 Preprocessing the Treebanks

Most state-of-the-art parsers are based on context-free grammars. However, both treebanks do not completely adhere to the requirements of a CFG: Apart from NEGRA's crossing branches, both treebanks contain sentences that consist of more than one tree. For all sentences, a virtual root node that groups all trees is inserted, and parenthetical trees are attached to the surrounding tree. The virtual root also ensures that the grammar has a single start symbol. In order to resolve NEGRA's crossing branches, a script was used that is provided with the graphical annotation tool, which was used to annotate both treebanks[2]. The script isolates crossing constituents and attaches the non-head constituents higher up in the trees. After the conversion, the sentence in Figure 1 receives the tree structure shown in Figure 3. Both modifiers of the verb phrase have been reattached at the clause level in order to resolve the crossing branches. Unfortunately, the modified tree does not contain any information on the scope of the modifiers, which has previously been shown by the low attachment in the VP. Since crossing branches occur in approximately 30% of the sentences, we use a modified script to keep trace of the original phrase from which the constituent was moved. In this version, NEGRA+traces, the crossing modifier PPs in Figure 1 are assigned the function label *MO<VP* specifying that they are extracted from the verb phrase. Thus, the tree would be the same as in Figure 3, except for the function labels of the two reattached PPs.

5 Comparing Treebanks for Parsing

For the experiments, the statistical left-corner parser LoPar (Schmid 00) was used. Since the experiments are designed to show differences in parsing quality depending on the annotation decisions, the parser was

[2]Cf. `www.coli.uni-saarland.de/projects/sfb378/negra-corpus/annotate.html`

	NEGRA	NEGRA+traces	TüBa-D/Z
crossing brackets	1.07	n.a.	2.27
labeled recall	70.09%	n.a.	84.15%
labeled precision	67.82%	n.a.	87.94%
labeled F-score	68.94	n.a.	86.00
crossing brackets	1.04	1.03	1.93
function labeled recall	52.75%	49.03%	73.65%
function labeled precision	51.85%	50.49%	76.13%
function labeled F-score	52.30	49.75	74.87

Table 1: The results of comparing NEGRA and TüBa-D/Z.

used without (EM) training or lexicalization of the grammar.

For all the experiments reported here, only sentences with a length of maximally 40 words were used. These sentences were randomly split into 90% training data and 10% test data. The test data were kept fixed in order to enable error analysis. Since we did not want to have the results influenced by POS tagging errors, the parser was given the gold POS tags for the test sentences[3].

For each experiment, two different types of tests were performed: For one type, the data contained only syntactic constituents, i.e. the grammatical functions, which are shown as square boxes in the trees, were omitted. Thus, the rule describing the root node and its daughters in Figure 3 is represented as "S $\rightarrow$ PP VAFIN PP NP VP". These tests are reported below as "labeled precision" and "labeled recall". In the second type of tests, the syntactic categories were augmented by their grammatical function. Thus, the same rule extracted from the tree in Figure 3 now contains the grammatical function for each node: "S $\rightarrow$ PP-MO VAFIN-HD PP-MO NP-SB VP-OC". (Note that the root node is the only node in the tree that does not have a grammatical function.) These tests are reported below as "function labeled precision" and "function labeled recall".

The results of the experiments on the original treebanks after preprocessing are shown in Table 1. As reported above, NEGRA contains crossing branches in 30% of the sentences, which had to be resolved in preprocessing. Since in these sentences, attachment information is often not present, the experiment was repeated with the version of NEGRA that contains traces of moved constituents. This representation is closer to the TüBa-D/Z annotation which also contains such information for long distance relationships.

[3]Thus, the results are slightly better than in setting where the POS tags are assigned automatically.

The results show that the F-score for TüBa-D/Z is significantly higher than for NEGRA trees. In contrast, the number of crossing brackets is lower for NEGRA. The NEGRA results raise the question whether the low crossing brackets rate in NEGRA is only due to the low number of constituents in the trees. The percentage of nodes per words shows that while NEGRA trees contain on average 0.88 nodes per word, TüBa-D/Z trees contain 2.38 nodes per word. This leads to the question whether the deeper structures in TüBa-D/Z can be parsed reliably but may not be useful for further processing. Thus, a more detailed investigation is necessary.

This discussion leads to the question of how to evaluate the parsing results in a meaningful way. Generally, there are two possible evaluation methods that go beyond the calculation of precision and recall: an analysis of the different constituents and a task-based evaluation. The former approach can show for which categories there are differences between the annotation schemes. The latter approach tests the utility of the parser output for a specific task such as anaphora resolution or question answering. While this would provide valuable insight, the results would be difficult to generalize from the specific task. For the former approach, the equivalence of the different syntactic and functional categories must be presupposed. Such a comparison is only meaningful if both annotation schemes describe the same phenomena with the same categories. Unfortunately, for NEGRA and TüBa-D/Z, this assumption often does not hold. The most obvious area in which the two treebanks differ is the treatment of unary nodes: while TüBa-D/Z annotates such constituents, NEGRA does not allow unary branching. The differences in annotation are shown in Figure 4 for NEGRA and in Figure 5 for TüBa-D/Z. In these trees, it becomes obvious that the differences in annotation are widespread and do not only concern verbal phrases but also, for example, noun phrases,

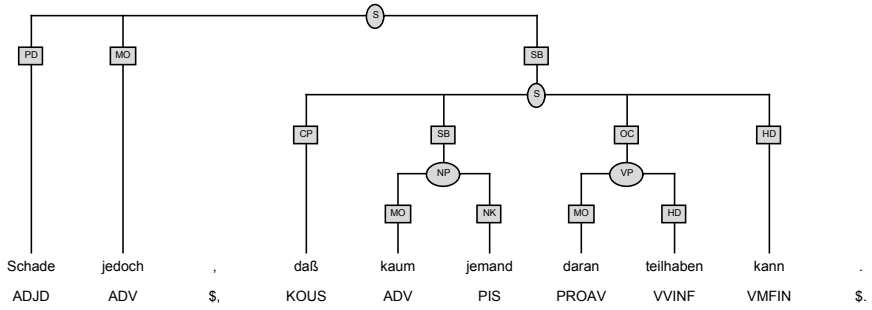


Figure 4: A sentence from NEGRA without the annotation of unary nodes.

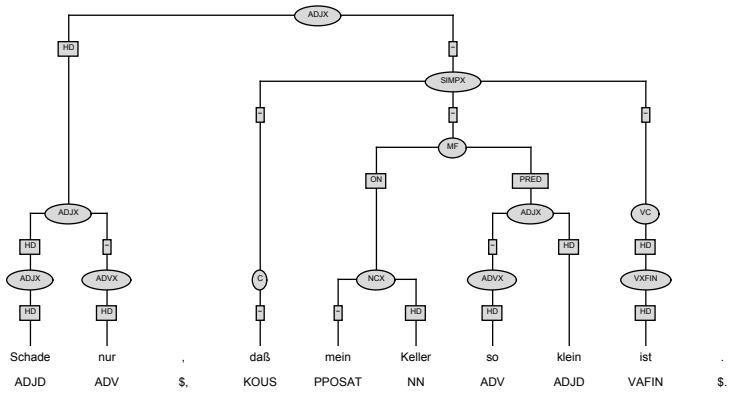


Figure 5: A sentence from TüBa-D/Z, in which unary nodes are annotated.

adverbial phrases, and prepositional phrases. Due to these great differences, a comparison of single constituents cannot be meaningful since one would compare, for example, all NPs in TüBa-D/Z to complexer NPs (with two words or more) in NEGRA.

Other differences concern the use of the POS tagset which are also reflected in phrase structure, e.g. stative passives, the attachment of relative clauses, and the treatment of comparative particles. For example, NEGRA treats comparative particles without a comparative semantic interpretation as prepositions, thus annotating such phrases as PPs. In TüBa-D/Z, in contrast, the presence of a comparative particle does not change the phrase type.

In the absence of more detailed methods of comparison, testing the effect of modifying individual annotation decisions gives insight into the factors that influence parsing results. As mentioned above, NEGRA and TüBa-D/Z differ in three major points (the fourth difference, crossing branches in NEGRA, is already addressed in preprocessing): flatter phrases and no unary nodes in NEGRA, and flatter structures on the clause level in TüBa-D/Z. In order to test the individual decisions, the opposite treebank is modified to also follow the respective decision. So in order to test the influence of not annotating unary nodes, all such nodes were removed from TüBa-D/Z while the other differences remained unchanged.

Consequently, the following modifications of the treebanks were executed:

- To test the influence of not annotating unary nodes (such as in NEGRA), all nodes with only one daughter were removed from TüBa-D/Z, preserving the grammatical functions. In the following section, this version will be named Tü_NU.

- To test the influence of NEGRA's flat phrase structure, phrases in TüBa-D/Z were flattened. This version will be named Tü_flat.

- In a third test, both modifications, the removal of unary nodes and the flattening of phrases were applied to TüBa-D/Z. The resulting tree for the sentence in Figure 2 is shown in Figure 6. This version will be named Tü_flat_NU.

- In order to test the influence of the flatter TüBa-D/Z structure on the clause level, topological fields were introduced into the NEGRA annotations. The topological fields were automatically extracted from the NEGRA corpus by the DFKI Saarbrücken. Since the NEGRA annotation in

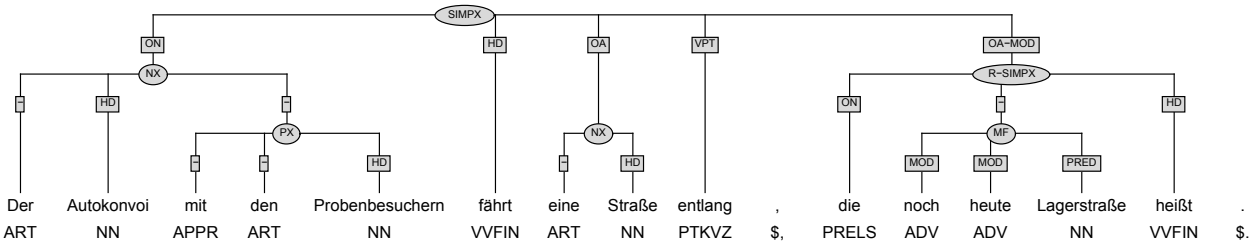


Figure 6: The sentence from Figure 2 in the flattened version without unary branches.

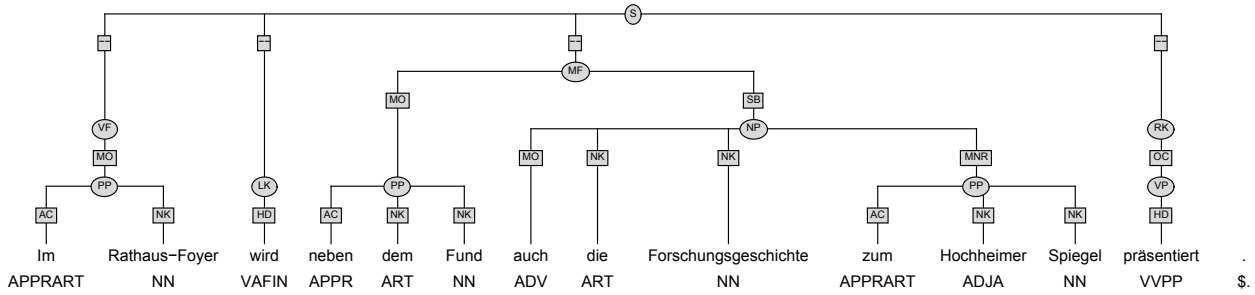


Figure 7: The sentence from Figure 1 with fields.

some cases does not contain enough information about the correct topological field, the conversion algorithm needs to use heuristics, which lead to a small number of errors in the field annotation.

The original annotation of NEGRA had to be modified when the topological fields were introduced. In many cases, the topological fields cross phrasal boundaries: These phrasal nodes were removed[4]. The resulting tree for the sentence in Figure 1 is shown in Figure 7. This version of NEGRA will be named NE_field.

The resulting modified treebanks were split into training and test data so that these sets contained the same sentences as the data sets for the baseline experiments. These data sets were then used for training and testing the parser on the modifications. The results of these experiments are shown in Table 2.

6 Discussion of the Results of the Comparison

Table 2 gives the results for the evaluation of the two types of tests: the upper half of the table gives resutls for parsing with syntactic node labels only and the lower half of the table gives results for parsing syntactic categories and grammatical functions. The results show that every transformation of the treebank annotations changes the results approximating those of the other treebank.

6.1 Modification of NEGRA

The modification of NEGRA, which **introduces topological fields** in order to flatten the clause structure, leads to an improved F-score but also to more crossing brackets. A first hypothesis would be that the improvement is due to the reliable recognition of the new field nodes. This hypothesis can be rejected by an evaluation of the parsing results for single syntactic categories. This evaluation shows that the introduction of topological fields gives high F-scores for the major fields, but it also improves both precision and recall for adverbial phrases, noun phrases, prepositional phrases, and almost all types of coordinated phrases, For adjectival phrases, precision improves from 55.95% to 64.46% - but at the same time, recall degrades from 56.38% to 50.97%. In contrast, the F-score for verb phrases deteriorates. This is probably due to the fact that only such verb phrases are annotated which do not cross field boundaries.

One reason for the improvement in the overall F-score is the change in the number of rules for a specific syntactic category. A look at the rules extracted from the training corpus shows a dramatic drop in numbers: for adjectival phrases, the number drops from more than 3900 rules containing AP to approximately3400 – even though new rules were added for the treatment of topological fields.

[4]We also tested a version in which the phrases were split into two to fit under the topological fields. However, this change resulted in lower precision and recall values.

	NEGRA	NE_field	TüBa-D/Z	Tü_NU	Tü_flat	Tü_flat_NU
crossing brackets	1.07	1.30	2.27	1.87	1.09	1.15
labeled recall	70.09%	75.21%	84.15%	77.41%	85.63%	77.43%
labeled precision	67.82%	77.17%	87.94%	81.52%	86.24%	76.44%
labeled F-score	68.94	76.18	86.00	79.41	85.93	76.93
sentences not parsed (%)	0.55%	0.05%	0.48%	1.91%	0.62%	2.26%
crossing brackets	1.04	1.21	1.93	2.17	1.07	1.29
function labeled recall	52.75%	69.85%	73.65%	62.11%	73.80%	53.63%
function labeled precision	51.85%	69.53%	76.13%	65.43%	74.66%	58.87%
function labeled F-score	52.30	69.19	74.87	63.73	74.23	56.13
sentences not parsed (%)	12.59%	2.17%	1.03%	9.98%	3.55%	18.87%
ratio nodes/words (in treebank)	0.88	1.38	2.38	1.33	2.00	1.06

Table 2: The results of comparing the modified versions of NEGRA and TüBa-D/Z.

6.2 Modification of TüBa-D/Z

Each modification of TüBa-D/Z results in a loss in F-score, but also in an improvement concerning crossing brackets. While flattening phrase structure only leads to minor changes, **deleting unary nodes** has a detrimental effect: the F-score drops from 86.00 to 79.41 when parsing syntactic constituents, and from 74.83 to 63.73 when parsing syntactic constituents including grammatical functions. Especially when parsing grammatical functions, deleting unary nodes leads to an increase of sentences that could not be parsed by a factor of almost 10. These sentences would have required additional rules not present in the training sentences. This leads to the question whether the deterioration is only due to the high number of sentences which were assigned no parse. However, an evaluation of only those sentences that did receive a parse shows only slightly better results in recall (obviously, precision remains the same): 68.18% for parsed sentences as compared to 62.11% for all sentences. This result, however, may also be caused by missing rules, which is corroborated by a look at the rules extracted from the test sentences: Approximately 24.0% of the rules needed for correctly parsing the test sentences in the modification without unary nodes are not present in the training set, as compared to 18.2% in the original version of the TüBa-D/Z treebank.

A closer look at the different constituents shows that the syntactic categories that are affected most by the **deletion of unary nodes** are noun phrases, finite verb phrases, adjectival phrases, adverbial phrases, and infinitival verb phrases. All those categories suffer losses in the F-score between 1.81% (for infinitival verb phrases) and 57.28% (for adverbial phrases). Since both precision and recall are similarly affected, this means that the parser does not only annotate spurious phrases but also misses phrases which should be annotated.

Flattening phrases in TüBa-D/Z has a negative effect on precision but it causes a slight increase in recall. The latter effect is a consequence of the bias of the PCFG parser, which prefers small trees. A comparison of the average number of nodes per word in a sentence shows that for all models, the parsed trees contain significantly fewer nodes than the gold standard trees. For the original TüBa-D/Z grammar including grammatical functions, the parsed tree contains 54.6% of the nodes in the gold standard; in the flattened version, the ratio is 58.6% (and for NEGRA, it is 62.5%).

The category that profits most from this modification is the category of named entities (*EN-ADD*). This is not surprising considering the fact that this node type does not serve a syntactic function, it is inserted above the syntactic category, which spans the named entity (cf. e.g. the named entity "Lagerstraße" in Figure 2). Flattening the structure often deletes the internal node and consequently allows the parser to base the annotation of named entities on more information than just a noun phrase node. This result is even more pronounced when also unary nodes are deleted. Other syntactic categories that profit from a flattening of the trees are prepositional phrases and relative clauses.

The **combination of both modifications** in TüBa-D/Z, flat phrase structure and deleted unary nodes, leads to a dramatic loss in the F-score for functional parsing as compared to the experiment in which only the unary nodes were deleted. A look at the unlabeled F-scores shows that this loss is not only due to incorrect labels for constituents, it also affects the recognition of phrase boundaries: the unlabeled F-score degrades from 91.34 for the original version of Tüba-

D/Z, to 81.06 for the version without unary nodes, and to 71.65 for the combination of both modifications.

7 Conclusion and Future Work

We have presented a method for comparing different annotation schemes and their influence on PCFG parsing. It is impossible to compare the performance of a parser on single syntactic categories since even rather similar annotation schemes apply different definitions for different phrase types. As a consequence, the comparison must be based on modifications within one annotation scheme to make it more similar to the other. The experiments presented here show that annotating unary nodes and structured phrases improve parsing results. On the clause level, however, a flatter structure incorporating topological fields is helpful for German.

The experiments presented here were conducted with a standard PCFG parser. The next logical step is to extend the comparison to different probabilistic parsers with different probability models and different biases. The (Charniak 00) parser or in the (Klein & Manning 03) parser use extensions of the probability model which were very successful for English. It is, however, unclear what the effect of these extensions is on German data.

Another area to be explored is lexicalization. Here, the picture is also unclear: Studies on the Penn treebank show that parsing results improve with lexicalized trees (cf. e.g. (Collins 97; Charniak 00)). The results on German (Dubey & Keller 03), however, show a detrimental effect of lexicalization for the NEGRA data. Thus, a comparison of treebank annotation schemes based on lexicalization only makes sense if a method of lexicalization can be found for both annotation schemes that does not overly decrease performance.

Another unexplored area for the two treebanks used here is the difference in grammatical functions. A comparison of grammatical functions, however, cannot be performed on the basis of a modification from one set to the other since there is no straightforward conversion from one set of grammatical functions to the other. For such a comparison, task-based evaluations of the parser trained on the two treebanks will be necessary.

Acknowledgments

We are grateful to the DFKI Saarbrücken for making their topological field annotation for the NEGRA treebank available to us. We would also like to thank Wolfgang Maier, Julia Trushkina, Holger Wunsch, and Tylman Ule for providing scripts for the conversion and evaluation of the data, and Erhard Hinrichs and Tylman Ule for many fruitful discussions.

References

(Bosco *et al.* 00) Cristina Bosco, Vincenzo Lombardo, D. Vassallo, and Leonardo Lesmo. Building a treebank for Italian: a data-driven annotation scheme. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation, LREC-2000*, Athens, Greece, 2000.

(Charniak 00) Eugene Charniak. A maximum-entropy-inspired parser. In *Proceedings of the 1st Conference of the North American Chapter of the Association for Computational Linguistics and the 6th Conference on Applied Natural Language Processing, ANLP/NAACL'2000*, pages 132–139, Seattle, WA, 2000.

(Charniak 01) Eugene Charniak. Immediate head parsing for language models. In *Proceedings of the 39th Annual Meeting of the ACL and the 10th Conference of the European Chapter of the ACL, ACL/EACL 2001*, pages 116–123, Toulouse, France, 2001.

(Collins 97) Michael Collins. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the ACL (jointly with the 8th Conference of the EACL)*, Madrid, Spain, 1997.

(Drach 37) Erich Drach. *Grundgedanken der Deutschen Satzlehre*. Diesterweg, Frankfurt/M., 1937.

(Dubey & Keller 03) Amit Dubey and Frank Keller. Probabilistic parsing for German using sister-head dependencies. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 96–103, Sapporo, Japan, 2003.

(Höhle 86) Tilman Höhle. Der Begriff "Mittelfeld", Anmerkungen über die Theorie der topologischen Felder. In *Akten des Siebten Internationalen Germanistenkongresses 1985*, pages 329–340, Göttingen, Germany, 1986.

(Klein & Manning 03) Dan Klein and Christopher Manning. Accurate unlexicalized parsing. In *Proceedings of ACL-2003*, pages 423–430, Sapporo, Japan, 2003.

(Marcus *et al.* 94) Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. The Penn Treebank: Annotating predicate argument structure. In *Proceedings of the ARPA Human Language Technology Workshop, HLT 94*, Plainsboro, NJ, 1994.

(Montegmagni *et al.* 00) S. Montegmagni, F. Barsotti, M. Battista, N. Calzolari, O. Corazzari, A. Zampolli, F. Fanciulli, M. Massetani, R. Raffaelli, R. Basili, M. T. Pazienza, D. Saracino, F. Zanzotto, N. Mana, F. Pianesi, and R. Delmonte. The Italian syntactic-semantic treebank: Architecture, annotation, tools and evaluation. In *Proceedings of the Workshop on Linguistically Interpreted Corpora LINC-2000*, pages 18–27, Luxembourg, 2000.

(Sampson 93) Geoffrey Sampson. The SUSANNE corpus. *ICAME Journal*, 17:125 – 127, 1993.

(Schmid 00) Helmut Schmid. LoPar: Design and implementation. Technical report, Universität Stuttgart, 2000.

(Skut *et al.* 97) Wojciech Skut, Brigitte Krenn, Thorsten Brants, and Hans Uszkoreit. An annotation scheme for free word order languages. In *Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP)*, Washington, D.C., 1997.

(Telljohann *et al.* 04) Heike Telljohann, Erhard Hinrichs, and Sandra Kübler. The TüBa-D/Z treebank: Annotating German with a context-free backbone. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004)*, pages 2229–2235, Lisbon, Portugal, 2004.

(Thielen & Schiller 94) Christine Thielen and Anne Schiller. Ein kleines und erweitertes Tagset fürs Deutsche. In Helmut Feldweg and Erhard Hinrichs, editors, *Lexikon & Text*, pages 215–226. Niemeyer, Tübingen, 1994.

Multilingual Lexical Acquisition by Bootstrapping Cognate Seed Lexicons

Kornél Markó and **Stefan Schulz**
Freiburg University Hospital
Medical Informatics Department
Stefan-Meier-Strasse 26
D-79104 Freiburg, Germany
www.imbi.uni-freiburg.de/medinf

Udo Hahn
Jena University
Language & Information Engineering Lab
Fürstengraben 30
D-07743 Jena, Germany
www.coling.uni-jena.de

Abstract

We present a methodology by which multilingual dictionaries (for Spanish, French and Swedish) emerge automatically from simple seed lexicons. The seed lexicons for the target languages are automatically generated by cognate mapping from (previously manually constructed) Portuguese, German as well as English sources. Lexical and semantic hypotheses are then validated by processing parallel corpora. In a last step, we use the cleaned list of 'approved' cognates in order to augment, step by step, the target dictionaries by processing the parallel corpora in terms of co-occurrence patterns of hypothesized translation equivalents which are not cognates.

1 Introduction

Applications of NLP to medical language up until now have mainly focused on monolingual tasks involving document retrieval or information extraction. The reason for widening their scope to include *multilingual* considerations as well is fairly evident. While clinical documents are typically written in the country's native language, searches in major bibliographic databases and the Web require sophisticated knowledge of English medical terminology. Hence, for cross-language information retrieval (CLIR) some sort of bridging between synonymous or, at least, related terms from different languages has to be done to make use of the information these sources hold.

Dictionaries for CLIR provide explicit lexical links within and between the languages involved. However, manually built lexical resources often lack coverage, since their construction and maintenance is costly and error-prone. Therefore, we propose a mechanism by which comprehensive dictionaries for CLIR can be automatically set up, relying on simple techniques and easily available resources. In previous studies (Schulz *et al.* 04), we showed how lexical cognates can be identified using unrelated (i.e., non-parallel, non-aligned) corpora. We here enhance this approach by relating non-cognate lexical items from different language pairs as well. In particular, we examine a bootstrapping approach in order to acquire Spanish, French, and Swedish lexicons, starting from already available Portuguese, English, and German lexicons.

2 Subwords as Basic Indexing Units

Our work starts from the assumption that neither fully inflected nor automatically stemmed words constitute the appropriate granularity level for lexicalized content description. Especially in scientific sublanguages, we observe a high frequency of domain-specific suffixes (e.g., *'-itis'*, *'-ectomia'* in the medical domain) and the construction of complex word forms such as in *'pseudo⊕hypo⊕para⊕thyroid⊕ism'*, or *'gluco⊕corticoid⊕s'*.[1]

In order to properly account for these particularities of 'medical' morphology, we developed the MORPHOSAURUS system.[2] It is centered around a lexicon, in which the entries are *subwords*, i.e., self-contained, semantically minimal units (cf. (Schulz *et al.* 02) for a distinction between subwords and linguistically motivated morphemes). We have found empirical evidence that subword-based document indexing improves the performance of cross-lingual document retrieval in the medical domain (Hahn *et al.* 04).

Subwords are assembled in a multilingual lexicon and thesaurus, which contain their entries, special attributes and semantic relations between them, according to the following considerations:

- Subwords are listed, together with their attributes such as language (English, German, Portuguese) and subword type (stem, prefix, suffix, invariant). Each lexicon entry is assigned one MORPHOSAURUS identifier representing one synonymy class, the MID.
- Synonymy classes which contain intralingual synonyms and interlingual translations of subwords are fused. Intra- and interlingual semantic equivalence are judged within the context of medicine only.
- Semantic links between synonymy classes are added. We subscribe to a shallow approach in which semantic relations are restricted to a single paradigmatic relation *has-meaning*, which

[1] '⊕' denotes the concatenation operator.
[2] http://www.morphosaurus.net

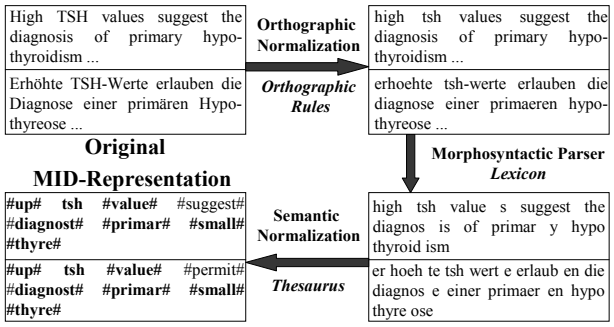


Figure 1: Morpho-Semantic Normalization Pipeline

Lang.	Seed Lexicon		Corpus	
	Stems	Affixes	Types	Tokens
POR	14,004	858	133,146	13,400,491
GER	21,705	680	17,151	161,952
ENG	21,501	540	11,349	56,317
SPA	-	824	82,431	3,979,051
FRE	-	197	43,105	2,284,646
SWE	-	633	47,823	957,904

Table 1: Resources Used for the Generation of Cognates

relates one ambiguous class to its specific readings,[3] (cf. (Markó *et al.* 05) for the disambiguation of subwords) and a syntagmatic relation *expands-to*, which consists of predefined segmentations in case of utterly short subwords.[4]

Figure 1 depicts how source documents (top-left) are converted into an interlingual representation by a three-step procedure. First, each input word is orthographically normalized in terms of lower case characters and according to language-specific rules for the transcription of diacritics (top-right). Next, words are segmented into sequences of subwords or left as is when no subwords can be decomposed (bottom-right). The segmentation results are checked for morphological plausibility using a finite-state automaton in order to reject invalid segmentations (e.g., segmentations without stems or beginning with a suffix). Finally, each meaning-bearing subword is replaced by a language-independent semantic identifier, its MID, thus producing the interlingual output representation of the system (bottom-left). A comparison of the original input (top-left) and the interlingual representation (bottom-left) already reveals the degree of (hidden) similarity uncovered by the overlapping MIDs.

3 Generation of Cognate Pairs

The manual construction of a trilingual lexicon and the thesaurus has consumed four person years. The combined subword lexicon contains (as of July 2005) 57,210 entries,with 21,501 for English, 21,705 for German, and 14,004 for Portuguese. In an effort to further expand the language coverage of the MORPHOSAURUS by Spanish, French, and Swedish, we wanted to reuse the already available resources for Portuguese, English, and German in order to speed up and to ease the lexicon acquisition process. The procedure for doing so can be divided into three separate steps. First, cognate pairs for typologically related languages such as Portuguese-Spanish are generated. Second, the generated lexical hypotheses are checked for validity considering simple corpus statistics. In a last step, we use the cleaned list of validated cognates to augment, step by step, the target lexicons by processing parallel corpora in terms of co-occurrence patterns of hypothesized translation equivalents which are *not* cognates. Table 1 lists the resources we used for the generation of cognate pairs:

- Manually established PORtuguese, ENGlish and GERman subword lexicons (stems and affixes).
- Manually created lists of SPAnish, FREnch, and SWEdish affixes. They were assembled by medical linguists based on introspection and heuristic support from various dictionaries.
- Medical corpora for all languages involved, all acquired from heterogeneous WWW sources.
- Word frequency lists, which were automatically generated from these corpora.

3.1 Subword Candidates

For the initialization of the target subword lexicons we pursued the following strategy: From the Portuguese (alternatively, English and German) lexicon, identical and similarly spelled Spanish (French, Swedish) subword candidates were generated. As an example, the Portuguese word stem *'estomag' ['stomach']* is identical with its Spanish cognate, while *'mulher' ['woman']* (Portuguese) is similar to *'mujer'* (Spanish). Similar subword candidates were generated by applying a set of string substitution rules, some of which are listed in Table 2. In total, we used 44 rules for Portuguese-Spanish, 26 rules for German-French, 18 rules for English-French, 19 rules for German-Swedish, and 6 rules for English-Swedish. These rules were all formulated by medical linguists based on introspection, also using various dictionaries for

[3] For instance, {*head*} ⇒ {*zephal,kopf,caput,cephal,cabec, cefal*} *OR* {*leader,boss,lider,chef*}

[4] For instance, {*myalg*} ⇒ {*muscle,muskel,muscul*} ⊕ {*pain, schmerz,dor*}

44 Rules:	*Portuguese*	*Spanish*
lh → j	mulher	mujer
+ca → za	cabeca	cabeza
26 Rules:	*German*	*French*
or → eur	tumor	tumeur
s → z	gas	gaz
18 Rules:	*English*	*French*
o → ou	movement	mouvement
ve → f	nerve	nerf
19 Rules:	*German*	*Swedish*
ei → e	bein	ben
+aa+ → a	saal	sal
6 Rules:	*English*	*Swedish*
ph → f	phosphor	fosfor
ce → s	iceland	island

Table 2: Some String Substitution Rules

heuristic guidance. Some of these substitution patterns cannot be applied to starting or ending sequences of characters in the source subword. This constraint is captured by a wildcard ('+' in Table 2), which stands for at least one arbitrary character.

Based on these string substitution rules and the already available (Portuguese, English, German) lexicons, for each entry (excluding affixes) of these sources, all possible Spanish, French and Swedish variant strings were generated. This led, on the average, to 8.8 Spanish variants per Portuguese subword (ranging from 2.7 for high-frequent four-character words to 355.2 for low-frequent 17-character words). Since the rule set is much smaller for the other language pairs, their average is far less than for Portuguese-Spanish (cf. Table 3).

All generated Spanish, French, and Swedish variants were subsequently compared with the target language word frequency list previously compiled from the text corpora. Wherever a (purely formal) prefix string match (in the case of stems) or an exact match (for invariants) occurred, the matching string was listed as a potential Spanish (French, Swedish) cognate of the Portuguese (alternatively, English and German) subword it originated from. Whenever several substitution alternatives for a source subword had to be considered that particular alternative was chosen which had the most similar lexical distribution in the corpora considered.

Similarity was measured as follows: Let S be the source lexical item, C_S the source language corpus containing n tokens and V_1, V_2, ..., V_p the hypotheses

Language Pair	String Variants			
	#Variants	4-chars	17-chars	over-all
POR-SPA	123,235	2.7	355.2	8.8
GER-FRE	68,999	2.0	9.1	3.2
ENG-FRE	46,122	1.6	5.6	2.2
GER-SWE	145,423	2.7	14.6	6.7
ENG-SWE	68,803	1.8	15.3	3.2

Table 3: Variant Generation: For each language pair (first column), the total number of variants is depicted in the second column. Columns three to five show variant averages per length.

generated from S that match the target language corpus C_T, containing m tokens. With $f(x, y)$ denoting the frequency of a word x in a corpus y, that particular V_j ($1 \leq j \leq p$) was chosen for which

$$\left| \frac{f(S, C_S)}{n} - \frac{f(V_j, C_T)}{m} \right|$$

was minimal. All other candidates were discarded.

As a result, we obtained a list of putative Spanish (French, Swedish) subwords each linked by the associated MID to their grounding source cognate in the Portuguese (alternatively, English and German) lexicon. We refer to these lists of cognate candidates as CC_{SPA} for Spanish, CC_{FRE} for French, and CC_{SWE} for Swedish.

As an example, starting from 14,004 Portuguese, 21,705 German and 21,501 English subwords (cf. Table 1), a total of 123,235 Spanish subword variants were created using the string substitution rules (cf. Table 3). Matching these variants against the Spanish corpus and allowing for a maximum of one candidate per source subword, we identified 8,644 tentative Spanish cognates. Combining English and German evidence, 9,536 French and 6,086 tentative Swedish cognates were found (cf. Table 4). Spanish candidates are linked to a total of 6,036 MIDs from their Portuguese correlates (hence, 2,608 synonym relationships have also been hypothesized), whilst French (Swedish) candidates are associated with 6,622 (4,157) MIDs from their German and English correlates (cf. Table 4).

3.2 Validation Using Parallel Corpora

We take advantage of the availability of large parallel corpora in the biomedical domain in order to identify *false friends*, i.e., similar words in different languages with different meanings. In our experiments, we found, e.g., the Spanish subword candidate **'crianz'* for the Portuguese *'crianc' ['child']* (the normalized stem of *'criança'*). The correct translation of

Language Pair	Source Lexicon	Selected Cognates	Linked MIDs
POR-SPA	14,004	8,644	6,036
GER-FRE	21,705	6,817	5,398
ENG-FRE	21,501	7,861	6,023
Combined Evidence		9,536	6,622
GER-SWE	21,705	4,249	3,308
ENG-SWE	21,501	4,140	3,208
Combined Evidence		6,086	4,157

Table 4: Selected Cognates

Portuguese *'crianc'* to Spanish, however, would have been *'nin'* (the stem of *'niño'*), whilst the Spanish *'crianz'* refers to *'criac' ['breed']* (stem of *'criação'* in Portuguese).

The corpora are made available by the *Unified Medical Language System* (UMLS 04), an umbrella system which currently combines more than one hundred heterogeneous medical terminologies (thesauri, classifications), most of them available in a couple of languages. Entries of these different nomenclatures are linked to each other via the UMLS *Metathesaurus*, which makes it possible to extract parallel corpora for various languages. Unfortunately, word-to-word translation occurs only in very few cases. More often one encounters rather complex noun phrases with a similarly complex semantic structure. Examples for typical English-Spanish alignments are *"Cell Growth"* aligned with *"Crecimiento Celular"*, or *"Heart transplant, with or without recipient cardiectomy"* aligned with *"Trasplante cardiaco, con o sin cardiectomia en el receptor"*.

We use English as the pivot language for our experiments, since it has the broadest coverage in the UMLS. The size of the corpora derived from the linkages of the English UMLS to other languages amounts to 60,526 alignments for English-Spanish,[5] 17,130 for English-French, and 10,953 alignments for English-Swedish. In order to determine the false friends in the list of the generated cognate pairs — CC_{SPA}, CC_{FRE} and CC_{SWE} — the parallel corpora of the aligned UMLS expressions were then morpho-semantically processed as described in Section 2. Whenever the same MID occurred on both sides after this simultaneous bilingual processing, the appropriate Spanish (French or Swedish, alternatively) subword entry that led to this particular MID is taken to be a valid entry. We think that this approach is reasonable, since it is highly unlikely that a false friend occurs within the same translation context.

[5] We only focused on the so-called *preferred entries*.

Language Pair	Hypotheses	Valid
POR-SPA	8,644	3,230 (37.4%)
GER/ENG-FRE	9,536	3,540 (37.1%)
GER/ENG-SWE	6,086	1,565 (25.7%)

Table 5: Cognates Matching the UMLS Alignments

Those hypotheses which never matched in this validation procedure were rejected from the candidate lexicons. As a result (cf. Table 5), 37% of the Spanish and French as well as 26% of the Swedish hypotheses are kept. These now serve as the seed lexicons (in the following, $L(0)$) for acquiring additional lexical entries, which are *not* cognates to elements of any of the source lexicons.

4 Lexical Learning Using Parallel Corpora

The parallel corpora derived from the UMLS and the lexicons with validated cognates both serve as starting points for a continuation of the lexical acquisition process, as described in Algorithm 1. In order to illustrate this process, assume the Swedish subword *'blod'* was identified as being a cognate to the English subword *'blood'* (and, therefore, is included in $L(0)$). Then, the yet unknown Swedish word *'blodtryck'*, which has the English translation *'blood pressure'* in the UMLS Metathesaurus gets segmented into [ST:*blod*|UK:*t*|SF:*r*|UK:*yck*], with ST being a marker for a stem, SF for a suffix and UK for an unknown sequence, thus satisfying the condition in line 12 of the algorithm. At the same time, the morpho-semantic normalization of *'blood pressure'* leads to the sequence of MIDs [#blood #tense], whilst the normalization of *'blodtryck'* leads to [#blood], since *'tryck'* is not yet part of the Swedish lexicon. Comparing these two representations, the condition in line 13 of the algorithm is satisfied, since there is exactly one more MID resulting from English which cannot be found in the Swedish normalization result. The invalid segment is then reconstructed (*'t⊕r⊕yck'*) by eliminating those substrings that led to a matching MID (*'blod'*) in the aligned unit (*'blodtryck'*) (line 15). The supernumerary MID resulting from the English normalization is assigned to that remaining substring (line 17 in the algorithm). After processing all UMLS alignments, this new entry is then incorporated in the Swedish lexicon as a stem, resulting in the lexicon L(1) (line 26). In the next run, in which all UMLS alignments are processed once again, this newly derived lexicon entry may serve for extracting, e.g., the Swedish word *'luft'* with its iden-

1: MSI: morpho-semantic indexing procedure from Section 2 (maps sequences of words to sequences of MIDs and remainders)
2: $current \leftarrow 0$
3: $quiescence \leftarrow$ false
4: **while** not $quiescence$ **do**
5: the lexicon for MSI is set to $L(current)$
6: the list of $new\_entries$ is empty
7: **for all** $AU_i, i \in$ [1,n] (UMLS alignment units) **do**
8: $AU_S \leftarrow$ source language part of AU_i
9: $AU_T \leftarrow$ target language part of AU_i
10: $MID_S \leftarrow MSI(AU_S)$
11: $MID_T \leftarrow MSI(AU_T)$
12: **if** for exactly one word there is an invalid segmentation (checked by the FSA) in MID_T **then**
13: **if** there is exactly one more MID in MID_S than in MID_T **then**
14: $mid \leftarrow$ supernumerary MID from MID_S
15: $entry \leftarrow$ restore the invalid segment and remove substrings that led to a matching MID in MID_S and MID_T;
16: strip off potential suffixes from $entry$, if the remaining substring is longer than 4 (thus, avoiding too short entries);
17: add $entry$ together with the associated mid to $new\_entries$
18: **end if**
19: **end if**
20: **end for**
21: **if** $new\_entries$ is empty **then**
22: $quiescence \leftarrow$ true
23: **else**
24: $current \leftarrow current + 1$
25: copy $L(current - 1)$ to $L(current)$
26: add all entries from $new\_entries$ to the lexicon $L(current)$
27: **end if**
28: **end while**

Algorithm 1: Bootstrapping Algorithm for Lexical Acquisition

tifier #aero from the UMLS entry *'air pressure'* (English, indexed to [#aero #tense]) linked to *'lufttryck'* (Swedish). When no new entries can be generated using this method (quiescence), the algorithm stops.

Table 6 depicts the growth steps of the target lexicons for the entire bootstrapping process (new entries in comparison to each previous step are in brackets). In the first run, for Spanish, 3,587 new lexemes are added to the lexicon which comes to a size of 6,817 including those lexemes already generated by the cognate identification routines (cf. Table 5). For French, 2,023 new lexemes were generated in the first step and for Swedish only 759. Remarkably, these Swedish entries lead to the acquisition of 1,361 new lexemes in the next step. After 14 runs, learning comes to an end with 7,154 lexemes generated for Spanish, while after 6 runs, 5,734 lexicon entries for French (Swedish, respectively) are acquired. Finally, for Swedish, 4,148 lexemes were learned after 9 iteration steps.

	Spanish	French	Swedish
L(0)	3,230	3,545	1,565
L(1)	6,817 (3,587)	5,568 (2,023)	2,324 (759)
L(2)	7,001 (184)	5,720 (152)	3,685 (1,361)
L(3)	7,094 (93)	5,730 (10)	4,013 (328)
L(4)	7,108 (14)	5,733 (3)	4,119 (106)
L(5)	7,109 (1)	5,734 (1)	4,136 (17)
...	...	...	...
L(14)	7,154 (45)	5,734 (0)	4,148 (12)

Table 6: Lexicon Growth Steps (Δ in brackets)

5 Quality Checking of Derived Lexicons

For lexicon generation, we referred to English-Spanish, English-French, and English-Swedish corpora compiled out of the UMLS Metathesaurus. To estimate the quality of the interlingual connections between the newly derived lexicons, we now compare the results after running the morpho-semantic indexing system (the function *MSI* from Algorithm 1) on these collections, at each stage of the lexical acquisition. We are aware that these results probably include overfitting phenomena.

Therefore, we additionally extracted Spanish-French (13,158), Spanish-Swedish (8,993) and French-Swedish (6,713) aligned entities from parallel corpora from the UMLS. The alignments range, again, from word-to-word translations (e.g., Spanish *'pierna'* to Swedish *'ben' ['leg']*) to complex noun phrases, which sometimes correspond to a single word in the other language, e.g., the Spanish phrase *'enfermedad virica transmitida por artropodos, no especificada'* maps to the Swedish *'arbovirusinfektioner" ['arbovirus infections']* in the UMLS.

Rather than only examining the coverage of the acquired lexicons, we wanted to estimate the quality of the generated lexicons (admitting that their status is far from being complete), i.e. the validity of the interlingual synonymy relations we stipulate. For this goal, we indexed the English-Spanish, English-French, and English-Swedish corpora on which the lexical acqui-

Lexicon	C	Cov.(%)	Ident.(%)	C	Cov.(%)	Ident.(%)	C	Cov.(%)	Ident.(%)
	English-Spanish (n = 60,526)			English-French (n = 17,130)			English-Swedish (n = 10,953)		
L(0)	39.6	87.6	6.1	39.2	78.3	16.1	27.4	60.0	11.7
L(1)	47.5	95.5	9.7	52.5	90.5	27.3	29.8	63.3	18.4
L(2)	51.0	95.6	11.8	53.2	90.8	27.9	50.7	81.8	39.9
L(5)	52.0	95.7	12.4	53.2	90.9	27.9	56.3	85.6	42.6
	Spanish-French (n = 13,158)			Spanish-Swedish (n = 8,993)			French-Swedish (n = 6,713)		
L(0)	34.9	73.6	17.2	21.4	53.8	8.9	32.4	66.7	17.9
L(1)	45.4	86.4	26.7	29.8	77.1	18.4	45.4	79.2	30.0
L(2)	45.7	86.7	27.0	40.6	80.1	23.9	45.8	79.5	30.0
L(5)	45.8	86.9	27.0	45.9	83.8	26.9	45.8	79.6	30.0

Table 7: Indexing Consistency (C), Coverage (Cov.) of Lexicons and Number of Identical Indexes (Ident.) at each Stage of Lexicon Generation. English-German Reference (n = 34,296): 56.9 Consistency, 96.9% Coverage, 29.8% Identical MIDs.

sition was based employing the MSI routines for all lexicon levels, L(0)-L(14). Furthermore, the Spanish-Swedish, Spanish-French, and French-Swedish corpora – previously unseen by the learning algorithm – were processed accordingly. For each alignment unit of the corpora, we then compared the resulting MIDs using the following measure of indexing consistency:

$$C_{AU_i} = (100A)/(A + N + M)$$

The indexing *consistency* of one alignment unit (AU_i) of the parallel corpus, C_{AU_i}, is dependent on A, the number of MIDs that co-occur on both sides of that unit in the parallel corpus and the number of MIDs that occur only on one of its sides, N or M. To express the overall consistency, the mean over all alignment units (C_{AU_i}) of the corpus is calculated.

Table 7 depicts the over-all consistency values (columns 2, 5 and 8) starting from lexicon L(0) (only validated cognates) to lexicon , L(1), L(2), up to L(5) for all target languages (improvements after that step are only marginal, cf. Table 6). When processing the English-Spanish corpus, consistency is already about 40%, only considering cognates using the C measure. This surprisingly high value is due to the high amount of overlapping medical terms in different Western European languages. Adding those entries acquired from bootstrapping the same corpus, consistency climbs to a maximum of 52%. As a reference item, the processing of an English-German corpus, which is also derived from UMLS, yields 57% consistency – keeping in mind that English and German lexicons were generated manually and provide a real good coverage (as shown, e.g. in (Hahn *et al.* 04)). The processing of Spanish-French, Spanish-Swedish, and French-Swedish is particularly interesting, since the underlying corpora were not involved at all in the lexical acquisition. With consistency starting from 35% for cognates (Spanish-French), 46% is reached after 5 cycles of generating the target lexicons, for each these language pairs.

Coverage was measured by counting those cases in which at least one MID occurs on both sides of the alignment units considered. For Spanish cognates only (L(0) in Table 7), (incomplete) alignments to English can be observed for 88% of the corpus. This value increases to 96% after 5 runs of bootstrapping the Spanish lexicon. For English-French, coverage reaches 91% (for English-Swedish 86%). For Spanish-French, Spanish-Swedish, and French-Swedish, surprisingly enough, coverage yields 87%, 84%, and 80%, respectively. Again, as a reference, the processing of the English-German corpus yields 97% coverage. The number of cases in which both sides are indexed identically, are depicted in Table 7, Columns four, seven, and ten. The reference data for these values is 30% for English-German.

6 Related Work

The rise of the empirical paradigm in the field of machine translation is, to a large degree, due to the widespread availability of parallel corpora.They also constitute an important resource for the automated acquisition of translational lexicons (Turcato 98). Most approaches to multilingual lexical acquisition employ statistical methods, such as context vector comparison (Rapp 99; Widdows *et al.* 02; Déjean *et al.* 02) or mutual information (Fung 98) and require a seed lexicon of trusted translations. (Koehn & Knight 02) derived such a seed lexicon from German-English cognates which were selected by using string similarity criteria (a method also favored by (Ribeiro *et al.* 01)). (Barker & Sutcliffe 00) propose an alternative *generative* approach where Polish cognate candidates are created from an English word list using string map-

Thesaurus	#	Subject
Eurovoc	13	European Communities activities: science, politics, law, culture, economics, etc.
GEMET	19	
UNESCO	3	
OECD	4	
Eurodicautom	12	technical terminology
Europ. Education	18	education, teaching, individual development research, etc.
Europ. Schools Treasury Browser	13	
AGROVOC	6	agriculture
Astronomy Thes.	5	astronomy

Table 8: Overview of Selected Multilingual Resources (`http://sky.fit.qut.edu.au/~middletm/cont_voc.html`, last visited in January 2005)

ping rules, an approach to cognate mapping also discussed by (MacWhinney 95) for 2nd language acquisition of human learners.

The second issue concerns the processing of suitable corpora. Whilst (Widdows *et al.* 02) deal with parallel German-English corpora to enrich an existing multilingual lexicon (also taken from the UMLS Metathesaurus), (Rapp 99), (Déjean *et al.* 02) and (Fung 98) propose methods that require only weaker comparable corpora (cf. (Fung 98) for a linguistic distinction between both types of corpora). Furthermore, (Déjean *et al.* 02) incorporate hierarchical information from an external thesaurus for combining different evidence for lexical acquisition.

In contradistinction to these precursors, we propose a fully heuristic method for acquiring translations of subwords, instead of using statistics. This is made possible by the availability of relatively large and well aligned parallel corpora, as provided within the UMLS Metathesaurus. Finally, rather than acquiring bilateral word translations, our focus lies on assigning subwords to interlingual semantic identifiers.

7 Conclusions

We have shown that a significant amount of Portuguese, English and German subwords from the medical domain can be mapped to Spanish, French, and Swedish cognates by simple string transformations. With these seeds, we further enlarge the cognate lexicons by subwords which are *not* cognates. For the latter task, we used a specific aligned corpus, the UMLS Metathesaurus, and extracted those non-cognates in a bootstrapping way.

In what concerns the generality of our approach, we rely on large aligned thesaurus corpora. Fortunately, large-coverage multilingual thesauri are already available for several relevant domains (cf. Table 8), both in terms of the number of languages covered and the number of alignment units available (e.g., on the order of 5 million for Eurodicautom). Hence, this approach bears further potential for lexicon acquisition tasks.

8 Acknowledgements

This work was partly supported by Deutsche Forschungsgemeinschaft (DFG), grant KL 640/5-2, and the European Network of Excellence *Semantic Mining* (NoE 507505).

References

(Barker & Sutcliffe 00) Gosia Barker and Richard F. E. Sutcliffe. An experiment in the semi-automatic identification of false-cognates between English and Polish. In *AICS 2000 – Irish Conference on Artificial Intelligence and Cognitive Science*. National University of Ireland Galway, 24-25 August, 2000, 2000.

(Déjean *et al.* 02) Hervé Déjean, Éric Gaussier, and Fatiha Sadat. An approach based on multilingual thesauri and model combination for bilingual lexicon extraction. In *COLING 2002 – Proceedings of the 19th International Conference on Computational Linguistics*, pages 218–224. Taipei, Taiwan, August 24 - September 1, 2002. Association for Computational Linguistics, 2002.

(Fung 98) Pascale Fung. A statistical view on bilingual lexicon extraction: From parallel corpora to non-parallel corpora. In David Farwell, Laurie Gerber, and Eduard H. Hovy, editors, *Machine Translation and the Information Soup. Proceedings of the 3rd Conference of the Association for Machine Translation in the Americas – AMTA 98*, volume 1529 of *Lecture Notes in Computer Science*, pages 1–17. Langhorne, PA, USA, October 28-31, 1998. Berlin: Springer, 1998.

(Hahn *et al.* 04) Udo Hahn, Kornél Markó, Michael Poprat, Stefan Schulz, Joachim Wermter, and Percy Nohama. Crossing languages in text retrieval via an interlingua. In *RIAO 2004 – Conference Proceedings: Coupling Approaches, Coupling Media and Coupling Languages for Information Retrieval*, pages 100–115. Avignon, France, 26-28 April 2004. Paris: Centre de Hautes Etudes Internationales d'Informatique Documentaire (CID), 2004.

(Koehn & Knight 02) Philipp Koehn and Kevin Knight. Learning a translation lexicon from monolingual corpora. In *Unsupervised Lexical Acquisition. Proceedings of the Workshop of the ACL Special Interest Group on the Lexicon (SIGLEX)*, pages 9–16. Philadelphia, PA, USA, July 12, 2002. Association for Computational Linguistics, 2002.

(MacWhinney 95) Brian MacWhinney. Language-specific prediction in foreign language learning. *Language Testing*, 12(3):292–320, 1995.

(Markó *et al.* 05) Kornél Markó, Stefan Schulz, and Udo Hahn. Unsupervised multilingual word sense disambiguation via an interlingua. In *AAAI'05 – Proceedings of the 20th National Conference on Artificial Intelligence & IAAI'05 – Proceedings of the 17th Innovative Applications of Artificial Intelligence Conference*, pages 1075–1080. Pittsburgh, Pennsylvania, USA, July 9-13, 2004. Menlo Park, CA; Cambridge, MA: AAAI Press & MIT Press, 2005.

(Rapp 99) Reinhard Rapp. Automatic identification of word translations from unrelated English and German corpora. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 519–526. College Park, MD, USA, 20-26 June 1999. San Francisco, CA: Morgan Kaufmann, 1999.

(Ribeiro *et al.* 01) António Ribeiro, Gaël Dias, Gabriel Lopes, and João Mexia. Cognates alignment. In *Proceedings of Machine Translation Summit VIII*, pages 287–293. Santiago de Compostela, Spain, September 18-22, 2001., 2001.

(UMLS 04) UMLS. *Unified Medical Language System*. Bethesda, MD: National Library of Medicine, 2004.

(Schulz *et al.* 02) Stefan Schulz, Martin Honeck, and Udo Hahn. Biomedical text retrieval in languages with a complex morphology. In Stephen Johnson, editor, *Proceedings of the ACL/NAACL 2002 Workshop on 'Natural Language Processing in the Biomedical Domain'*, pages 61–68. University of Pennsylvania, Philadelphia, PA, USA, July 11, 2002. New Brunswick, NJ: Association for Computational Linguistics (ACL), 2002.

(Schulz *et al.* 04) Stefan Schulz, Kornél Markó, Eduardo Sbrissia, Percy Nohama, and Udo Hahn. Cognate mapping: A heuristic strategy for the semi-supervised acquisition of a Spanish lexicon from a Portuguese seed lexicon. In *COLING Geneva 2004 – Proceedings of the 20th International Conference on Computational Linguistics*, volume 2, pages 813–819. Geneva, Switzerland, August 23-27, 2004. Association for Computational Linguistics, 2004.

(Turcato 98) Davide Turcato. Automaticaly creating bilingual lexicons for machine translation from bilingual text. In *COLING/ACL'98 – Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics & 17th International Conference on Computational Linguistics*, volume 2, pages 1299–1306. Montréal, Quebec, Canada, August 10-14, 1998. San Francisco, CA: Morgan Kaufmann, 1998.

(Widdows *et al.* 02) Dominic Widdows, Beate Dorow, and Chiu-Ki Chan. Using parallel corpora to enrich multilingual lexical resources. In M.G. Rodriguez and C. Paz Suarez Araujo, editors, *LREC 2002 – Proceedings of the 3rd International Conference on Language Resources and Evaluation. Vol. 1*, pages 240–245. Las Palmas de Gran Canaria, Spain, 29-31 May, 2002. Paris: European Language Resources Association (ELRA), 2002.

Term Representation with Generalized Latent Semantic Analysis

Irina Matveeva and **Gina-Anne Levow**
Department of Computer Science, the University of Chicago
Chicago, IL 60637
Ayman Farahat and **Christiaan Royer**
Palo Alto Research Center
Palo Alto, CA 94304
{matveeva,levow}@cs.uchicago.edu {farahat,royer}@parc.com

Abstract

Document indexing and representation of term-document relations are very important issues for document clustering and retrieval. In this paper, we present Generalized Latent Semantic Analysis as a framework for computing semantically motivated term and document vectors. Our focus on term vectors is motivated by the recent success of co-occurrence based measures of semantic similarity obtained from very large corpora. Our experiments demonstrate that GLSA term vectors efficiently capture semantic relations between terms and outperform related approaches on the synonymy test.

1 Introduction

Document indexing and representation of term-document relations are crucial for document classification, clustering and retrieval (Salton & McGill 83; Ponte & Croft 98; Deerwester *et al.* 90). Since many classification and categorization algorithms require a vector space representation for the data, it is often important to have a document representation within the vector space model approach (Salton & McGill 83). In the traditional bag-of-words representation (Salton & McGill 83) of the document vectors, words represent orthogonal dimensions which makes an unrealistic assumption about the independence of terms within documents.

Modifications of the representation space, such as representing dimensions with distributional term clusters (Bekkerman *et al.* 03) and expanding the document and query vectors with synonyms and related terms as discussed in (Levow *et al.* 05), improve the performance on average. However, they also introduce some instability and thus increased variance (Levow *et al.* 05). The language modelling approach (Salton & McGill 83; Ponte & Croft 98; Berger & Lafferty 99) used in information retrieval uses bag-of-words document vectors to model document and collection based term distributions.

Since the document vectors are constructed in a very high dimensional vocabulary space, there has also been a considerable interest in low-dimensional document representations. Latent Semantic Analysis (LSA) (Deerwester *et al.* 90) is one of the best known dimensionality reduction algorithms used in information retrieval. Its most appealing features are the ability to interpret the dimensions of the resulting vector space as semantic concepts and the fact that the analysis of the semantic relatedness between terms is performed implicitly, in the course of a matrix decomposition. LSA often does not perform well on large heterogeneous collections (Ando 00). Different related dimensionality reduction techniques proved successful for document clustering and retrieval (Belkin & Niyogi 03; He *et al.* 04; Callan *et al.* 03).

In this paper, we introduce Generalized Latent Semantic Analysis (GLSA) as a framework for computing semantically motivated term and document vectors. As opposed to LSA and other dimensionality reduction algorithms which are applied to documents, we focus on computing term vectors; document vectors are computed as linear combinations of term-vectors. Thus, unlike LSA (Deerwester *et al.* 90), Iterative Residual Rescaling (Ando 00), Locality Preserving Indexing (He *et al.* 04) GLSA is not based on bag-of-words document vectors. Instead, we begin with semantically motivated pair-wise term similarities to compute a representation for terms. This shift from dual document-term representation to term representation has the following motivation.

Terms offer a much greater flexibility in exploring similarity relations than documents. The availability of large document collections such as the Web offers a great resource for statistical approaches. Recently, co-occurrence based measures of semantic similarity between terms have been shown to improve performance on such tasks as the synonymy test, taxonomy induction, and document clustering (Turney 01; Terra & Clarke 03; Chklovski & Pantel 04; Widdows 03). On the

other hand, many semi-supervised and transductive methods based on document vectors cannot yet handle such large document collections and take full advantage of this information.

In addition, content bearing words, i.e. words which convey the most semantic information, are often combined into semantic classes that correspond to particular activities or relations and contain synonyms and semantically related words. Therefore, it seems very natural to represent terms as low dimensional vectors in the space of semantic concepts.

In this paper, we use a large document collection to extract point-wise mutual information, and the singular value decomposition as a dimensionality reduction method and compute term vectors. Our experiments show that the GLSA term representation outperforms related approaches on term-based tasks such as the synonymy test.

The rest of the paper is organized as follows. Section 2 contains the outline of the GLSA algorithm, and discusses the method of dimensionality reduction as well as the term association measures used in this paper. Section 4 presents our experiments, followed by conclusion in section 5.

2 Generalized Latent Semantic Analysis

2.1 GLSA Framework

The GLSA algorithm has the following setup. We assume that we have a document collection C with vocabulary V. We also have a large Web based corpus W.

1. Construct the weighted term-document matrix D based on C
2. For the vocabulary words in V, obtain a matrix of pair-wise similarities, S, using the large corpus W
3. Obtain the matrix U^T of a low dimensional vector space representation of terms that preserves the similarities in S, $U^T \in R^{k \times |V|}$
4. Compute document vectors by taking linear combinations of term vectors $\hat{D} = U^T D$

The columns of $\hat{D}$ are documents in the k-dimensional space.

The motivation for the condition on the low dimensional representation in step 3 can be explained in the following way. Traditionally, cosine similarity between term and document vectors is used as a measure of semantic association. Therefore, we would like to obtain term vectors so that their pair-wise cosine similarities correspond to the semantic similarity between the corresponding vocabulary terms. The extent to which these latter similarities can be preserved depends on the dimensionality reduction method. Some techniques aim at preserving all pair-wise similarities, for example, the singular value decomposition used in this paper. Some graph-based approaches, on the other hand, preserve the similarities only locally, between the pairs of most related terms, e.g. Laplacian Eigenmaps Embedding (Belkin & Niyogi 03), Locality Preserving Indexing (He *et al.* 04).

The GLSA approach can combine any kind of similarity measure on the space of terms with any suitable method of dimensionality reduction. The traditional term-document matrix is used in the last step to provide the weights in the linear combination of term vectors.

In step 2, it is possible to compute the matrix S for the vocabulary of the large corpus W and use the term vectors to represent the documents in C. In addition to being computationally demanding, however, this approach would suffer from noise introduced by typos and infrequent and non-informative words. Finding methods of efficient filtering of the core vocabulary and keeping only content bearing words would be another way of addressing this issue. This is subject of future work.

2.1.1 Document Vectors

One of the advantages of the term-based GLSA document representation is that it does not have the out-of-sample problem for new documents. It does have this problem for new terms, but new terms appear at a much lower rate than documents. In addition, new rare terms will not contribute much to document classification or retrieval. Since the computation of the term vectors is done off-line, the GLSA approach would require occasional updates of the term representation.

GLSA provides a representation for documents that reflects their general semantics. Since GLSA does not transform the document vectors in the course of computation, the GLSA document representation can be easily extended to contain more specific information such as presence of proper names, dates, or numerical information.

2.2 Low-dimensional Representation

2.2.1 Singular Value Decomposition

In this section we outline some of the basic properties of the singular value decomposition (SVD) which we use as a method of dimensionality reduction. SVD is applied to the matrix S that contains pair-wise similarities between the vocaburaly terms.

First, consider the eigenvalue decomposition of S. Since S is a real symmetric matrix, it is diagonizable, i.e. it can be represented as

$$S = U\Sigma U^T$$

The columns of U are the orthogonal eigenvectors of S. Σ is a diagonal matrix containing the corresponding eigenvalues of S.

If in addition, S is positive semi-definite, it can be represented as a product of two matrices $S = \hat{U}\hat{U^T}$, and in this case $\hat{U} = U\Sigma^{1/2}$. This means that the entries of S, which in the GLSA case represent pair-wise term similarities, are inner products between the eigenvectors of S scaled with the corresponding eigenvalues.

The singular value decomposition of S is $S = U\bar{\Sigma}V^T$, where U and V are column orthogonal matrices containing the left and right singular vectors of S, respectively. $\bar{\Sigma}$ is a diagonal matrix with the singular values sorted in decreasing order.

Eckart and Young, see (Golub & Reinsch 71), have shown that given any matrix S and its singular value decomposition $S = U\Sigma V^T$, the matrix $S_k = U_k\Sigma_k V_k^T$ obtained by setting all but the first k diagonal elements in Σ to zero is

$$S_k = \text{argmin}_X ||S - X||_F^2,$$

where X is a matrix of rank k. The minimum is taken with respect to the Frobenius norm, where $||A||_F^2 = \sum_{ij} A_{ij}^2$.

The SVD of a symmetric matrix of pair-wise term similarities S is the same as its eigenvalue decomposition. Therefore, the method for computing a low-dimensional term representation that we used in this paper is to compute the eigenvalue decomposition of S and to use k eigenvectors corresponding to the largest eigenvalues as a representation for term vectors. Thus, the cosine similarities between the low dimensional GLSA term vectors preserve the semantic similarities in the matrix S for each pair of terms.

LSA is one special case within the GLSA framework. Although it begins with the document-term matrix, it can be shown that LSA uses SVD to compute the rank k approximation to a particular matrix of pair-wise term similarities. In the LSA case, these similarities are computed as the inner products between the term vectors in the space of documents, see (Bartell *et al.* 92) for details. If the GLSA matrix S is positive semi-definite, its entries represent inner products between term vectors in a feature space. Thus, GLSA with the eigenvalue decomposition can be interpreted as kernelized LSA, similar to the kernel PCA (Schölkopf *et al.* 98). Since S contains co-occurrence based similarities which have been shown to reflect semantic relations between terms, GLSA uses semantic kernels.

2.2.2 PMI as Measure of Semantic Association

We propose to obtain the matrix of semantic associations between all pairs of vocabulary terms using a number of well-established methods of computing collection-based term associations, such as point-wise mutual information, likelihood ratio, χ^2 test etc. (Manning & Schütze 99). In this paper we use point-wise mutual information (PMI) because it has been successfully applied to collocation discovery and semantic proximity tests such as the synonymy test and taxonomy induction (Manning & Schütze 99; Turney 01; Terra & Clarke 03; Chklovski & Pantel 04; Widdows 03). It was also successfully used as a measure of term similarity to compute document clusters (Pantel & Lin 02), and to extract semantic relations between verbs (Chklovski & Pantel 04).

The point-wise mutual information between random variables representing two words, w_1 and w_2, is computed as

$$PMI(w_1, w_2) = \log \frac{P(W_1 = 1, W_2 = 1)}{P(W_1 = 1)P(W_2 = 1)}.$$

The similarity matrix S with pair-wise PMI scores may not be positive semi-definite. Since such matrices work well in practice (Cox & Cox 01) one common approach is to use only the eigenvectors corresponding to the positive eigenvalues (Cox & Cox 01). This is the approach which we use in our experiments.

3 Related Approaches

As mentioned above, most related approaches compute a dual document-term representation based on the same document collection. Iterative Residual Rescaling (Ando 00) tries to put more weight on documents from underrepresented clusters of documents to improve the performance of LSA on heterogeneous collections. Random Indexing (Sahlgren & Coester 04) projects the document vectors on random low-dimensional vectors. Locality Preserving Indexing (He *et al.* 04) is a graph-based dimensionality reduction algorithm which preserves the similarities only locally. LPI differs from LSA due to the notion of locality, which is incorporated through a linear transformation of the term-document matrix. GLSA can be used with semantically motivated non-linear kernel matrices S.

Recent applications of LSA tried to compute term vectors using large collections. Document vectors for other collections are constructed as linear combinations of LSA term vectors. As mentioned above, LSA uses only one particular measure of term similarity. The Word Space Model for word sense disambiguation developed by Schütze (Schütze 98) is another special case of GLSA which computes term vectors directly. Instead of using document co-occurrence statistics, it uses term co-occurrence in the contexts of the most frequent informative terms, then SVD is applied. One particular kind of co-occurrence based similarities, namely normalized counts, are used (Schütze 98; Widdows 03). Latent Relational Analysis (Turney 04) looks at pair-wise relations between selected terms and not at term vectors for the whole vocabulary and uses co-occurrence counts within context patterns. SVD is applied to the matrix of similarities between the context patterns as a method of smoothing the similarity information.

The probabilistic LSA (Hofmann 99) and Latent Dirichlet Allocation (Blei *et al.* 02) use the latent semantic concepts as bottleneck variables in computing the term distributions for documents. The probabilities are estimated using the EM algorithm which can suffer from local minima and has a large space requirement. This limits the use of these approaches for large document collection.

4 Experiments

The goal of the experimental evaluation of the GLSA term vectors was to demonstrate that the GLSA vector space representation for terms captures their semantic relations. We used the synonymy and term pairs tests for the evaluation. Our results demonstrate that similarities between GLSA term vectors achieve better results than the latest approaches based on PMI scores (Terra & Clarke 03).

To collect the co-occurrence for the matrix of pair-wise term similarities S, in all experiments presented here we used the English Gigaword collection (LDC), containing New York Times articles. We only used the documents that had the label "story". Thus, we used a collection comprised of 1,119,364 documents with 771,451 terms. We used the Lemur toolkit[1] to tokenize and index all document collections used in our experiments; we used stemming and a list of stop words.

The similarities matrix S was constructed using the PMI scores. In our preliminary experiments we used some other co-occurrence based measures of similarities, such as likelihood ratio and χ^2 test but obtained results which were below those for PMI. Therefore, we do not report them here. We used the PMI matrix S in combination with SVD (denoted as $GLSA$) to compute GLSA term vectors. Unless stated otherwise, for the GLSA method we report the best performance over different numbers of embedding dimensions. We used the PLAPACK package[2] to perform the SVD (Bientinesi *et al.* 03).

4.1 Synonymy Test

The synonymy test represents a list of words and for each of them, there are 4 candidate words. The task is to determine which of these candidate words is a synonym to the word in question. This test was first used to demonstrate the effectiveness of LSA term vectors (Landauer & Dumais 97). More recently, the PMI-IR approach developed by Turney (Turney 01) was shown to outperform LSA on this task (Turney 01) and (Terra & Clarke 03).

We evaluated the GLSA term vectors on the synonymy test and compared the results to the latest results with the PMI-IR approach (Terra & Clarke 03). Terra et al. (Terra & Clarke 03) com-

[1] http://www.lemurproject.org/

[2] http://www.cs.utexas.edu/users/plapack/

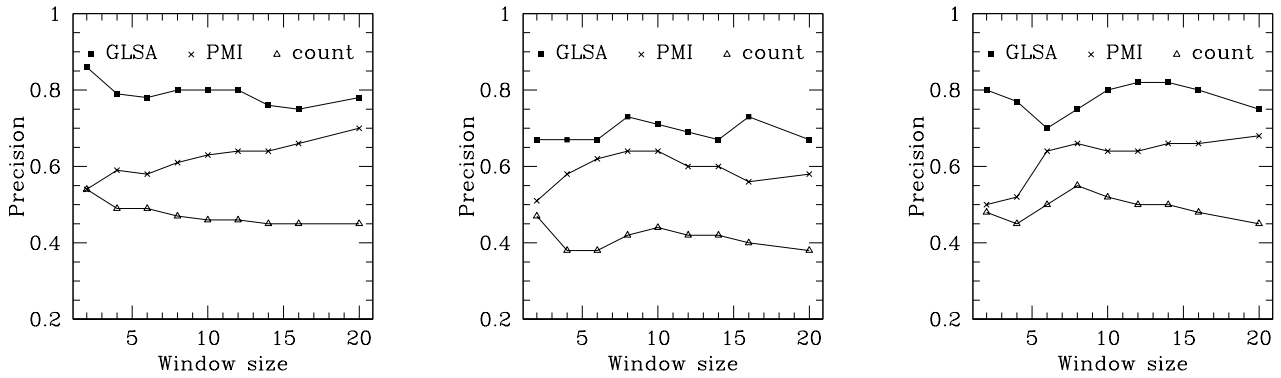


Figure 1: Precision with GLSA, PMI and count over different window sizes, for the TOEFL(left), TS1(middle) and TS2(right) tests.

pared the performance of different co-occurrence based measures of term similarity on the synonymy test and came to the conclusion that PMI yielded the best results.

Following (Terra & Clarke 03), we used the TOEFL, TS1 and TS2 synonymy tests. The TOEFL test contains 80 synonymy questions. We also used the preparation tests called TS1 and TS2. Since GLSA in its present formulation cannot handle multi-word expressions, we had to modify the TS1 and TS2 tests slightly. We removed all test questions that contained multi-word expressions. From 50 TS1 questions we used 46 and from 60 TS2 questions we used 49. Thus, we would like to stress that the comparison of our results on TS1 and TS2 to the results reported in (Terra & Clarke 03) is only suggestive. We used the TS1 and TS2 test without context. The only difference in the experimental setting for the TOEFL test between our experiments and the experiments in (Terra & Clarke 03) is in the document collections that were used to obtain the co-occurrence information.

4.1.1 GLSA Setting

To have a richer vocabulary space, we added the 2000 most frequent words from the English Gigaword collection to the vocabularies of the TOEFL, TS1 and TS2 tests. We computed GLSA term vectors for the extended vocabularies of the TOEFL, TS1 and TS2 tests and selected the term t^* whose term vector had the highest cosine similarity to the question term vector $\vec{t_q}$ as the synonym. We computed precision scores as the ratio of correctly guessed synonyms.

The co-occurrence counts can be obtained using either term co-occurrence within the same document or within a sliding window of certain fixed size. In our experiments we used the window-based approach which was shown to give better results (Schütze 98; Terra & Clarke 03). Since the performance of co-occurrence based measures is sensitive to the window size, we report the results for different window sizes.

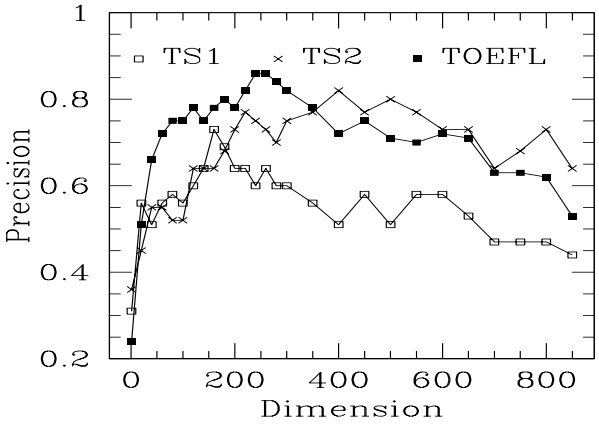


Figure 2: Precision at different numbers of GLSA dimensions with the best window size.

4.1.2 Results on the Synonymy Test

Figure 1 shows the precision using different window sizes. The baselines are to choose the candidate with the highest co-occurrence count or PMI score. For all three data sets, GLSA significantly outperforms PMI scores computed on the same collection. The results that we obtained using just the PMI score are below those reported in Terra and Clarke (Terra & Clarke 03). One explanation for this discrepancy is the size and the composition of the document collections used for the co-occurrence statistics. The English Gigaword collection that we used is smaller and, more im-

portantly, less heterogeneous than the web based collection in (Terra & Clarke 03). Nonetheless, on the TOEFL data set GLSA achieves the best precision of 0.86, which is much better than our PMI baseline as well as the highest precision of 0.81 reported in (Terra & Clarke 03). GLSA achieves the same maximum precision as in (Terra & Clarke 03) for TS1 (0.73)and a much higher precision on TS2 (0.82 vs. 0.75 in (Terra & Clarke 03)).

Figure 2 shows the precision for the GLSA terms only, using different number of dimensions. The number of dimensions is important because it is one of the parameter in the GLSA setting. LSA-based approaches usually perform best with 300-400 resulting dimensions. The variation of precision at different numbers of embedding dimensions within the 100-600 range is somewhat high for TS1 but much smoother for the TOEFL and TS2 tests.

4.2 Term Pairs Test

Some of the terms on the synonymy test are infrequent (eg. "wig") and some are usually not considered informative (eg. "unlikely"). We used the following test to evaluate how the cosine similarity between GLSA vectors captures similarity between terms which are considered important for such tasks as document classification.

We computed GLSA term vectors for the vocabulary of the 20 news groups document collection. Using the Rainbow software[3] we obtained the top N words with the highest mutual information with the class label. We also obtained the probabilities that each of these words has with respect to each of the news groups. We assigned the group in which the word has the highest probability as the word's label. Some of the top words and their labels can be seen in Table 3. Although the way we assigned labels may not strictly correspond to the semantic relations between words, this table shows that for this particular collection and for informative words (e.g., "bike","team") they do make sense.

We computed pair-wise similarities between the top N words using the cosine between the GLSA vectors representing these words and also used just the PMI scores. Then we looked at the pairs of terms with the highest similarities. Since for this test we selected content bearing words, the intuition is that most similar words should be semantically related and are likely to appear in documents belonging to the same news group. Therefore, they should have the same label. Each word can also be considered a query, and in this test we are trying to retrieve other words that are semantically most related to the it.

This task is better suited to demonstrate the advantage of GLSA over PMI-IR. In the synonymy task the comparisons are made between the PMI scores of a few carefully selected terms that are synonymy candidates for the same word. While PMI-IR performs quite well on the synonymy task, it is in general difficult to compare PMI scores across different pairs of words. Apart from this normalization issue, PMI scores for rare words tend to be very high, see (Manning & Schütze 99). Our experiments illustrate that GLSA significantly outperforms the PMI scores on this test.

We used $N = \{100, 1000\}$ top words by the MI with the class label. The top 100 are highly discriminative with respect to the news group label whereas the top 1000 words contain many frequent words. Our results show that GLSA is much less sensitive to this than PMI.

First we sort all pairs of words by similarity and compute precision at the k most similar pairs as the ratio of word pairs that have the same label. Table 1 shows that GLSA significantly outperforms the PMI score. PMI has very poor performance, since here the comparison is done across different pairs of words.

The second set of scores was computed for each word as precision at the top k nearest terms, similar to precision at the first k retrieved documents used in IR. We report the average precision values for different values of k in Table 2. GLSA achieves higher precision than PMI. GLSA performance has a smooth shape peaking at around 200-300 dimension which is in line with results for other SVD-based approaches (Deerwester *et al.* 90; He *et al.* 04). The dependency on the number of dimensions was the same for the top 1000 words.

In Table 3 we show the individual results for some of the words. GLSA representation achieves very good results for terms that are not very frequent in general document collections but are very good indicators of particular news groups, such as "god" or "bike". For much more frequent words, and words which have multiple senses,

[3]http://www-2.cs.cmu.edu/ mccallum/bow/rainbow/

	top 100		top 1000	
k	Pmi	Glsa	Pmi	Glsa
1	0.0	1.0	0.0	1.0
5	0.0	1.0	0.0	1.0
10	0.0	1.0	0.0	0.8
50	0.32	0.88	0.12	0.8
100	0.24	0.76	0.1	0.8

Table 1: Precision for the term pairs test at the top k most similar pairs.

	top 100		top 1000	
k	Pmi	Glsa	Pmi	Glsa
1	0.27	0.67	0.08	0.43
5	0.40	0.48	0.8	0.40
10	0.35	0.37	0.1	0.37
50	0.14	0.13	0.16	0.20
100	0.08	0.08	0.16	0.18

Table 2: Average precision for the term pairs test at the top k nearest words.

word	nn=1	nn=2	nn=3	Prec
god (18)	jesus (18)	bible (18)	heaven (18)	1
bike (15)	motorcycle (15)	rider (15)	biker (15)	1
team (17)	coach (17)	league (20)	game (17)	0.6
car (7)	driver (1)	auto (7)	ford (7)	0.6
windows (1)	microsoft (1)	os (3)	nt (1)	0.4
dod (15)	agency (10)	military (13)	nsa (10)	0
article (15)	publish (13)	fax (4)	contact (5)	0

Table 3: Precision at the 5 nearest terms for some of the top 100 words by mutual information with the class label. The table also shows the first 3 nearest neighbors. The word's label is given in the brackets. (1=os.windows; 3=hardware; 4=graphics; 5=forsale; 7=autos; 10=crypt; 13=middle-east;15=motorcycles; 17=hokey; 18=religion-christian; 20=baseball.)

such as "windows" or "article", the precision is lower. The pair "car", "driver" is semantically related for one sense of the word "driver", but the word "driver" is assigned to the group "windows-os" with a different sense.

5 Conclusion and Future Work

Our experiments have shown that the cosine similarity between the GLSA term vectors corresponds well to the semantic similarity between pairs of terms. Interesting questions for future work are connected to the computational issues. As other methods based on a matrix decomposition, GLSA is limited in the size of vocabulary that it can handle efficiently. Since terms can be divided into content-bearing and function words, GLSA computations only have to include content-bearing words. Since the GLSA document vectors are constructed as linear combinations of term vectors, the inner products between the term vectors are implicitly used when the similarity between the document vectors is computed. Another interesting extension is therefore to incorporate the inner products between GLSA term vectors into the language modelling framework and evaluate the impact of the GLSA representation on the information retrieval task.

We have presented the GLSA framework for computing semantically motivated term and document vectors. This framework allows us to take advantage of the availability of large document collection and recent research of corpus-based term similarity measures and combine them with dimensionality reduction algorithms. Using the combination of point-wise mutual information and singular value decomposition we have obtained term vectors that outperform the state-of-the-art approaches on the synonymy test and show a clear advantage over the PMI-IR approach on the term pairs test.

Acknowledgements We are very grateful to Paolo Bientinesi for his extensive help with adopting the PLAPACK package to our problem. The TOEFL questions were kindly provided by Thomas K. Landauer, Department of Psychology, University of Colorado. This research has been funded in part by contract #MDA904-03-C-0404 to Stuart K. Card and Peter Pirolli from the Advanced Research and Development Activity, Novel Intelligence from Massive Data program.

References

(Ando 00) Rie Kubota Ando. Latent semantic space: iterative scaling improves precision of inter-document similarity measurement. In *Proc. of the 23rd ACM SIGIR*, pages 216–223, 2000.

(Bartell *et al.* 92) Brian T. Bartell, Garrison W. Cottrell, and Richard K. Belew. Latent semantic indexing is an optimal special case of multidimensional scaling. In *Proc. of the 15th ACM SIGIR*, pages 161–167. ACM Press, 1992.

(Bekkerman *et al.* 03) Ron Bekkerman, Ran El-Yaniv, and Naftali Tishby. Distributional word clusters vs. words for text categorization, 2003.

(Belkin & Niyogi 03) Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

(Berger & Lafferty 99) Adam Berger and John Lafferty. Information retrieval as statistical translation. In *Proc. of the 22rd ACM SIGIR*, 1999.

(Bientinesi *et al.* 03) Paolo Bientinesi, Inderjit S. Dhilon, and Robert A. van de Geijn. A parallel eigensolver for dense symmetric matrices based on multiple relatively robust representations. *UT CS Technical Report TR-03-26*, 2003.

(Blei *et al.* 02) David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. In *Proc. of 14th NIPS*, New York, 2002. ACM.

(Callan *et al.* 03) Jamie Callan, Gordon Cormack, Charles Clarke, David Hawking, and Alan Smeaton. Document clustering based on non-negative matrix factorization. In *Proc. of the 26rd ACM SIGIR*, New York, 2003. ACM.

(Chklovski & Pantel 04) Timothy Chklovski and Patrick Pantel. Verbocean: Mining the web for fine-grained semantic verb relations. In *Proc. of EMNLP*, 2004.

(Cox & Cox 01) Trevor F. Cox and Micheal A. Cox. *Multidimensional Scaling*. CRC/Chapman and Hall, 2001.

(Deerwester *et al.* 90) Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

(Golub & Reinsch 71) G. Golub and C. Reinsch. *Handbook for Matrix Computation II, Linear Algebra*. Springer-Verlag, New York, 1971.

(He *et al.* 04) Xiaofei He, Deng Cai, Haifeng Liu, and Wei-Ying Ma. Locality preserving indexing for document representation. In *Proc. of the 27rd ACM SIGIR*, pages 96–103. ACM Press, 2004.

(Hofmann 99) Thomas Hofmann. Probabilistic latent semantic analysis. In *Uncertainty in Artificial Intelligence*, 1999.

(Landauer & Dumais 97) Thomas K. Landauer and Susan T. Dumais. A solution to platos problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*, 1997.

(Levow *et al.* 05) Gina-Anne Levow, Douglas W. Oard, and Philip Resnik. Dictionary-based techniques for cross-language information retrieval. *Information Processing and Management: Special Issue on Cross-language Information Retrieval*, 2005.

(Manning & Schütze 99) Chris Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press. Cambridge, MA, 1999.

(Pantel & Lin 02) Patrick Pantel and Dekang Lin. Document clustering with committees. In *Proc. of the 25th ACM SIGIR*, pages 199–206. ACM Press, 2002.

(Ponte & Croft 98) Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proc. of the 21st ACM SIGIR*, pages 275–281, New York, NY, USA, 1998. ACM Press.

(Sahlgren & Coester 04) Magnus Sahlgren and Rickard Coester. Using bag-of-concepts to improve the performance of support vector machines in text categorization. In *Proc. of the 20th COLING*, pages 487–493, 2004.

(Salton & McGill 83) Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

(Schölkopf *et al.* 98) Bernhard Schölkopf, Alex J. Smola, and Klaus-Robert Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10, 1998.

(Schütze 98) Hinrich Schütze. Automatic word sense discrimination. *Computational Linguistics*, 24(21):97–124, 1998.

(Terra & Clarke 03) Egidio L. Terra and Charles L. A. Clarke. Frequency estimates for statistical word similarity measures. In *Proc.of HLT-NAACL*, 2003.

(Turney 01) Peter D. Turney. Mining the web for synonyms: PMI–IR versus LSA on TOEFL. *Lecture Notes in Computer Science*, 2167:491–502, 2001.

(Turney 04) Peter D. Turney. Human-level performance on word analogy questions by latent relational analysis. Technical report, Technical Report ERB-1118, NRC-47422, 2004.

(Widdows 03) Dominic Widdows. Unsupervised methods for developing taxonomies by combining syntactic and statistical information. In *Proc. of HLT-NAACL*, 2003.

Using NLP to build the hypertextual network of a back-of-the-book index

Touria Aït El Mekki and Adeline Nazarenko
LIPN —University of Paris13 & CNRS UMR 7030
Av. J.B. Clément, F-93430 Villetneuse, France
{taem,nazarenko}@lipn.univ-paris13.fr

Abstract

Relying on the idea that back-of-the-book indexes are traditional devices for navigation through large documents, we have developed a method to build a hypertextual network that helps the navigation in a document. Building such an hypertextual network requires selecting a list of descriptors, identifying the relevant text segments to associate with each descriptor and finally ranking the descriptors and reference segments by relevance order. We propose a specific document segmentation method and a relevance measure for information ranking. The algorithms are tested on 4 corpora (of different types and domains) without human intervention or any semantic knowledge.

1 Introduction

Helping readers to get access to the document content is a text-mining challenge. Back-of-the-book indexes are traditional devices that provide an overview of the document content and help the reader to navigate through the document. An index[1] is "an alphabetical list of persons, places, subjects, etc., mentioned in the text of a printed work, usually at the back, and indicating where in the work they are referred to"[2]. More formally, an index is made of a nomenclature, which is a (structured) list of descriptors, and of a large set of references that link the descriptors to document segments. Such indexes are also designed for electronic documents and for web sites[3].

We have designed a method for automating the building of indexes. Our IndDoc system relies on the text of the document 1) to select the descriptors that are worth mentioning in the final index and 2) to link each descriptor to document segments. We do not address the first point here[4]. We rather focus on the elaboration of the hypertextual network.

Building such a network raises two problems. The first one is the *segmentation problem*. For each relevant descriptor, it is necessary to identify the relevant document segments to refer to. The difficult point is not to identify the various text occurrences of a descriptor, but to determine, for a given occurrence of a descriptor, to which span of text (short paragraph or whole section) it is necessary to refer. There is also a *relevance-ranking problem*. Linking all descriptors to all their occurrences would introduce too many links and work against navigation. A relevance measure must be defined to select the most important links.

Section 2 presents the previous works on navigational tools and segmentation or ranking methods. Our method is described in section 3. The section 4 presents our experiments and results.

2 Previous works

2.1 Existing indexing tools

Existing computer-aided indexing tools are either embedded in word processing or stand-alone software such as Macrex[5] and Cindex[6]. They are designed to assist a human indexer. They locate the various occurrences of a descriptor, automatically compute the page numbers for references, rank the entries in alphabetic order and format the resulting index according to a given index style sheet. However, the indexer still has to choose the relevant descriptors. In the best case, the indexing tool proposes a huge list of all the noun phrases to the indexer (*e.g.* Indexing online[7], Syntactica[8]). The indexer also has to identify the various forms under which a given descriptor is mentioned in the document and to select the descriptor occurrences that are worth referring to.

2.2 Navigation through a document

Various approaches have been developed to help readers to visualise large document bases (Byrd, 1999) but these methods are usually designed to handle IR results, *i.e.* rather large and potentially heterogeneous set of documents.

[1] In the following, the term *index* is always used with the same meaning.
[2] Collins 1998 dictionary définition.
[3] A web site can be considered as a special type of document and indexed in the same way as traditional printed books.
[4] It is based on a terminological analysis and includes the recognition of variant descriptors (Nazarenko & Aït El Mekki 2005).

[5] http://www.macrex.cix.co.uk/
[6] http://www.indexres.com
[7] http://www.indexingonline.com/index.php
[8] http://www.syntactica.com/login/login1.htm

Less attention has been paid to the problem of navigating through a single document, which requires a finer grained content description due to the relative document homogeneity. Some document and collection browsers rely on the list of the key phrases extracted from documents (Anick 01, Wacholder 01) but these works do not consider the document side of the index hypertextual network. (Gross & Assadi 97) presents a navigation system for a technical document but the method relies on a pre-existing ontology of the document domain. The indicative summaries (Saggion & Lapalme 02), which present the list of the keywords occurring in the most relevant phrases of the document, are close to traditional indexes but coarser-grained.

Independently of indexes, however, the segmentation and relevance-ranking problems are traditional ones in NLP and IR.

2.3 Segmentation approaches

Segmentation methods are usually based on the physical structure of the documents (typography, sectionning), on the lexical cohesion (Morris & Hirst 91; Hearst 97, Ferret *et al*. 98) and/or the linguistic markers expressing local continuity (Litman & Passonneau 95). The lexical cohesion approach gives interesting results on large and heterogeneous documents, but is less adapted to the segmentation of homogeneous documents. The structural and linguistic approaches are more relevant for our purposes. Our segmentation algorithm combines both methods (see Section 3).

However, traditional segmentation algorithms propose an absolute segmentation of documents, whereas, in indexes, the segmentation may vary from one entry to another. A whole set of paragraphs can be considered as a coherent Documentary Unit (UD) for a given entry and a smaller fragment be more relevant for another one.

2.4 Relevance measures

Ranking a set of documents is a well-known problem in IR. We adapted the traditional IR relevance tf.idf score (Salton 89) to rank the various paragraphs of a document instead of a set of documents.

The relevance problem is also addressed for document summarisation, to extract the more relevant sentences from the original document. The relevance score is based on the word weights, document structure and linguistic or typographical emphasis markers. Our relevance measure takes those parameters into account.

3 Method

For each descriptor, it is necessary to identify the relevant segments of the document that are worth referring to. This implies to detect its occurrences (not addressed here), identify the span of the segments to be referred to and to rank the results in relevance order.

3.1 Identifying reference segments

3.1.1 Segmentation cues

Our segmentation method relies on the presence of markers of integration of structural, linguistic and typographical kind. The algorithm takes the following cues into account:

The physical structure of texts (sectioning);

The presence of markers of linear integration (*if, then, secondly, on the other hand, thus, moreover, in addition...*) at the beginning of a paragraph; IndDoc relies on a core dictionary of generic markers, which can be tuned and extended for any specific corpus;

The presence of an anaphoric pronoun at the beginning of a paragraph: *this, this, these, it, its;*

The lexical cohesion of contiguous paragraphs, which is based on the recurrence of the index descriptors and their variant and thesaurus relations for a fine-grained segmentation as opposed to (Hearst 97);

The typographical homogeneity between contiguous paragraphs (two paragraphs in italics or several items of the same list, for instance).

3.1.2 Segmentation algorithm

Our algorithm (Figure 1) is made up of two phases, which correspond to an absolute segmentation in *documentary units* (DU) and a relative segmentation in *reference segments.*

The *absolute segmentation* phase only depends on the document. We start with a rough segmentation of the document in minimal DUs (MDU) (step 1). These MDUs are then widened in DUs (step 2) according to the linguistic and typographical markers and to the logical structure of the document (a DU cannot cross a section frontier for instance). At the end of this phase, the document is represented as a list of DUs.

The *relative segmentation* phase depends on a given descriptor. It comprises three more steps. The segments of reference are first identified (DUs which contain an occurrence of the descriptor or of one of its variants) (step 3). The segments that are contiguous in the text of the document are then merged (step 4), which results in a simplified list of segments. The segments belonging to a same section are finally generalised into in a single reference to the whole section (step 5), if a significant part of the section is represented in the list of the segments established in step 2.

3.2 Relevance ranking

Our relevance measure is based on the tf.idf score. We apply it to the paragraphs of a text rather than to the documents of a given collection. We also adapted the tf.idf score to take into account, in addition to the weight of a word in the whole document and its frequency in the segment, the weight of a particular occurrence (which can be typographically emphasised, for example).

Two scores are taken into account: the descriptor score (d-score(i) for the descriptor d_i) and the segment score (s-score(i,j) for the the j^{th} occurrence segment of d_i). A segment score is higher if it contains some important descriptors and a descriptor score is higher if it is mentioned in informative part of the document. We solve this traditional authority circularity problem by

	Monographs		Collections	
	LI	**AI**	**KE01**	**KE04**
Corpus size (# words occurrences)	42 260	111 371	185 382	122 229
Vocabulary size (without empty words)	3 018	9 429	38 962	32 334
Nomenclature size(# descriptors)	615	1 361	10 008	8 259
Corpus size (# paragraphs)	793	7 386	4 929	5 110

Table 1: Corpus profiles

distinguishing in the following an intrinsic segment weight and a relative segment score.

Let MDU be the list of MDUs.
Let Σ be the list of the all document sections and subsections.
Let D = {d_1,…,d_m} be the set of extracted descriptors.
Let DU be the list of DUs.
Begin
DU = MDU
// Document Units
For each du_i de DU
Widen ud_i to the next ud_{i+1} of DU
if there is no section frontier between ud_i and ud_{i+1}
and if there is a linguistic or typographical continuity between ud_i and ud_{i+1}.
// Plain segments
For each d_i descriptor of D:
Compute d_i^+, the class formed by d_i and its variant forms.
For each d_i^+ class of D^+:
Compute S_i^+, the list of the DUs in which the d_i^+ descriptors occur.
// Simplified segments
Compute SS_i^+ from S_i^+ by merging the contiguous segments.
//Generalised segments
For each σ_j of Σ
Identify the set e_{ij} of all segments of SS_i^+ belonging to σ_j.
if the proportion of occurrences of the d_i^+ descriptors *per* paragraph in the section σ_j is higher than a given threshold,
then the section σ_j as a whole is considered as a reference segment for d_j^+ and the e_{ij} paragraph sublist is substituted by σ_j in Σ.
else each paragraph of e_{ij} is considered as an individual reference segment for d_j^+.
End.
The **linguistic continuity** is marked by the presence of a marker listed in the dictionary of linear integration
The **typography continuity** is marked by italic, bold or list structure

Figure 1: The segmentation algorithm

3.2.1 Segment score

The *s-score(i,j)* is defined by the following formula:

$$s-score(i,j) = siw_j . \sum_{k=1}^{D} (\alpha . sdw(i,j))$$

where D is the total number of descriptors in the document and $\alpha = 1$ if d_k is d_i or one of its variants and 0,5 otherwise.

The score of the segment s_{ij}, *s-score(i,j)* is based on two elementary weights. (1) The *segment informational weight* (siw_j) is intrinsic to the segment s_j. It is high if s_j contains some typographical markers (bold, italics…) or new descriptors (first occurrence in s_j). It also depends on the status of the segment in the document: titles are more relevant segments than the summary or the conclusion. (2) The *segment discriminating weight* of the segment s_j relatively to the descriptor d_i (sdw_{ij}) depends on the number of occurrences of d_i in s_j and of its distribution over the document. ssw_{ij} is high if d_i has several occurrences in s_j and if it mainly occurs in s_j. This weight is a revised tf.idf measure:

$$sdw_{ij} = occ_{ij} . \log(p/p_i)$$

where occ_{ij} is the number of occurrences of d_i in s_j, P is the total number of paragraphs in the document and P_i is the number of paragraphs in which d_i occurs.

3.2.2 Descriptor score

The d-score(i) is defined by the following formula:

$$d-score(i) = dsw_i . \sqrt{ddw_i . diw_i . \sum_{j=1}^{p_i} s-score(i,j)/p_i}$$

The score of the descriptor d_i, *d-score(i)* is based on three elementary weights. (1) The *descriptor informational weight* (diw_i) depends on the typographical characteristics of individual occurrences of d_i and of the weights of the segments in which it occurs. diw_i is high if some occurrences of d_i are typographically emphasised or if d_i appears in special document parts (such as the titles, summary, introduction…). (2) The *descriptor discriminating weight* (ddw_i) depends on the normalised number of occurrences d_i and of its distribution over the document. dsw_i is high if d_i occurs more often than the other descriptors and if it is irregularly distributed. This weight is a revised tf.idf measure.

$$ddw_i = {occ_i}/{occ'} . \log(p/p_i)$$

where *occ'* is the mean number of occurrences per descriptor. (3) The *descriptor semantic weight* (dsw_i) depends on the number of descriptors to which d_i is linked in the semantic network of the index nomenclature.

Relevance is thus computed from a large set of cues. Besides frequency, typography, document structure, distribution and semantic network density are exploited.

4 Experiments and results

4.1 Corpora

Our first experiments are based on four different French corpora (Table 1): 2 handbooks in artificial intelligence (AI) and linguistics (LI) and 2 collections of scientific papers dealing with Knowledge Engineering (in the following: KE01 and KE04).

4.2 Segmentation

4.2.1 Example

The Figure 2 presents a segmentation example. The initial text is divided into 4 paragraphs (4 MDUs).

Unit types		Unit number			
		KE04	KE01	AI	LI
1	Min. Doc. Units	5110	4929	7386	793
2	Doc. Units	4272	4698	7245	634
3	Plain segments	14585	9863	8823	2569
4	Simplified segments	13876	9786	5157	1893
5	Generalised segments	13345	9728	4469	950
6	Paragraph occurrences	39089	18974	9897	3983

	Reduction factors			
	KE04	KE01	AI	LI
1->2	-20%	-10%	-0%	30%
3->4	-10%	-0%	-40%	-30%
4->5	-10%	-10%	-20%	-50%
5->6	-33%	-50%	-45%	-25%

Table 2: Segmentation results

Because of the presence of markers of linear integration (*Actually, Moreover*), the MDU corresponding to the paragraph §i is widened to cover *§i-§i+2*. The absolute segmentation thus gives 2 DUs : *§i-§i+2* and *§i+3*. For the relative segmentation, let us consider the descriptor "contexte d'insertion" (*insertion context*). The only occurrence of that descriptor in the whole document appears in paragraph *§i* (DU *§i-§i+2*). This single reference segment is finally generalised to the whole section because the segment of reference covers three of the four paragraphs of the section.

section k : Begin

$§_i$ *Le* ***contexte d'insertion*** *d'une ACCA a nécessairement des incidences ….*

$§_{i+1}$ **En effet** (Actually)*, pour atteindre …*

$§_{i+2}$ **De plus** (Moreover)*, même si dans notre cas le domaine est une variable libre, il faut qu'il ….*

$§_{i+3}$ *Ces différentes considérations nous ont conduit à proposer une activité,….*

section k : End

Figure 2: A segmentation example

4.2.2 Global segmentation behaviour

We applied the segmentation algorithm to our four corpora. The results are given in Table 2. The left part of the table describes the lists of textual units obtained at each step. The segmentation reduces the number of references for each corpus. The 6th line (size of the corpus in terms of paragraph number) is added for comparison: we consider the number of paragraphs as a basic segmentation reference. The comparison between the lines 5 and 6 shows that our segmentation algorithm actually reduces the number of references (from 25% to 50%) but we observe that:

The reduction factors (right part of the table) depend on the nature of the document (monograph *vs* collection) and of their style;

The simplification of segments (line 3->4) has a stronger effect on monographs due to lexical homogeneity;

For the KE corpora, which are rather heterogeneous, the first step (line 1->2) is the more important.

There are proportionally more integration markers in LI than in AI.

The segment generalisation has a stronger impact on LI, which is more strictly structured in sections and subsections.

The diversity of the segmentation cues makes our segmentation algorithm robust to various types of documents.

4.3 Relevance ranking

Our relevance ranking algorithm behaves as expected on our experimental corpora.

4.3.1 Example

Let us consider the descriptor "contrainte temporelle" (*temporal constraint*). The 12 initial occurrences of this descriptor in LI corpus are grouped into 3 reference segments during the segmentation phase:

S1 contains the first occurrence of the descriptor which is written in bold and which is a definition but it is a small segment.

S2 is composed of three subsections. "contrainte temporelle" occurs in the title of the first one and is mentioned in the two others. The descriptors "concordance des temps" (*sequence of tenses*) and "relation temporelle" (*temporal relation*") which are semantically close[9] to "contrainte temporelle" occur in the titles of the second and third subsections.

The descriptor appears at the beginning of the third segment but S3 itself belongs to a conclusion.

The ranking gives the references in the following order: S2, S1 and S3. S2 is given first because it is the most informative and it contains a title occurrence of the descriptor. Even if S1 contains the first occurrence of the descriptor and if it is typographically emphasized, it is considered as less informative. The segment S3 is last because it is a conclusion part.

It is interesting to consider the "contrainte temporelle" entry in the index of the published LI book. The published index gives exactly the same segments (along with an empty and probably erroneous reference), in textual order, which is less informative.

4.3.2 Segment ranking evaluation

To evaluate our segment ranking measure, we have selected a sample of 30 descriptors that have numerous reference segments among the 110 descriptors of the original published LI index. For each descriptor, the author of the book was asked to analyse the quality of the segment ranking.

[9] These semantic relations are computed during the terminological analysis that is note presented here (Nazarenko & Aït El Mekki 05).

The results are given in Table 3. We distinguished the descriptors whose segment list is correctly ranked (group 1), those for which the ranking is only partially correct (but the top list is good, group 2), those whose ranking is globally incorrect (group 3) and the undecidable cases (group 4).
Table 3 shows that the top of the segment lists are correct in 77% of the cases and that the ranking algorithm fails in less than 15% of the cases. A detailed analysis shows that defining occurrences tend to get high-ranking scores: for polysemous descriptors (such as *origine* (Engl. *origin*)), the technical occurrences are better ranked than the common sense ones (*à l'origine de/to begin with*).

Correct ranking : 17%		Incorrect ranking: 23%	
Group 1	Group 2	Goup 3	Group 4
17	6	4	3

Table 3: Segment ranking for 30 descriptors

4.3.3 Descriptor ranking evaluation

The ranking of the descriptors does not have direct impact on navigation functionalities but the ranking of segments and descriptors are interdependent.

For evaluation purposes, an independent indexer was asked to choose the most relevant descriptors in the flat list of 615 LI descriptors. She decided to keep 203 descriptors. If we consider the ranking of those 203 relevant descriptors, we observe that the mean rank is 126,5, which is much higher than the 307,5 median rank. The precision at the 203rd position in the ranking is 83%. For the KE04 experiment, only the 1500 top ranked descriptors have been validated and the precision rate is 70%. For test purposes, 500 descriptors with low scores have been artificially added. All but one of these "bad" descriptors were actually eliminated (less than 0.01% of precision).

Those figures confirm the rather good performance of our knowledge-poor ranking algorithm.

5 Conclusion

We propose a knowledge poor method to automatically build the hypertextual network that helps the navigation through the document. The resulting device is similar to a back-of-the-book index. We show that, given a document and a list of descriptors, it is possible to automatically compute a network of reference links that connect the list of descriptors to the text of the document. Two interrelated problems must be solved: What are the spans of text that are worth referring to for each descriptor? What are the most relevant pieces of information (descriptors and references) for navigation? We adapted the traditional techniques developed for text segmentation and document ranking. The originality of our method is the large variety of cues that are taken into account: typography, document logical structure, linguistic markers of linear integration, lexical cohesion, etc. The impact of each type of cue depends on the document style but the combination of all make our segmentation and ranking algorithm more robust.

References

(Anick 01) P. Anick, The automatic construction of faceted terminological feedback for interactive document retrieval, In D. Bourigault *et al.* (ed.) *Recent Advances in Computational Terminology*, John Benjamins, Amsterdam, 2001.

(Byrd 99) D. Byrd, A Scrollbar-based Visualization for Document Navigation. Proc. of Digital Libraries 99 Conf., ACM, New York, 1999.

(Ferret *et al.* 98) O. Ferret, B. Grau, N. Masson, Thematic segmentation of texts: two methods for two kinds of texts. Proc. of COLING-ACL Conf., Montreal, 392–396, 1998.

(Gross *et al.* 96) C. Gross, H. Assadi, N. Aussenac, A. Courcelle, Task Models for Technical Documentation Accessing. Proc. of EKAW Conf, 1996.

(Hearst 97) M. Hearst, TextTiling: Segmenting Text into Multi-Paragraph Subtopic Passages, *Computational Linguistics* , 23 (1), 33-64, March 1997

(Jacquemin *et al.* 97) C Jacquemin, J.L. Klavans, E. Tzoukermann, Expansion of multiword terms for indexing and retrieval using morphology and syntax. Proc. of the COLING/EACL Conf, 24-31, Madrid, 1997.

(Litman & Passonneau 95) D.J. Litman, R. Passonneau, Combining Multiple Knowledge Sources for Discourse Segmentation, Proc. of the ACL Conf., 1995.

(Mandar *et al.* 1997) M. Mandar, C. Buckley, A. Singhal, C. Cardie, An analysis of statistical and syntactic phrases. Proc. of the Intelligent Multimedia Information Retrieval Systems and Management Conf. (RIAO'97), Montreal, 200-214, 1997.

(Morris & Hirst 91) J. Morris, G. Hirst, Lexical Cohesion Computed by Thesaural Relations as an Indicator of the Structure of Text, *Computational Linguistic*, 17 (1), 21-48, 1991.

(Nazarenko & Aït El Melli 05) A. Nazarenko and T. Aït El Mekki, Building back-of-the-book indexes, *Terminology*, vol. 11(1), 199-224, 2005.

(Saggion & Lapalme 02) H. Saggion, G. Lapalme, Generating Indicative-Informative Summaries with SumUM. *Computational Linguistics*, vol. 28, 2002.

(Salton 89) G. Salton, *Automatic text processing, the transformation, analysis, and retrieval of information by computer*, Addison-Wesley, Reading, 1989.

(Washolder 01) N. Wacholder, The Intell-Index System: Using NLP Techniques to organize a dynamic text browser, *Proc. of the Technology of Browsing Applications Work. 2001.*

Using the Essence of Texts to Improve Document Classification

Rada Mihalcea and **Samer Hassan**
Department of Computer Science
University of North Texas
rada@cs.unt.edu, samer@unt.edu

Abstract

This paper explores the possible benefits of the interaction between automatic extractive summarization and text classification. Through experiments performed on standard test collections, we show that techniques for extractive summarization can be effectively combined with classification methods, resulting in improved performance in a text categorization task. Moreover, comparative results suggest that the synergy between text summarization and text classification can be regarded as a new application-oriented evaluation testbed for automatic summarization.

1 Introduction

Text categorization is a problem typically formulated as a learning task, where a classifier learns how to distinguish between categories in a given set, using features automatically extracted from a collection of training documents. In addition to the learning methodology itself, the accuracy of the text classifier also depends to a large extent upon the classification granularity, and on how well separated are the training or test documents belonging to different categories. For instance, it may be a relatively easy task to learn how to classify documents in two distinct categories such as *computer science* and *music*, but it may be significantly more difficult to distinguish between documents pertaining to more closely related topics such as *operating systems* and *compilers*.

Intuitively, if the gap between categories could be increased, the classification performance would raise accordingly, since the learning task would be simplified by removing features that represent potential overlap between categories. This is in fact the effect achieved through feature weighting and selection (Yang & Pedersen 97), (Ng *et al.* 97), which was found to improve significantly over the case where no weighting or selection is performed. We propose a new approach for reducing the potential overlap between documents belonging to different categories, by using a method that extracts the **essence** of a text prior to classification. In this way, only the important sections of a document participate in the learning process, and thus the performance of the text classification algorithm could be improved.

In this paper, we present a graph-based method for automatic summarization through sentence extraction, and show how it can be successfully integrated with a text classifier. Through experiments on standard test collections, we show that significant improvements can be achieved on a text categorization task by classifying extractive summaries, rather than entire documents. We believe that these results have implications not only on the problem of text classification, where the method proposed provides the means to improve the categorization accuracy with error reductions of up to 19.3%, but also on the problem of text summarization, by suggesting a new application-based evaluation of tools for automatic summarization.

2 Graph-based Algorithms for Sentence Extraction

Algorithms for extractive summarization are typically based on techniques for sentence extraction, and attempt to identify the set of sentences that are most important for the understanding of a given document. Some of the most successful approaches to extractive summarization consist of supervised algorithms that attempt to learn what makes a good summary by training on collections of summaries built for a relatively large number of training documents, e.g. (Hirao *et al.* 02), (Teufel & Moens 97). However, the price paid for the high performance of such supervised algorithms is their inability to easily adapt to new languages or domains, as new training data are required for each new type of data.

In this section, we shortly overview an efficient unsupervised extractive summarization method using graph-based ranking algorithms, as proposed in our previous work (Mihalcea & Tarau 04). Iterative graph-based ranking algorithms, such as Kleinberg's HITS algorithm (Kleinberg 99) or Google's PageRank (Brin & Page 98), have been traditionally and successfully used in Web-link analysis, social networks, and more recently in text processing applications (Mihalcea *et al.* 04), (Mihalcea & Tarau 04). In short, a graph-based ranking algorithm is a way of deciding on

the importance of a vertex within a graph, by taking into account global information recursively computed from the entire graph, rather than relying only on local vertex-specific information. The basic idea implemented by the ranking model is that of "voting" or "recommendation". When one vertex links to another one, it is basically casting a vote for that other vertex. The higher the number of votes that are cast for a vertex, the higher the importance of the vertex.

These graph ranking algorithms are based on a random walk model, where a walker takes random steps on the graph, with the walk being modeled as a Markov process – that is, the decision on what edge to follow is solely based on the vertex where the walker is currently located. Under certain conditions, this model converges to a stationary distribution of probabilities associated with vertices in the graph, representing the probability of finding the walker at a certain vertex in the graph. Based on the Ergodic theorem for Markov chains (Grimmett & Stirzaker 89), the algorithms are guaranteed to converge if the graph is both aperiodic and irreducible. The first condition is achieved for any graph that is a non-bipartite graph, while the second condition holds for any strongly connected graph. Both these conditions are achieved in the graphs constructed for the sentence extraction application considered in this paper.

Let $G = (V, E)$ be a directed graph with the set of vertices V and set of edges E, where E is a subset of $V \times V$. For a given vertex V_i, let $In(V_i)$ be the set of vertices that point to it (predecessors), and let $Out(V_i)$ be the set of vertices that vertex V_i points to (successors).

PageRank (Brin & Page 98) is perhaps one of the most popular ranking algorithms, and was designed as a method for Web link analysis.

$$PR(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} \frac{PR(V_j)}{|Out(V_j)|} \quad (1)$$

where d is a parameter set between 0 and 1.

HITS (Hyperlinked Induced Topic Search) (Kleinberg 99) makes a distinction between "authorities" (pages with a large number of incoming links) and "hubs" (pages with a large number of outgoing links). For each vertex, *HITS* produces two sets of scores.

$$HITS_A(V_i) = \sum_{V_j \in In(V_i)} HITS_H(V_j) \quad (2)$$

$$HITS_H(V_i) = \sum_{V_j \in Out(V_i)} HITS_A(V_j) \quad (3)$$

For each of these algorithms, starting from arbitrary values assigned to each node in the graph, the computation iterates until convergence below a given threshold is achieved. After running the algorithm, a score is associated with each vertex, which represents the "importance" or "power" of that vertex within the graph. Note that the final values are not affected by the choice of the initial value, only the number of iterations to convergence may be different. Somewhat faster or more compact (but significantly more complex) implementations have been recently suggested (Kamvar *et al.* 03; Raghavan & Garcia-Molina 03), but the additional time or memory savings are more relevant in cases like the complete Web graph than it would be in the case of moderately large graphs like a text-based graph.

When the graphs are built starting with natural language texts, it may be useful to integrate into the graph model the *strength* of the connection between two vertices V_i and V_j, indicated as a weight w_{ij} added to the corresponding edge. Consequently, the ranking algorithm is adapted to include edge weights, e.g. for *PageRank* the score is determined using the following formula (a similar change can be applied to the *HITS* algorithm):

$$PR^W(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} w_{ji} \frac{PR^W(V_j)}{\sum_{V_k \in Out(V_j)} w_{kj}} \quad (4)$$

While the final vertex scores (and therefore rankings) for weighted graphs differ significantly as compared to their unweighted alternatives, the number of iterations to convergence and the shape of the convergence curves is almost identical for weighted and unweighted graphs.

For the task of sentence extraction, the goal is to rank entire sentences, and therefore a vertex is added to the graph for each sentence in the text. To establish connections (edges) between sentences, we are defining a similarity relation, where "similarity" is measured as a function of content overlap. Such a relation between two sentences can be seen as a process of "recommendation": a sentence that addresses certain concepts in a text, gives the reader a "recommendation" to refer to other sentences in the text that address the same concepts, and therefore a link can be drawn between any two such sentences that share common content.

The overlap of two sentences can be determined simply as the number of common tokens between the lexical representations of the two sentences, after removing the stopwords. Moreover, to avoid promoting long sentences, we are using a normalization factor, and divide the content overlap of two sentences with the length of each sentence.

[1] Watching the new movie, "Imagine: John Lennon," was very painful for the late Beatle's wife, Yoko Ono.
[2] "The only reason why I did watch it to the end is because I'm responsible for it, even though somebody else made it," she said.
[3] Cassettes, film footage and other elements of the acclaimed movie were collected by Ono.
[4] She also took cassettes of interviews by Lennon, which were edited in such a way that he narrates the picture.
[5] Andrew Solt ("This Is Elvis") directed, Solt and David L. Wolper produced and Solt and Sam Egan wrote it.
[6] "I think this is really the definitive documentary of John Lennon's life," Ono said in an interview.

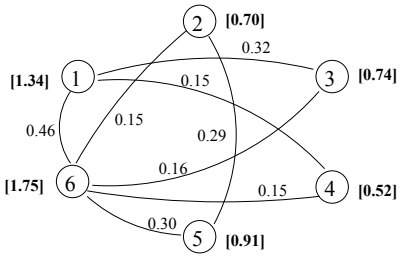


Figure 1: Graph of sentence similarities built on a sample text. Scores reflecting sentence importance, obtained with the graph-based ranking algorithm, are shown in brackets next to each sentence.

The resulting graph is highly connected, with a weight associated with each edge, indicating the strength of the connections between various sentence pairs in the text. The graph can be represented as: (a) simple *undirected* graph; (b) directed weighted graph with the orientation of edges set from a sentence to sentences that follow in the text (*directed forward*); or (c) directed weighted graph with the orientation of edges set from a sentence to previous sentences in the text (*directed backward*).

After the ranking algorithm is run on the graph, sentences are sorted in descending order of their score, and the top ranked sentences are selected for inclusion in the extractive summary. Figure 1 shows an example of a graph built for a sample text of six sentences.

Evaluation. The performance of the sentence extraction method was evaluated in the context of a single-document summarization task, using 567 news articles provided during the Document Understanding Evaluations 2002 (DUC 02). For each article, the method was used to generate a 100-words summary, which is the task undertaken by other systems participating in this single-document summarization task. The evaluation was run using the ROUGE toolkit, which is an evaluation method based on Ngram statistics, found to be highly correlated with human evaluations (Lin & Hovy 03). For each document, two manually produced reference summaries were provided, and used in the evaluation process. Table 1 shows results using the *HITS* and *PageRank* algorithms on graphs that are: (a) undirected, (b) directed forward, or (c) directed backward.

Algorithm	Graph		
	Undirected	Forward	Backward
$HITS_A^W$	0.4912	0.4584	**0.5023**
$HITS_H^W$	0.4912	**0.5023**	0.4584
PageRank	0.4904	0.4202	**0.5008**

Table 1: Results for extractive summarization using graph-based sentence extraction. Graph-based ranking algorithms: *HITS* and *PageRank*. Graphs: undirected, directed forward, directed backward.

By ways of comparison, a competitive baseline proposed by the DUC evaluators – consisting of a 100-word summary constructed by taking the first sentences in each article – achieves a ROUGE score of 0.4799. The best performing system in DUC 2002 was a *supervised* system which achieved a score of 0.5011.

The results are encouraging: the sentence extraction method applied on a *directed backward* graph structure exceeds the performance achieved through a simple (but competitive) baseline, and competes with the best performing systems from DUC 2002. Unlike other supervised systems, which attempt to learn what makes a good summary by training on collections of summaries built for other articles, the graph-based method is fully unsupervised, and relies only on the given text to derive an extractive summary. Moreover, due to its unsupervised nature, the algorithm can be easily adapted to other languages, genres, or domains.

3 Text Categorization Using Extractive Summarization

Provided a set of training documents, each document assigned with one or more categories, the task of text categorization consist of finding the most probable category for a new unseen document, based on features extracted from training examples. The classification process is typically performed using information drawn from entire documents, and this may sometime result in noisy features. To lessen this effect, we propose to feed the text classifier with summaries rather than entire texts, with the goal of removing the less-important, noisy sections of a document prior to classification.

The text classification process is thus modified to integrate an extractive summarization tool

that determines the top N most important sentences in each document. Starting with a collection of texts, every document is replaced by its summary, followed by the application of a regular text categorization algorithm that determines the most likely category for a given test document. Figure 2 illustrates the classification process based on extractive summarization.

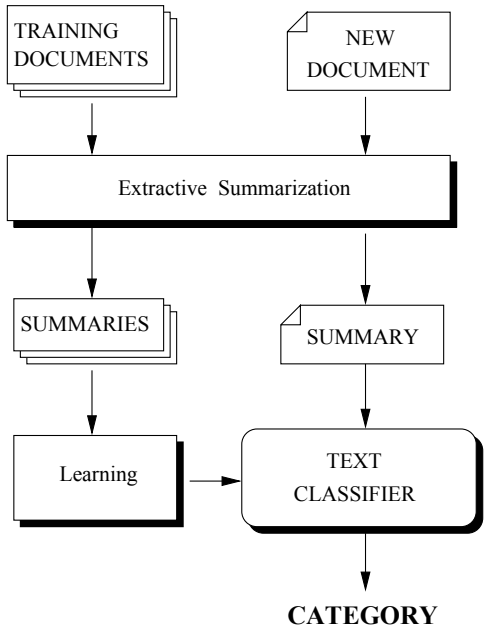


Figure 2: Text classification using extractive summarization.

3.1 Extractive Summarization

To summarize documents, we use the $HITS_A$ *directed backward* graph-based sentence extraction algorithm described in Section 2, which generates extractive summaries by finding the most important sentences in the text. The choice of the algorithm is motivated by several reasons. First, the decision to use an extractive summarization tool, versus more complex systems that include sentence compression and text generation, is based on the fact that in our experiments text summarization is not an end per se, but rather an intermediate step for document classification. The informativeness of a summary is thus more important than its coherence, and summarization through sentence extraction is sufficient for this purpose. Second, through evaluations conducted on standard data sets, the algorithm was found to work best among other graph-based algorithms for sentence extraction, and was demonstrated to be competitive with the state-of-the-art in text summarization. Finally, it is an algorithm that can produce a ranking over sentences in a text, and thus it is well suited for our experiments where we want to measure the impact on text classification of summaries of various lengths.

3.2 Algorithms for Text Classification

There is a large body of algorithms previously tested on text classification problems, due also to the fact that text categorization is one of the testbeds of choice for machine learning algorithms. In the experiments reported here, we compare results obtained with two frequently used text classifiers – Rocchio and Naïve Bayes, selected for the diversity of their learning methodologies.

Rocchio. This is an adaptation of the relevance feedback method developed in information retrieval (Rocchio 71). It uses standard $TFIDF$ weighted vectors to represent documents, and builds a prototype vector for each category by summing up the vectors of the training documents in each category. Test documents are then assigned to the category that has the closest prototype vector, based on a cosine similarity. Text classification experiments with different versions of the Rocchio algorithm showed competitive results on standard benchmarks (Joachims 97), (Moschitti 03).

Naïve Bayes. The basic idea in a Naïve Bayes text classifier is to estimate the probability of a category given a document using joint probabilities of words and documents. Naïve Bayes assumes word independence, which means that the conditional probability of a word given a category is assumed to be independent of the conditional probability of other words given the same category. Despite this simplification, Naïve Bayes classifiers perform surprisingly well on text classification (Joachims 97), (Schneider 04). While there are several versions of Naïve Bayes classifiers (variations of multinomial and multivariate Bernoulli), we use the multinomial model (McCallum & Nigam 98), which was shown to be more effective.

3.3 Data

For the classification experiments, we use the *Reuters-21578*[1] and *WebKB*[2] data sets – two of the most widely used test collections for text classification. For *Reuters-21578*, we use the standard *ModApte* data split (Apte *et al.* 94), obtained by eliminating unlabeled documents, and selecting only categories that have at least one document in the training set and test set (Yang & Liu 99). For *WebKB*, we perform a four-fold

[1]Publicly available at http://www.daviddlewis.com/ /resources/testcollections/reuters21578/

[2]Publicly available at http://www-2.cs.cmu.edu/afs/ /cs.cmu.edu/project/theo-20/www/data/

cross validation after removing the *other* category, using the 3 : 1 training/test split automatically created with the scripts provided with the collection, which separates the data into training on three of the universities plus a miscellaneous collection, and testing on a fourth held-out university. Both collections are further post-processed by removing all documents with less than 10 sentences, resulting into final data sets of 1277 training documents, 436 test documents, 60 categories for *Reuters-21578*, and 350 training documents, 101 test documents, 6 categories for *WebKB*. This last step is motivated by the goal of our experiments: we want to determine the impact of various degrees of summarization on text categorization, and this is not possible with very short documents.

3.4 Evaluation Metrics

The evaluation is run in terms of accuracy, defined as the number of correct assignments among the document–category pairs in the test set. For the *WebKB* data set, since the classifiers assign exactly one category to each document, and a document can belong to only one category, this definition of accuracy coincides with the measures of precision, recall, and F-measure. For the *Reuters-21578* data set, multiple classifications are possible for each document, and thus the accuracy coincides with the classification precision. Evaluations figures are reported as *micro-average* over all categories in the test set.

4 Experimental Results

Classification experiments are run using each of the two learning algorithms, with documents in the collection represented by extracts of various lengths. The classification performance is measured for each experiment, and compared against a traditional classifier that performs text categorization using the original full-length documents.

Table 2 shows the classification results obtained using the *Reuters-21578* and the *WebKB* test collections. Note that these results refer to data subsets somewhat more difficult than the original test collections, and thus they are not directly comparable to results previously reported in the literature. For instance, the average number of documents per category in the *Reuters-21578* subset used in our experiments is only 25, compared to the average of 50 documents per category available in the original data set[3]. Similarly, the *WebKB* subset has an average number of 75 documents per category, compared to 750 in the original data set.

Figures 3 and 4 plot the classification accuracy on the two data sets for each learning algorithm, using extractive summaries of various lengths generated with the graph-based sentence extraction method. For a comparative evaluation, the figure plots results obtained with methods that create an extract by: (a) selecting the N most important sentences using the graph-based sentence extraction algorithm; (b) selecting the first N sentences in the text; (c) randomly selecting N sentences; (d) selecting the N *least* important sentences using the low-end of the ranking obtained with the same graph-based sentence extraction algorithm; (e) selecting the last N sentences in the text.

From a **text categorization** perspective, the results de-monstrate that techniques for text summarization can be effectively combined with text classification methods to the end of improving **categorization performance.** On the *Reuters-21578* data set, the classification performance using the Rocchio classifier improves from an accuracy of 74.90% to 77.00% obtained for summaries of 6 sentences. Similar improvements are observed for the Naïve Bayes classifier, where the highest accuracy of 78.38% is obtained again for summaries of 6 sentences, and is significantly better than the accuracy of 75.01% for text classification with full-length documents. The impact of summarization is even more visible on the *WebKB* data set, where summaries of 5 sentences result in a classification accuracy of 79.24% using a Naïve Bayes classifier, significantly higher than the accuracy of 74.25% obtained when the classification is performed with entire documents. Similarly, the categorization accuracy using a Rocchio classifier on the same data set improves from 63.36% for full-length documents to 66.35% for summaries of 5 sentences[4]. The highest error rate reduction observed during these experiments was 19.3%, achieved with a Naïve Bayes classifier applied on 5-sentence extractive summaries.

Another aspect worth noting is the fact that these results can have important implications on the problem of text classification for documents for which only abstracts are available. As it was previously suggested (Hulth 03), many documents on the Internet are not available as full-texts, but only as abstracts. Similarly, documents that are typically stored in printed form, such as books, journals, or magazines, may have an abstract available in electronic format, but not the entire

[3]Using the same Naïve Bayes and Rocchio classifier implementations on the entire *Reuters-21578* data set results in a classification accuracy of 77.42% for Naïve Bayes and 79.70% for Rocchio, figures that are closely comparable to results previously reported on this data set.

[4]The improvements observed in all classification settings are statistically significant at $p < 0.05$ level (paired t-test).

Summary length	*Reuters-21578*		*WebKB*	
	Rocchio	Naïve Bayes	Rocchio	Naïve Bayes
1 sentence	70.18%	72.94%	60.40%	61.39%
2 sentences	72.48%	73.85%	62.38%	71.29%
3 sentences	73.17%	75.46%	60.40%	74.26%
4 sentences	72.94%	75.46%	63.37%	76.24%*
5 sentences	75.23%	75.92%	66.35%*	79.24%*
6 sentences	77.00%*	78.38%*	65.37%*	77.24%*
7 sentences	76.38%*	77.61%*	65.34%*	76.24%*
8 sentences	76.38%*	77.38%*	64.36%	76.21%*
9 sentences	75.23%	77.61%*	65.34%*	75.25%
10 sentences	75.23%	77.52%*	65.35%*	75.25%
full document	74.91%	75.01%	63.36%	74.25%

Table 2: Classification results for *Reuters-21578* and *WebKB*, using Rocchio and Naïve Bayes classifiers, for full-length documents and for extractive summaries of various lengths. Statistically significant improvements ($p < 0.05$, paired t-test) with respect to full-document classification are also indicated (*).

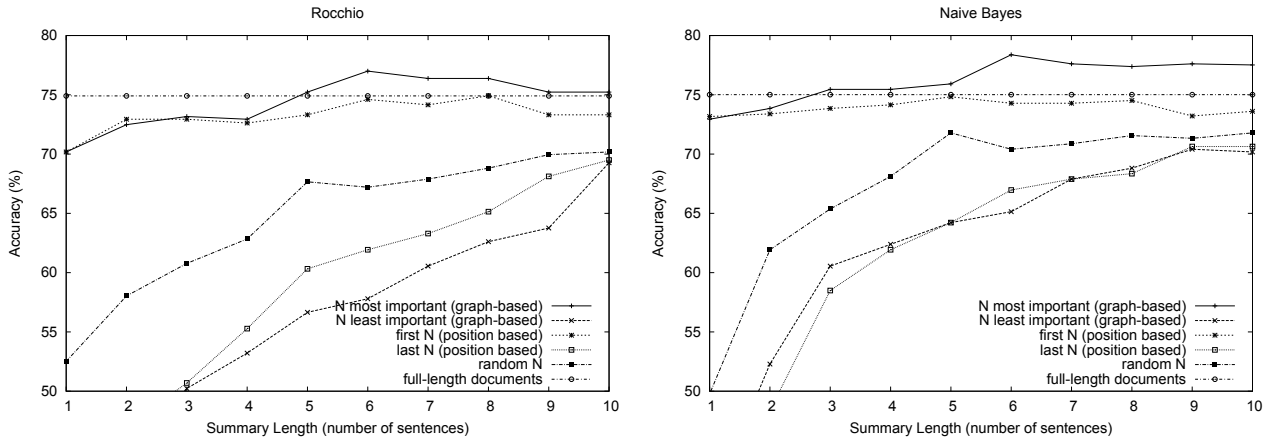


Figure 3: Classification accuracy for *Reuters-21578* for extractive summaries of various lengths.

text. This means that an eventual classification of such documents has to rely on **abstract categorization**, rather than traditional full-text categorization. The results obtained in the experiments reported here suggest that the task of text classification can be efficiently performed even when only abstracts are available for the documents to be classified.

Classification efficiency is also significantly improved when the classification is based on summaries, rather than full-length documents. A clear computational improvement was observed even for the relatively small test collections used in our experiments. For instance, the Naïve Bayes classifier takes 22 seconds to categorize the full-length documents in the *Reuters-21578* subset on a Pentium IV 3GHz 2GB computer, compared to only 6 seconds when the classification is done using 5-sentence summaries[5]

The results have also important implications on the problem of **text summarization.** As seen in Figures 3 and 4, regardless of the classifier, a performance peak is observed for summaries of 5-7 sentences, which give the highest increase in classification accuracy. This result can have an interesting interpretation in the context of text summarization, as it indicates the optimal number of sentences required for "grasping" the content of a text. From this perspective, the task of text classification can be regarded as an **objective** way of defining the "informativeness" of an abstract, and

[5]One could argue that the summarization-based classification implies an additional summarization overhead. Note however that the summarization process can be parallelized and needs to be performed only once offline. The resulting summaries can be then used in multiple classification tasks.

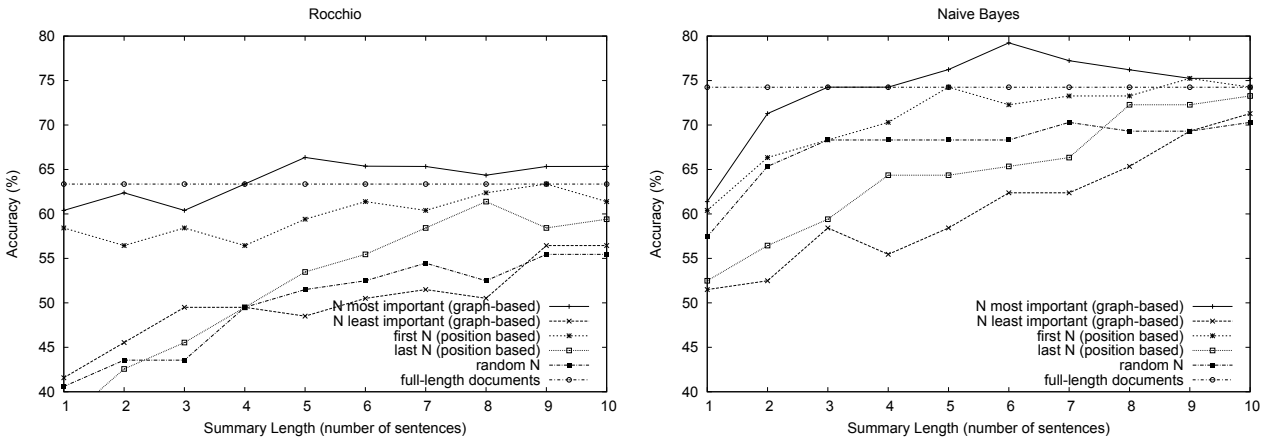


Figure 4: Classification accuracy for *WebKB* for extractive summaries of various lengths.

could be used as an alternative to more subjective human-assessed evaluations of summary content.

The comparative plots from Figures 3 and 4 also reveal another interesting aspect of the synergy between text summarization and document classification. Different automatic summarization tools – graph-based extractive summarization, heuristics that extract sentences based on their position in the text, or a simple random sentence selection baseline – have different but **consistent** impact on the quality of a text classifier, which suggests that text categorization can be used as an application-oriented evaluation testbed for automatic summarization. The graph-based summarization method gives better text classification accuracy as compared to the position-based heuristic, which in turns performs better than the simple sentence selection baselines. This comparative evaluation correlates with rankings of these methods previously reported in the literature (Erkan & Radev 04), which were based on human judgment or other automatic evaluation metrics such as ROUGE (Lin & Hovy 03).

5 Related Work

To our knowledge, the impact of content-based text summarization on the task of text categorization was not explored in previous studies. Sentence importance was considered as an additional factor for feature weighting in work reported in (Ko *et al.* 02), where words in a text were weighted differently based on the score associated with the sentence they belong to. Experiments with four different text categorization methods have shown that a weighting scheme based on sentence importance can significantly improve the classification performance. In (Kolcz *et al.* 01), text summarization is regarded as a feature selection method, and is shown to improve over alternative algorithms for feature selections. Finally, another related study is the polarity analysis through subjective summarization reported in (Pang & Lee 04), where the main goal was to distinguish between positive and negative movie reviews by first selecting those sentences likely to be more subjective according to a min-cut algorithm. The focus of their study was the analysis of text style (subjective versus objective), rather than classification of text content. We consider instead the more general text classification problem, combined with a typical text summarization task, and evaluate the role that text summaries can play in document categorization.

6 Conclusions

In this paper, we investigated the interaction between document summarization and text categorization, through comparative classification experiments relying on full-length documents or summaries of various lengths. First, we showed how graph-based algorithms can be successfully applied to the task of extractive summarization, resulting in state-of-the-art results as measured on standard data sets. Next, we showed that techniques for automatic summarization can be used to improve the performance of a text categorization task, with error rate reductions of up to 19.3% obtained with a Naïve Bayes or a Rocchio classifier when applied on short extractive summaries rather than full-length documents. Finally, we suggested that the interaction between text summarization and document categorization can be regarded as an application-oriented evaluation testbed for tools for automatic summarization, as summaries produced by different summarization tools were shown to have different impact on the performance of a text classifier.

References

(Apte *et al.* 94) C. Apte, F. Damerau, and S. M. Weiss. Towards language independent automated learning of text categorisation models. In *Proceedings of the 17th ACM SIGIR Conference on Research and Development in Information Retrieval*, 1994.

(Brin & Page 98) S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7), 1998.

(DUC 02) DUC. Document understanding conference 2002, 2002. http://www-nlpir.nist.gov/projects/duc/.

(Erkan & Radev 04) G. Erkan and D. Radev. Lexpagerank: Prestige in multi-document text summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain, July 2004.

(Grimmett & Stirzaker 89) G. Grimmett and D. Stirzaker. *Probability and Random Processes*. Oxford University Press, 1989.

(Hirao *et al.* 02) T. Hirao, Y. Sasaki, H. Isozaki, and E. Maeda. Ntt's text summarization system for duc-2002. In *Proceedings of the Document Understanding Conference 2002*, 2002.

(Hulth 03) A. Hulth. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, Japan, August 2003.

(Joachims 97) T. Joachims. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In *Proceedings of ICML-97, 14th International Conference on Machine Learning*, Nashville, US, 1997.

(Kamvar *et al.* 03) S. Kamvar, T. Haveliwala, C. Manning, and G. Golub. Extrapolation methods for accelerating PageRank computations. In *Proceedings of the 12th International World Wide Web Conference*, 2003.

(Kleinberg 99) J.M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

(Ko *et al.* 02) J. Ko, J. Park, and J. Seo. Automatic text categorization using the importance of sentences. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002)*, Taipei, Taiwan, August 2002.

(Kolcz *et al.* 01) A. Kolcz, V. Prabakarmurthi, and J. Kalita. Summarization as feature selection for text categorization. In *Proceedings of the 10th International conference on Information and knowledge management*, Atlanta, Georgia, 2001.

(Lin & Hovy 03) C.Y. Lin and E.H. Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of Human Language Technology Conference (HLT-NAACL 2003)*, Edmonton, Canada, May 2003.

(McCallum & Nigam 98) A. McCallum and K. Nigam. A comparison of event models for Naive Bayes text classification. In *Proceedings of AAAI-98 Workshop on Learning for Text Categorization*, 1998.

(Mihalcea & Tarau 04) R. Mihalcea and P. Tarau. TextRank – bringing order into texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, Barcelona, Spain, 2004.

(Mihalcea *et al.* 04) R. Mihalcea, P. Tarau, and E. Figa. PageRank on semantic networks, with application to word sense disambiguation. In *Proceedings of the 20st International Conference on Computational Linguistics (COLING 2004)*, Geneva, Switzerland, 2004.

(Moschitti 03) A. Moschitti. A study on optimal paramter tuning for Rocchio text classifier. In *Proceedings of the European Conference on Information Retrieval*, Pisa, Italy, 2003.

(Ng *et al.* 97) H.T. Ng, W.B. Goh, and K.L. Low. Feature selection, perceptron learning, and a usability case study for text categorization. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Philadelphia, PA, July 1997.

(Pang & Lee 04) B. Pang and L. Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics*, Barcelona, Spain, July 2004.

(Raghavan & Garcia-Molina 03) S. Raghavan and H. Garcia-Molina. Representing Web graphs. In *Proceedings of the IEEE International Conference on Data Engineering*, March 2003.

(Rocchio 71) J. Rocchio. *Relevance feedback in information retrieval*. Prentice Hall, Ing. Englewood Cliffs, New Jersey, 1971.

(Schneider 04) K. Schneider. A new feature selection score for multinomial naive bayes text classification based on kl-divergence. In *The Companion Volume to the Proceedings of 42st Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain, July 2004.

(Teufel & Moens 97) S. Teufel and M. Moens. Sentence extraction as a classification task. In *ACL/EACL workshop on "Intelligent and scalable Text summarization"*, pages 58–65, Madrid, Spain, 1997.

(Yang & Liu 99) Y. Yang and X. Liu. A reexamination of text categorization methods. In *Proceedings of the 22nd ACM SIGIR Conference on Research and Development in Information Retrieval*, 1999.

(Yang & Pedersen 97) Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, Nashville, US, 1997.

Automatic Import of Verbal Syntactic Relations Using Parallel Corpora

Verginica Barbu Mititelu and **Radu Ion**
Romanian Academy Research Institute for Artificial Intelligence
13 Septembrie, 13, Bucharest 050711, Romania
{vergi,radu}@racai.ro

Abstract

The data that we present in this paper are only some preliminary results of an experiment that aims at testing whether it is possible (and also to what degree) to automatically transfer syntactic relations contracted by verbs (as they are lexicalized in a corpus) from a resource-rich language into another language with fewer resources, using parallel corpora.

1. Introduction

Artificial Intelligence applications make great use of linguistic resources. As the development of such resources is time- and money-consuming, lately, the AI community has started using alternative strategies for getting the necessary resources. One such strategy is the use of knowledge in one language to help solving tasks in another language. One example of knowledge transfer is to take advantage of the resources built for one language to induce knowledge in a resource-poor language. This is made possible by the existence of aligned parallel corpora.

What we present below are some preliminary results of an experiment in which we test the possibility of automatic transfer of syntactic relations from a resource-rich language (English) into a resource-poor one (Romanian).

2. Assumption

We started from the Direct Correspondence Assumption (DCA) (Hwa et al. 2002b) that applies to parallel treebanks. However, we have modified it as follows, so that it serves our purpose:

> Given a pair of sentences E and F, that are (literal) translations of each other, if words x_E and y_E of E are aligned with words x_F and y_F of F, respectively, and if syntactic relationship $R_{(xE \text{ and } yE)}$ holds in E, then $R_{(xF \text{ and } yF)}$ holds in F.

The reformulated DCA ensures the cross-lingual transfer of syntactic relations existent between two lexical items into the same syntactic relations between the translation equivalents of those lexical items in a parallel corpus.

3. Resources and Tools

The parallel corpus that we use is George Orwell's *1984*, which was developed during the (MULTEXT-EAST 1998) project. This corpus is rather small as one can see in Table 1.

	English	Romanian
Translation units	6411	
Unique lemmas	7359	7248
Unique word forms	10152	15112

Table 1: Quantitative data about the *1984* parallel corpus

1984 is XML encoded obeying a simplified form of the XCES standard (Ide et al. 2000) and is sentence aligned, tokenized and morpho-syntactically annotated with the same tagset over the language in the corpus in order to provide a direct correspondence between the parts of speech (POS) of the two languages at hand. Besides, the above-mentioned annotation, we have also used a simple chunker to mark the constituents of a given sentence: noun phrases, prepositional phrases, adjectival and adverbial groups and verbal groups. Two separate grammars have been written (one for English and the other for Romanian) that generate PERL regular expressions over sequences of POS tags of English and Romanian for each type of phrase.

1984 has also been word aligned using a combined word aligner (COWAL) (a program that for every index of a word in a source language sentence gives the index of a word in the target language sentence to which the source word aligns) described in (Tufiş et al. 2005). The above-mentioned chunks were successfully used in reducing the ambiguities that a word aligner has to face, assuming that in most cases, the chunks align as units between Romanian and English.

For the transfer of the syntactic relations between English and Romanian, another annotation of the English part of *1984* was needed: the syntactic analysis using a functional dependency grammar (FDG). In particular, we had at our disposal the output of the FDG parser described in (Tapanainen, Järvinen, 1997a) and (Tapanainen, Järvinen, 1997b) on the English version of *1984*. The dependency between two words is marked by specifying the index of the governor along with the function name at the dependant position as in the Figure 1 (where 0 represents the root of the syntactic tree):

```
IDX     WORDFORM      FDG ANNOTATION
 0
 1      It            subj:2
 2      was           main:0
 3      a             det:6
 4      bright        attr:5
 5      cold          attr:6
 6      day           tmp:2
. . .
```

Figure 1: Representation of dependencies

The FDG parser was applied on an older version of the English *1984* and due to the fact that the tokenization of the English *1984* parsed with the FDG and the tokenization of our current version of 1984 are different, we were forced to use a subset of the translation units from our corpus, so that the following conditions held:

1. The selected translation unit contains sentences that are in a 1:1 correspondence, meaning that the English sentence is translated by a single Romanian sentence;
2. The tokenization of the English sentence from the English part of our corpus is the same as the tokenization of that sentence in the FDG annotated variant.

After making this selection, there were 1537 translation units left in which only 1:1 sentence alignments exist. This selection also favors COWAL because the shorter the sentences, the better the accuracy of the word alignment.

4. Problems for DCA

The parallel corpus that we have made use of raises some problems due to the strategy adopted by the translator: his/her aim was giving a literary translation of Orwell's novel, not a literal one which keeps as close as possible to the original version.

The sentences we focused on for the transfer are those where the translator kept close to the original both semantically and syntactically, trying to use the most appropriate Romanian equivalents of the English words, and also in similar syntactic structures.

While making the selection of the units that are worth taking into consideration for our task, we manually corrected the alignments that were wrongly identified by the COWAL aligner. As similar previous experiments proved (see for instance (Hwa et al. 2002a)), the quality of the alignment results influences the quality of the syntactic transfer.

5. The transfer procedure

Having ensured that the English sentence is as closed as possible translated into Romanian with respect to the syntactic realization of its content, we pursued the following course of action for every syntactic dependency relation (`srel`) of the English sentence:

1. extract the alignment indexes of the governor and dependant of the English `srel` in Romanian. We thus obtain two indexes sets: `G(ro)` and `D(ro)`;
2. if `|D(ro)|` is equal to `|G(ro)|` and equal to 1 (`|·|` being the set cardinality function) and if `d(ro)∈D(ro)` and `g(ro)∈G(ro)`, and `d(ro)≠g(ro)` then transfer the relation `g(ro) srel d(ro)` (we simply transfer the relation from English to Romanian provided that "both ends of the (relation) arrow" point to single (different) indexes in Romanian);
3. if either `|D(ro)| > 1` or `|G(ro)| > 1`, we employ a rule-based algorithm for the extraction of the group head from the Romanian alignment indexes set that has more that one index in it. For instance, if the alignment 'went' – 'se duse' is encountered, one such rule extracts the Romanian verb 'duse' as the head of the construction (the index of whom comprises the new, reduced set, `D(ro)` or `G(ro)`) and the transfer algorithm continues with step 2.

For the purpose of identifying the verbal syntactic relations in Romanian, we decided to transfer all the available relations from English and to assess their generality over English and Romanian. Table 2 gives the percentages of the relations transferred in Romanian from the total of relations present in the English part of the bitext (for the description of these relations, as well as for examples, one can see (Tapanainen and Järvinen 1997b)). We assume, in concordance with the DCA that the higher the transfer percentage is, the more chances there are that the relation also holds in Romanian.

In addition to these relations, we discovered some relations (see the LOST column) that are, in some cases, English syntax tailored. That is, at step 2 in the transfer algorithm, if `d(ro)=g(ro)`, the relation is lost (because the "relation arrow" will start and point to the same index in Romanian). Obviously, these relations were not transferred.

Rel	RO	LOST	EN	Transfer Percent
Pth	1	0	1	100%
Pccomp	1	0	1	100%
agt	4	0	5	80%
neg	10	0	13	76.92%
oc	3	0	4	75%
dat	3	0	4	75%
cnt	8	0	11	72.72%
ad	25	0	35	71.42%
pcomp	218	9	316	68.98%
loc	26	0	39	66.66%
meta	40	0	63	63.49%
comp	70	1	112	62.5%
attr	151	4	245	61.63%
cc	94	2	155	60.64%
pm	44	1	75	58.66%
obj	79	2	137	57.66%
mod	114	1	201	56.71%
ha	41	0	74	55.4%
cla	8	0	15	53.33%
tmp	23	0	46	50%
man	16	0	32	50%
goa	7	0	14	50%
subj	121	2	319	37.93%
frq	8	0	22	36.36%
det	126	173	355	35.49%
dur	1	0	3	33.33%
cnd	1	0	4	25%
v-ch	35	48	143	24.47%
phr	3	0	15	20%
ins	0	0	1	0%

Table 2: Percent of transferred relations

6. Comments on the transfer possibilities

6.1. Perfect import

Some preliminary results showed us that the cross-lingual transfer of syntactic relations is possible most of the times, thus confirming the DCA.

6.2. Import with some amendments

Sometimes, although the syntactic structures in the two languages are not similar, some relations can be transferred. It is the case of the "active" constructions in English which are translated into Romanian with their "passive" counterparts:

(1) En: It was partly the unusual geography of the room that had suggested to him the thing that he was now about to do.
Ro: Lucrul pe care avea de gând să-l facă îi fusese sugerat, în parte, de această geografie neobişnuită a camerei.

Had suggested is in the active voice and enters the following relations: `subj` (with *that*), `dat`[1] (with *to him*), and `obj`[2] (with *the thing*). Its Romanian counterpart, *fusese sugerat*, establishes the `dat` relation with *îi* (the equivalent of *him*), the `subj` relation with *lucrul* (the equivalent of *the thing*) and `phr` relation with the group headed by the preposition *de*. From the morpho-syntactic annotation we can get the information that the Romanian sentence is in the passive voice, so we can create a rule for the import of syntactic functions, a rule which may help the conditioned "inverse" import of some functions: the `subj` is imported as an `obj`, and the `obj` as a `subj`.

Another such example is represented by those situations when the translator simply cannot keep close to the original, as the target language does not permit it. This case justifies the low percent of `subj` relation. Consider the following example:

(2) En: It was a peculiarly interesting book.
Ro: Era o carte deosebit de frumoasă.

Such cases are rather frequent: Romanian lacks an equivalent for the English dummy (anticipatory) *it*, so the subject relations existing in English has no Romanian counterpart. However, the `comp` relation existing in such sentences cannot be transferred as such in Romanian, but it would be appropriate to transfer it as a `subj` relation (this involves ignoring the `subj` relation in the source language and the transfer of the `comp` relation as `subj` in the target language).

6.3. Language specific phenomena

The typological differences between the two languages considered make idiosyncrasies unavoidable. Unlike English, Romanian is a *pro-drop* language, thus many `subj` relations from the source language remain without an equivalent in the target one. Consider the following example, where the Romanian sentence lacks a lexicalized subject for the verb *erai*, the equivalent of *had*:

(3) En: You had to live.
Ro: Erai obligat să trăieşti.

Another "peculiarity" of Romanian is the doubling phenomenon: a direct or indirect object lexicalized as an NP with some semantic and/or syntactic characteristics (Guțu Romalo 1973) is obligatorily doubled by a pronominal clitic with which it shares the grammatical information of case, gender, person, and number.

A further step (at the target language level only) would be taking a decision concerning the treatment of the clitics in such situations. The possibilities would be either to treat them at the morphological level, so part of the verbal morphology, or to treat them at the syntactic level and postulate a language-specific relation (which we may call `anaph`) holding between the clitic and its co-referent NP. The grammatical information shared by the two would ease the resolution.

6.4. Impossibility of import

Besides such idiosyncrasies due to the typological differences between the chosen languages there are also cases when the equivalent verbs display a different syntactic behavior.

(4) En: I like to see them kicking.
Ro: Îmi place să-i văd dând din picioare.

Like takes a `subj` (*I*) and an `obj` (*see*), while *place* is involved in a `dat` (*îmi*) and an `obj` (*văd*) relations.

7. Conclusions and further work

The preliminary results of the syntactic annotation transfer justifies our belief that an automatic procedure of extracting verb frames in Romanian is reliable provided that all the resources with the required level of annotation are present. However, language specific structures and grammatical phenomena require the pre- and post-processing of the data. That is why, our very next step is the implementation of linguistic rules for eliminating the noise obtained after the transfer.

We are perfectly aware that our corpus is too small. As we needed a corpus as well as possibly aligned at the word level, we restricted our analysis to a limited number of sentences for which we could manually check the results of the COWAL aligner. For the future, as we will have a better version of the COWAL, we will be able to extend the corpus.

Through this study we aim at enriching the Romanian WordNet (Tufiş et al. 2004) developed during the BalkaNet project with the verb frames obtained via word alignment and syntactic relation transfer.

[1] This is the relation established between the indirect object (in Dative) and the verb whose argument it is.

[2] This relation is established between the verb and its object. According to (Tapanainen and Järvinen 1997b) the notion of object comprises essentially all types of second arguments, except subject complements.

References:

(Guţu Romalo 1973) V. Guţu Romalo, *Sintaxa limbii române. Probleme şi interpretări*, Bucharest, Editura Didactică şi Pedagogică, 1973.

(Hwa et al. 2002a) R. Hwa, Ph. Resnik, A. Weinberg, Breaking the Bottleneck for Multilingual Parsing. In *Workshop on "Linguistic Knowledge Acquisition and Representation: Bootstrapping Annotated Language Data"*, Third International Conference on Language resources and Evaluation (LREC-2002), Las Palmas, Canary Islands, Spain.

(Hwa et al. 2002b) R. Hwa, Ph. Resnik, A. Weinberg, and O. Kolak, Evaluating Translational Correspondence using Annotation Projection. In *Proceedings of the 40th Annual Meeting of the ACL*, Philadelphia, PA, 2002b.

(Ide et al. 2000) N. Ide, P. Bonhomme , L. Romary, XCES: An XML-based Standard for Linguistic Corpora. In *Proceedings of LREC2000*, Athens, Greece.

(Marinov 2004) S. Marinov, *(Semi-)Automatic Transfer of Syntactic Information*, 2004, www.gslt.hum.gu.se/~svet/courses/treebank/paper.pdf

(MULTEXT-EAST 1998) L. Dimitrova, T. Erjavec, N. Ide, H. Kaalep, V. Petkevic, D. Tufiş: *Multext-East: Parallel and Comparable Corpora and Lexicons for Six Central and Eastern European Languages*, COLING, Montreal, 1998.

(Tapanainen and Järvinen 1997a) P. Tapanainen and T. Järvinen, A non-projective dependency parser. In *Proceedings of the 5th Conference on Applied Natural Language Processing* (ANLP'97), ACL, Washington, D.C., 1997a..

(Tapanainen and Järvinen 1997b) Tapanainen, P. and T. Järvinen, *A dependency parser for English*. Technical Report no. TR-1, Department of General Linguistics, University of Helsinki, Finland, 1997b.

(Tufiş et al. 2004) D. Tufiş, E. Barbu, V. Barbu Mititelu, R. Ion, L. Bozianu. The Romanian Wordnet. In *Romanian Journal on Information Science and Technology*, D. Tufiş (ed.) Special Issue on BalkaNet, Romanian Academy, vol7, no. 2-3, 2004.

(Tufiş et al. 2005) D. Tufiş, R. Ion, Al. Ceauşu, D. Ştefănescu, Combined Word Alignments. To appear in the *Proceedings of the ACL 2005 Workshop on Parallel Text*, Ann Arbor, Michigan, USA.

Selecting Features for Semantic Roles in QA Systems

P. Moreda and **M. Palomar**
Grupo de investigación del Procesamiento del Lenguaje y Sistemas de Información
Departamento de Lenguajes y Sistemas Informáticos. Universidad de Alicante
Alicante, Spain
{moreda,mpalomar}@dlsi.ua.es

Abstract

In this paper a methodology to select one of the best set of features in Semantic Roles annotation process based on Machine Learning method, is proposed. So, this paper will present how the selected set of features can be applied on two different Machine Learning systems, Maximum Entropy and TiMBL. The results will show the importance of a features selection process. In addition, the necessity of a semantic role annotation process in a Question Answering System will be shown.

1 Introduction

The use of Machine Learning (ML) strategies in Natural Language Processing tasks is growing more and more every day. ML is a field focused on making machines learning to make predictions from examples.

One of the difficulties of ML strategies is selecting the best attributes (also named features) to be uses when learning from a large set of candidate attributes. Ideally, a learning algorithm's generalization performance would improve when it is given the information supplied by additional attributes. Unfortunately, the opposite often occurs: additional attributes can interfere with other more useful attributes.

In this paper a methodology to select the best set of features on ML strategies to annotation of Semantic Roles is presented. In order to do this, two different ML approaches will be used, Maximum Entropy and TiMBL.

In addition, the necessity of a semantic role annotation process in a Question Answering (QA) System will be shown.

A semantic role is the relationship between a syntactic constituent and a predicate. So, the semantic role is the role given by the predicate to its arguments. For instance, in the next sentence

(E0) The executives gave the chefs a standing ovation

The executives has the Agent role, *the chefs* the Recipient role and *a standing ovation* the Theme role.

To achieve high precision QA systems, recognizing and labeling semantic arguments is a key task for answering "Who", "When", "What", "Where", "Why", etc. For instance, the following questions could be answered with the sentence (E0). The Agent role answers the question (E1) and the Theme role answers the question (E2).

(E1) Who gave the chefs a standing ovation?

(E2) What did the executives give the chefs?

The remaining paper is organized as follows: firstly, the two ML strategies used in this semantic role annotation process are described in section 2. Secondly, the principal issues of the feature selection process and the most important search algorithms are shown in section 3. Then the tuning process applied in order to obtain the best set of features for a semantic role annotation process and the results obtained making use of the selected features are presented in sections 4 and 5, respectively. Next, the importance of a semantic role annotation process in a QA system is shown in section 6. Finally, section 7 concludes.

2 Machine Learning Approaches

Statistical approaches to process natural language texts have become dominant in recent years. Therefore, different approaches have been developed. In our semantic role annotation process two of them will be used: Maximum Entropy and TiMBL, which are briefly explained next.

2.1 Maximum Entropy Models

Maximum Entropy (ME) modelling provides a framework to integrate information for classification from many heterogeneous information

sources (Manning & Schütze 99). ME probability models have been successfully applied to some Natural Language Processing tasks, such as part-of-speech (POS) tagging or sentence boundary detection (Ratnaparkhi 98).

The method presented in this paper is based on conditional ME probability models. It has been implemented using a supervised learning method that consists of building classifiers using a tagged corpus. A classifier obtained by means of an ME technique consists of a set of parameters or coefficients which are estimated using an optimization procedure. Each coefficient is associated with one feature observed in the training data. The main purpose is to obtain the probability distribution that maximizes the entropy, that is, maximum ignorance is assumed and nothing apart from the training data is considered. Some advantages of using the ME framework are that even knowledge-poor features may be applied accurately; the ME framework thus allows a virtually unrestricted ability to represent problem-specific knowledge in the form of features (Ratnaparkhi 98).

Let us assume a set of contexts X and a set of classes C. The function $cl : X \rightarrow C$ chooses the class c with the highest conditional probability in the context x: $cl(x) = \arg\max_c p(c|x)$. Each feature is calculated by a function that is associated with a specific class c', and it takes the form of equation (1), where $cp(x)$ is some observable characteristic in the context[1]. The conditional probability $p(c|x)$ is defined by equation (2), where α_i is the parameter or weight of the feature i, K is the number of features defined, and $Z(x)$ is a constant to ensure that the sum of all conditional probabilities for this context is equal to 1.

$$f(x,c) = \begin{cases} 1 & \text{if } c' = c \text{ and } cp(x) = true \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

$$p(c|x) = \frac{1}{Z(x)} \prod_{i=1}^{K} \alpha_i^{f_i(x,c)} \quad (2)$$

2.2 TiMBL

TiMBL (Daelemans *et al.* 03) is a program implementing several memory-based learning algorithms. All implemented algorithms have in common that they store some representation of the training set explicitly in memory. During testing, new cases are classified by extrapolation from the most similar stored cases.

Memory-based learning (MBL) is founded on the hypothesis that performance in cognitive tasks is based on reasoning on the basis of similarity of new situations to stored representations of earlier experience, rather than on the application of mental rules abstracted from earlier experiences.

A MBL system contains two components:

- A learning component which is memory-based and
- A performance component which is similarity-based.

The learning component of MBL is memory-based as it involves adding training instances to memory (the instance base or case base); it is sometimes referred to as *lazy* since memory storage is done without abstraction or restructuring. An instance consists of a fixed-length vector of n feature-value pairs, and an information field containing the classification of that particular feature-value vector.

In the performance component of an MBL system, the learning component is used as a base for mapping input to output: this usually takes the form of performing classification. During classification, a previously unseen test example is presented to the system. The similarity between the new instance X and all examples Y in memory is computed using some distance metric $\Delta(X,Y)$ (see equations (3) and (4)). The extrapolation is done by assigning the most frequent category within the found set of most similar example(s) (the k-nearest neighbors) as the category of the new test example. In case of a tie among categories, a tie breaking resolution is used.

$$\Delta(X,Y) = \sum_{i=1}^{K} |\delta(x_i, y_i)| \quad (3)$$

where:

$$\delta(x_i, y_i) = \begin{cases} abs(\frac{x_i - y_i}{max_i - min_i}) & \text{if numeric, else} \\ 0 & \text{if } x_i = y_i \\ 1 & \text{if } x_i \neq y_i \end{cases} \quad (4)$$

[1]The ME approach is not limited to binary functions, but the optimization procedure used for the estimation of the parameters, the *Generalized Iterative Scaling* procedure, uses this feature.

3 Methodology for Features Selection

The selection of relevant features, and the elimination of irrelevant ones, is a central problem in ML. The task of features selection can be viewed as a search problem (Langley 94). Any feature selection method must consider four basic issues that determine the nature of the heuristic search problem:

- *The starting point of the search.* For example, it is possible to start with no features and successively add attributes, or to start with all attributes and successively remove them.
- *The organization of the search.* An exhaustive search is impractical, as there exist 2^a possible subsets of a attributes. A more realistic approach relies on a greedy method. At each point in the search, local changes to the current set of attributes are considered, selecting or eliminating one of them, and then iterates, never reconsidering the choice.
- *The strategy used to evaluate.* Some strategies, named filter methods, consider attributes independently of the machine learning algorithm that will use them, relying on general characteristics of the training set to select some features and exclude others. Other approaches with wrapper methods generate a set of candidate features, run the ML algorithm on the training data and use the accuracy of the resulting description to evaluate the feature set.
- *The criterion for halting search.* For example, search could stop when none of the alternatives improves the estimation of classification accuracy, or search could revise the feature set as long as accuracy does not degrade.

3.1 The organization of the search

Perhaps the most important issue of a feature selection method is the organization of the search. Most of ML methods generalize worse when dealing with too many attributes, instead of a good subset of those attributes. There are several methods that greedily search attribute subsets that generalize well when given to a learning procedure. These methods (Caruana & Freitag 94), which are explained next, differ only in the particular hillclimbing strategy they employ.

3.1.1 Forward Selection (FS)

The method starts with the empty set and greedily adds attributes, one at a time, until all attributes are added.

First, the attribute which results in the best fit is selected. Next, this attribute is used to test all combinations with the remaining attributes in order to find the best pair of attributes. In all further steps, additional attributes are added until either all attributes are used up, or some stopping criterion is reached. Once an attribute is added FS cannot remove it later.

This algorithm can be summarized in four steps:

1. Calculate all partial values for each independent attribute
2. Select the best fit
3. Calculate all combinations with the remaining attributes
4. Proceed with step 2

3.1.2 Backward Elimination (BE)

It starts with all attributes in the attribute set and greedily removes them one at a time until no attributes remain. The algorithm is defined as follows:

1. Calculate all partial values for each independent attribute
2. Calculate a model including all available attributes
3. Remove the attribute with the lowest independent value, if it falls below a predefined limit
4. Proceed with step 2

Like FS, once BE removes an attribute, it cannot add it back to the set later again.

3.1.3 Forward Stepwise Selection (FSS)

Like FS, FSS is greedy attribute hillclimbing initialized with the empty attribute set. However it considers the whole set of attributes at any step of the search.

3.1.4 Backward Stepwise Elimination (BSE)

BSE is attribute hillclimbing initialized with the complete set of attributes and at each step one attribute is eliminated. In addition, all sets of attributes, eliminated in a previous step or not, are candidates to be eliminated at any step of the search.

3.1.5 Backward Stepwise Elimination - SLASH (BSE-SLASH)

BSE-SLASH starts will the full attribute set, but after taking a step, eliminates any attribute not used in what was learned at that step.

4 Tuning the Semantic Role Annotation Tool

In this section the tuning of the set of features for the Semantic Roles Disambiguation process is presented. The Semantic Role Disambiguation is part of our SemRol method (Moreda *et al.* 04). SemRol is a Semantic Role Labelling tool based on ML. It consists of three main phases: i) Verb Sense Disambiguation phase, ii) Argument Boundaries Disambiguation phase, and iii) Semantic Role Disambiguation phase. Each phase is independent. First of all, the sense of the verb has to be obtained because different senses of a verb will have different sets of semantic roles. Secondly, the argument boundaries are determined. And finally, the semantic roles that fill these arguments are obtained.

In this paper we focus on Semantic Role Disambiguation phase. All of the three modules need a feature selection process in order to obtain a set of features that maximizes the results but the process is the same one in the three cases.

Taking into account the previous section, section 3, the feature selection process to our semantic role annotation tool has been defined as follow:

- The starting point of the search will be the empty set. It is determined by the algorithm used in the organization of the search.
- The organization of the search. In order to obtain one of the best set of features the FS algorithm will be applied. It has been selected because it reduces the set of possible subsets considered. Once an attribute is added FS cannot remove it latter. However, FSS, BSE and BSE-SLASH consider the whole set of attributes at any step of the search. On the other hand, FS is not needed to determine a predefined limit to remove or not an attribute like as BE. If the limit was wrong, the process could be wrong also.
- The strategy used to evaluate. It will be used a wrapper method because we consider there are not any advantages to an independent evaluation strategy. So, both ML algorithms presented in section 2, ME and TiMBL, will be used.
- The criterion for halting search. In order to reduce still more the number of possible subsets, the search will stop when the results are not improved.

4.1 Feature set

The initial set of features has used partial syntactic information, such as part of speech tag, base chunks, clauses and named entities (see section 5.1). This initial feature set, which consists of 25 features, is the following:

- Features based on arguments

  Predicate position (F6). The position of the argument with respect of the verb, before (-1) or after (+1) the predicate.

  Clause position (F7). It indicates if the argument is inside (-1), outside (+1) or in the same (0) clause which contains the predicate.

  Distance in words (F8), phrases (F9) and arguments (F10). Distance from the argument to the predicate as a number of words, phrases or arguments. The possible values are 0, 1 or 2, when the number of words is 0, or is between 1 or 2, or is more than 2, respectively.

  Number of words (F11), phrases (F12), and arguments (F13). Number of words, phrases or arguments between the argument and the predicate.

- Features based on Named Entities (NE)

  Kind/List of Named Entities (F14), (F16). Different kinds/list of NE in the argument.

- Features based on phrases

  List of Phrases (F17), (F18). List of phrases in the argument including or not the position in the phrase.

  Prepositions (F19), (F51). If the argument begins with a preposition, the preposition and the part of speech tag of the preposition.

  Headwords (F20). Headwords of the phrases included in the argument. Heads in syntactic phrases refer to words with part of speech related to noun, in a noun phrase; or related to verb, in a verb phrase.

  Lemma of Headwords (F109). The first four letters of each headwords of the phrases included in the argument.

- Features based on Part of Speech tag

  Content-words (F30). Words in the argument with part of speech related to noun, adjective, adverb or verb.

  PoS/Lemma of Content-words (F112),(F107), (F108). Part of speech/lemma of content-words in the argument.

  Words (F111). Part of speech of the words in the argument.

  Headwords (F22). Headwords of the phrases included in the argument.

  Nouns (F27), Adjectives (F28) or Adverbs (F29). Nouns, adjectives or adverbs in the argument.

- Features based on sentence

  Voice (F2). Voice of the sentence. The possible values are P or A, depending on the voice will be passive or active, respectively.

5 Results and Discussion

Before showing the obtained results, a brief description about the used experimental data is presented.

5.1 Experimental Data

Our methodology presented on section 3 has been applied on the PropBank corpus (Palmer *et al.* 05), which is the Wall Street Journal part of the Penn Treebank corpus (Marcus *et al.* 93) enriched with predicate-arguments structures. To be preciset raining set matches with sections 15-18 and development set matches with section 20.

PropBank annotates the Penn Treebank with arguments structures related to verbs. The semantic roles considered in PropBank are the following (Carreras & Màrquez 04):

- Numbered arguments (A0-A5, AA): Arguments defining verb-specific roles. Their semantics depends on the verb and the verb usage in a sentence, or verb sense. In general, A0 stands for the *agent* and A1 corresponds to the *patient* or *theme* of the proposition, and these two are the most frequent roles. However, no consistent generalization can be made across different verbs or different senses of the same verb. PropBank takes the definition of verb senses from VerbNet, and for each verb and each sense defines the set of possible roles for that verb usage, called roleset.

- Adjuncts (AM-): General arguments that any verb may take optionally. There are 13 types of adjuncts:
  - AM-LOC: location
  - AM-EXT: extent
  - AM-DIS: discourse marker
  - AM-ADV: general-porpouse
  - AM-NEC: negation marker
  - AM-MOD: modal verb
  - AM-CAU: cause
  - AM-TEMP: temporal
  - AM-PRP: purpose
  - AM-MNR: manner
  - AM-DIR: direction

- References (R-): Arguments representing arguments realized in other parts of the sentence. The role of a reference is the same than the role of the referenced argument. The label is an R-tag preceded to the label of the referent, e.g. R-A1.

- Verbs (V): Participant realizing the verb of the proposition.

Training data consists of 8936 sentences, with 50182 arguments and 1838 distinct verbs. Development data consists of 2012 sentences, with 11121 arguments and 978 distinct verbs.

Apart from the correct output, both datasets contain the output of several annotation processors: PoS tags (Giménez & Màrquez 03), chunks and clauses (Carreras & Màrquez 03) and named entities (Chieu & Ng 03).

5.2 Results

The features have been evaluated about precision, recall and F1 measure. Precision (p) is the proportion of arguments predicted by the system which are correct. Recall (r) is the proportion of correct arguments which are predicted by the system. F1 measure computes the harmonic mean the precision and recall. It is formulated as $F_{\beta=1}=(2pr)/(p+r)$.

The results about TiMBL are shown in Table 1. These results show how additional attributes interfere with other more useful attributes. The precision using the complete set of features is 64.90%. This precision is exceeded by sets of four features (66.33%) and more. So, the highest precision is obtained with a set of eight features (68.00%). The last row of this table shows the results obtained by the eight features with the best individual results (62.76%).

On the other hand, the experiments using ME are shown in table 2. In this case, the precision using the complete set of features is 59.00%. This precision is exceeded by sets of five features (61.93%) and more. So, the highest precision is obtained with a set of seven features (62.41%). The last row of this table shows the results obtained by the seven features with the best individual results (58.95%).

In order to tune the features a reduced corpus has been used. So, the tables 3 and 4 show the results with the complete corpus. Talking about

Features	P	R	$F_{\beta=1}$
6	52.73	49.95	51.30
6,111	61.15	57.93	59.50
6,19,111	64.32	60.94	62.59
6,7,19,111	66.33	62.88	64.52
6,7,19,30,111	67.06	63.49	65.23
6,7,19,30,51,111	67.62	64.02	65.77
6,7,19,20,30,51,111	67.86	64.23	66.00
2,6,7,19,20,30,51,111	**68.00**	**64.31**	**66.10**
2,6,7,19,20,30,51,107,111	67.95	64.26	66.05
2,6,7,19,20,30,51,107,109,111	67.76	64.08	65.87
2,6,7,19,20,30,51,107,109,111,112	67.61	63.93	65.72
All	64.90	61.30	63.05
6,8,12,17,18,22,111,112	62.76	59.36	61.01

Table 1: Tunning using TiMBL

Features	P	R	$F_{\beta=1}$
6	51.80	51.16	51.48
6,111	57.64	59.63	57.28
6,19,111	56.64	55.94	56.29
6,7,19,111	56.55	55.87	56.21
6,7,19,30,111	61.93	62.50	62.21
6,7,19,30,51,111	61.75	62.32	62.03
6,7,19,20,30,51,111	**62.41**	**62.98**	**62.69**
2,6,7,19,20,30,51,111	60.99	61.54	61.26
2,6,7,19,20,30,51,107,111	61.66	62.21	61.93
2,6,7,19,20,30,51,107,109,111	61.80	62.35	62.07
All	59.00	59.53	59.26
6,8,12,17,18,22,111	58.95	59.47	59.21

Table 2: Tunning using ME

precision, the first rows show the results for the complete set of features (68.76% for TiMBL and 57.74% for ME). The second rows show the results for the best set of features for each ML method: F2,F6,F7,F19,F20,F30,F51,F111 for TiMBL (70.95%) and F6,F7,F19,F20,F30,F51,F111 for ME (62.25%). Finally, the third rows show the results for the eight/seven features which have the best individual results for each ML algorithm: F6,F8,F12,F17,F18,F22,F111,F112 for TiMBL (65.31%) and F6,F8,F12,F17,F18,F22,F111 for ME (61.69%).

Features	P	R	$F_{\beta=1}$
All	68.76	65.55	67.12
Best set	**70.95**	**67.84**	**69.36**
Set of eight	65.31	62.33	63.79

Table 3: Results of tunning with TiMBL

6 Applying Semantic Roles to Question Answering Systems

Other goal of this paper is to integrate Semantic Roles in a QA system, in order to achieve high precision QA systems.

The architecture of a QA system extended with the SemRol method is shown in Figure 1. It

Features	P	R	$F_{\beta=1}$
All	57.74	57.44	57.59
Best set	**62.25**	**61.91**	**62.08**
Set of seven	61.69	61.27	61.48

Table 4: Results of tunning with ME

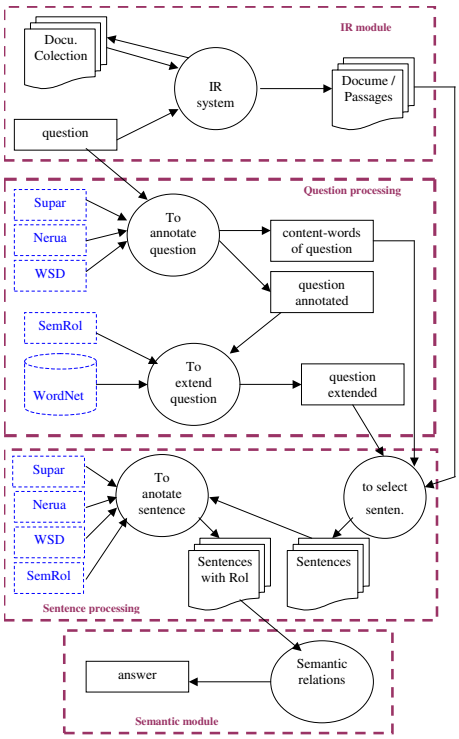


Figure 1: Architecture of a QA system based on Semantic Roles (SemRol method).

consists of four modules: Information Retrieval (IR) module, question processing module, sentence processing module and semantic module (Moreda *et al.* 05).

When a query is done the *IR module* retrieves a set of passages or documents, depending on the IR system used. It is supposed that these passages or documents contain the answer of the query.

Then, once the question has been annotated with different tools, such as, SUPAR (Palomar *et al.* 01), NERUA (Kozareva *et al.* 05) and a WSD system (Suárez & Palomar 02), (Montoyo *et al.* 05), the question is extended in the *Question Processing module*. So, a list of verbs related to the verb in the query is obtained. In order to do this, an electronic lexical database is used, WordNet (Miller *et al.* 90). In our system, the list of related verbs is extracted making use of synonymy and troponymy relations. In addition, the question is extended with semantic role information making use of the SemRol method.

Next, only passages containing sentences with one of these verbs are selected and those sentences are marked in the *Sentence Processing module*. These sentences are annotated with semantic information by using the SemRol method. So, the argument boundaries of the sentences are recognized and the semantic roles that fill this arguments are identified. As a result, a set of annotated sentences with the roles of the arguments of the verbs is obtained.

Finally, a set of semantic relationships are applied in the *Semantic module*. Depending on the kind of the question a different set of roles could be considered. So, it is possible to define a set of semantic relationships in order to establish a relationship between questions and semantic roles. For instance, questions such as "When", "What + time expression" or "In what + time expression" must be answered with the Temporal semantic role and must not be answered with the Agent, Patient, Location, Cause or Mode semantic role; and "Where", "In where + location expression" or "In what + location expression" must be answered with the Location semantic role and must not be answered with the Agent, Patient, Temporal, Cause or Mode semantic role.

Making use of these rules, only the sentence containing the right semantic roles is selected.

7 Conclusions

In this paper a methodology to select one of the best set of features in Semantic Roles annotation process based on ML methods has been proposed. As a result, the fact that additional attributes interfere with other more useful attributes has been demonstrated. So a tuning process has been applied starting with the empty set of features and greedily adding features one at a time (FS algorithm) until none of the alternatives improves the precision. In order to do this, two different kind of ML methods have been tested, ME and TiMBL. So, the best results in TiMBL have been obtained with a set of eight features (70.95% of precision) and in ME with a set of seven features (62.25% of precision) instead of the complete initial set of twenty five features (68.76% of precision for TiMBL and 57.74% precision for ME).

In addition, the necessity of a Semantic Role annotation process in a QA system in order to achieve high precision QA systems has been presented.

References

(Carreras & Màrquez 03) X. Carreras and L. Màrquez. Phrase recognition by filtering and ranking with perceptrons. In *Proceedings of Recent Advances in Natural Language Processing 2003*, Borovets, Bulgaria, Septiembre 2003.

(Carreras & Màrquez 04) X. Carreras and L. Màrquez. Introduction to the CoNLL-2004 Shared Task: Semantic Role Labeling. In *Proceedings of the Eighth Conference on Natural Language Learning (CoNLL-2004)*, Boston, MA, USA, Mayo 2004.

(Caruana & Freitag 94) R. Caruana and D. Freitag. Greedy attribute selection. In Morgan Kaufman, editor, *Proceedings of the 11th International Conference on Machine Learning*, pages 28–36, 1994.

(Chieu & Ng 03) H.L. Chieu and H.T. Ng. Named entity recognition with a maximum entropy approach. In *Proceedings of the Seventh Conference on Natural Language Learning (CoNLL)*, Edmonton, Alberta, Canada, Mayo-Junio 2003.

(Daelemans *et al.* 03) W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch. Timbl: Tilburg memory based learner, version 5.0, reference guide. ILK Research Group Technical Report Series 03-10, Tilburg, 2003. 56 pages.

(Giménez & Màrquez 03) J. Giménez and L. Màrquez. Fast and Accurate Part-of-Speech Tagging: The SVM Approach Revisited. In *Proceedings of Recent Advances in Natural Language Processing 2003*, Borovets, Bulgaria, Septiembre 2003.

(Kozareva *et al.* 05) Z. Kozareva, O. Ferrndez, A. Montoyo, R. Muoz, and A. Surez. Combining data-driven systems for improving named entity recognition. In *Proceedings of 10th International Conference on Natural Language Processing and Information Systems (NLDB2005)*, Alicante, Spain, Junio 2005.

(Langley 94) P. Langley. Selection of Relevant Features in Machine Learning. In AAAI Press, editor, *Proceedings of the AAAI Fall Symposium on Relevance (AAAI)*, New Orleans, LA, 1994.

(Manning & Schütze 99) C.D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts, 1999.

(Marcus *et al.* 93) M.P. Marcus, B. Santorini, and M.A. Marcinkiewicz. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, (19), 1993.

(Miller *et al.* 90) G. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. Five papers on wordnet. csl report 43. Technical report, Cognitive Science Laboratory, Princeton University, 1990.

(Montoyo *et al.* 05) A. Montoyo, A. Surez, G. Rigau, and M. Palomar. Combining knowledge- and corpus-based word-sense-disambiguation methods. *Journal of Artificial Intelligence Research*, 23:299–330, 2005.

(Moreda *et al.* 04) P. Moreda, M. Palomar, and A. Suárez. Semrol: Recognition of semantic roles. In *Proceedings of Espaa for Natural Language Processing (EsTAL)*, October 2004.

(Moreda *et al.* 05) P. Moreda, B. Navarro, and M. Palomar. Using semantic roles in information retrieval systems. In *Proceedings of 10th International Conference on Natural Language Processing and Information Systems (NLDB2005)*, Alicante, Spain, Junio 2005.

(Palmer *et al.* 05) M. Palmer, D. Gildea, and P. Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106, 2005.

(Palomar *et al.* 01) M. Palomar, A. Ferrndez, L. Moreno, P. Martnez-Barco, J. Peral, M. Saiz-Noeda, and R. Muoz. An Algorithm for Anaphora Resolution in Spanish Texts. *Computational Linguistics*, 27(4):545–567, 2001.

(Ratnaparkhi 98) A. Ratnaparkhi. *Maximum Entropy Models for Natural Language Ambiguity Resolution*. Unpublished PhD thesis, University of Pennsylvania, 1998.

(Suárez & Palomar 02) A. Suárez and M. Palomar. A maximum entropy-based word sense disambiguation system. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING)*, pages 960–966, Taipei, Taiwan, Agosto 2002.

Assigning Function Labels to Unparsed Text

Gabriele Musillo
Depts of Linguistics and Computer Science
musillo4@etu.unige.ch
University of Geneva
2 rue de Candolle
1211 Geneva 4
Switzerland

Paola Merlo*
Department of Linguistics
merlo@lettres.unige.ch
University of Geneva
2 rue de Candolle
1211 Geneva 4
Switzerland

Abstract

In this paper, we propose a novel solution to the problem of assigning function labels to syntactic constituents. This task is a useful intermediate step between syntactic parsing and semantic role labelling. What distinguishes our proposal from other attempts in function or semantic role labelling is that we perform the learning of function labels at the same time as parsing. We reach state-of-the-art performance both on parsing and function labelling. Our results indicate that function label information is located in the lower levels of the parse tree, and that, similarly to other function and semantic labelling results, the main difficulty lies in distinguishing constituents that bear a function label from constituents that do not.

1 Introduction

Recent successes in statistical parsing indicate that the time is ripe to solve deeper natural language understanding tasks using similar techniques (Collins 99; Charniak 00; Henderson 03). To achieve this goal, lexical semantic resources such as Framenet and Propbank are being annotated with semantic roles, as a form of shallow semantic annotation (Baker *et al.* 98; Kingsbury & Palmer 02), and a great deal of work has already been proposed to solve the problem of semantic role labelling (Gildea & Jurafsky 02; Nielsen & Pradhan 04; Xue & Palmer 04). See also the common task of (Senseval 04; CoNLL 04). Semantic information will be useful in information extraction applications (Surdeanu *et al.* 03), dialogue (Stallard 00), question-answering, and machine translation systems, among others.

A level of annotation similar to semantic role labels is already present in the the Penn Treebank (PTB) WSJ corpus (Marcus *et al.* 93) in the form of function labels. For instance, in the sentence *The Government's borrowing authority dropped at midnight Tuesday to 2.80 trillion from 2.87 trillion*[1], the constituent *The Government's borrowing authority* bears the function label SBJ and the PP *at midnight* the function label TMP. Table 1 provides the complete list of function labels in the PTB corpus. Function labels represent an intermediate level between syntactic phrase structure and semantic roles, and they have not yet been fully exploited, as observed in (Blaheta & Charniak 00). Function labels expressing grammatical roles, such as LGS, are useful in recovering argument structure. Semantically oriented labels, such as DIR, carry semantic role information.

Syntactic Labels		**Semantic Labels**	
DTV	dative	ADV	adverbial
LGS	logical subject	BNF	benefactive
PRD	predicate	DIR	direction
PUT	locative complement of *put*	EXT	extent
SBJ	surface subject	LOC	locative
VOC	vocative	MNR	manner
Miscellaneous Labels		NOM	nominal
CLF	*it*-cleft	PRP	purpose or reason
HLN	headline	TMP	temporal
TTL	title		
CLR	closely related	**Topic Labels**	
		TPC	topicalized

Table 1: Complete set of function labels in the Penn Treebank.

In this paper, we illustrate how to learn function labels during parsing, annotating parse trees with a richer set of non-terminal labels set than the standard PTB label set. Few other attempts have been made to automatically learn PTB function labels (Blaheta & Charniak 00; Blaheta 04; Jijkoun & deRijke 04)[2]. What distinguishes our

* We thank the Swiss National Science Foundation for supporting this research under grant number 101411-105286. We also would like to thank the reviewers for their helpful comments and James Henderson for his useful discussions.

[1] PTB, section 00.

[2] Recent attempts at automatically generating parsing systems consisting of a Lexical-Functional Grammar (LFG) have dealt with the problem of learning f-structures (Riezler *et al.* 02; Cahill *et al.* 04). Labels in LFG f-structures encode predicate-argument relations, similarly

proposal from previous attempts – and from most existing work on semantic role labelling – is that we perform the learning at the same time as parsing.

Our proposal tests the hypothesis that the function label of a constituent can be determined based only on the structural position it occupies in a labelled parse forest, and that a fully connected parse tree is not required to predict it. Also, it assumes that function labels depend on the same context as the usual non-terminal labels. This proposal is, therefore, more constrained than other methods that assign function or semantic role labels in two steps. These other methods have access to a full parse tree, including the context at the right of the node to be labelled. Moreover, the set of features they input to the function or semantic learner could be specialised and be very different from the input features for syntactic parsing.

It is interesting to test a more constrained hypothesis, because its results are of wide applicability. In particualr, since the function labelling is done incrementally, these results could be used in language modelling and interactive applications, where entire parse trees are not available.

2 The Learning Method

Our method is an extension of a robust statistical parser developed on the PTB, whose properties make it particularly adaptive to new tasks (Henderson 03).

2.1 The Function Label Set

The bracketting guidelines for the PTB II list 20 function labels, shown in Table 1 (Bies *et al.* 95). Based on their description in the PTB guidelines, we partition the set of function labels into four classes, as indicated in the table. Following (Blaheta & Charniak 00), we refer to the first class as syntactic function labels, and to the second class as semantic function labels. In the rest of the paper, we will ignore the other two classes, for they do not intersect with PropBank labels, and they do not form natural classes. Like previous work, we complete the sets of syntactic and semantic labels by labelling constituents that do not bear any function label with a NULL label.[3]

2.2 The Parser

Recall that our main hypothesis says that function labels can be successfully and automatically learned and recovered while parsing. It could be objected that this way the parsing task becomes more difficult. Moreover, the independence assumptions of parsing models might not be justified for this new task, rendering such models inappropriate and their parameters more difficult to estimate. It is therefore important to choose a statistical parser that can meet such objections. We use a family of statistical parsers, the Simple Synchrony Network (SSN) parsers (Henderson 03), which crucially do not make any explicit independence assumptions, and are therefore likely to adapt without much modification to the current problem. This architecture has shown state-of-the-art performance.

SSN parsers comprise two components, one which estimates the parameters of a stochastic model for syntactic trees, and one which searches for the most probable syntactic tree given the parameter estimates. As with many others statistical parsers (Collins 99; Charniak 00), the model of parsing is history-based. Its events are derivation moves. The set of well-formed sequences of derivation moves in this parser is defined by a Predictive LR pushdown automaton (Nederhof 94), which implements a form of left-corner parsing strategy.[4]

The probability of a phrase-structure tree can be equated to the probability of a finite (but unbounded) sequence of derivation moves. To bound the number of parameters, standard history-based models partition the set of well-formed sequences of transitions into equivalence classes. While such a partition makes the problem of searching for the most probable parse polynomial, it introduces hard independence assumptions: a derivation move only depends on the equivalence class to which its history belongs.

to our syntactic function labels, but no labels corresponding to the PTB semantic function labels are produced. While these attempts are indeed among the few that output richer annotations than the standard PTB labels, they can not be directly compared to our work.

[3] Strictly speaking, this label corresponds to two NULL labels: the SYN-NULL and the SEM-NULL. A node bearing the SYN-NULL label is a node that does not bear any other syntactic label. Analogously, the SEM-NULL label completes the set of semantic labels. Note that both the SYN-NULL label and the SEM-NULL are necessary, since both a syntactic and a semantic label can label a given constituent.

[4] The derivation moves include: projecting a constituent with a specified label, attaching one constituent to another, and shifting a tag-word pair onto the pushdown stack.

SSN parsers, on the other hand, do not state any explicit independence assumptions: they induce a finite history representation of an unbounded sequence of moves, so that the representation of a move $i-1$ is included in the inputs to the represention of the next move i, as explained in more detail in (Henderson 03). However, SSN parsers impose soft inductive biases to capture relevant properties of the derivation. The art of designing SSN parsers consists in selecting and introducing such biases. To this end, it is sufficient to specify features that extract some information relevant to the next derivation move from previous ones, or some set of nodes that are structurally local to the node on top of the stack. These features and these nodes are input to the computation of a hidden history representation of the sequence of previous derivation moves. Given the hidden representation of a derivation, a log-linear distribution over possible next moves is computed. Thus, the set D of structurally local nodes and the set f of predefined features determine the inductive bias of an SSN system. Unless stated otherwise, for each of the experiments reported here, the set D that is input to the computation of the history representation of the derivation moves $d_1, \ldots, d_{i-1}$ includes the following nodes: top_i, the node on top of the pushdown stack before the ith move; the left-corner ancestor of top_i; the leftmost child of top_i; and the most recent child of top_i, if any. The set of features f includes the last move in the derivation, the label or tag of top_i, the tag-word pair of the most recently shifted word, the leftmost tag-word pair that top_i dominates.

2.3 Evaluation Measures

To evaluate the performance of our function parsing experiments, we extend standard Parseval measures of labelled recall and precision to include function labels. Note that the maximal precision or recall score of function labelling is strictly smaller than one-hundred percent if the precision or the recall of the parser is less than one-hundred percent. Following (Blaheta & Charniak 00), incorrectly parsed constituents will be ignored (roughly 11% of the total) in the evaluation of the precision and recall of the function labels, but not in the evaluation of the parser. Of the correctly parsed constituents, some bear function labels, but the overwhelming majority do not bear any label, or rather, in our notation, they bear a NULL label. To avoid calculating excessively optimistic scores, constituents bearing the NULL label are not taken into consideration for computing overall recall and precision figures. NULL-labelled constituents are only needed to calculate the precision and recall of other function labels. (In other words, NULL-labelled constituents never contribute to the numerators of our calculations.) For example, consider the confusion matrix M in Table 2, which reports scores for syntactic labels of Model 1. Precision is computed as

$$\frac{\sum_{i=\text{DTV},\ldots,\text{VOC}} M[i,i]}{\sum_{j=\text{DTV},\ldots,\text{VOC}} M[\text{SUM}, j]}$$

Recall is computed by setting the denominator to $M[j, \text{SUM}_{gold}]$. Notice that $M[\text{NULL}, \text{NULL}]$ is never taken into account.

	DTV	LGS	PRD	PUT	SBJ	VOC	NULL	SUM_{gold}
DTV	0	0	0	0	0	0	12	12
LGS	0	98	0	0	0	0	19	117
PRD	0	0	482	0	0	0	62	544
PUT	0	0	0	0	0	0	7	7
SBJ	0	0	8	0	2590	0	97	2695
VOC	0	0	0	0	0	0	0	0
NULL	0	20	23	0	59	0	18825	18927
SUM	0	118	513	0	2649	0	19022	22302

Table 2: Confusion matrix for Model 1, calculated on the validation set. The NULL index in the matrix refers to the SYN-NULL label.

3 Experiments

In this section, we report the results of three experiments testing hypotheses concerning function labelling. All SSN function parsers were trained on sections 2-21 from the PTB and validated on section 24. All models are trained on parse trees whose labels include syntactic and semantic function labels. Both parsing results taking function labels into account (FLBL) and results not taking them into account (FLBL-less) are reported in Table 3. For the model that yields the best results on the validation set, we also report results on the test set, section 23 of the PTB. Results indicating performance on function labelling alone are reported in Table 4 below.

3.1 The Models

Model 1 Our hypothesis states that function labelling can be performed incrementally while parsing. First of all, we need to assess the complexity and relevance of the task. We need to

	Validation Set					
	FLBL			FLBL-less		
	F	R	P	F	R	P
Model 1	83.4	82.8	83.9	87.7	87.1	88.2
Model 2	83.8	83.2	84.4	87.9	87.3	88.5
Model 3	84.6	84.0	85.2	88.1	87.5	88.7
	Test Set					
	FLBL			FLBL-less		
Model 3	86.1	85.8	86.5	88.9	88.6	89.3

Table 3: Percentage F-measure (F), recall (R), and precision (P) of SSN parsers.

	Validation Set					
	Syntactic Labels			Semantic Labels		
	F	R	P	F	R	P
Model 1	95.3	93.9	96.7	73.1	70.2	76.3
Model 2	95.6	94.6	96.7	74.5	73.0	76.0
Model 3	95.7	95.0	96.5	80.1	77.0	83.5
	Test Set					
Model 3	96.4	95.3	97.4	86.3	82.4	90.5

Table 4: Results of different models for function labelling, separated for syntactic and semantic labels.

show that the function labelling problem is challenging, as it is not simply derivable from the parsing labels. To show this, we run a simple function parsing model that consists of the original SSN parser trained and tested on a more complex set of nonterminal labels which includes function labels. If function labelling is not easily predictable from parsing, we should have a degradation of the parser model with more complex labels.

For this model, 136 non-terminal labels were needed, in total. Of these labels, 103 consist of a standard non-terminal label and a sequence of one or more function labels. This SSN used all tag-word pairs which occur at least 200 times in the training set, resulting in 508 tag-word pairs.[5]

This first experiment yields two results that provide the starting point of our investigation, shown in the first lines of tables 3 and 4. First, it confirms that function labelling is not easily derived from parsing, as the difference in performance between function labelling (FLBL column) and parsing (FLBL-less column) in Table 3 illustrates. This motivates the task, as it shows that function labels require specific modelling to be properly learnt. The degradation in performance of the initial parser will have to be eliminated for our method to be competitive with other methods which learn function or semantic labels based on the output of a parser. These techniques do not modify the behaviour of the parser in any way, and therefore do not run the risk of improving their performance at the expense of the accuracy of the parser. Instead, we could trivially improve our function labeller by simply reducing the output of the parser to the few cases on which it is very confident.

[5]SSN parsers do not tag their input sentence. To provide the pre-terminal tags used in our first two models, we used (Ratnaparkhi 96)'s POS tagger.

	Syntactic Labels			Semantic Labels		
	F	R	P	F	R	P
BC00	95.7	95.8	95.5	79.0	77.6	80.4
B04 FT	95.9	95.3	96.4	83.4	80.3	86.7
B04 KP	98.7	98.4	99.0	78.0	73.2	83.5

Table 5: Results of Blaheta and Charniak's model for function labelling, separated for syntactic and semantic labels. The feature trees (FT) and kernel perceptrons (KP) are optimised separately for the two different sets of labels. Results are calculated on the test set of the PTB.

The second informative observation derives from a comparison with results reported by Blaheta and Charniak's paper and in Blaheta's dissertation, shown in Table 5. As can be noticed by comparing the results of Model 1 (Table 4), our results are lower.

For all these reasons, we develop two other models to improve performance, concentrating in particular on improving recall, which is particularly poor. We will see that the more function labelling improves, the more the parser improves, reducing the distance from the level of performance of the parser without function labels (Table 3, FLBL-less column).

Model 2 Our first SSN parser was designed to discriminate only among constituents bearing syntactic or semantic labels, and did not discriminate those constituents bearing the NULL label. Our second parser was designed to make such a distinction.

In this model, we hypothesize that the label NULL (i.e. the conjunction of the SYN-NULL and

SEM-NULL labels) is a mixture of types, which will be more accurately learnt separately. As can be observed by the confusion matrix in Table 2, most of the confusion occurs between the function labels and the NULL. If the label NULL is learnt more precisely, the recall of the other labels is expected to increase.

The NULL label was split into the mutually exclusive labels CLR, OBJ and OTHER. Constituents were assigned the OBJ label according to the conditions stated in (Collins 99).[6] As a result, 52 non-terminal labels were added, yielding a total of 188 non-terminals.[7]

As can be observed from the results concerning Model 2 in tables 3 and 4, our hypothesis is weakly confirmed. However, while it is true that all performance indicators increase, our method is still not as good as other methods. We think performance could be improved even further by finer-grained modelling of function labels.

Model 3 We observe that SSNs tend to project NULL labels more than any other label. Since SSNs decide the syntactic label of a non-terminal at projection, this behaviour indicates that the parser does not have enough information at this point in the parse to project the correct function label. We hypothesize that finer-grained labelling will improve parsing performance. This observation is consistent with results reported in (Klein & Manning 03), who showed that tags occurring in the Treebank are not fine-grained enough to discriminate between preterminals. For example, the tag TO labels both the preposition *to* and the infinitival marker. Extending (Klein & Manning 03)'s technique to function labelling, we split some POS tags into tags marked with semantic function labels. More precisely, the function labels DIR, LOC, MNR, PRP or TMP attached to a non-terminal were propagated down to the POS tag of the head word of the non-terminal, provided that the non-terminal is projected from the POS tag of its head.[8] As a result, 83 new part-of-speech (POS) tags were introduced to partition the original tagset of the Treebank. The vocabulary consists of 819 tag-word pairs. The non-terminal label set also includes the labels CLR, OBJ and OTHER introduced with our second model.

To provide Model 3 with tagged input sentences, we trained an SVM tagger whose features and parameters are described in detail in (Gimenez & Marquez 04). Trained on section 2-21, the tagger reaches a performance of 95.8% on the test set (section 23) of the PTB using our new tag set. As can be observed from the results concerning Model 3 in tables 3 and 4, this experiment indicates that function labelling of non-terminal labels can be done very accurately, if the parser is provided finer-grained POS tags. Concerning function labels, notice that our performance is better than the model in (Blaheta & Charniak 00) on all accounts. This is the only model which is trained on the same set of features for syntactic and semantic labels, like our model. The specialised models, reported in Table 5, optimise either their input features or their parameters separately for syntactic or semantic labels. They perform a little better than our model on syntactic labels, while they do worse than our model on semantic labels. In particular, the very time-consuming kernel models (Table 4, B04 KP) do not seem to provide any interesting added value for semantic labels. Also, the differential between the parser outputting complex labels (FLBL, Table 3) and the parser evaluated only on the standard non-terminal labels (FLBL-less, Table 3) has considerably decreased. Furthermore, the resulting parser achieves state-of-the-art parsing performance (88.9% F-measure).

4 Discussion and Comparison to Related Work

The work reported in the previous sections is directly related to a small number of other pieces of work on function labelling (Blaheta & Charniak 00; Blaheta 04), and more indirectly on all the recent work on semantic role labelling, of which we discuss the few who have reported results on function labelling (Jijkoun & deRijke 04) or who discuss issues relevant to ours here (Gildea & Palmer 02; Punyakanok *et al.* 05). In work that predates

[6] Roughly, an OBJ non-terminal is an NP, SBAR or S whose parent is an S, VP or SBAR. Any such non-terminal must not bear either syntactic or semantic function labels, or the CLR label. In addition, the first child following the head of a PP is marked with the OBJ label.

[7] This second SSN also used all tag-word pairs which occurs at least 200 times in the training set (508).

[8] In most cases, this condition was implemented by requiring that the non-terminal immediately dominates the POS tag. This condition was relaxed in a few cases to capture constructs such as coordinated PPs (e.g. [PP-LOC[PP[IN *at*]...][CC *and*][PP[IN *in*]...]...] or infinitival clauses (e.g. [S-PRP[VP[TO *to*][VP[VB...]...]...]).

the availability of Framenet and Propbank and explores many issues that also apply to semantic role labelling, (Blaheta & Charniak 00) define the task of function labelling (that they refer to as the *function tagging* task) for the first time, and highlight its relevance for NLP. Their method is in two-steps. First, they parse the PTB using a state-of-the-art parser (Charniak 00). Then, they assign function labels using features from the local context, mostly limited to two levels up the tree and only one next label. (Blaheta 04) extends on this method by developing specialised feature sets for the different subproblems of function labelling and slightly improves the results, as reported in Table 5. (Jijkoun & deRijke 04) approach the problem of enriching the output of a parser in several steps. The first step applies memory-based learning to the output of a parser mapped to dependency structures. This step learns function labels. Only results for all function labels, and not for syntactic or semantic labels alone, are provided. Although they cannot be compared directly to our results, it is interesting to notice that they are slightly better in F-measure than Blaheta's (88.5% F-measure).

In comparing different models for function and semantic role labelling, very important properties of the model are the features used by the learner and the domain of locality these features define in the tree. Both (Blaheta & Charniak 00; Blaheta 04) and (Jijkoun & deRijke 04) find that lexical heads are very useful features. (Blaheta 04) finds in particular that the head of the PP-internal noun improves results considerably for semantic function labels, which are often assigned to PPs. This is in contrast to our results. Our SSN parsers do not incorporate any inductive bias towards phrasal heads. This design was chosen because heads were not found to be useful. An SSN parser with an explicit representation of phrasal head was designed to investigate this issue directly.[9] Results are slightly worse than Model 2 above, both for syntactic and for semantic labels. (F-measure of 95.7%, recall 94.9%, and precision 96.8% for syntactic labels, F-measure of 74.0%, recall of 72.7%, and precision 75.3% for semantic labels.) While this result is in contradiction with other methods, it confirms published results on the parser we use (Henderson 03), and is therefore to be interpreted as an inherent property of the learning and parsing regime. It appears then that head-lexicalisation is not as essential as it was thought, as confirmed also by recent findings in (Bikel 04) and (Dubey & Keller 04).

The other interesting area of comparison lies in the locality of the nodes that are available to the learner. Since other methods take parsed trees as their input, in principle, they have access to all nodes in the tree. This differs crucially from assigning labels while parsing, where in most cases the parser has access only to the current level of recursion and the nodes to the right of the current node are not yet available. In practice, (Blaheta & Charniak 00; Blaheta 04) make limited use of context, and use the next label only to predict syntactic function labels. The domain of locality is therefore limited, and it defines topologies in the tree similar to ours. Such constrained methods are needed for language modelling and interactive applications.

Finally, our results provide some new insights into the discussion about the necessity of parsing for function or semantic role labelling (Gildea & Palmer 02; Punyakanok *et al.* 05). Comparing semantic role labelling based on chunked input to the better semantic role labels retrieved based on parsed trees, (Gildea & Palmer 02) conclude that parsing is necessary. In an extensive experimental investigation of the different learning stages usually involved in semantic role labelling, (Punyakanok *et al.* 05) find instead that sophisticated chunking can achieve state-of-the-art results. However, they identify pre-labelling pruning as the stage in which parsing provides an improvement that even sophisticated chunking techniques are not able to match. Pruning eliminates the nodes that almost certainly will not bear a semantic role, thus simplifying role labelling. These results are coherent with our findings. Our last experiment indicates that function labels tend to be situated very low in the tree and that tag-splitting techniques do a large amount of the work, if appropriately exploited. This suggests that most of the information is available, in principle, to a chunker, albeit a sophisticated one that recognises some phrase-internal structure. However, we also find that most errors are misclassifications between nodes that bear a function label and those that do not, affecting recall in particular. This indicates that, although a parser

[9] This model, which we do not have space to describe in detail, implements two additional derivation moves that project or attach head children of a constituent.

can identify nodes that do not need a label better than a chunker, argument identification remains the most difficult aspect of the task to be performed based on local information. Future research will lie in improving this stage of function and semantic role labelling.

5 Conclusions and Future Work

This paper has tested the hypothesis that function labelling can be successfully performed while parsing. The main result of the paper indicates that information related to function labels lies in lower level of syntactic trees and can be accurately projected from fine-grained POS tags. Future work lies in using function labels as input for semantic role labelling. Consider the semantic role labels of PropBank. Semantic function labels are straightforward predictors of the ARGM labels. Syntactic function labels, such as SBJ or LGS, encode grammatical function explicitly, and are therefore less noisy predictors of argument labels (ARG0..ARG6 in PropBank) than the indirect encoding of grammatical functions, like subject or object provided by the commonly used feature *path*.

References

(Baker *et al.* 98) Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The Berkeley FrameNet project. In Christian Boitet and Pete Whitelock, editors, *Proceedings of the Thirty-Sixth Annual Meeting of the Association for Computational Linguistics and Seventeenth International Conference on Computational Linguistics (ACL-COLING'98)*, pages 86–90, Montreal, Canada, 1998. Morgan Kaufmann Publishers.

(Bies *et al.* 95) Ann Bies, M. Ferguson, K.Katz, and Robert MacIntyre. Bracketing guidelines for Treebank II style. Technical report, University of Pennsylvania, 1995.

(Bikel 04) Dan Bikel. Intricacies of collins' parsing model. *Computational Linguistics*, pages 479–511, 2004.

(Blaheta & Charniak 00) Don Blaheta and Eugene Charniak. Assigning function tags to parsed text. In *Proceedings of NAACL-00*, 2000.

(Blaheta 04) Don Blaheta. *Function Tagging*. Unpublished PhD thesis, Department of Computer Science, Brown University, 2004.

(Cahill *et al.* 04) Aoife Cahill, Michael Burke, Ruth O'Donovan, Josef van Genabith, and Andy Way. Long-distance dependency resolution in automatically acquired wide-coverage pcfg-based lfg approximations. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL'04)*, pages 320–327, Barcelona, Spain, 2004.

(Charniak 00) Eugene Charniak. A maximum-entropy-inspired parser. In *Proceedings of the 1st Meeting of North American Chapter of Association for Computational Linguistics*, pages 132–139, Seattle, Washington, 2000.

(Collins 99) Michael Collins. *Head-Driven Statistical Models for Natural Language Parsing*. Unpublished PhD thesis, Department of Computer Science, University of Pennsylvania, 1999.

(CoNLL 04) CoNLL. Eighth conference on computational natural language learning (conll-2004). http://cnts.uia.ac.be/conll2004, 2004.

(Dubey & Keller 04) Amit Dubey and Frank Keller. Probabilistic parsing for german using sister-head dependencies. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 96–103, Sapporo,Japan, 2004.

(Gildea & Jurafsky 02) Daniel Gildea and Daniel Jurafsky. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3), 2002.

(Gildea & Palmer 02) Daniel Gildea and Martha Palmer. The necessity of parsing for predicate argument recognition. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, pages 239–246, Philadelphia, PA, 2002.

(Gimenez & Marquez 04) Jesus Gimenez and Lluis Marquez. Svmtool: A general pos tagger generator based on support vector machines. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal, 2004.

(Henderson 03) Jamie Henderson. Inducing history representations for broad-coverage statistical parsing. In *Proceedings of the Joint Meeting of the North American Chapter of the Association for Computational Linguistics and the Human Language Technology Conference (NAACL-HLT'03)*, pages 103–110, Edmonton, Canada, 2003.

(Jijkoun & deRijke 04) Valentin Jijkoun and Maarten de Rijke. Enriching the output of a parser using memory-based learning. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Barcelona,Spain, 2004.

(Kingsbury & Palmer 02) Paul Kingsbury and Martha Palmer. From TreeBank to PropBank. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC2002)*, Las Palmas, Spain, 2002.

(Klein & Manning 03) Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the ACL*, pages 423–430, Sapporo, Japan, 2003.

(Marcus *et al.* 93) Mitch Marcus, Beatrice Santorini, and M.A. Marcinkiewicz. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19:313–330, 1993.

(Nederhof 94) Mark Jan Nederhof. *Linguistic Parsing and Program Transformations*. Unpublished PhD thesis, Department of Computer Science, University of Nijmegen, 1994.

(Nielsen & Pradhan 04) Rodney Nielsen and Sameer Pradhan. Mixing weak learners in semantic parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2004)*, pages 80–87, Barcelona, Spain, July 2004.

(Punyakanok *et al.* 05) V. Punyakanok, D. Roth, and W. Yih. The necessity of syntactic parsing for semantic role labeling. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI'05)*, 2005.

(Ratnaparkhi 96) Adwait Ratnaparkhi. A maximum entropy part-of-speech tagger. In *Proceedings of the First Conference on Empirical Methods in Natural Language Processing*, pages 133–142, Philadelphia, PA, 1996.

(Riezler *et al.* 02) Stefan Riezler, Tracy H. King, Ronald M. Kaplan, Richard Crouch, John T. Maxwell, and Mark Johnson. Parsing the wall street journal using a lexical-functional grammar and discriminative estimation techniques. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, pages 271–278, Philadephia, PA, 2002.

(Senseval 04) Senseval. Third international workshop on the evaluation of systems for the semantic analysis of text (acl 2004). http://www.senseval.org/senseval3, 2004.

(Stallard 00) David Stallard. Talk'n'travel: A conversational system for air travel planning. In *Proceedings of the Sixth Applied Natural Language Processing Conference (ANLP'00)*, pages 68–75, 2000.

(Surdeanu *et al.* 03) Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. Using predicate-argument structures for information extraction. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, 2003.

(Xue & Palmer 04) Nianwen Xue and Martha Palmer. Calibrating features for semantic role labeling. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 88–94, 2004.

A Study of Using Search Engine Page Hits as a Proxy for n-gram Frequencies

Preslav Nakov and **Marti Hearst**
EECS and SIMS
University of California at Berkeley
Berkeley, CA 94720
nakov@cs.berkeley.edu, hearst@sims.berkeley.edu

Abstract

The idea of using the Web as a corpus for linguistic research is getting increasingly popular. Most often this means using Web search engine page hit counts as estimates for n-gram frequencies. While the results so far have been very encouraging, some researchers worry about what appears to be the instability of these estimates. Using a particular NLP task, we compare the variability in the n-gram counts across different search engines as well as for the same search engine across time, finding that although there are measurable differences, they are not statistically significantly different for the task examined.

1 Introduction

In 2001, (Banko & Brill 01) advocated for the creative use of very large text collections as an alternative to sophisticated algorithms and hand-built resources. They demonstrated the idea on a lexical disambiguation problem for which labeled examples are available "for free". The problem was to choose which of 2-3 commonly confused words (e.g., {*principle, principal*}) were appropriate for a given context. The labeled data was "free" because the authors could safely assume that in the carefully edited text in their training set the words are used correctly. They show that even using a very simple algorithm, the results continue to improve log-linearly with more training data, even out to a billion words. They conclude that getting more data may be a better idea than fine-tuning algorithms. Today, the obvious source of very large data is the Web.

Using the Web as a training and testing corpus is attracting ever-increasing attention. In 2003 the journal *Computational Linguistics* had a special issue (Kilgariff & Grefenstette 03), and in 2005 the Corpus Linguistics conference includes a special workshop on the Web as Corpus. The Web has been used as a corpus for a variety of NLP tasks including, but not limited to: machine translation (Grefenstette 98; Resnik 99; Cao & Li 02; Way & Gough 03), question answering: (Dumais *et al.* 02; Soricut & Brill 04), word sense disambiguation (Mihalcea & Moldovan 99; Rigau *et al.* 02; Santamaría *et al.* 03; Zahariev 04), extraction of semantic relations, (Chklovski & Pantel 04; Idan Szpektor & Coppola 04; Shinzato & Torisawa 04), anaphora resolution: (Modjeska *et al.* 03), prepositional phrase attachment: (Volk 01; Calvo & Gelbukh 03), language modeling: (Zhu & Rosenfeld 01; Keller & Lapata 03), and so on.

Despite the variability of applications, the most popular use of the Web as a corpus is as a means to obtain page hit counts as an estimate for n-gram word frequencies. (Keller & Lapata 03) demonstrate a high correlation between page hits and corpus bigram frequencies, as well as between page hits and plausibility judgments. They propose using Web counts as a baseline unsupervised method for many NLP tasks and experimented with eight NLP problems (machine translation candidate selection, spelling correction, adjective ordering, article generation, noun compound bracketing, noun compound interpretation, countability detection and prepositional phrase attachment), and show that variations on n-gram counts often perform nearly as well as more elaborate methods (Lapata & Keller 05). More recently, we have shown that the Web has the potential for more than just a baseline. Using various Web-derived surface features, in addition to paraphrases and n-gram counts, we demonstrated state-of-the-art results on the task of noun compound bracketing (Nakov & Hearst 05).

2 Problems and Limitations

Web search engines provide a convenient way for researchers to obtain statistics over an enormous corpus, but using them for this purpose is not without drawbacks.

First, there are limitations on what kinds of queries can be issued, mainly because of the lack of linguistic annotation. For example, if we want to estimate the probability that *health* precedes *care*: $\frac{\#(''health\ care'')}{\#(care)}$, we need the frequencies of

"health care" and *care*, where both words are nouns. The problem is that a query for *care* only will return many pages where it is used as a verb, while in *health care* it would nearly always occur as a noun. Even when both *health* and *care* are used as nouns and are adjacent, they may belong to different NPs but sit next to each other only by chance. Furthermore, since search engines ignore punctuation characters, the two nouns may also come from different sentences.

Other Web search engine restrictions prevent querying directly for terms containing hyphens or possessive markers such as *amino-acid sequence* and *protein synthesis' inhibition*. They also disallow querying for a term like *bronchoalveolar lavage (BAL) fluid*, which contains an internal parenthesized abbreviation. They also do not support queries that make use of generalized POS information such as

stem cells VERB PREP DET brain

in which the uppercase patterns stand for any verb, any preposition and any determiner, e.g., *stem cells derived from the brain*.

Furthermore, using page hits as a proxy for n-gram frequencies can produce some counter-intuitive results. Consider the bigrams w_1w_4, w_2w_4 and w_3w_4 and a page that contains each bigram exactly once. A search engine will contribute a page count of 1 for w_4 instead of a frequency of 3; thus the number of page hits for w_4 can be smaller than that for the sum of the bigrams that contain it. See (Keller & Lapata 03) for more potential problems with page hits.

Another potential problem is instability of the n-gram counts. Today Web search engines are too complex to be run on a single machine, and instead the queries are served by hundreds, sometimes thousands of servers, which collaborate to produce the final result. In addition, the Web is dynamic, since at any given time some pages disappear, some appear for the first time, and some change frequently. Thus search engines need to update their indexes frequently, and in fact the different engines compete on how "fresh" their indexes are. As a result, the number of page hits for a given query changes over time in unpredictable ways.

The indexes themselves are too big to be stored on a single machine and so are spread across multiple ones (Brin & Page 98). For availability and efficiency reasons, there are also multiple copies of the same part of the index, and these are not always synchronized with one another since the different copies are updated at different times. As a result, if we issue the same query multiple times in rapid succession, we may connect to different physical machines and get different results. This is known as search engine "dancing".

From a research perspective, "dancing" and dynamics over time are potentially undesirable, as they preclude the exact replicability of any results obtained using search engines. At best, one could reproduce the same initial conditions, and expect similar outcomes.

Another potentially undesirable aspect of using Web search engines is that two of the major ones (Google and Yahoo) do not provide exact numbers of page hits, but instead show rounded estimates. For example, at the moment of preparation of this paper, Google returns 79,000,000 page hits for the exact phrase query *"search engine"*, and Yahoo Search returns 127,000,000. Google and Yahoo provide exact numbers of page hits only in case these numbers are relatively small. MSN Search, by contrast, does not round its page hits, and it returns 46,502,549 for the "search engine" query. This rounding is probably done because for most users' purposes, exact counts are not necessary once the numbers get somewhat large, and computing the exact numbers is expensive if the index is distributed and continually changing. It might also indicate that under high load search engines sample from their indexes, rather than performing an exact computation.

It is unclear what the implications of these inconsistencies are on using the Web to obtain n-gram frequencies. If the estimates are close to accurate and consistent across queries, this should not have a big impact for most applications, since they only need the ratios of different n-grams.

We decided that the best way to determine the impact of rounding and inconsistencies was to design a suit of experiments organized around a real NLP task. We chose noun compound bracketing, which, while being a simple task, can be solved using several different methods which make use of n-grams of different lengths. In the next two sections we first describe the noun compound bracketing problem, and then report the results of comparative experiments on this problem.

3 Noun Compound Bracketing

Consider the following contrastive pair of noun compounds:

(1) *liver cell antibody*
(2) *liver cell line*

In example (1) an *antibody* targets a *liver cell*, while (2) refers to a *cell line* which is derived from the *liver*. Although equivalent at the part of speech (POS) level, these two noun compounds have different syntactic trees. The distinction can be represented as a binary tree or, equivalently, as a binary bracketing:

(1b) [[*liver cell*] *antibody*] (left bracketing)
(2b) [*liver* [*cell line*]] (right bracketing)

3.1 Unigrams and Bigrams

The problem of choosing the correct bracketing has been traditionally addressed using unigram and bigram frequencies (Marcus 80; Pustejovsky *et al.* 93; Resnik 93; Lauer 95; Lapata & Keller 05). In related work, a distinction is often made between what is called the *dependency model* and the *adjacency model* (Lauer 95). The main idea is as follows. For a given 3-word NC $w_1w_2w_3$, there are two reasons it may take on right bracketing, $[w_1[w_2w_3]]$. Either (a) w_2w_3 is a compound (modified by w_1), or (b) w_1 and w_2 independently modify w_3. This distinction can be seen in the examples *home health care* (*health care* is a compound modified by *home*) versus *adult male rat* (*adult* and *male* independently modify *rat*).

The adjacency model checks (a), whether w_2w_3 is a compound (i.e., how strongly w_2 modifies w_3 as opposed to w_1w_2 being a compound) to decide whether or not to predict a right bracketing. The dependency model checks (b) whether w_1 modifies w_3 (as opposed to w_1 modifying w_2).

Adjacency and dependency could be computed via frequencies, but we can also use probabilities. Let $\Pr(w_i \rightarrow w_j|w_j)$ be the probability that the word w_i precedes a given word w_j. So in a dependency model we can compare $\Pr(w_1 \rightarrow w_3|w_3)$ to $\Pr(w_1 \rightarrow w_2|w_2)$. The adjacency model compares $\Pr(w_2 \rightarrow w_3|w_3)$ to $\Pr(w_1 \rightarrow w_2|w_2)$, i.e., the association strength between the last two words vs. that between the first two. If the first probability is larger than the second one, the model predicts right bracketing.

The probability $\Pr(w_1 \rightarrow w_2|w_2)$ can be estimated as $\#(w_1, w_2)/\#(w_2)$, where $\#(w_1, w_2)$ and $\#(w_2)$ are the corresponding bigram and unigram frequencies. They can be approximated as the number of pages returned by a search engine in response to queries for the exact phrase "w_1 w_2" and for the word w_2. In our experiments below we smoothed[1] each of the frequencies by adding 0.5 to avoid problems caused by nonexistent n-grams.

In both models, $\Pr(w_i \rightarrow w_j|w_j)$ can be replaced by some (possibly symmetric) measure of association between w_i and w_j. Below we use *Chi squared* (χ^2) and mutual information (MI). See (Nakov & Hearst 05) for details on how to compute χ^2 on the Web.

3.2 Longer n-grams

Since the Web is a very big corpus, we can hope to obtain reliable estimates for longer n-grams too. Below we list some other kinds of statistics that can be computed from the Web that we have found helpful in other work (Nakov & Hearst 05), and that are used in the experiments in the next section.

First, the genitive ending, or *possessive* marker, can be a useful indicator. The phrase *brain's stem cells* suggests a right bracketing for *brain stem cells*, while *brain stem's cells* favors a left bracketing. In some cases, we can query for this directly: although search engines drop the apostrophe, they keep the *s*, so we can query for *"brain's"* (but not for *"brains' "*). We then compare the number of times the possessive marker appeared on the second versus the first word, to make a bracketing decision.

Abbreviations are another important feature. For example, *"tumor necrosis factor (NF)"* suggests a right bracketing, while *"tumor necrosis (TN) factor"* would favor left. We would like to issue exact phrase queries for the two potential abbreviation patterns and see which one is more frequent. Unfortunately, the search engines drop the brackets and ignore the capitalization, so we issue queries with the parentheses removed, as in *"tumor necrosis factor nf"*. This produces highly accurate results, although errors occur when the abbreviation is an existing word (e.g., *me*), a state (e.g., *CA*), a Roman digit (e.g., *IV*), etc.

Another reliable feature is *concatenation*. Consider the NC *health care reform*, which is left-bracketed. Now, consider the bigram *"health*

[1] Zero counts sometimes happen for $\#(w_1, w_3)$, but are rare in general for unigrams and bigrams on the Web, and there is no need for a more sophisticated smoothing.

care". At the time of writing, Google estimates 80,900,000 pages for it as an exact term. Now, if we try the word *healthcare* we get 80,500,000 hits. At the same time, *carereform* returns just 109. This suggests that authors sometimes concatenate words that act as compounds. We find below that comparing the frequency of the concatenation of the left bigram to that of the right (adjacency model for concatenations) often yields accurate results. We also try the dependency model for concatenations, as well as the concatenations of two words in the context of the third one (i.e., compare frequencies of *"healthcare reform"* and *"health carereform"*).

Further, we try to look inside the *internal inflection variability*. The idea is that if *"tyrosine kinase activation"* is left-bracketed, then the first two words probably make a whole and thus the second word can be found inflected elsewhere but the first word cannot, e.g., *"tyrosine kinase**s** activation"*. Alternatively, if we find different internal inflections of the first word, this would favor a right bracketing.

Finally, we try switching the word order of the first two words. If they independently modify the third one (which implies a right bracketing), then we could expect to see also a form with the first two words switched, e.g., if we are given *"adult male rat"*, we would also expect *"male adult rat"*.

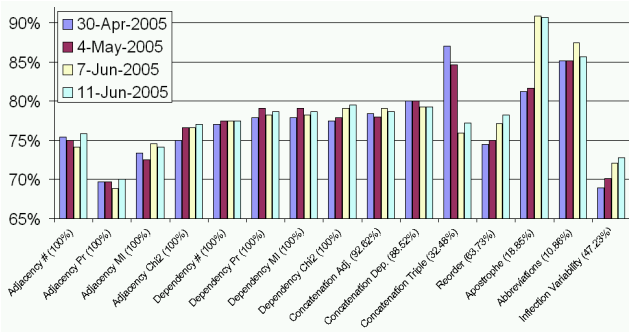


Figure 1: **Comparison over time for Google.** Precision for any language, no inflections. Average recall is shown in parentheses.

4 Experiments and Results

We performed series of experiments comparing the accuracy of the methods described above across four dimensions: (1) *search engine* (Google vs. Yahoo vs. MSN), (2) *time*, (3) *language filter*

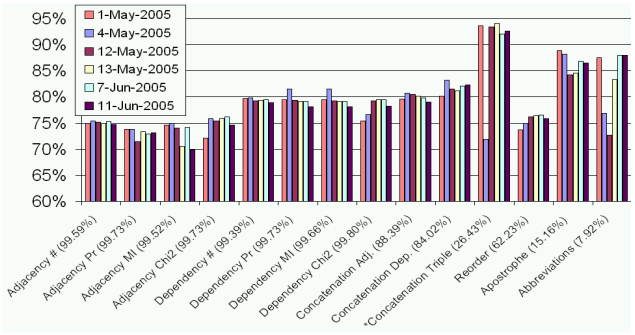


Figure 2: **Comparison over time for MSN Search.** Precision for any language, no inflections. Average recall is shown in parentheses.

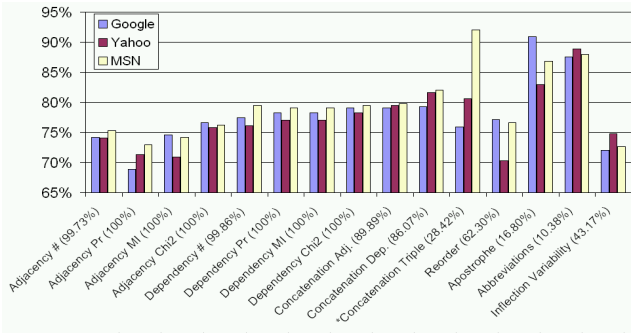


Figure 3: **Comparison by search engine.** *Precision* (in %) for any language, no inflections. All results are for 6/6/2005. Average recall is shown in parentheses.

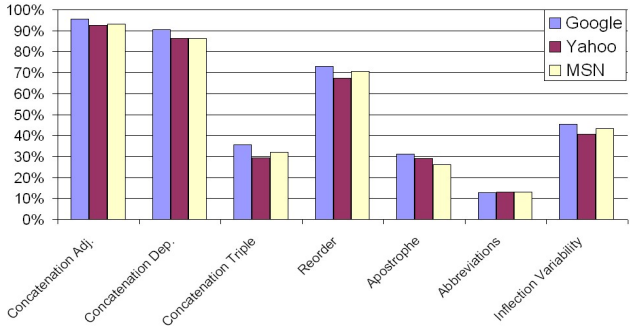


Figure 4: **Comparison by search engine.** *Recall* (in %) for any language, no inflections. All results are for 6/6/2005.

(English only vs. any), and (4) *inflected word-forms* usage.

In these experiments we compared the results using the Chi squared test for statistical significance as computed by (Lapata & Keller 05). In nearly every case we found that the differences were not statistically significant. The only exceptions are observed for *concatenation triple* in tables 2 and 3 (marked with a *).

We experimented with the dataset from (Lauer 95), in order to produce results comparable to those of both Lauer and Keller & Lapata. The set consists of 244 unambiguous 3-word noun compounds extracted from *Grolier's encyclopedia*; however, only 216 of these NCs are unique.

(Lauer 95) derived n-gram frequencies from the *Grolier's* corpus and tested the dependency and the adjacency models using this text. To help combat data sparseness issues he also incorporated a taxonomy and some additional information.

At the time of writing, the Google search engine reportedly indexes over 8 billion pages, i.e., about 8 trillion words, which is about 80,000 times the size of the British National Corpus (100 million words), thus confirming it as a gateway to a very large corpus. We were unable to find official information about the sizes of Yahoo and MSN Search, but they probably index a similar number of pages. When still in Beta version, MSN announced indexing over 5 billion pages.

For all n-grams, we issued exact phrase queries within a single day. Unless otherwise stated, the queries were not inflected and no language filter was applied. We used a threshold of 5 for the difference between the left- and the right-predicting n-gram frequencies: we did not make a decision when the module of that difference was below that threshold. This slightly lowers the recall but potentially increases the precision.

Figures 1 and 2 show the variability over time for Google and for MSN Search respectively. (As Yahoo behaves similarly to Google, it is omitted here due to space limitations.) We chose time samples at varying time intervals in an attempt to capture index changes, in case they happen in the same fixed time intervals. For Google (see Figure 1), we observe a low variability in the adjacency- and dependency-based models and a more sizable variability for the other models and features. The variability is especially high for *apostrophe* and *concatenation triple*: while in the first two time snapshots the precision of the apostrophes is much lower than in the last two, it is the reverse for concatenation.

MSN Search exhibits a more uniform behavior overall (see Figure 2), however while the variability in the adjacency- and dependency-based models is still a little bit lower than that of the last five features, it is bigger than Google's. We think that this is due to the rounding: because Google's counts are rounded, they change less over time, especially for very large counts. By contrast, these counts are exact for MSN Search, which makes its unigram and bigram counts more sensitive to variation. For the higher order n-grams, both engines exhibit a higher variability: these counts are smaller, and so are more likely to be represented by exact numbers in Google, and thus they are also more sensitive to index updates for both search engines. However, the difference between the precision for May 4, 2005 and that for the other five dates is statistically significant for MSN Search only.

Figure 3 compares the three search engines at the same fixed time point. The biggest difference in precision is exhibited by *concatenation triple* which in MSN Search achieves a precision of 92%, which is better than the others' by 11% (statistically significant). Other large variations (not statistically significant) are seen in *apostrophe*, *reorder*, and to a lesser extent in the adjacency- and dependency-based models. As we expected, MSN Search looks best overall (especially on the unigram- and bigram-based models), which we attribute to the better accuracy of its n-gram estimates. Google is almost 5% ahead of the others on *apostrophes* and *reorder*. Yahoo leads on *abbreviations* and *inflection variability*. The fact that different search engines exhibit strength on different kinds of queries and models shows the potential of combining them: in a majority vote combining some of the best models, we would choose *concatenation triple* from MSN Search and *apostrophe* from Google and *abbreviations* from Yahoo (together with *concatenation dependency*, χ^2 *dependency* and χ^2 *adjacency*). Figure 4 shows the corresponding recall for some of the methods (it is about 100% for the rest). We can see that Google exhibits a slightly higher recall, which suggests it might have a bigger index compared to Yahoo and MSN Search.

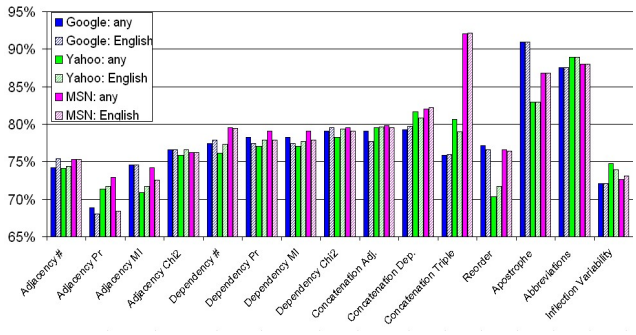


Figure 5: **Comparison by search engine: any language vs. English.** *Precision* shown in %, no inflections. All results are for 6/6/2005.

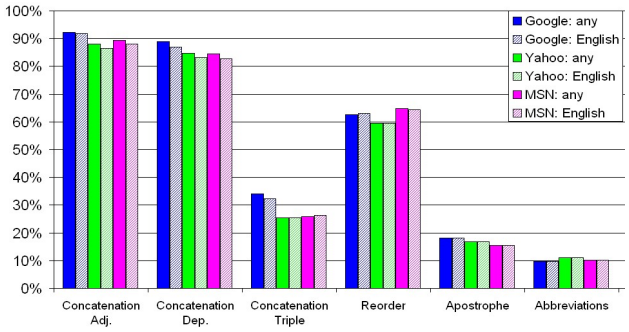


Figure 6: **Comparison by search engine: any language vs. English.** *Recall* shown in %, no inflections. All results are for 6/6/2005.

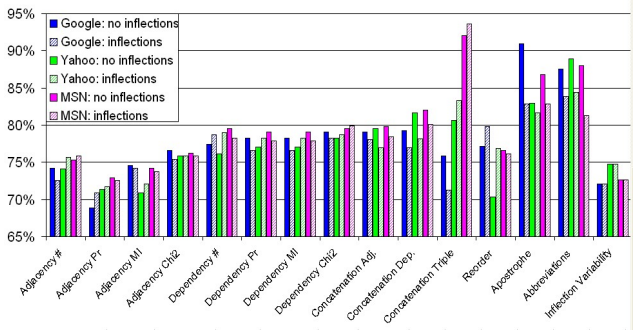


Figure 7: **Comparison by search engine: no inflections vs. using inflections.** *Precision* shown in %, any language. All results are for 6/6/2005.

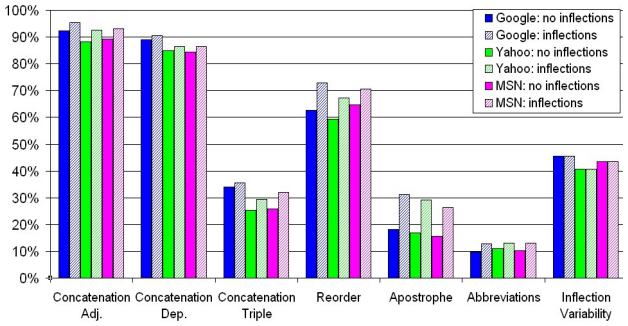


Figure 8: **Comparison by search engine: no inflections vs. using inflections.** *Recall* shown in %, any language. All results are for 6/6/2005.

Figure 5 compares, on a fixed date (6/6/2005), for all the three search engines the impact of language filtering, meaning requiring only documents in English versus no restriction on language. The impact of the language filter on the precision seems minor and inconsistent for all three search engines: sometimes the results are improved slightly and sometimes they are negatively impacted. Figure 6 compares the corresponding recall for some of the models (the rest are omitted as the recall for them is about 100%). As we can see, using English only leads to a drop in recall, as one could expect, but this drop is small.

Finally, Figure 7 compares for the three search engines the impact of using inflections[2]. When we estimate the frequency of a word, e.g., *tumor*, we also add up the frequencies of all possible variants, e.g., *tumors*, *tumour*, *tumours*. For bigrams, we inflect only the second word, and for n-grams only the last one. The results are again mixed, but the impact on precision is more significant compared to that of the language filter, especially on the high-order n-grams (of course, there is no impact on *inflection variability*). Figure 8 compares the corresponding recall for some of the models (for the rest it is about 100%). As one would expect, the recall goes up when using inflection. The change for *apostrophe*, *reorder* and *concatenation triple* is again the biggest.

5 Conclusions and Future Work

Using a real NLP task, we have shown that effects of variability over time and across search engines,

[2]We made use of Carroll's morphological tools: http://www.cogs.susx.ac.uk/lab/nlp/carroll/morph.html.

as well as using language filters and morphologically inflected wordforms, do not significantly effect the results of an NLP application and thus do not greatly impact the interpretation of results obtained using Web-derived n-gram frequencies.

In order to further bolster these results we will need to perform similar studies for other NLP tasks, which make use of Web-derived n-gram estimates. We would also like to run similar experiments for languages other than English, where the language filter could be much more important, and where the impact of the inflection variability may differ, especially in case of a morphologically rich language.

Acknowledgements This research was supported by NSF DBI-0317510, and a gift from Genentech.

References

(Banko & Brill 01) Michele Banko and Eric Brill. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of ACL*, 2001.

(Brin & Page 98) Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117, 1998.

(Calvo & Gelbukh 03) Hiram Calvo and Alexander Gelbukh. Improving prepositional phrase attachment disambiguation using the web as corpus. In *Progress in Pattern Recognition, Speech and Image Analysis: 8th Iberoamerican Congress on Pattern Recognition, CIARP 2003*, 2003.

(Cao & Li 02) Yunbo Cao and Hang Li. Base noun phrase translation using web data and the EM algorithm. In *COLING*, pages 127–133, 2002.

(Chklovski & Pantel 04) Timothy Chklovski and Patrick Pantel. VerbOcean: Mining the web for fine-grained semantic verb relations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 33–40, 2004.

(Dumais *et al.* 02) Susan Dumais, Michele Banko, Eric Brill, Jimmy Lin, and Andrew Ng. Web question answering: Is more always better? In *Proceedings of SIGIR*, pages 291–298, 2002.

(Grefenstette 98) Gregory Grefenstette. The world wide web as a resourcefor example-based machine translation tasks. In *Proceedings of the ASLIB Conference on Translating and the Computer.*, 1998.

(Idan Szpektor & Coppola 04) Ido Dagan Idan Szpektor, Hristo Tanev and Bonaventura Coppola. Scaling web-based acquisition of entailment relations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 41–48, 2004.

(Keller & Lapata 03) Frank Keller and Mirella Lapata. Using the Web to obtain frequencies for unseen bigrams. *Computational Linguistics*, 29:459–484, 2003.

(Kilgariff & Grefenstette 03) Adam Kilgariff and Gregory Grefenstette. Introduction to the special issue on the web as corpus. *Computational Linguistics*, 29(3):333–347, 2003.

(Lapata & Keller 05) Mirella Lapata and Frank Keller. Web-based models for natural language processing. *ACM Transactions on Speech and Language Processing*, 2005.

(Lauer 95) Mark Lauer. *Designing Statistical Language Learners: Experiments on Noun Compounds.* Unpublished PhD thesis, Department of Computing Macquarie University NSW 2109 Australia, 1995.

(Marcus 80) Mitchell Marcus. *A Theory of Syntactic Recognition for Natural Language.* MIT Press, 1980.

(Mihalcea & Moldovan 99) Rada Mihalcea and Dan Moldovan. A method for word sense disambiguation of unrestricted text. In *ACL*, pages 152–158, 1999.

(Modjeska *et al.* 03) Natalia Modjeska, Katja Markert, and Malvina Nissim. Using the web in machine learning for other-anaphora resolution. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 176–183, 2003.

(Nakov & Hearst 05) Preslav Nakov and Marti Hearst. Search engine statistics beyond the n-gram: Application to noun compound bracketing. In *Proceedings of CoNLL-2005, Ninth Conference on Computational Natural Language Learning*, 2005.

(Pustejovsky *et al.* 93) James Pustejovsky, Peter Anick, and Sabine Bergler. Lexical semantic techniques for corpus analysis. *Computational Linguistics*, 19(2):331–358, 1993.

(Resnik 93) Philip Resnik. *Selection and information: a class-based approach to lexical relationships.* Unpublished PhD thesis, University of Pennsylvania, UMI Order No. GAX94-13894, 1993.

(Resnik 99) Philip Resnik. Mining the web for bilingual text. pages 527–534, 1999.

(Rigau *et al.* 02) German Rigau, Bernardo Magnini, Eneko Agirre, and John Carroll. Meaning: A roadmap to knowledge technologies. In *Proceedings of COLING Workshop on A Roadmap for Computational Linguistics*, 2002.

(Santamaría *et al.* 03) Celina Santamaría, Julio Gonzalo, and Felisa Verdejo. Automatic association of web directories with word senses. *Computational Linguistics*, 29(3):485–502, 2003.

(Shinzato & Torisawa 04) Keiji Shinzato and Kentaro Torisawa. Acquiring hyponymy relations from web documents. In *Proceedings of HLT-NAACL*, pages 73–80, 2004.

(Soricut & Brill 04) Radu Soricut and Eric Brill. Automatic question answering: Beyond the factoid. In *Proceedings of HLT-NAACL*, pages 57–64, 2004.

(Volk 01) Martin Volk. Exploiting the WWW as a corpus to resolve PP attachment ambiguities. In *Proceedings of Corpus Linguistics*, 2001.

(Way & Gough 03) Andy Way and Nano Gough. wEBMT: developing and validating an example-based machine translation system using the world wide web. *Computational Linguistics*, 29(3):421–457, 2003.

(Zahariev 04) Manuel Zahariev. *A (Acronyms).* Unpublished PhD thesis, School of Computing Science, Simon Fraser University, USA, 2004.

(Zhu & Rosenfeld 01) Xiaojin Zhu and Ronald Rosenfeld. Improving trigram language modeling with the world wide web. In *Proceedings of ICASSP*, pages I:533–536, 2001.

Building a Cross-lingual Referential Knowledge Database using Dictionaries

Shigeko Nariyama*˙, Takaaki Tanaka*, Eric Nichols$^{\upsilon}$, Francis Bond*, Hiromi Nakaiwa*

Communication Science Lab, NTT Kyoto, Japan

*{shigekon, takaaki, bond, hiromi}@cslab.kecl.ntt.co.jp

˙ The University of Melbourne, Australia shigeko@unimelb.edu.au

$^{\upsilon}$ Nara Institute of Science and Technology, Nara, Japan eric-n@is.naist.jp

Abstract

Referential knowledge is vital for resolving various problems in NLP, such as anaphora resolution. For example, we have the referential knowledge that *diagnose* is most likely a member of the referential relation '*doctor* diagnose *patient's illness'*. Nariyama et al. (2005) presented an inventory of such referents as *doctor,* collected from Japanese dictionary definition sentences. Such referential information is based on world knowledge and is applicable across languages. This paper describes our work using the inventory to build a cross-lingual referential database for multilingual applications.

1 Introduction

Natural language can be highly ambiguous. Utterances tend to avoid repeating information that is deducible from context or world knowledge. Furthermore, individual words include multiple senses (as opposed to being limited to one sense per word).

Various problems in NLP deriving from these ambiguities, such as anaphora resolution and word sense disambiguation, have been known to be prohibitively difficult to solve. The difficulties lie in the fact that the resolutions of these problems rely heavily on contextual information and world knowledge, for which even the state of the art in NLP cannot adequately account.

Nonetheless, words contain in their lexical semantics a large amount of inferences and entailments. When we hear words, we tend to make a strong association with certain referents. For example, the word *diagnose* prototypically appears in the referential relation, '*doctor* diagnose *patient's illness'*. With the word *diagnose* alone, we strongly associate two referents: one referent that is *doctor* as the subject of the sentence and another *patient's illness* as its object. Similarly, *arrest* prototypically has the referential relation as '*police* arrest *criminal*'. We refer to such relations of referents with a predicate as *referential knowledge.*

We contend that referential knowledge is a kind of contextual information or world knowledge, and it can be extracted from dictionary definition sentences. As such, referential knowledge captures referential relations of words based on heuristic and provides what we term 'representative arguments'.

Nariyama et al. (2005) presented an inventory of such representative arguments as *doctor* and *patient's illness* for *diagnose,* and *police* and *criminal* for *arrest,* collected from Japanese dictionary definition sentences. This referential knowledge makes a great contribution to resolving the aforementioned problems in NLP, such as zero pronoun resolution for languages, including Japanese and Chinese, that do not verbalise many referents (Isozaki and Hirano 2003, Nariyama 2003).

Since much of such referential information has its basis in world knowledge, it is quite possible to apply them across languages. In other words, if a language has an equivalent word of *arrest,* then the language is likely to use it with two referents: *police or person with a related authority* as the agent and *criminal or person with a suspect of crime* as the patient.

Arrest = arrestation (French), anhalten (German),
арестование (Russian),
검거 (Korean), 拘捕(Chinese), …

This paper describes our work using this inventory of representative arguments towards building a cross-lingual referential knowledge database that can be used in various languages. This database will save an enormous amount of work by eliminating the necessity to go through various steps for extracting representative arguments for each language. Moreover, the transfer of representative arguments to other languages automatically creates links among the languages, which is useful for multilingual applications.

Section 2 reviews the work on the inventory of the Japanese representative arguments (Nariyama et al. 2005). Section 3 examines the feasibility and the methods of building a cross-lingual referential knowledge database using the inventory. Section 4 describes related research, followed by Conclusions.

2 Inventory of representative arguments

Nariyama et al. (2005) presented an inventory of representative arguments collected from the Japanese semantic database, Lexeed (Bond et al. 2004). This is a hand-built self-contained lexicon, consisting of words and their definitions for the most familiar 28,000 words, as measured by native speakers, comprising a total of 46,347 different senses. This set is large enough to include most basic level words and covers 72.2% of the words in a typical Japanese newspaper.

Lexeed has been enhanced by manual word sense disambiguation of all the open class words. Furthermore, the senses are linked in an ontology (Nichols et al. 2005), which allows us to measure the semantic distance between words or senses using a variety of methods.

We see several advantages in using dictionary definition sentences for collecting referential knowledge. Dictionaries are created to provide information about words from cross-domain in lay terms with little contextual information to be comprehensible, while often providing world knowledge as well. For example, Lexeed provides the following definition about the word *taiho* 逮捕 'arrest', whereby we extract the referential information *police officer* and *criminal*. These extracted referents are 'representative arguments', prototypical examples of the real-world referents that are likely to fill the argument slots.

(1) *Taiho: Keisatsu ga hannin o toraeru koto.*
逮捕: 警察が、犯人 を 捕える こと 。
'Arrest: A police officer captures a criminal.'

It is a fact about the real-world that things like *police* are likely to be the subject of the verb *arrest* and things like *criminal* (or *someone with a suspect of crime*) are likely to be its object. These representative arguments can be used as the basis for selectional preferences, which allow room for any rhetorical and other deviated usages.

In general, we should prefer an interpretation where the referents of the arguments are semantically similar to the representative arguments. Because arguments only have to be similar, not subsumed by, it is possible for the representative arguments to be actual words, although word senses would be preferred.

In contrast, processing using selectional restrictions must use broader semantic classes, otherwise non-typical sentences would be rejected. For example, *Goi-Taikei*'s valency dictionary (Ikehara et al. 1997) has the semantic classes *agent* and *person* as selectional restrictions for *taiho* 'arrest'. These semantic classes are derived by most research (see Section 4). These subsume the words *police* and *criminal* but are much less informative.

2.1 Process of extracting arguments

We created an inventory of the representative arguments that are more specific than what is available in *Goi-Taikei (GT)*, currently the most informative resource available in Japanese. The process involved:

1) Automatic extraction of the representative arguments of definition words (i.e. words being defined) that are predicates (i.e. verbs, verbal nouns, and adjectives) from definition sentences in Lexeed, using both Shallow and Deep parsing;
2) Hand-selecting representative arguments from those extracted to make a reliable list;
3) Selecting only those that are more specific than what is provided by *GT*.

Deep parsing (DP) gives us the information we want immediately, but only for those sentences that can be parsed. Shallow parsing (SP), on the other hand, allows us to extract information from more data, but with less precision. For the optimal results, we combined DP and SP to extract arguments for greater quality and quantity. This technique of combining DP and SP has been proposed in the Deep Thought project and proved to be effective (Nichols et al. 2005, inter alia). For DP, we used a combination of the PET parsing system (Callmeier 2002) and the JaCY Japanese HPSG grammar (Siegel and Bender 2002).[1]

2.2 Results

The total number of extracted arguments was 10,076. Of these 6,550 (65.0%) were manually verified as representative arguments that are more specific than those in *Goi-Taikei* or new to *Goi-Taikei*. Table 1 gives the precision (the rate of representative arguments extracted over total extraction) per POS and parsing method. The results are promising.

[1] PET is an open source, highly efficient unification parser. JaCY is broad-coverage, freely available HPSG grammar that produces semantic analysis in Robust Minimal Recursion Semantics (RMRS, Frank 2004). See Nariyama et al. (2005) for detail.
PET can be downloaded at: http://wiki.delph-in.net/moin/PetTop

	Adjective	Verb	Verbal N	All
DP only	69.3%	76.6%	72.8%	74.1%
SP only	49.9%	63.7%	49.9%	55.3%
Extracted by Both	56.8%	72.4%	72.9%	70.0%
Total (number)	**57.8%** (841/1455)	**71.5%** (3041/4252)	**66.0%** (2883/4370)	**67.1%** (6765/10076)

Table 1: Precision per POS and parsing method

Filtering by *Goi-Taikei*

We compared the specificity of the extracted arguments with that of the corresponding words in *Goi-Taikei* (GT) with the following classification. The results are shown in Table 2.

- ▪ > GT: more specific than GT
- ▪ = GT: same as GT
- ▪ no entry of the definition word in GT
- ▪ no sense entry of the definition word in GT
- ▪ < GT: less specific than GT

	Adj.	Verb	VN	All
> GT ▪	**48.8%**	**57.4%**	**46.6%**	***51.7%***
= GT ▪	.8%	3.4%	3.1%	2.9%
no GT entry ▪	41.1%	22.7%	39.8%	32.3%
No sense GT ▪	9.3%	16.2%	10.2%	12.8%
< GT ▪	0%	.3%	.3%	.3%
∑	100% (841)	100% (3,041)	100% (2,883)	100% (6,765)
N(▪ + ▪ + ▪) / ∑ extracted	98.9%	98.2%	98.5%	*96.8% (6,550)*

Table 3: Comparing specificity of extracted arguments with that in Goi-Taikei (GT)

The results show that 51.7% of the arguments we selected provide more specific referential information than those in GT. If those arguments that are not listed in GT are to be included, i.e. ▪ + ▪ , it goes up to 96.8%. In other words, virtually every argument extracted from the proposed method provides new or more specific referential information than what exists in GT.

While there remain many areas of improvements that will increase the precision as discussed in Nariyama et al. (2005), we have extracted 6,550 referents. This result is a promising first step towards building an inventory of representative arguments.

Proportions of sense use

The definition words in Lexeed have one or more senses, with 53 senses being the highest. Lexeed has been enhanced through manual word sense disambiguation. This enables us to measure the proportion of usage of a particular sense of a definition word among the other senses. That tells us how often a definition word is likely to come with the representative arguments extracted. For example, *taiho* 'arrest' has a single sense, while *umu* 'give birth' has two senses, and the sense with the representative arguments *mother* and *child/egg* is used 87.0% of the time in our corpus.

Table 3 shows the average proportions of sense use per POS. 'Mono' refers to words with a single sense (i.e. unambiguous) and the rest having multiple senses. '1st' refers to the most frequently used sense, '2nd' the next, and so forth. We accounted for up to the '3rd' most common sense, where the proportions plateau after the 2nd, 3rd for Verbs.

Figure 1 reports the cumulative frequencies computed from Table 3 by using the ordering: mono>1st>2nd; namely, '+1st' means the total proportion of 'mono' and '1st', and those plus '2nd' is shown by '+2nd'. It shows that many of the representative arguments we extracted have a single sense. For those words with multiple senses, the great majority of the representative arguments appear for either the most frequently used sense or the second highest sense, and few appear with the senses less frequent than the 3rd sense.

\ senses	Mono	1st	2nd	3rd	∑
Adjective	19.4 %	45.2%	16.9%	6.9%	88.4%
Verb	8.1%	28.3%	19.7%	19.7%	75.8%
Verb noun	34.6%	50.8%	12.6%	1.7%	99.7%

***Table 3: The proportions of sense use per POS**: mono-sense, 1st (most frequently used sense), 2nd, 3rd, and Total*

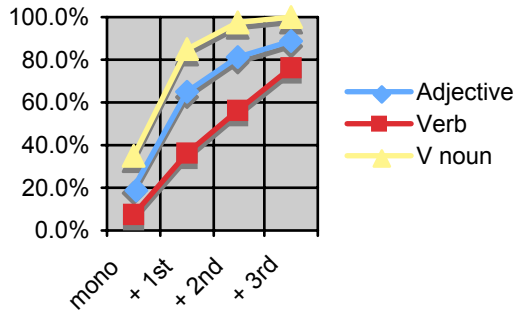


Figure 1: Representation of Table 3

3 Building a cross-lingual referential knowledge database

We aim to create a cross-lingual referential database, whereby the representative arguments for a word are shared across languages. There are two ways to approach this task. The first is to extract

representative arguments from each language individually, analogous to the way we did for Japanese. Then we list those definition words that take the same representative arguments shared by other languages. The other approach is to select only those representative arguments from Japanese that have their basis in world knowledge, and to transfer the information across to other languages.

We show in Subsection 3.1 that the observation from English dictionaries and a (Mandarin) Chinese dictionary indicates that the first option is not viable, so the second option should be taken. Accordingly, Subsection 3.2 discusses the process of classifying the Japanese representative arguments into two classes: Language independent referents (i.e. representative arguments that are shared across languages) and Language specific referents. Subsection 3.3 gives verification of the classification in two stages: first through human judgement, and secondly by hand-checking the Language independent referents using the English dictionary. The results are presented in Subsection 3.4.

3.1 English dictionaries

To make a comparison with Lexeed, we examined definition sentences from three machine-readable English electric dictionaries: Oxford Advanced Learner's dictionary 2000, Webster's dictionary 1913 (GCIDE http://www.ibiblio.org/webster/ in the public domain; 130,633 definition words), and Collins Cobuild Advance learner's English dictionary (Fourth edition 2003). In addition, a Chinese dictionary (现代汉语词典 *Xiandai hanyu cidian* 2002 by *Shang-wu-yin-shu-guan*) was referred to in comparison.

As an example, Figures 2a and 2b list the definition sentences for two definition words: *diagnose* and *marry* respectively. It is clear from the definition sentences for English and Chinese that Lexeed provides referential information more concisely, and that automatic extraction of the representative arguments in other languages will be not only difficult but also not fruitful. Most explanations are not sentences, but phrases with infinitive forms. This means that the subjects of sentences, one of the most important sources of representative arguments, are not expressed.

Furthermore, the referents are often very general as 'somebody' and 'something'. Many of the explanations also provide encyclopaedic information, which further complicates the process for automatically extracting representative arguments.

Lexeed: 診断：医師が患者を調べその病状を判断すること.
Shindan: Ishi-ga kanja-o shirabe sono byoujou-o handansuru.
'A doctor examines a patient and gives an option about the patient's illness.'

Oxford: To say exactly what an illness or the cause of a problem is.

Webster: To ascertain by diagnosis; to diagnosticate.

Cobuild: (1) To diagnose an illness or a problem means to discover and identify exactly what is wrong. e.g. 'Doctor has diagnosed it as rheumatism.'

Chinese: 诊断 *zhen-duan* 'diagnose'
在 检查 病人 的 症状 之后 判定 病人 的 病症 及 其 发展 情况
zai- jian-cha bing-ren de zheng-chuang zhi-hou pan ding bing-ren de bing-zheng ji qi fa-zhan qing-kuang
'(lit.) at examine patient DE symptom after decide patient DE disease and its development status'

Figure 2a: Definition sentences for 'diagnose'

Lexeed: 結婚：男女が夫婦になること。
Kekkon: Danjo-ga fuufu-ni naru koto.
'A man and a woman become a married couple.'

Oxford: To become the husband or wife of somebody; to get married to somebody.

Webster: To unite in wedlock or matrimony; to perform the ceremony of joining, as a man and a woman, for life.

Cobuild: If you marry someone, or if you get married, you form a legal relationship with a person of the opposite sex in a ceremony during which you make particular promises to that person and become their husband or wife. EG. 'I want to marry him.'

Chinese: 结婚 *jiehun* 'marriage'
男子 和 女子 经过 合法 手续 结合 成为 夫妻
nan-zi he nv-zi jing-guo he-fa shou-xu jie-he cheng-wei fu-qi
'(lit.) man and women through legal procedure combine become husband and wife'

Figure 2b: Definition sentences for 'marry'

Thus, we opted for taking the second approach; that is, to take those Japanese representative arguments that are judged as language independent referents from the experiments using English dictionaries (see Subsections 3.2 and 3.3) as seeds, and to use the resources from Japanese to bootstrap coverage of other languages.

Although the work up to this stage involves a certain amount of manual work, benefits of this approach are substantial:

1) It enables collection of representative arguments otherwise not possible, because they are not specifically written in the dictionaries of the other languages, or for languages with no dictionaries.

2) It lessens the amount of work on other languages, eliminating the various stages of extractions and hand verifications.

3) It will become even more cost effective as the amount of resources available increases and the number of language transfer increases.

4) It can automatically create a cross-lingual link that is useful for multi-lingual applications.

3.2 Classifying representative arguments

We manually classified the Japanese representative arguments that we extracted into two classes:

(a) **Language independent referents**
(b) **Language dependent referents**

The criteria for the distinction used in this verification are as follows, although they should be more objective and clear, requiring improvements. (a) was classified as such if the representative arguments in Japanese are either [1] not (b), or [2] based on scientific facts (e.g. physics, biology of animals, physiology of humans) and common knowledge (believed to be commonly known or agreed by adults of the world). For example, a word *umu* 生む 'give birth' is likely to have *mother* as the subject of the verb and *child/egg* as its object, and this referential relation is cross-linguistically valid.

The referents under (a) includes 'referent incorporation'. For example, *kyuukon* 求婚 'propose (a marriage)' includes the object 'a marriage' in the definition word in itself, whereas the English equivalent does not.

(b) Language dependent referents mainly involve three types. The definition words express:

(b-1) *Language specific concepts and idioms*: e.g. *katazukeru*（娘を）かたずける 'to get rid of' implies 'to get rid of (one's daughter by marrying her off');

(b-2) *Honorifics and other social ranking*: e.g. *insotsu* 引率 'to take' is used as 'a (higher ranked) person takes a (lower ranked) person', instead of the neutral form *tsurete-iku* 連れて行く 'to take';

(b-3) *Specialised or domain specific terms* e.g. *aisatsu* あいさつ「俳優が観客に」 'to greet' is used as '(In performing arts and theatre plays), the actors/actresses greet the audience'.

3.3 Verification

Using the criteria described in Subsection 3.2, we conducted the following hand verification in two stages in order to ascertain how many of the Japanese representative arguments are identified as 'language independent' referents.

[1] Among the 4,099 representative arguments we extracted that are more specific than *Goi-Taikei*, we have identified only 421 are language specific, i.e. 3,678 arguments (89.7%) are judged as language independent.
[2] 10% of those arguments were randomly selected and hand-checked in Cobuild[2] CD-ROM and/or Google search in order to further verify.

The results are classified as follows:

Language independent
i. Lexeed's referent is found in the Cobuild definition sentence.
ii. When not i, Lexeed's referent is found in the Cobuild example sentence.
iii. When not i or ii, Cobuild lists a referent that is of the same semantic class as Lexeed's referent.
iv. Lexeed's referent is not found in Cobuild, but the collocation of the Lexeed's referent with its definition word exceeds 100,000[3] hits in Google search (which indicates that the referent is most likely to appear with its definition word).

Language specific
v. Lexeed's referent is not found in Cobuild, and the collocation of the referent with its definition word is less 100,000 hits in Google search.
vi. No match of English translation of the definition word, or referent.

We make an assumption that the referents under i, ii, iii, and iv are likely to be language independent for the definition words and that they are readily

[2] Cobuild was chosen for this verification, as it is somewhat different from other English dictionaries. It uses full sentences, not phrases, and focuses on providing frequently used examples taken from a corpus (a collection of British and American newspapers, books, TV programs, real-life conversations, etc). In other words, Cobuild entries explain their usage in discourse, unlike the traditional dictionaries that focuses on precise definitions of words. In addition, Cobuild has been used in various NLP work (e.g. Hoelter 1999).
[3] The figure of 100,000 was heuristically chosen as the cut-off point.

transferable to other languages. The referents under v are not verified in English as language independent referents, requiring another mode of verification. Many of the referents under vi should have been classified as language dependent in the first stage of verification (see the next subsection).

3.4 Results and discussion

The results were significant, as shown in Table 4. Types i - iv (i.e. ∑ i-iv for all POSs) amount to 90.7%, which are confirmed to be language independent in the second verification.

	i	ii	iii	iv	∑ i-iv	v	vi
Adj	44.4	7.1	12.2	24.4	87.8%	7.8	4.4
Verb	44.2	3.3	20.8	16.7	85.0%	5.8	9.2
VN	58.1	3.2	21.8	13.7	96.8%	2.4	.8
All POS	50.1	4.0	19.0	17.6	***90.7%***	4.2	5.1

Table 4: Types of Japanese representative arguments compared with English per POS

We find that when there is a good match of English translation that is expressed by one or two words of English and semantically maps well, Lexeed's referent is generally found in the Cobuild or confirmed in Google search.

This gives rise to two implications.

One is that the exact semantic transfer between languages can be difficult for some words, and those words that don't transfer cleanly in translation tell us much about the culture of the speakers and are, thus, language specific (Bond 2005).

On this note, iv includes the cases where there is a translation but it takes, say, more than three words to *explain* the Japanese definition word; e.g. 論外 *rongai* is translated as 'be out of question'. Practically speaking, one word has to be chosen to be able to look it up in the Cobuild dictionary, and even then the arguments listed there are often too general or unrelated. In other words, referents can be automatically determined as language independent referents discerning from specific, when the appropriate translation of words are found and they comprise no more than three.

The other implication is that finding the optimal match of translation requires high command of the two languages: English and Japanese in this case. The inadequacy of this increased the number for vi ' no match', which could have been eliminated on the first verification. Nonetheless, the results from the two methods of verification showed that approximately 80% of arguments can be expected to be language independent.[4]

We can, thus, conclude that we can expect the majority of the representative arguments extracted from Lexeed to be language independent. Although this experiment is based on only two languages, considering the fact that Japanese and English are linguistically and culturally quite distant, the results are intriguing and promising. If arguments are manually confirmed as being language independent by the two very distinct languages, it is likely that those referents are shared across languages, although further verifications using a third language will be more assuring.

Interesting to note that some referents showed a case of 'partial mismatch of referents'. For example. the word *iroppoi* refers to women in Japanese, while the English equivalent 'sexy' is used for both sexes. The reverse is also true: 'pretty' in English is generally referred to girls, while in Japanese both sexes.

4 Related work

The series of our work is summarised as involving:

- the extraction of referential knowledge in the form of representative arguments,
- from Japanese dictionary definition sentences,
- using machine-readable dictionaries,
- investigating the feasibility for extending its referential knowledge across languages.

Since the inception of electronic lexical databases, such as WordNet (for English), *Goi-Taikei* (for Japanese), and HowNet (for Chinese), the use of machine-readable dictionaries for acquiring ontology has been the method taken by many in various languages (e.g. Tsurumaru 1991, Wilks et al. 1996, Nichols et al. 2005, inter alia). The majority of work, however, has concentrated on extracting semantic relations of words, such as synonym, hypernym, and meronym (Wilks et al. 1996, Fellbaum 1998).

In terms of work that focuses on extracting referential information, many studies use newspaper corpora instead of dictionaries. The two notable works in Japanese are the new EDR *Verb valency dictionary* (Hagino et al. 2003, listing verbs only) and a case frame dictionary (Kawahara and Kurohashi 2004). Utsuro et al. (1992) use bilingual corpora to acquire lexical knowledge.

Similar work for English has also been reported (Resnik 1997, inter alia). Slightly different is the work by Agirre and Martinez (2002) that focuses on class-to-

[4] We manually examined 3,678 representative arguments, 421 of these are found to be language specific by the first verification and another 342 from the second verification, that amount to 20.7% total (421+342/3,678).

class (class of verbs – class of nominals) relations instead of usual word-to-class (verb – a nominal class) relations.

The notable works that aim to process knowledge are CYC (Lenat 1995), Harabagiu and Moldovan (1998), and MindNet (Richardson et al. 1998). Although all of them are designed for English, we can improve our work from their approaches, which is our future work. Different in its approach but closer to the interest of our work is the work by Elouazizi (2004). It tries to formalise a universal ontology of referring modes to capture an optimal referential relations from the perspective of cognitive semantics.

What is different about our research is that while others extract general semantic classes of referents (e.g. 'person'), we extract specific referents that are representative for the predicate (e.g. 'police').

Our approach, however, has one disadvantage in terms of coverage. It cannot hold for all referents, since not every definition word has 'representative' arguments or dictionaries ensure to list them. As no single method is perfect per se, it is deemed beneficial that we consider merging the positives from various methods to further improve.

5 Conclusions

The output of this paper is that following the work on extracting the referential knowledge in the form of representative arguments from a machine readable Japanese dictionary (Nariyama et al. 2005), we examined the feasibility for extending its application across languages. The initial results show substantial promise.

Accounting for contextual information and world knowledge seems a prohibitive task at present. This paper has made a first step forward towards dealing with these issues by proposing a method to create a cross-lingual referential knowledge database. This linkage is of significant importance for multi-lingual applications, such as machine translation systems.

As another future work, we plan to formulate additional inferences drawing from representative arguments. For example, 'Mary gave birth to a baby' entails that Mary is the *mother* of the *baby*, and this knowledge is cross-linguistically true. This knowledge is particularly of importance for Question and answering tasks. It enables to find the answer for questions, such as 'Who is the mother of the baby?'

References

E Agirre and D. Martinez. 2002. Integrating selectional preferences in WordNet. The first *International WordNet Conference*.

F. Bond. 2005. *Translating the untranslatable: A solution to the problem of generating English determiners.* CSLI Publications. California

F. Bond et al. 2004. The Hinoki Treebank: A Treebank for Text Understanding, In Proc. of the First *IJCNLP*, *Lecture Notes in Computer Science.* Springer Verlag

U. Callmeier. 2002. Preprocessing and encoding techniques in PET. In Oepen et al. (eds), *Collaborative Language Engineering*, 127–143. CSLI, Stanford

N. Elouazizi. 2004. Towards an ontology of referential relations in natural language and its implication for NLP. Workshop on the Potential of Cognitive semantics for ontology (FOIS 2004)

C. Fellbaum (ed.). 1998. *WordNet: An electronic lexical database*. The MIT Press, Cambridge

A. Frank. 2004. Constraint-based RMRS construction from shallow grammars. In Proc. of *COLING*. 1269–1272, Geneva

T. Hagino et al. 2003. *Japanese verb valency dictionary.* Sanseido Publishing

S. M. Harabagiu and D. Moldovan. 1998. Knowledge processing on an extended WordNet. In Fellbaum (ed). 379-405

M. Hoelter. 1999. *Lexical-semantic information in Head-Driven Phrase Structure grammar and natural language processing.* Lincom Theoretical Linguistics

S. Ikehara et al. (eds). 1997. *Japanese Lexicon.* Iwanami

H. Isozaki and T. Hirano. 2003. Japanese zero pronoun resolution based on ranking rules and machine learning. In Proc. of *EMNLP*.184-191

D. Kawahara and S. Kurohashi. 2004. Improving Japanese zero pronoun resolution by global word sense disambiguation. In Proc. of *COLING*. 343-349. Geneva

D. B. Lenat. 1995. CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11), 32-38

S. Nariyama. 2003. *Ellipsis and Reference-tracking in Japanese*. SLCS 66, Amsterdam: John Benjamins

S. Nariyama et al. 2005. Extracting Representative Arguments from Dictionaries for Resolving Zero Pronouns. MT Summit X. Phuket Thailand

E. Nichols, F. Bond, and D. Flickenger. 2005. Robust Ontology Acquisition from Machine-Readable Dictionaries. In Proc. of *the International Joint Conferences on Artificial Intelligence.* Edinburgh

P. Resnik. 1997. Selectional preference and sense disambiguation. In proceedings of the ANLP Workshop *Tagging text with lexical semantics: Why What and How?*

S. Richardson et al. 1998. MindNet: acquiring and structuring semantic information from text. In Proc. of *COLING*

M. Siegel and E M. Bender. 2002. Efficient deep processing of Japanese. In Proc. of the 3rd Workshop on Asian Language Resources and International Standardization at *COLING*, Taipei

H. Tsurumaru et al. 1991. An approach to thesaurus construction from Japanese language dictionary. In IPSJ SIG Notes Natural Language, vol.83-16. 121–128 (in Japanese)

T.Utsuro, Y.Matsumoto, M.Nagao. 1992. Lexical knowledge acquisition from bilingual corpora. COLING. 581-587

Y.A. Wilkes et al. 1996. *Electric Words*. MIT Press

Approaching Sequential NLP Tasks with an Automata Acquisition Algorithm*

Muntsa Padró and Lluís Padró
TALP Research Center
Universitat Politècnica de Catalunya
{mpadro,padro}@lsi.upc.edu

Abstract

In this work, Causal-State Splitting Reconstruction algorithm, which learns a finite state automaton from data sequences, is applied to NLP tasks, namely Named Entity Recognition and Chunking. The obtained results are slightly below the best state-of-the-art systems, but can be considered competitive, and given the simplicity of the used features, they are really promising.

Keywords: Finite State Automaton, Sequential NLP Tasks, Pattern Acquisition, Named Entity Recognition, Chunking.

1 Introduction

Many NLP basic tasks have a sequential structure and may be modelled as pattern matching problems. These tasks can be approached using techniques based on linguistic knowledge and hand-built rules, or can be also approached using Machine Learning (ML) techniques that must take into account the sequence information.

There are several ML techniques devoted to sequential tasks. One group are generative models such as Hidden Markov Models (Rabiner 90) or Stochastic Grammars (Lari & Young 90) among others. Another group are conditional models that solve some problems related with generative models. Some examples are Maximum Entropy Markov Models (McCallum *et al.* 00) and Conditional Random Fields (Lafferty *et al.* 01).

The main problem of these kinds of algorithms is that they use graphical models with a predefined structure, so they are not very flexible and can not adapt well to some kinds of problems. One alternative are those algorithms that learn finite state automata from data. There are many different algorithms of this kind, for example Regular Positive and Negative Inference (RPNI) (Garcia *et al.* 00), Error Correction Grammatical Inference (ECGI) (Rulot 92), extended ECGI (Pla 00) and reconstruction algorithms such as Variable-Length Markov Models (VLMM) (Rissanen 83) and State-Merging ϵ-Machine Inference algorithms (Crutcheld & Young 90). The algorithm used in this work, Causal-State Splitting Reconstruction (CSSR) (Shalizi & Shalizi 04), is also a reconstruction algorithm.

CSSR is more powerful and widely applicable than VLMMs because the later is a particular case of causal state reconstruction algorithms. Comparing CSSR with ϵ-machines leads to the conclusion that while both have the same domain of applicability, CSSR is more well-behaved and converges faster than merging methods. It also makes use of some properties about causal states to guide the automata construction. Furthermore, since CSSR builds the automaton using statistical information, it is expected to capture better the behaviour of the process than other algorithms (such RPNI or ECGI) that do not use statistical information to build the automata.

This work is focused on the study of CSSR algorithm and on its applicability to some NLP tasks, in this case Named Entity Recognition (NER) and Chunking.

2 CSSR algorithm

Given a discrete alphabet Σ, consider a sequence x^- (history) and a future Z^+. Z^+ can be observed after x^- with a probability $P(Z^+|x^-)$. Two histories, x^- and y^-, are equivalent when $P(Z^+|x^-) = P(Z^+|y^-)$, i.e. when they have the same probability distribution for the future.

The different future distributions build the equivalence classes, named *causal states* of the process. Each causal state is a set of suffixes (sequences of symbols drawn from alphabet) that represent histories (up to a preestablished maximum length) with the same probability distribution for the future. The causal states of a process form a deterministic machine and are recursively calculable.

*This research is being funded by the Catalan Government Research Department (DURSI), by the Spanish Ministry of Science and Technology (ALIADO TIC2002-04447-C02) and by the European Commission projects Meaning (IST-2001-34460) and CHIL (IST-2004-506909).

Causal-State Splitting Reconstruction (CSSR) estimates an HMM inferring the causal states from sequence data. The main parameter of this algorithm is the maximum length (l_{max}) the suffixes can reach. That is, the maximum length of the considered histories. The algorithm starts by assuming the process is an identically-distributed and independent sequence with a single causal state, and then iteratively adds new states when statistical tests show that the current states set is not sufficient. Once the causal states are built, it is necessary to make the machine deterministic. For details about the algorithm, see (Shalizi & Shalizi 04) or (Padró & Padró 05).

3 Approaching NLP tasks with CSSR

In this work an approach to apply CSSR algorithm to different NLP tasks is presented. The tasks approached so far are Named Entity Recognition (NER) and Chunking. Both tasks are tasks where some subsequences of words, with determined properties, have to be detected. For the NER tasks, these subsequences are the Named Entities (NE) and for Chunking, they are the different kind of phrases.

Following CoNLL 2000, 2002 and 2003 shared tasks[1], we worked with the "B-I-O" approach (Ramshaw & Marcus 95) to mark these word subsequences. Each word has a B, I or O tag, being B the tag for a word where a NE or a chunk begins, I when the word is part of a NE or chunk but not the beginning, and O the tag for the words not belonging to any NE or chunk.

The general idea of our approach is to use CSSR to learn an automaton for NE or chunks structure. Once the automaton is learnt, it can be applied to detect NEs or chunks in untagged text. From now on, to explain our approach to use CSSR for these NLP tasks, we will refer to the chunk detection task as a general reference to NER and Chunking.

3.1 Learning the automaton

To learn the automaton that must reproduce a chunk structure, different information about the words is used. This information can be orthographic, morpho-syntactic, about the position in the sentence, etc[2]. Using the chosen features, the alphabet of the automaton is built as a closed set of symbols, and the words in a sentence are translated into this alphabet. The sentences in the train corpus translated in such a way will be the sequence used by CSSR to learn the automaton. The learnt automaton will reproduce the sequence behaviour in terms of the chosen features.

A problem of using that algorithm for this kind of tasks is that it models stationary processes, but chunk patterns are not in this category. So, what we did was to regard a text sequence as a stationary process in which chunks occur once a while. Doing so implies the automaton is modelling the pattern of the whole sequence (the text).

To allow CSSR to learn the pattern of the chunks, it is necessary to introduce in the alphabet the information of the chunk-tag (B, I or O) available in the supervised training corpus. In this way, we have information encoded in the transitions about B-I-O tags for chunks in the text. Thus, we can latter use this information to compute the best path for a sequence and use it to tag chunks in a new text.

For instance, let's suppose an approach to tag NEs where the only feature taken into account is whether a word is capitalized or not. Let's say that a capitalized word will have the feature "A" and a non-capitalized word the feature "a". In this case, the alphabet will consist of six symbols, which are the possible combinations of a capitalization value and a B-I-O tag (A_B, A_I, A_O, a_B, a_I, a_O). Each word will be translated into one of these symbols depending on whether it is capitalized and on its NE-tag.

3.2 Using the learned automaton to tag chunks

When the chunks in a sentence have to be tagged, the information about the correct chunk tag is not available, so there are several possible alphabet symbols for that word. It is only possible to know the part of the translation that depends on the word or sentence features. In our example, it would be possible to translate each word to an "A" or to an "a", but not to know the part of the symbol that depends on the chunk-tag, which is, in fact, what we want to know.

To find this most likely tag for each word in a sentence –that is, to find the most likely symbol of the alphabet (e.g. G_B, G_I, G_O for a G word)–, a Viterbi algorithm is applied. For each word, the best path for this word having each tag is stored.

[1] CoNLL 2000 shared task was devoted to text chunking, and CoNLL 2002 and 2003 shared tasks approached Named Entity Recognition and Classification

[2] In our approach, the features used to tag NEs are basically orthographic, while those used for the chunking task, are syntactic.

At the end of the sentence, the best probability is chosen and the optimal path is backwards recovered. In this way, the most likely sequence of B-I-O tags for each word in the sentence is obtained. There are some forbidden paths, which are those that lead to the *OI* tag-combination. The paths including this combination are pruned out.

When performing the tagging of chunks given a text, it is possible to find symbol sequences that haven't been seen in the training corpus. It implies the automaton falls in a *sink* state, which receives all the unseen transitions. When it happens, the automaton can not follow the input sequence using transition information because, as the transitions weren't seen, they are not defined.

To allow the system to continue tagging the text when the automaton falls into *sink* state, the suffix of length l_{max} is built concatenating the last $l_{max}-1$ symbols with the next symbol from the input. A state containing this new suffix is searched over the automaton and, if found, the automaton goes to this state and continues its normal functioning. If not, the process is repeated, getting more symbols from the input sequence, until a state containing the new suffix is found.

4 Experiments and Results

In this section, the specific settings, the performed experiments and the obtained results for NER and Chunking tasks are presented.

4.1 NER with CSSR

For the experiments in NER task, the data for the CoNLL-2002 shared task (Tjong Kim Sang 02) for Spanish were used. These data contain three corpora: one for the train and two for the test: one for the development of the system and the other one for the final test.

The used alphabet to learn an automaton via CSSR encoded some relevant information about the word position in the sentence, its capitalization and some information extracted from a dictionary. The chosen alphabet was the following:

- **G:** Beginning of the sentence, capitalized, not containing numbers, not in the dictionary[3].
- **S:** Beginning of the sentence, capitalized, not containing numbers, one of its possible analysis being a noun.
- **M:** Not at the beginning of the sentence, capitalized.
- **a:** Not at the beginning of the sentence, non-capitalized, functional word[4].
- **w:** Other.

In this way, the alphabet for CSSR will be the combination of these four features with the three possible NE tags (G_B, G_I, G_O, S_B, S_I, etc.).

CSSR algorithm has three important parameters. One is the chosen maximum length (l_{max}), which is the most significant parameter. The other two are the test used to check the null hypothesis and the parameter α, controlling the test significance degree. We made several experiments for different l_{max} values and with two different statistical test: χ^2 and Kolmogorov-Smirnov. For each test, the experiments were performed with several α values.

The results show that the significance degree value is not as influent as l_{max} value. In fact, for α under 0.01 the reached results and the size of the built automata don't vary significantly with α. The behaviour of the system using the different statistical tests is not qualitatively different, although Kolmogorov-Smirnov leads to slightly better results.

About the influence of l_{max}, best results are obtained with small l_{max} (three and four, in this case), likely caused by the limited size of the training corpus. For a more detailed discussion about the behaviour of the system depending on those parameters, see (Padró & Padró 05).

The best performance is obtained with $l_{max} = 3$ and $\alpha = 1e-5$. Table 1 shows the obtained results whith these values over the two test corpora. The baseline column corresponds to a baseline stablished using FreeLing analyzer (Carreras *et al.* 04) NER system. This system uses a simple hand-built rule-based automaton of four states.

	Baseline		**CSSR**	
	Test a	Test b	Test a	Test b
Precision	79.83	79.14	89.81	90.03
Recall	88.32	90.02	88.22	88.81
F_1	83.86	84.23	89.01	89.42

Table 1: Baseline and obtained results using CSSR with best parameter settings

In this table, it is shown that the results obtained with CSSR are clearly better than the baseline, what means that our system is able to

[3]The used dictionary is the one provided by FreeLing (Carreras *et al.* 04)

[4]Functional words are articles or prepositions that are often found inside a NE

capture more sophisticated patterns than those recognized FreeLing.

These results can be also compared with the winner system of CoNLL-2002 shared task (Carreras *et al.* 02) which performs the NE recognition and classification separately , so it is possible to compare our system with the part that performs the NE recognition. This NER system obtained a F_1 of 91.66% for the Spanish development corpus and a 92.91% for the test corpus. These results are higher than the results presented in this work, which was expected since the feature set used by that system is much richer (bag of words, disambiguated PoS tag, many orthographic features, etc.) than the used in our experiments.

Furthermore, it is possible to apply the NEC system used by (Carreras *et al.* 02) to the output of our NE detector. Doing so over our best results yields to a $F_1 = 76.30\%$, which would situate our system in the fifth position in CoNLL-2002 ranking table for complete NER systems in Spanish.

4.2 Chunking with CSSR

In this section, the performed experiments in detecting different syntactic chunks in a text and the obtained reults are presented. For these experiments, the data for the CoNLL-2000 shared task (Tjong Kim Sang & Buchholz 00) are used. These data contain two corpora: one for the train and one for the test. There are eleven different chunk types, but for the moment we only worked with the seven more frequent chunks. These chunk types are: Noun Phrase (NP), Verb Phrase (VP), Prepositional Phrase (PP), Adverb Phrase (ADVP), Subordinated Clause (SBAR), Adjective Phrase (ADJP) and Particles (PRT).

To perform the tagging of the different kind of chunks, an automaton is learnt using CSSR for each phrase type. Each automata is expected to reproduce the pattern of each chunk kind. Once the automata are built, it is necessary to combine the application of all the automata over the test corpus to obtain a global text chunking system.

The alphabet used to train the automata are the Part of Speech (PoS) tags available in the CoNLL-2000 corpora. The total number of different tags is 44, what means that CSSR alphabet will have 132 symbols (each PoS tag combined with the B, I or O tag). In these experimets, the same vocabulary is used for all the chunk types.

First of all, in order to know how the chunkers are performing, all the chunkers are tested separatelly over the test corpus. Regarding that automata built via CSSR are very dependent on the maximum choosen length, it is necessary to perform a search over different l_{max}. Since in previous section it has been seen that α value doesn't affect too much to the system performance, this value is fixed in these experiments. Firsts columns of table 2 show the best results obtained for each kind of chunk tested separately, and the maximum lengths used to obtain these results.

Secondly, a method to combine the different chunkers was implemented. To do so, the automata were not built and applied to the task separately, as they were independent, but they were applied sequentially, each one taking into account the information produced by the previous chunkers. Therefore, it is necessary to design previously the architecture of the whole system, deciding in which order the chunkers will be applied. Once it is decided, all the chunkers must be trained in order to build automata that take into account the previous tags. So each chunker is built via CSSR, with a vocabulary that includes the information about the previously tagged chunks.

The chosen order to apply this method is the order of better performance when the systems are applied separately. This order is: VP, PP, NP, ADVP, ADJP, SBAR, PRT. In this case, as the automata are learned with a different vocabulary, the search for the best l_{max} has to be performed again. Table 2 shows both, the obtained performance of the separated chunkers and the obtained results when combining the different chunkers in order. In both cases, the maximum length for which these results are obtained is shown.

Although these results are not very good, they are in the range of the lower systems presented in CoNLL-2000 shared task. Taking into account the simplicity of our approach, being competitive with the state-of-the-art systems is a good result and states a basis to work on. In fact, this system would reach the tenth position in that shared task, with two systems behind it. Furthermore, these results are expected to improve by adapting the alphabet to the different kind of chunks.

5 Conclusions and Further Work

In this work a finite automata acquisition algorithm has been applied to NLP sequential tasks. Firstly, it has been shown that this algorithm can

Chunk Type	Separated Automata				Combined System			
	Precision	Recall	F_1	l_{max}	Precision	Recall	F_1	l_{max}
NP	89.42	85.36	87.34	2	90.58	90.05	90.32	2
VP	91.31	91.80	91.55	1	91.34	91.67	91.50	1
PP	87.33	95.26	91.12	1	85.99	94.91	90.23	2
ADVP	74.81	66.86	70.61	1	74.88	69.52	72.10	1
SBAR	59.33	23.18	33.33	1	55.06	25.42	34.78	1
ADJP	65.33	63.24	64.27	1	61.54	65.75	63.58	1
PRT	23.08	16.98	19.57	1	33.33	29.25	31.16	1
Overall					**88.03**	**88.37**	**88.20**	

Table 2: Results obtained separately and combining the predictions of the different kind of chunkers one after the other.

build automata that give quite good results when applied to recognize the NEs of a text. In fact, the system results are not too far from those obtained by the winner system on CoNLL 2002 shared task and they may be expected to improve by introducing more information in the system, since we use less knowledge than all CoNLL 2002 participants.

Secondly, CSSR has been also applied to text Chunking, with a performance that is comparable to the state-of-the-art systems. Nevertheless, we think that there is still a wide range for improvement given that the performed experiments are really preliminary and there is already a lot of work to do in searching which are the important features for each type of chunk and which is the best way to combine the learned automata.

The main conclusion of this work, is that CSSR algorithm can be satisfactorily applied to NER tasks and that it performs quite well in Chunking task also. Taking into account that our approach is very simple and uses few features, this work opens a door to continue our research line by improving the systems presented here and exploring the application of the system to other NLP sequential tasks.

The future lines to be developed are:

- Improve NER and Chunking systems by developing more suitable alphabets and introducing more information into the system.
- Use conditional models to include richer features. This could solve one of the limitations of our approach, which is that all the information to be introduced in the automaton has to be codified as part of the alphabet.
- Apply this algorithm to other NLP tasks such as Part of Speech Tagging or Verb Diathesis Acquisition. The latter is interesting because in this case CSSR will not be used to learn automata to perform a tagging task, but to extract the subcategorization pattern of a verb. So CSSR will be directly used to perform knowledge extraction from data.

References

(Carreras *et al.* 02) Xavier Carreras, Lluís Màrquez, and Lluís Padró. Named entity extraction using adaboost. In *Proceedings of CoNLL Shared Task*, pages 167–170, Taipei, Taiwan, 2002.

(Carreras *et al.* 04) Xavier Carreras, Isaac Chao, Lluís Padró, and Muntsa Padró. Freeling: An open-source suite of language analyzers. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal, 2004.

(Crutcheld & Young 90) James P. Crutcheld and Karl Young. Computation at the onset of chaos. In *In Wojciech H. Zurek, editor, Complexity, Entropy, and the Physics of Information, volume 8 of Santa Fe Institute Studies in the Sciences of Complexity, pages 223-269*, Massachusetts, 1990.

(Garcia *et al.* 00) P. Garcia, A. Cano, and J. Ruiz. A comparative study of two algorithms for automata identification. In *Proceedingf of the 5th International Colloquium on Grammatical Inference, ICGI 2000*, 2000.

(Lafferty *et al.* 01) John Lafferty, A. McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *In Proceedings of the Eighteenth International Conference on Machine Learning (ICML-2001)*, 2001.

(Lari & Young 90) K. Lari and S. J. Young. The estimation of stochastic context-gree grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56, 1990.

(McCallum *et al.* 00) Andrew McCallum, Dayne Freitag, and Fernando Pereira. Maximum entropy Markov models for information extraction and segmentation. In *Proc. 17th International Conf. on Machine Learning*, pages 591–598. Morgan Kaufmann, San Francisco, CA, 2000.

(Padró & Padró 05) Muntsa Padró and Lluís Padró. Applying a finite automata acquisition algorithm to named entity recognition. In *Proceedings of 5th International Workshop on Finite-State Methods and Natural Language Processing (FSMNLP)*, Helsinki, Finland, September 2005.

(Pla 00) Ferran Pla. *Etiquetado Léxico y Análisis Sintáctico Superficial basado en Modelos Estadísticos.* Unpublished PhD thesis, Departament de Sistemes Informàtics i Computació, Universitat Politècnica de València, 2000.

(Rabiner 90) L. R. Rabiner. *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition.* Readings in Speech Recognition (eds. A. Waibel, K. F. Lee). Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1990.

(Ramshaw & Marcus 95) L. Ramshaw and M. P. Marcus. Text chunking using transformation-based learning. In *Proceedings of the Third ACL Workshop on Very Large Corpora*, 1995.

(Rissanen 83) Jorma Rissanen. A universal data compression system. In *IEEE Transactions in Information Theory, IT-29:656-664*, 1983.

(Rulot 92) H. Rulot. *ECGI: Un algoritmo de inferencia gramatical basado en la corrección de errores.* Unpublished PhD thesis, Universitat de València, 1992. Advisor(s): Dr. E. Vidal and Dr. F. Casacuberta.

(Shalizi & Shalizi 04) Cosma Shalizi and Kristina Shalizi. Blind construction of optimal nonlinear recursive predictors for discrete sequences. In *Uncertainty in Artificial Intelligence: Proceedings of the Twentieth Conference*, 2004.

(Tjong Kim Sang & Buchholz 00) Erik F. Tjong Kim Sang and Sabine Buchholz. Introduction to the conll-2000 shared task: Chunking. In Claire Cardie, Walter Daelemans, Claire Nedellec, and Erik Tjong Kim Sang, editors, *Proceedings of CoNLL-2000 and LLL-2000*, pages 127–132. Lisbon, Portugal, 2000.

(Tjong Kim Sang 02) Erik F. Tjong Kim Sang. Introduction to the conll-2002 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2002*, pages 155–158. Taipei, Taiwan, 2002.

The Use of Causal Expressions for Abstracting and Question Answering

Christopher Paice and **William Black**
Computing Department, Lancaster University, UK
cdp@comp.lancs.ac.uk
School of Informatics, University of Manchester, UK
w.black@manchester.ac.uk

Abstract

This paper describes a method for identifying causal expressions and extracting their semantic components, using the CAFETIÈRE information extraction system. After further processing and ranking, the extracted structures can be used for generating output statements. The method is suitable for generating abstracts of scientific 'experiment papers', and for the generation of answers to questions involving causal relations, and is not domain specific.

1 Background and Motivation

Approaches to developing systems for the automatic generation of abstracts and summaries fall under two broad heads. In the first approach, the program attempts to 'understand' the source text and construct a representation of its meaning, or at least those facets of its meaning which are judged important. This representation is then processed to generate textual output. In the second, an 'extract' is produced by selecting passages from the source text (often sentences, but sometimes clauses or paragraphs), and outputting them, verbatim or with superficial changes. Discussions of these approaches in terms of their effectiveness and practicality may be found elsewhere (Mani 01),), but here we simply note that neither is very satisfactory.

In this paper we outline an approach which in some ways falls between these two extremes. What we describe is a development of the concept-based abstracting (CBA) approach of (Paice & Jones 93) and (Oakes & Paice 99). The CBA method focused on the abstracting of papers on crop husbandry, and involved defining a set of contextual expressions which, when found in a text, are likely to contain entity names of specific types or 'roles' – names of crops, crop properties or parts, pests, environmental influences, etc. After scanning a complete text, the various entity names assigned to each distinct role were compared, and those occurring repeatedly were adopted as key terms. Information about the key-terms and their roles were then expressed as stylised output statements.

The CBA method suffered from various drawbacks; in particular:

- although semantic roles were assigned to key-terms, any relationships linking these terms were not extracted; and
- the defined patterns were domain specific.

The method described in this paper addresses both of these shortcomings. It relies in particular on the identification of relational expressions, and the extraction of the textual fragments which form the semantic components of these expressions. The expressions are identified by looking for characteristic markers in the text. The expressions we want to use are ones which are not associated with any specific domain, but can occur regardless of the subject area. Of course, such a method is only likely to be useful only for texts of certain genres. Our current work is focused on scientific texts, and in particular 'experiment papers', which introduce and describe experiments and tests, and discuss their results. Whilst various kinds of relational expression may be important in such texts (definitions, associations, observations, etc.), we have chosen to concentrate on causal expressions (causations), since these constitute the explanatory core of a scientific investigation. We believe that if causations can be identified and their components extracted and collated, the resulting data-sets can be used to generate relevant output statements. Causations are also likely to be useful for question answering, particularly where the question calls for an explanation, as discussed briefly in Section 5.5.

We are by no means the first researchers to develop a system for extracting causal expressions. Most notably, (Khoo *et al.* 00) have developed a system for extracting causations based on 68 'causality patterns' which are used to construct conceptual graphs. Their system was ap-

plied to a collection of abstracts in various areas of medicine, and they obtained recall and precision values in the region of 50% – 58%. (Girju & Moldovan 02a; Girju & Moldovan 02b) have concentrated on data mining and question answering, using a method based on detection of causal verbs. Garcia's COATIS system(Garcia 97) locates causal expressions in French texts.

In the present paper we provide an overview of our proposed abstracting system. We have already developed a repertoire of rules for identifying causations and extracting their main features, as described in the next section. The rules are expressed and applied using the CAFETIÈRE system, outlined in section 3. In section 4 we explain how the system output contains a semantic template for each extracted causation, and in section 5 we outline the operations (still to be implemented) which will adjust and collate the information in the templates, and generate the final output.

2 Causal Expressions in English

2.1 Scope

The range of causal expressions which concern us here are restricted to what might be called 'material causations', obtaining between entities in the natural world. We do not deal with human goals and motivations, indicated by expressions such as "in order to", nor logical entailments, expressed using "hence", "therefore", etc.

Secondly, we are concerned only with intra-sentential expressions ('causations'). Causal links between propositions in neighbouring sentences or clauses form a perfectly valid topic for study, but they are not amenable to our present approach.

Whilst, as previously explained, we expect causations to occur over a wide range of domains, we expect that they will only occur frequently in certain text genres. Our work is focused in particular on experimental papers in the physical and biological sciences and technology. Causations will doubtless occur in texts of various other genres e.g., descriptions of systems or devices.

2.2 Types of Causal Expression

Causal expressions may be conveniently classified as abstract or concrete. An *abstract causation* typically takes the form of an extended noun phrase, consisting of a distinctive head noun followed by postmodifiers. Typical examples are:

(1) The effect of fluxoids on self-recombination

(2) The response of chickpea seedlings to abnormally low temperatures

An abstract causation refers to the possibility of a causal relationship between two entities, but does not say anything about the nature of the relationship, other than its presumed direction. It thus defines a topic or theme, which is discussed further elsewhere. Abstract causations are often used as the titles of documents or sections of documents. Otherwise, the expression exists within a longer sentence, which presents or comments on the theme. For instance, it may be preceded by an indicator construct such as "In this paper we discuss ..."(Paice 81).

A *concrete causation* normally contains a finite verb, and expresses some definite information about the relationship concerned. For example

(3) The growth of chickpea seedlings is seriously impeded by night-time temperatures below 5°C.

It is noticeable that the distinction between abstract and concrete causations corresponds closely to the traditional distinction between indicative (topic-indicating) and informative abstracts.

Another useful classifying feature of causations is their direction. Example (1) is a *forward causation*, in which the causal factor ("fluxoids") is mentioned before the affected property ("self-recombination"). By contrast, examples (2) and (3) are *backward causations*.

To give an impression of the range of English causal expressions, Table 1 shows eight typical forms – four abstract and four concrete. In each case, C stands for 'cause' and P for 'effect or 'affected property'. Note that the second example, though abstract, contains a finite verb in a relative clause.

The effect of C on P
The effect that C has on P
The response of P to C
The change in P due to C
C changes/causes P
C has an effect on P
P is changed by C
P is a result of C

Table 1: Causation and association expressions

2.3 Detailed Features of Causations

A glance at example (3), which belongs to the '*P* is changed by *C*' pattern, will at once convince us that the real situation is really quite complicated. Several additional factors and features need to be accommodated in our model:

1. Many of the 'clue words' in Table 1 stand for a range of alternatives "effect" for "influence" and "impact"; "due to" for "because of", and so on.

2. The verbs in the concrete expressions may be replaced by elaborate verb groups containing modals, adverbs and negations e.g., "seems to have no effect on", "may be significantly reduced by", etc.

3. Additional prepositional phrases may be inserted within expressions – typically after the left-hand argument, or else at the very end of the expression. Often these are time expressions such as "after harvest", but can include other conditions such as "in the absence of insect predation".

Aside from these elaborations, it turns out that the simple 'two-argument' model of Table 1 is inadequate. Referring again to example (3), we see that the affected entity is "chickpea seedlings", but that the affected property is "growth". This leads to a decision to split the 'effect side' of a causation into the particular *property* which is changed, and the *object* which possesses that property. If no object is specified in the causation, that 'slot' is left unfilled.

By a similar argument, it is sometimes possible to split the 'cause side' into the direct cause, and some remoter entity, called the origin, to which it pertains. The presence of an explicit origin is relatively unusual, but when it does occur it is helpful to record it (see example (5)).

It is important to emphasise that the splits between property and object, and between cause and origin, do not correspond to specific semantic relations. Thus, in some cases a 'property' may actually be a physical part of the object ("leaves"), or a stage in its development ("germination"). The splitting merely provides a convenient framework within which to organise the information extracted from a causation.

We need to clarify the nature of the conceptual labels which may fill these four slots. Since prepositions are important for partitioning an expression into its components, we would expect the components to be compound nouns – that is, noun phrases without prepositional postmodifiers. This should include co-ordinate noun phrases, as in example (4).

(4) Endophytic fungi have been shown to affect reproduction and/or growth of some grasses.

Finally, we note that causes and affected properties are often expressed using such general-purpose modifiers as "number of ...", "type of ...", and we therefore bind such modifiers to the following noun-group to serve as the property. In example (5), "loss of potential nutrients" is the causal factor, and "droppings" the origin; had "loss of" not been bound to the following noun group, the meaning of the extracted text would have been distorted.

(5) The loss of potential nutrients from droppings had no effect on the eventual yield.

Note also in this case that, had the causal side not been split into cause and origin, the cause would have been recorded, inadequately and misleadingly, as "droppings".

2.4 Part-Causations

A part-causation is one in which either the cause side, or the effect side, cannot be retrieved from the immediate context. Although incomplete, part-causations can provide useful information, and are retrieved by our rules.

Some causations are only partially extracted due to the limitations of our rules. For example, our current rules only deal fully with compact expressions, where all the defined components are contiguous. Thus, in example (6) intervening material prevents retrieval of "yield components" as the affected entity.

(6) The yield components determined later in the spring (seeds per pod and seed weight) were not sensitive to planting dates.

This qualifies as a retrieval error.

On the other hand, some part-causations occur because one of the components is not available within the sentence e.g., the missing element is implicit, or is referred to by means of an anaphor. These are counted as special cases, but not actual retrieval errors.

3 Template Extraction Mechanism

The current experimental work uses the CAFETIÈRE framework (Black *et al.* 03), which has been further developed from one first described in (Black *et al.* 97).

CAFETIÈRE, like most information extraction systems is modular and filters texts through a sequence of analyses at different levels. After tokenizing a text, a part-of-speech tagger (Black & Vasilakopoulos 02) following Brill's algorithm (Brill 95) is applied.

Following tagging, ontology lookup (Black *et al.* 04) associates semantic information with matching text elements. The ontology is intentionally small, due to the present focus on domain-independent extraction of causal relations.

At present, all further analysis uses the rule-based (partial) parsing component of CAFETIÈRE. This rule formalism was designed to allow a partial syntactic analysis to draw upon any attributes of token, whether linguistic or not, and rules can also be constrained by attributes of neighbouring tokens.

Phrases and their constituents are described by sets of attribute-value pairs. Attributes range over orthographic (e.g. `orth=punct`), morpho-syntactic (e.g. `pos=NN`) and semantic/conceptual (e.g. `lookup=title`) properties. Attributes are used both to constrain and to construct representations by means of feature unification (`_var` is a variable that unifies as in Prolog,[1] `__var` is a variable whose values on the right-hand side are concatenated to instantiate the instance on the left). Both negation (!= operator) and disjunction of values (| operator) are supported. There is a mechanism to identify simple forms of co-reference.

Examples of rules are (7), (8) and (9).[2] Rule (7) is a heuristic rule that labels a span as the proper name of a person on the basis of two clues: the orthographic form of the constituent tokens, and the preceding title. This rule illustrates how items can be required to be present in the immediate context of a phrase in order to confirm the phrase's semantic class. Items on the right-hand side of the rule that precede the \ or follow the / are not part of the phrase described by the left-hand side, but items required to be found in the left or right context respectively.

(7)
```
[syn=NNP, sem=PERSON] =>
   [sem=title]{1,2}
 \ [orth=capitalized],
   [orth=upperinitial]?,
   [orth=capitalized] / ;
```

(8)
```
# Rule 40
# Noun Group
[pos=NG, sem=nounGroup, label=__NP,
type=pnamex, rulid=NounGroup40] =>
\
[pos=DT]?,
[pos=CD]?,
[pos=RB, token=__NP]?,
[pos=JJ|JJR|VBG|VBN, token=__NP]?,
[sem=name_group, token=__NP]?,
(( [pos=NN|NNP, token=__NP],
   [token="-"|"/", token=__NP])?,
 [pos=NN|JJ|VBN|VBG|NNP, token=__NP]){0,3},
[pos=NN|NNP|NNS|NNPS, token=__NP]
/;
```

(9)
```
# Rule 67
# An 'effect that' a cause has on something
[sem=causation, label=__X, cause=_C,
effect=_E, object=_O, type=pnamex,
rulid=Causation67] =>
\
[sem=effect_that, token=__X],
[sem=thing, token=__X, token=_C],
[sem=have_group, token=__X],
[token="on", token=__X],
[sem=thing, token=__X, token=_E],
([token="of"|"in", token=__X],
[sem=thing, token=__X, token=_O])?
/;
```

Rules like (7) can be used to label spans according to semantic category, i.e. to classify named entities. On the other hand, syntactic chunking, or minimal phrase structure analysis, can be done with rules like (8). The use of regular expression quantifiers allows several productions to be grouped together in a single rule.

Rule (9) illustrates the use of feature value assignments to fill templates representing relations or as in this case facts. The instantiated features `cause, effect` and `object` which can be seen on the left-hand side (before the `=>` operator) are the slots of a causation template instance.

4 The Causation Templates

The application of our set of causation rules to a text results in an XML-formatted output file, in which details of each causation and part-causation are recorded. Each output record is in effect a template recording the various features of the causation concerned. In the case of an abstract causation, only the four main features al-

[1] The implementation is in Java.

[2] Rules (8) and (9) are from a set of causal relation-extracting rules discussed in this paper, whereas Rule (7) illustrates context-sensitivity in rules.

Table 2: Output templates for 3 causation expressions

Sentence	*slot fills*
"The growth of chickpea seedlings is seriously impeded by night-time temperatures below 5°C"	cause=night-time temperatures; verb=impeded; adverb=seriously; property=growth; object=chickpea seedlings
"Endophytic fungi have been shown to affect reproduction and/or growth of some grasses"	cause=endophytic fungi; verb=affect; auxiliary=shown to; object=some grasses; property=reproduction and/or growth
"The loss of potential nutrients from droppings had no effect on the eventual yield"	cause=loss of potential nutrients; origin=droppings; verb=had; polarity=no; property=eventual yield

ready described – origin, cause, affected property and object – are expected to be present (and in practice usually only two or three of these). For concrete causations, however, the template normally contains various other features which refer to what is known about the relationship. Semantically, these represent degrees of certainty or significance, directions of change (of cause or of property), quality evaluations, etc., but the output templates simply contain words or word groups (adverbs, negation words, modals, etc.) extracted from the causal context.

As an illustration, Table 2 shows the instantiations of various features from concrete causations (3) to (5) , given previously (note that, in the first example, the words "below 5°C" are not picked up by our current rules). The features are here shown as three groups: the cause side, the effect side, and between them the 'relational details' (verb, polarity, etc.). Any missing feature is set by default to a NULL value. Note that the verb features "affect" and "had" provide no information beyond showing that these are concrete causations, whereas "impeded" provides information which can be used during later processing.

5 Processing of Causation Templates

5.1 Semantic Standardisation

The first stage in processing the output templates is to convert the relational details into appropriate semantic indicators. For instance, an effect will normally have a effect-type (increase/decrease, benefit/harm, etc.) and an intensity (absent, slight, moderate, large), and this information may be known with a certain confidence or level of significance. The rules for these transformations remain to be fully developed and tested, but Table 3 shows some examples of indicative words or expressions, found among the relational details or in the premodifiers of properties.

5.2 Ranking of Causations

We are interested in those causations which talk about the key topics of the text being processed. In order to identify these topics we use a procedure, outlined in (Paice & Black 03), which extracts from a text all sequences of from 1 to N contiguous words (where N is typically 3..5). Deletion of stopwords from the sequences, followed by stemming and sorting of the remaining words, allows variant phrase forms to be merged. An ad hoc formula, based on word- and phrase-frequencies and phrase length, is then used to assign a score, supposedly representing importance, to each phrase. The top-ranking phrases are retained as key-phrases for the text.

The extracted causations can then be assigned an overall score by looking for any key-phrases which match the contents of the cause, origin, property and object features, and summing these scores for the whole causation. Any partial overlap (e.g., between "growth" and "growth rate") may be reduced according to the proportion of words which match. This stage thus results in the causations being ranked in order of apparent importance.

5.3 Editing of Causation Records

Given a ranked set of causation records, it would be possible to retain just the top few records, and generate an output sentence from each one. This is likely to give poor results, partly because different records may both contain virtually the same information, but also because some high-ranking records may contain feature strings which are only

Table 3: Assignments to Semantic Categories

effect	increase	increased, increase in, growth, greater, enlarge
	decrease	decreasing, reduce, reduction in, loss of, smaller
	assist	promoting, encouraged, stimulation of
	hinder	inhibit, impeded, suppression of
	benefit	improve, better, improvement in, enhanced
	harm	damaging, deterioration, harmful
intensity	absent	no, not, none, absence of
	slight	negligible, small, almost no, minimal, slightly
	moderate	moderate. (Also assigned as default)
	large	large, considerable, greatly, seriously, severely
confidence	low	possibly, may, could be
	normal	probably, seems to, is believed to. (Also assigned as default)
	high	significant, certainly, is known to

partly specified, or else require definition. For instance, a feature such as "size" might need to be expanded to "leaf size", and a feature "RL" to be replaced (at least on its first occurrence) by "root length".

The rules for these processes remain to be worked out, but in general terms we would expect to start with the highest-ranking record R_1, and compare its cause/effect fillers with those of later records, in order to decide whether any of the fillers of R_1 need to be expanded or replaced; evidence from part-causations can also be valuable here. Moving to the second record R_2 we would repeat this process, but would then compute the similarity between the features of R_2 and R_1. If the two are highly similar, this implies that R_2 is redundant and can be deleted. This cycle of actions (feature editing, followed by comparison with the already-accepted records) is repeated for subsequent records, until a sufficiently long abstract can be generated.

5.4 Output Generation

There are two possibilities for generating output statements from the selected causation records, neither of which presents any great difficulty. One is to take each record and use a simple output template to generate a suitable concatenation of prescribed words and fillers. The other, which would give more variation in style, would take the original form of words of the extracted passage, but would include any expansions or substitutions of fillers.

To illustrate the last few stages, let us take the first output template shown in Table 2, derived from causation example (3) repeated as (10).

(10) The growth of chickpea seedlings is seriously impeded by night-time temperatures below 5°C.

Suppose that semantic standardisation assigns standardised features in place of "impeded" and "seriously", and that comparison with other records allows the cause feature to be expanded to "low night-time temperatures". This gives the updated record (11):

(11) cause=low night-time temperatures;
effect=hinder; intensity=large;
property=growth;
object=chickpea seedlings

Use of a standard output format might then allow generation of a statement such as (12):

(12) Low night-time temperatures greatly hinder the growth of chickpea seedlings.

Note that the semantic flag "large" is rendered here as "greatly". Inclusion of such transformations within the output template means that a repertoire of templates can be defined, allowing more variety in the output. Thus, it would be easy to generate an equivalent passive form based around "is greatly hindered by".

The alternative approach, in which the original form of words is retained, would (assuming "below 5°C" is not extracted) result in the output form (13):

(13) The growth of chickpea seedlings is seriously impeded by low night-time temperatures.

This approach appears to make the semantic standardisation stage redundant (since the relational details are expressed in the original words), but would reduce the accuracy of the similarity comparisons between pairs of causations.

5.4.1 Abstract generation

The contribution our approach can make to generating high-quality abstracts of scientific papers is that by focusing on causal relationships, it becomes possible to extract the key content that distinguishes an *informative* abstract from a merely *indicative* one. Complementary techniques are needed to generate additional content for a well-formed abstract, indicating e.g. the broad topic of the study and details of method.

5.5 Question Answering

The field of open-domain question answering (ODQA) is evolving to deal with more complex kinds of relation between questions and answers, but has not yet systematically considered causal relation questions. Example question forms answerable from our templates include:

(14) What factors influence *property* of/in *class*?

(15) Why does *change* occur in *class*?

(16) Under what conditions does *change* occur in *class*?

Filled causation templates form a database against which wh-questions involving causal relations can be answered. Previously retrieved texts are being analyzed with the causation templates described here in order to answer "why" questions in ongoing work on ODQA.

6 Summary

We have outlined our system for generating scientific abstracts by extracting causal expressions. Our system for identifying causations is currently being evaluated, using texts across a range of domains. The detailed rules for processing the output templates and generating the final abstracts are still being developed. Aside from completing the system as described, we are interested (a) in augmenting our rule set to extract other expression types, such as associations and definitions; (b) in using the extracted data for other applications, such as question answering and term discovery; and (c) in studying the applicability of this approach to information extraction in other text genres, such as historical narrative and news reportage.

7 Bibliography

References

(Black & Vasilakopoulos 02) W.J. Black and A. Vasilakopoulos. Language Independent Named Entity Classification by modified Transformation-based Learning and by Decision Tree Induction. In *Proceedings of the 6th Conference on Natural Language Learning (CoNLL2002)*, Academica Sinica, Taipei, August 2002.

(Black *et al.* 97) W J Black, L Gilardoni, F Rinaldi, and R Dressel. Integrated text categorisation and information extraction using pattern matching and linguistic processing. In *Proceedings of RIAO97*, pages 321–335, Montreal, 1997.

(Black *et al.* 03) William J. Black, John M^{c}Naught, Argyris Vasilakopoulos, Kalliopi Zervanou, Babis Theodoulidis, and Fabio Rinaldi. CAFETIÈRE: Conceptual Annotations for Facts, Events, Terms, Individual Entities, and RElations. Technical Report TR-U4.3.1, Department of Computation, UMIST, Manchester, 2003. `http://www.crim.co.umist.ac.uk/parmenides`.

(Black *et al.* 04) William J Black, Simon Jowett, Thomas Mavroudakis, John McNaught, Babis Theodoulidis, Argyrios Vasilakopoulos, Gian-Piero Zarri, and Kalliopi Zervanou. Ontology-enablement of a system for semantic annotation of digital documents. In Siegfried Handschuh, editor, *Proceedings of Semannot 2004, ISWC 2004 Workshop on Semantic Annotation*, Sanibel, Hiroshima, Japan, November 8 2004.

(Brill 95) Eric Brill. Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging. *Computational Linguistics*, 21(4):543–566, December 1995.

(Garcia 97) D. Garcia. COATIS, an NLP system to locate expressions of actions connected by causality links. In *Knowledge Acquisition, Modeling and Management, Proceedings of the Tenth European Workshop EKAW '97*, pages 347–352, 1997.

(Girju & Moldovan 02a) R. Girju and D. Moldovan. Mining answers for causation questions. In *Proceedings of the AAAI Spring conference*, 2002.

(Girju & Moldovan 02b) R. Girju and D. Moldovan. Text mining for causal relations. In *Proceedings of the FLAIRS 2002 conference*, pages 360–364, 2002.

(Khoo *et al.* 00) C. Khoo, S. Chan, and Y. Niu. Extracting causal knowledge from a medical database using graphical patterns. In *Proceedings of the ACL conference*, pages 336–343, Hong Kong, 2000.

(Mani 01) I. Mani. *Automatic Summarization*. John Benjamins, Amsterdam/Philadelphia, 2001.

(Oakes & Paice 99) M.P. Oakes and C.D. Paice. Automatic generation of templates for automatic abstracting. In *Proc. 21st British Computer Society Information Retrieval Specialist Group Colloquium (IRSG99)*, University of Strathclyde, Scotland, May 1999.

(Paice & Black 03) C.D. Paice and W.J. Black. A Three-pronged Approach to the Extraction of Key Terms and Semantic Roles. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2003)*, pages 357–363, Borovets, Bulgaria, 2003.

(Paice & Jones 93) C.D. Paice and P.A. Jones. The identification of important concepts in highly structured technical papers. In *Proc. 16th ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 93)*, pages 69–78, New York, 1993.

(Paice 81) C.D. Paice. The automatic generation of literature abstracts: an approach based on self-indicating phrases. In R. N. Oddy, S.E. Robertson, C.J. van Rijsbergen, and P.W. Williams, editors, *Information Retrieval Research*, pages 172–191. Butterworths, London, 1981.

Optimizing the subtasks in the double classification approach to Information Extraction

Viktor Pekar and **Richard Evans**
Research Group in Computational Linguistics
HLSS, University of Wolverhampton
MB114 Stafford Street
Wolverhampton, WV1 1SB, UK
{v.pekar,r.j.evans}@wlv.ac.uk

Abstract

In Information Extraction, a very common task is to extract facts about a single event or entity from an entire document such as a personal homepage, a job or a seminar announcement. The double classification method approaches this task with two automatic classifiers. The first one classifies larger document fragments to roughly indicate which of them are likely to contain template fillers. The second classifies text tokens inside promising fragments to more precisely pinpoint the filler. In this study we show how the effectiveness of the method can be considerably increased by optimizing the task difficulty of each of the classifiers. We then consider the related problem of identifying the best filler per template field among all the text tokens labeled as positive instances of that field by the second classifier. We present a method to delimit a token or sequence of tokens among them as the best filler for the field.

1 Introduction

A common information extraction (IE) task is to extract facts about a single event or entity from an entire document such as a person's name and contact details from her home page. Correspondingly, for each IE task, there is one template to be filled for each document. Previous research has developed a range of IE methods based on automatic classification techniques to address this kind of task (e.g., (Freitag 98), (Soderland 99), (McCallum *et al.* 00), (Califf & Mooney 98), (Chieu & Ng 02)).

The double classification method uses two different automatic classifiers to extract information. The method is inspired by the observation that when humans need to extract some facts from a document they scan it quickly and only read closely those parts that look most relevant. In a similar manner, the double classification method uses the first classifier to identify document fragments that are likely to contain template fillers. The second classifier classifies tokens inside the promising fragments to more precisely pinpoint the filler. The study by (Sitter & Daelemans 03) has shown that this is a very promising approach to pursue. By first performing classification of fragments, the unbalanced data problem at the token level is greatly reduced, i.e. the fact that a document contains much fewer positive examples of field fillers than negative ones. De Sitter and Daelemans' study demonstrated that the method is both more efficient and effective than methods that extract template fillers by examining the immediate context of each token in the text.

This paper examines the idea that the effectiveness of the method can be increased by carefully choosing the classification problems for the two classifiers. We show that by using an appropriate fragment size and applying thresholding and instance selection techniques, the token-level classifier is able to locate template fillers more accurately than the original method proposed by (Sitter & Daelemans 03).

Another contribution of the paper is the proposal of a new method for accurate identification of one single filler for a field which may consist of a single token or a sequence of tokens. The method identifies such fillers in the output of the double classification method. It takes advantage of such evidence for the best filler as the relative position of tokens labeled as positive by the second classifier, the frequency of the token sequences, and the frequency of their subparts.

The paper is organized as follows. The next section discusses the double classification method in more detail, presents the techniques that we investigate in order to adjust the two classification problems and proposes an algorithm for the identification of the best fillers in the output of the method. Section 3 describes the settings used for experimental evaluation. In Section 4, we present and discuss the results of the evaluation. Finally, in Section 5, we summarize the results and draw conclusions.

2 The Double Classification Method

2.1 Task Definition

The IE procedure performed by the double classification method can be formally described as follows. Suppose we have a corpus annotated in terms of a predefined IE template, i.e., certain text tokens (words and punctuation) have a tag signifying that they instantiate a field of the template. The corpus is randomly divided into a training set of documents D_{TR} and a test set of documents D_{TS}. The first step is to split documents in D_{TR} and D_{TS} into sets of document fragments (say, sentences or paragraphs) F_{TR} and F_{TS}, respectively.

At each classification stage, n binary classifiers are

built, one for each template field s_i. At the fragment classification stage, classifier C_1 is learned from F_{TR}. Positive instances are fragments, which contain at least one token annotated as s_i. Features for representing an instance are all tokens found inside the fragment. C_1 is evaluated on F_{TS}. Its output is F_{out}, the set of fragments that it has labeled as positive.

At the second stage, classifier C_2 is learned only from those fragments in F_{TR} which contain tokens annotated as a filler for s_i. Positive instances here are tokens annotated as s_i, negative ones are those that do not have any tags or have tags other than s_i. Features are tags and tokens appearing within a certain window around the token. C_2 is evaluated on tokens contained in F_{out}. The output from C_2 is a list of tokens W_{out} which it has labeled as positive instances of s_i.

In computing evaluation metrics for the method as a whole, true positives are those tokens in W_{out} that have been manually annotated as positive, false positives are tokens in W_{out} that have not been manually annotated as positive, and false negatives are tokens have been manually annotated as positive, but did not make it into either F_{out} or W_{out}.

2.2 Adjusting the Task Difficulty

We hypothesize that the effectiveness of the method greatly depends on how the difficulty of the entire IE task (i.e. the search space in which template fillers should be found) is distributed between the two classifiers. We would like to find a balance that will avoid two kinds of situations leading to poor overall results:

1. C_1 achieves high precision, but low recall. The search space for C_2 is greatly shrunk, but many relevant tokens are missing from it.

2. C_1 achieves high recall, but low precision. Most of the relevant tokens do appear at the second stage, but there are also very many irrelevant ones and the search space for C_2 is too large.

In this paper, we examine whether or not the factors described in Sections 2.2.1 to 5 can help in the discovery of this optimal balance:

2.2.1 Thresholding

The most suitable balance is not simply the most balanced precision/recall ratio at C_1. It may depend on the nature of the documents at hand, namely on how indicative of template fillers: (1) larger contexts of tokens (e.g., paragraphs) and (2) their local contexts (e.g., neighboring tokens) are relative to each other. If, for example, it is easy to recognize fillers in their local contexts, but larger contexts are difficult to distinguish as relevant or irrelevant, one should be cautious about classifications at C_1 and aim to maximize C_1's recall in order to increase the number of promising tokens passed on to C_2. If, on the contrary, larger contexts are highly indicative of template fillers, then it makes sense to try to narrow down the search space for C_2 by preferring greater precision on the part of C_1.

Thresholding is a technique that allows one to boost either precision or recall by looking at the confidence score of the classifier, such as the class membership probability output by Naïve Bayes classifiers or the distance in the vector space output by k nearest neighbors and Support Vector Machine classifiers. Various thresholding techniques exist (see (Sebastiani 02)). In this study we prioritize recall by determining the average confidence score for false negatives on held-out data and during proper testing we retrieve all instances above that threshold. To increase precision, we find the average confidence score for false positives and during testing treat all instances below that threshold as negative even if the classifier assigns them the positive label.

2.2.2 Fragment Size

The size of the fragments may have a strong effect on how many relevant and irrelevant tokens will be passed from C_1 to C_2. If a document is split into many smaller fragments, such as lines, classifying them will be more difficult but this will allow for the greatest reduction of the search space for C_2. If instead one chooses larger fragments like paragraphs, relevant ones will be found with greater ease, but C_2 will suffer more from the unbalanced data problem.

Again, the best solution is not necessarily to simply choose average sized fragments; the usefulness of the fragments depends on the specific characteristics of the documents.

Note that thresholding and fragment size are orthogonal factors, so that their combination may maximize the desired effect and at the same time compensate for one another's drawbacks. For example, aiming for good recall of smaller fragments may lead to better results overall than increasing precision in the classification of larger fragments, and vice versa.

2.2.3 Selection of Instances

As in the work described in (Sitter & Daelemans 03), we also consider whether or not the problem of too many negative instances can be alleviated by performing instance selection. In order to build a more effective classification model, we train it on data from which some negative instances have been removed so that a certain desired proportion between negative and positive instances is achieved. After the model is learned from the balanced training data, it is evaluated on the unbalanced test data.

In this study we look at how instance selection interacts with the other two parameters. In particular, we wish to find out whether the problem of increased search space resulting from maximized recall or from using larger fragments can be remedied by performing instance selection.

2.3 Identifying the Best Filler

The double classification method tries to find all occurrences of a filler in the document. Obviously, this

task is difficult and seldom error-free. In many situations, however, extracting all field instantiations is not necessary, since the template field has to be filled with one single filler. One possibility to perform this is to choose out of all candidates the one that the token-level classifier extracted with the greatest confidence. However, this approach will not be very helpful in the case when fillers consist of multiple tokens, as it may easily select one part of the filler, but miss out others.

We propose a new method to identify the best filler for a template field. In addition to the classifier confidence score, it incorporates information about whether the positively labeled tokens make up sequences, the count of these sequences, and the use of general surface constraints on the appearance of the filler.

The algorithm (see Algorithm 1) consists of two major steps. The first step (lines 1-7) is to extract N, a set of all possible token ngrams from C_2's output and compute the initial score for each n. The ngrams are uninterrupted sequences of tokens labeled as positive instances of a template field. Those ngrams are eliminated that consist of tokens bearing no content such as stopwords and punctuation (line 4). Semantically empty tokens are also removed from the beginnings and ends of the ngrams (line 5). The initial score for n is computed as the sum of the weights of its occurrences, where the weight of each occurrence n_{occ} is the average classifier score C of its constituents (line 6).

To illustrate with an example, consider the hypothetical output from the classifier in Figure 1 (the first column shows the number of the token in the document, the second the token itself, the third the label assigned by the classifier, and the last the classifier confidence score). The algorithm will first extract the ngrams "by John Doe.", "John", and "Doe". Stripping stopwords and punctuation at the beginning and end of each of them, three ngrams will be obtained: "John Doe", "John", and "Doe". Their initial scores will be computed as follows:

$score(\text{"}John\ Doe\text{"}) = (0.75+0.5)/2 = 0.625$
$score(\text{"}John\text{"}) = 0.35$
$score(\text{"}Doe\text{"}) = 0.7$

```
...
18   authored   NIL      0.5
19   by         AUTHOR   0.5
20   John       AUTHOR   0.75
21   Doe        AUTHOR   0.5
22   .          NIL      0.5
...
44   writes     NIL      0.6
45   John       AUTHOR   0.35
46   .          NIL      0.45
...
72   John       NIL      0.3
73   Doe        AUTHOR   0.7
74   ,          NIL      0.9
...
```

Figure 1: Example of the output from C_2.

Data: a list of positively classified document tokens T, each attached with a classifier confidence score C
Result: a list of token ngrams ranked according to their relevance as template fillers

1 Extract a set of token sequences S from T;
2 Create a set of unique token ngrams N by extracting all subsequences from each $s \in S$;
3 **for** *each* n *in* N **do**
4 | discard n, if it contains only punctuation or stopwords;
5 | remove punctuation and stopwords in the beginning and end of n;
6 | $score(n) = \sum_{n_{occ} \in n} \frac{1}{length(n_{occ})} \sum_{i \in n_{occ}} C(i)$;
7 **end**
8 **for** *each* n *in* N **do**
9 | discover N' in N such that n' is a subsequence of n;
10 | **for** *each* n' **do**
11 | | $score(n)$ += $score(n') \times \frac{length(n')}{length(n)}$;
12 | **end**
13 **end**
14 Rank N according to *score*;

Algorithm 1: The algorithm for identifying the best filler per template field.

The second step (lines 8-13) is to add further weight to those ngrams whose subsequences exist in N. Thus, the final score for "John Doe" will be increased by the initial scores of its subsequences "John" and "Doe" appearing as distinct ngrams, each weighted by the proportion of its length to the length of the greater ngram, i.e. by 0.35*0.5 + 0.7*0.5 = 0.525. In this way, the algorithm aims to further take into account those cases, when only a part of a relevant ngram has been labeled positively by the classifier.

As it is reliant upon the count of the ngram, the method may have important interaction with the particular thresholding and fragmentation methods used. Specifically, we hypothesize that the best filler identification works best with double classification settings that achieve the greatest recall while maintaining high precision.

3 Evaluation

3.1 Experimental Task

The documents we would like to extract information from are web pages describing NLP resources including software (part-of-speech taggers, parsers, various corpus tools) and data (evaluation corpora and datasets, frequency lists, gazetteers). The IE template consists of the following fields: NAME, CREATOR, AREA (application area), TGTLANG (target language), PLATFORM, PROGLANG (programming language), and EMAIL (contact email). All the fields take single fillers, except TGTLANG and PLATFORM. Some slots are mandatory

(e.g., NAME), while others are not (e.g., TGTLANG). It should be emphasized that although some of the fields are filled by a closed class of words (e.g., PLATFORM), the IE method is a machine learning procedure that extracts fillers by examining only the context of tokens in the documents.

3.2 Data

The evaluation is carried out on 100 web pages that had been manually downloaded using the link collection on the topic at the Language Technology World web site[1]. The documents are preprocessed in the following steps:

Irrelevant HTML code (e.g., tags for images, forms, various scripts) are removed. The HTML structure is standardized and converted to XML. The documents were tagged for paragraph and sentence boundaries, parts-of-speech and syntactic chunks using the *LT Chunker* program (Mikheev 96).

3.3 Classification Method

At C_1 each fragment was represented as a feature vector, where features corresponded to the tokens found in it. All words were stemmed, stopwords and words appearing in less than 5 different fragments in the entire corpus were discarded. At C_2, to represent each token t, the following features were used:

- `token_itself`: the string corresponding to t
- `tags_itself`: XML tags (layout, PoS, phrase chunking tags) inside which t appears
- `token_before`: the token directly before t
- `tags_before`: XML tags on the token before t
- `token_after`: the token directly after t
- `tags_after`: XML tags on the token before t
- `token_window`: the tokens appearing within the context window of 2 around t
- `tags_window`: all XML tags appended on the tokens within the context window.

In the experiments we used the WEKA implementation of the multinomial Naïve Bayes learner (Witten & Frank 99). To assess the accuracy of classifications, we use 10-fold cross-validation, computing precision, recall and F-measure for each field and then averaging the results.

4 Results and Discussion

4.1 Fragmentation Method

We experimented with four types of fragments: Sections (*Sec*), Paragraphs (*Par*), Sentences (*Sent*) and Lines (*Lin*). Table 1 characterizes each type of fragments.

[1] http://www.lt-world.org

	Sec	Par	Sent	Lin
Tokens per fragment	68.5	18.8	9.4	7.3
Fragments per doc	15	54.2	101.6	142.1

Table 1: The average size of fragments and the number of fragments per document for the four fragmentation methods.

	No thresholding	Boosted P	Boosted R
Sec	0.02	0.03	0.01
Par	0.05	0.06	0.04
Sent	0.81	0.08	0.77
Lines	0.09	0.1	0.08

Table 3: Search space at C_2: the proportion of positive and negative training instances for different fragmentation methods and thresholding settings.

Table 2 describes the effectiveness of classifications at both levels (C_1 and C_2) resulting from the use of each fragmentation method (*Sec*, *Par*, *Sent*, *Lin*). The best results across fragmentation methods are shown in bold. Column 1 in Table 3 characterises the search space for each fragmentation method as the corresponding proportion of positive and negative instances at the token level.

We see that at C_1 larger fragments do indeed result in an easier classification task: the highest effectiveness at the first stage is achieved for the *Sec* and *Par* methods. Looking at C_2, we notice that the proportion of positive instances increases as the fragment size decreases. This accounts for the fact that notwithstanding good performance at the first stage, the *Sec* method is often the worst when these fragments are taken as the source from which fillers are extracted. Although *Lin* has the greatest positive/negative ratio, it performs poorly compared with other methods, because of inaccurate classifications at the initial stage. *Par* exhibited the best overall performance at the second level, outdoing *Sent* by a large margin.

4.2 Thresholding

We looked at how maximizing recall or precision interacts with different fragment sizes. We would like to see if thresholding can help to compensate for the weaknesses in a particular fragmentation method. Thus, we expect that overall performance of small fragments which greatly reduce the search space for C_2 can be improved by increasing recall for C_1. This will increase the search space for C_2, but the increase might be smaller than the one resulting from simply using larger fragments. Table 4 describes the results of these runs at C_2. In bold are the figures showing better performance than the runs without thresholding.

As will be noted from Table 3, boosting recall at C_1 does increase C_2's search space somewhat, but for the smaller fragments, *Sent* and *Lin*, the search space is still smaller than for *Sec* and *Par* without thresholding. As figures in Table 4 show, this leads to an

	Fragments			Tokens			Fragments			Tokens		
	P	R	F	P	R	F	P	R	F	P	R	F
	Sections						Sentences					
NAME	0.937	0.789	0.857	0.299	0.452	0.360	0.666	0.867	0.753	0.277	0.811	0.413
AREA	0.857	0.666	**0.750**	0.080	0.378	0.132	0.500	0.375	0.429	0.177	0.810	0.291
CREATOR	0.500	1	0.667	0.032	0.568	0.061	0.242	0.444	0.313	0.073	0.964	0.136
PLATFORM	0.466	1	0.636	0.291	0.388	0.333	0.818	0.692	0.750	0.288	1	0.447
PROGLANG	0.666	1	0.800	0	0	0	0.666	0.666	0.666	0.115	1	0.206
TGTLANG	0.400	1	**0.571**	0.233	0.304	0.264	0.103	0.800	0.183	0.041	0.761	0.078
EMAIL	0.304	1	**0.466**	0.169	0.907	0.285	0.134	1	0.236	0.173	0.975	0.294
AVERAGE	0.590	0.922	**0.720**	0.158	0.428	0.231	0.447	0.692	0.543	0.163	0.903	0.276
	Paragraphs						Lines					
NAME	0.924	0.910	0.917	0.525	0.792	**0.631**	0.875	0.913	**0.894**	0.277	0.480	0.351
AREA	0.363	0.500	0.421	0.301	0.814	**0.439**	0.600	0.375	0.462	0.245	0.285	0.263
CREATOR	0.476	0.625	0.540	0.319	1	**0.484**	0.636	1	**0.778**	0.029	0.509	0.055
PLATFORM	1	0.692	**0.818**	0.375	1	**0.545**	0.437	1	0.608	0.225	0.388	0.285
PROGLANG	1	1	**1**	0.636	0.700	**0.666**	0.400	1	0.571	0	0	0
TGTLANG	0.303	1	0.465	0.201	0.652	0.307	0.272	0.857	0.413	0.243	0.454	**0.317**
EMAIL	0.214	1	0.353	0.524	0.981	**0.683**	0.296	1	0.457	0.142	0.962	0.247
AVERAGE	0.611	0.818	0.848	0.700	0.412	**0.555**	0.502	0.878	0.639	0.166	0.440	0.241

Table 2: The accuracy of the fragment- and token-level classifiers resulting from each fragmentation method.

improvement in performance: both precision and recall rates frequently rose for *Sent*. In a similar fashion, larger fragments profit from increased precision. Boosting precision at C_1 narrows down the search space for C_2, which often improves the accuracy for larger fragments such as Sections.

4.3 Instance Selection

We examined instance selection techniques as an alternative way to relax the unbalanced data problem at C_2. We look at whether they are especially helpful for C_2 in situations when high recall at C_1 is achieved. The instance selection was carried out by randomly discarding negative instances from the training data until their count was the same as that of positive ones. Table 5 describes the results for different thresholding and fragmentation settings with instance selection applied. In bold are the results that are higher than those achieved using the same configuration but without instance selection.

We see that instance selection very often significantly improves recall, notably for *Sec* and *Lin*; the recall averages have gone up in all but one configurations. However, this is sometimes achieved at the cost of a considerable decrease in precision (e.g., for *Lin* from 0.114 to 0.038). This may be due to the fact that the model is induced from data that contains a greater proportion of positive instances. This causes the classification of a larger proportion of test instances as positive, hence higher recall, but lower precision. At the same time, in some cases, instance selection also resulted in an improved precision. In five out of the twelve configurations, the precision averages have increased. In general, it can be noted that instance selection helps to achieve greater effectiveness: for many configurations the averages of the F-measure have gone up. Cases, when the averages of the F-measure have deteriorated, are usually those when a large improvement in recall was achieved at the expense of very low precision. We believe that these unwanted situations can be avoided by finding a better proportion of positive and negative instances in the training data during instance selection.

4.4 Field-Specific Fine-Tuning

Using binary classifiers gives one the opportunity to adjust the classification problems for each template field separately. Table 6 compares the results achieved for each field using the most optimal configuration for that field (the last but one column) against the typical configuration of the double classification method, i.e. using sentence fragments without performing any thresholding or instance selection (the last column). The results indicate that fine-tuning the classification problem for each field separately offers a significant improvement over the traditional approach in terms of precision (by 0.2) and F-measure (by 0.26).

4.5 Identifying the Best Fillers

We evaluated the best filler identification algorithm against the performance of hand-crafted IE rules. The rules trigger the extraction of a particular field filler based on a variety of orthographic, linguistic, and page formatting cues. The hand-crafted rules were prepared by two domain experts; the construction of the rules took 4 person/weeks in total. As gold standard, we used the same evaluation data as in the previous experiments: a database was prepared by filling each template field for each document with the most frequent unique filler tagged by annotators in that document. The evaluation of the both IE methods consisted of 10-fold cross-validation, at each fold both methods were evaluated on the same set of documents.

We examined the effect of varying the parameters of the double classification method (the fragment size, thresholding and instance selection) on the perfor-

	Sec			Par			Sent			Lines		
	P	R	F	P	R	F	P	R	F	P	R	F
Boosted precision												
NAME	**0.305**	**0.456**	**0.366**	0.420	**0.828**	0.557	0.296	0.811	0.434	0.220	**0.481**	0.302
AREA	**0.085**	**0.400**	**0.140**	0.218	0.814	**0.344**	0.177	0.810	0.291	**0.274**	0.285	**0.279**
CREATOR	**0.018**	**0.551**	**0.035**	0.190	1	0.319	0.073	0.964	0.136	0.002	**0.142**	0.004
PLATFORM	**0.666**	0.200	0.308	0.230	1	0.374	0.288	1	0.447	**0.250**	0.272	0.261
PROGLANG	0	0	0	0.583	0.700	0.636	0.115	1	0.206	0	0	0
TGTLANG	**0.241**	**0.318**	**0.274**	0.141	0.652	0.232	0.041	0.761	0.078	0.243	0.454	0.317
EMAIL	0.142	**1**	0.249	0.358	0.981	0.525	0.173	0.975	0.294	0.063	**1**	0.119
AVERAGE	**0.208**	0.418	0.278	0.306	**0.854**	0.451	**0.166**	0.903	**0.280**	0.150	**0.376**	0.214
Boosted recall												
NAME	0.299	0.452	0.360	0.253	0.780	0.382	0.271	**0.900**	0.417	0.277	0.480	0.351
AREA	0.080	0.378	0.132	0.032	0.803	0.062	0.288	0.810	**0.425**	**0.041**	**0.213**	**0.069**
CREATOR	0.032	0.568	0.061	0.190	1	0.319	0.008	**1**	0.016	0.029	0.509	0.055
PLATFORM	0.291	0.388	0.333	0.230	1	0.374	0.428	0.923	**0.585**	0.225	0.388	0.285
PROGLANG	0	0	0	0.583	0.700	0.636	0.750	1	**0.857**	0	0	0
TGTLANG	0.233	0.304	0.264	0.096	0.652	0.167	0.003	**1**	0.006	**0.087**	0.391	**0.142**
EMAIL	0.169	0.907	0.285	0.358	0.981	0.525	0.041	**1**	0.079	0.142	0.962	0.247
AVERAGE	0.158	0.428	0.231	0.249	0.845	0.385	0.256	**0.948**	**0.403**	0.114	0.420	0.179

Table 4: The effect of boosting precision vs. recall at C_1 on the accuracy of C_2

.

mance of the best filler identification algorithm. Table 7 describes the results achieved with the most optimal parameter settings for each field (the last but one column) and compares them with the performance of the hand-crafted rules (the last column). We find that the performance of the proposed algorithm is consistently superior to that of the hand-crafted rules, and often by a considerable margin (e.g., by 0.83 for TGTLANG).

5 Conclusion

The double classification method provides convenient means to perform information extraction tasks where there is one template to be filled from an entire document. In this paper we presented an investigation into a number of parameters of the method in order to optimize its two classification subproblems and eventually improve its overall performance.

In general, these experiments have shown that finding appropriate settings for the three factors influencing the distribution of the task difficulty between the two classifiers helps to improve the performance of the method. In particular, doing so increased F-measure by 0.26 in comparison with using fragmentation of documents into sentences without applying thresholding and instance selection as was done in the original study by (Sitter & Daelemans 03).

The double classification method aims to extract all tokens instantiating of template fields, which is a very difficult and error-prone task. However, what is often needed instead is accurate extraction of one single filler which may consist of a single token or a sequence of tokens. We have presented a new method for the identification of such fillers in the output of the double classification method. The proposed method takes advantage of the evidence for the best filler in form of the relative position of tokens labeled as positive by the second classifier, the frequency of the token sequences, and the frequency of their subparts. Our evaluation shows that the method coupled with the double classification performs consistently better than hand-crafted extraction rules.

6 Acknowledgements

This study was conducted within the research project "An Automatic System for Resource Databases for Researchers" (ESRC grant RES-000-23-0010).

References

(Califf & Mooney 98) M. E. Califf and R. J. Mooney. Relational learning of pattern-match rules for information extraction. In *Working Notes of AAAI Spring Symposium on Applying Machine Learning to Discourse Processing*, pages 6–11, Menlo Park, CA, 1998. AAAI Press.

(Chieu & Ng 02) Hai L. Chieu and Hwee T. Ng. A maximum entropy approach to information extraction from semi-structured and free text. In *Proceedings of AAAI-02*, pages 768–791, 2002.

(Freitag 98) Dayne Freitag. Information extraction from html: application of a general learning approach. In *Proceedings of AAAI-98*, 1998.

(McCallum *et al.* 00) Andrew McCallum, Dayne Freitag, and Fernando Pereira. Maximum entropy Markov models for information extraction and segmentation. In *Proc. 17th International Conf. on Machine Learning*, pages 591–598. Morgan Kaufmann, San Francisco, CA, 2000.

(Mikheev 96) Andrei Mikheev. LT_CHUNK V 2.1. Language Technology Group, University of Edinburgh,UK, 1996.

(Sebastiani 02) Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

(Sitter & Daelemans 03) Ann De Sitter and Walter Daelemans. Information extraction via double classification. In *Proceedings of ATEM-2003*. Dubrovnik,Croatia, 2003.

(Soderland 99) Stephen Soderland. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34:233–272, 1999.

(Witten & Frank 99) Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999. ISBN: 1–558–60552–5.

	Sec			Par			Sent			Lines		
	P	R	F	P	R	F	P	R	F	P	R	F
no thresholding												
NAME	0.150	**0.857**	0.255	0.407	**0.867**	0.554	0.271	**0.900**	**0.417**	0.195	**0.880**	0.319
AREA	0.029	**0.864**	0.056	0.274	0.685	0.391	**0.288**	0.810	**0.425**	0.283	**0.775**	**0.415**
CREATOR	0.008	**1**	0.016	0.208	0.947	0.341	**0.097**	0.964	**0.176**	0.009	**1**	0.018
PLATFORM	0.005	**1**	0.010	**0.647**	0.916	0.758	**0.428**	0.923	**0.585**	0.006	**1**	0.012
PROGLANG	**0.714**	**0.500**	**0.588**	**1**	0.600	**0.750**	**0.750**	1	**0.857**	0.040	**1**	0.077
TGTLANG	0.003	**1**	0.006	0.009	**1**	0.018	0.016	**1**	0.031	0.004	**1**	0.008
EMAIL	0.008	**1**	0.016	0.050	**1**	0.095	0.041	**1**	0.079	0.008	**1**	0.016
AVERAGE	0.131	**0.889**	0.228	0.371	**0.859**	0.518	**0.270**	**0.942**	**0.420**	0.078	**0.951**	0.144
boosted precision												
NAME	0.162	**0.876**	0.273	**0.465**	**0.881**	**0.609**	**0.309**	**0.891**	**0.459**	0.159	**0.796**	0.265
AREA	0.033	**0.885**	0.064	**0.305**	0.666	**0.418**	**0.288**	0.810	**0.425**	**0.302**	**0.734**	**0.428**
CREATOR	0.006	**1**	0.012	**0.208**	0.947	**0.341**	**0.097**	0.964	**0.176**	**0.004**	**1**	**0.008**
PLATFORM	**0.800**	**0.400**	**0.533**	**0.647**	0.916	**0.758**	**0.428**	0.923	**0.585**	**0.636**	**0.636**	**0.636**
PROGLANG	**1**	**0.125**	**0.222**	**1**	0.600	**0.750**	**0.750**	1	**0.857**	**0.250**	**0.800**	**0.381**
TGTLANG	0.003	**1**	0.006	0.010	**1**	0.020	0.016	1	0.031	0.004	**1**	0.008
EMAIL	0.008	**1**	0.016	0.050	**1**	0.095	0.041	**1**	0.079	0.004	1	0.008
AVERAGE	**0.287**	**0.755**	**0.416**	**0.384**	**0.859**	0.531	**0.276**	**0.941**	**0.427**	**0.194**	**0.852**	**0.316**
boosted recall												
NAME	0.150	**0.857**	**0.255**	0.022	**1**	0.043	0.271	0.900	0.417	0.195	**0.880**	0.319
AREA	0.029	**0.864**	0.056	0.012	**1**	0.024	0.288	0.810	0.425	0.008	**1**	0.016
CREATOR	0.008	**1**	0.016	**0.208**	0.947	**0.341**	0.008	1	0.016	0.009	**1**	0.018
PLATFORM	0.005	**1**	0.010	**0.647**	0.916	**0.758**	0.428	0.923	0.585	0.006	**1**	0.012
PROGLANG	**0.714**	**0.500**	**0.588**	**1**	0.600	**0.750**	0.750	1	0.857	0.040	**1**	0.077
TGTLANG	0.003	**1**	**0.006**	0.006	**1**	**0.012**	0.003	1	0.006	0.003	**1**	0.006
EMAIL	0.008	**1**	**0.016**	0.050	**1**	**0.095**	0.041	1	0.079	0.008	**1**	0.016
AVERAGE	0.131	**0.889**	0.228	0.278	**0.923**	**0.427**	0.256	0.948	0.403	0.038	**0.983**	0.073

Table 5: The effect of instance selection on different fragmentation and thresholding configurations.

	Settings			Best settings			Typical		
	Thresholding	Fragmentation	Inst. sel.	P	R	F	P	R	F
NAME	boosted P	paragraph	yes	**0.47**	**0.88**	**0.61**	0.277	0.811	0.413
AREA	boosted P	paragraph	yes	**0.31**	0.67	**0.418**	0.177	0.81	0.291
CREATOR	boosted P	paragraph	yes	**0.21**	0.95	**0.341**	0.073	0.964	0.136
PLATFORM	boosted P	paragraph	yes	**0.65**	0.92	**0.758**	0.288	1	0.447
PROGLANG	boosted P or none	paragraph	yes	**1**	0.6	**0.75**	0.115	1	0.206
TGTLANG	boosted P	line	no	**0.24**	0.45	**0.317**	0.041	0.761	0.078
EMAIL	boosted P	paragraph	no	**0.36**	**0.98**	**0.524**	0.173	0.975	0.294
AVERAGE	-	-	-	**0.46**	0.78	**0.53**	0.163	0.903	0.276

Table 6: Comparison of accuracy using the best settings for each field against the typical parameter settings.

	Settings			One-best filler selection	Hand-crafted rules
	Thresholding	Fragmentation	Inst. sel.		
NAME	boosted P	paragraph	no	**0.527**	0.424
AREA	boosted P	sentence	yes	**0.705**	0.211
CREATOR	irrelevant	sentence	no	**0.639**	0.402
PLATFORM	irrelevant	sentence	yes	**1**	0.472
PROGLANG	boosted P or none	sentence	yes	**1**	0.443
TGTLANG	irrelevant	paragraph	no	**0.849**	0.016
EMAIL	irrelevant	paragraph	no	**0.276**	0.108
AVERAGE	-	-	-	**0.714**	0.129

Table 7: The F-measures of the best filler identification algorithm vs. hard-crafted rules.

About the effects of using Anaphora Resolution in assessing free-text student answers

Diana Pérez[1], **Oana Postolache**[3], **Enrique Alfonseca**[1], **Dan Cristea**[2] and **Pilar Rodriguez**[1*]

[1]Dpt. of Computer Science U. Autonoma Madrid (Spain)
[2]Dpt. of Computer Science U. of Iasi (Romania)
[3]Dpt. of Computational Linguistics U. of Saarland (Germany)

{Diana.Perez,Enrique.Alfonseca,Pilar.Rodriguez}@uam.es
dcristea@infoiasi.ro and oana@coli.uni-sb.de

Abstract

In this paper we present a possibility for integrating Anaphora Resolution (AR) in a system to automatically evaluate students' free-text answers. An initial discussion introduces some of the several methods that can be tried out. The implementation makes use of the AR-Engine RARE (Cristea *et al.* 02), integrated into the free-text answers assessor Atenea (Alfonseca & Pérez 04) to test these methods. RARE has been applied to find coreferential chains, and it has been found useful to extend the set of reference answers used by Atenea, by generating automatically new correct answers.

1 Introduction

Computer Assisted Assessment (CAA) is a field that studies how a computer can be used to assess students. One of its subfields, that has recently attracted much attention, focuses on assessing free-text answers. It is a quite complex task, still far from being completely solved. Thus, many systems are being developed, relying on various techniques. A classification of these techniques with examples of existing systems that use them is given in (Perez 04):

- **Statistical techniques**: they are based on some kind of statistical analysis, such as word frequency counts, or Latent Semantic Analysis (LSA) (Landauer *et al.* 01).
- **Text Categorisation Techniques (TCT)**: they are applicable when the student's answer can be classified as right or wrong, or inside a category in a scale of grades, e.g. bad, intermediate, good and very good (Larkey 98).
- **Information Extraction techniques**: they are used by systems which acquire structured information from free text, for example dependencies between concepts as in Automark (Mitchell *et al.* 02).
- **Full Natural Language Processing (NLP)**: NLP techniques, such as parsing and rhetorical analysis, can be used to gather more information about the student's answer. A system that applies NLP techniques is C-rater (Burstein *et al.* 01).
- **Clustering**: these techniques group essays that have similar words patterns to form a cluster with the same score. This is the approach followed by the Intelligent Essay Marking System (Ming *et al.* 00).
- **Hybrid approaches**: they combine several techniques to achieve better results. For instance, E-rater (Burstein *et al.* 98) and Atenea (Alfonseca & Pérez 04) use statistical and NLP techniques.

Although the techniques may seem very different, the general idea that underpins all these systems is the same: to compare the student's answer (or candidate answer) with the teacher's ideal answer (or reference answer). The closer they are, the higher the student's score is.

A problem to be able to compare the results of all these systems with each other is that, currently, there are not any standard evaluation corpora and metrics. Concerning the evaluation metrics, the one that is commonly used is the Pearson correlation between the teachers' and the system's scores on the same data set (Valenti *et al.* 03; Perez 04). The state-of-the-art results are between 30% and 93%, because the corpora used have very different degrees of difficulty.

Among the NLP techniques that can be employed to improve the automatic assessing of open-ended questions, Anaphora Resolution (AR), the process of finding the antecedent of an anaphora, could be considered as well. This language phenomenon, consisting of referring to a previously mentioned entity, is quite common in written language (Vicedo & Ferrández 00). Moreover, it has been successfully applied to other fields (Cristea *et al.* 05).

Previous authors have also mentioned that AR will probably be useful for free-text CAA (Valenti

*This work has been sponsored by Spanish Ministry of Science and Technology, project number TIN2004-0314.

et al. 03). However, to our knowledge, still there are no studies indicating the impact of applying AR to automatic assessment of free-text answers. Therefore, the main motivation of this paper is to study the effects of using AR integrated with the Atenea system. The AR-engine chosen is RARE (Cristea *et al.* 02). Our initial hypothesis was that somehow it would improve the accuracy of the assessment.

The first step to accomplish our aim has been to decide the way in which AR will be integrated with Atenea. The experimental framework given by the integration of RARE in Atenea has made possible to try several different uses of AR for free-text CAA. The indicator of the appropriateness of the procedure has been measured with the Pearson correlation between the teachers' and the system's scores. The results show that the application of AR directly on the student's answers does not improve the results in our case. On the other hand, AR has been found useful for generating automatically many alternative references and in this way, it slightly increases Atenea's assessment accuracy.

The paper is organised as follows: Section 2 presents the description of the possible uses that AR has in CAA of free-text answers. In Section 3, the implementation used in the experiments to test the previously mentioned methods is shown. Finally, in Section 4 several conclusions are drawn and future work is outlined.

2 Possible uses of AR in free-text CAA

Most of the systems for evaluating open-ended questions compare the student's candidate answer with reference answers written by the teachers. Therefore, the system will not be able to evaluate correctly an answer if the word choice or the expression used by the student and the teacher are different. We can try to solve this problem on both sides:

- Reducing the possible paraphrasings of each text, for instance, by eliminating all the pronouns and some definite NPs, using Anaphora Resolution.
- Extending the set of references with alternative paraphrasings. This can be done manually by asking several teachers to write alternative answers for the same question, or automatically, for instance, expanding the text with synonyms of the words used, or using AR, as described below.

Concerning the **reduction of paraphrasing** in a text, it is well known that there are many different expressions that have the same meaning. One of the sources for paraphrasing stems from the fact that there are many ways to refer to a previously mentioned entity by using an anaphoric expression. AR could help by identifying the referential expressions (REs) for the same referents, and gathering them in coreferential chains. Once coreferential chains are found, we have designed three ways in which they can be used:

1. *First-NP*: Each NP in the candidate and in the reference answers is substituted for the first NP in the coreferential chain. The aim is to filter the paraphrasing by substituting all NPs which refer to the same concept for the first NP used.
   For instance, let us suppose that we are scoring the candidate answer

   (1) Unix is an operating system. It is multiuser.

   and we apply this method to help in the comparison between this text and the references. The AR-engine RARE says that *Unix*, *operating system* and *It* are coreferential REs. Therefore, all of them will be substituted by the first RE (*Unix*). Therefore, the answer will be transformed into

   (2) Unix is Unix. Unix is multiuser

   Note that RARE considers the relationship between the subject and the predicative noun as coreferential as indicated in the MUC annotation guidelines (Hirschman *et al.* 97).
2. *All-NPs*: Each NP in the candidate and the reference answers is substituted for the whole coreferential chain to which it belongs. In this way, the candidate and reference answers will match if the intersection between the coreferential chains, considered as sets, is not empty. The third person singular personal pronouns *it* are excluded from these chains because most of the coreferential chains contain them.
   Thus, the candidate answer (1) will be transformed into

   (3) {an operating system,Unix} is {an operating system,Unix}. {an operating system,Unix} is multiuser

3. *Only-it*: Only the *it* pronouns in the candidate and the reference answers are substituted for the first NP in the coreferential chain which is not an *it*. This has been considered relevant enough to be studied given the extremely high frequency of this pronoun in the student answers in our test sets. This technique will also avoid the problem mentioned before with the predicative NPs.
   Thus, the resulting candidate answer for (1) would be
   (4) Unix is an operating system. Unix is multiuser.

Concerning the creation of new **reference answers** with alternative paraphrasings, we have also considered the possibility of applying AR in this task. While in the previous methods AR was applied to both the candidate and the reference answers, in this method it only affects the reference answers. The motivation is that the quality of the references is crucial, since they are the texts to which the students' answers are compared. Therefore, the usual practise of getting new references is to ask teachers to write these references.

However, as this is very cost and time consuming, we have also considered the automatic generation of new reference answers. It can be done by replacing automatically the NPs in the coreferential chains with other referential entities of those NPs. For instance, if we consider that (1) is a reference written by a teacher, two new references can be generated from its coreferential chain [Unix,an operating system,it]:"Unix is an operating system. Unix is multiuser" and "Unix is an operating system. An operating system is multiuser".

3 Implementation

3.1 Atenea

Atenea (Alfonseca & Pérez 04) is a CAA system for automatically scoring students' short answers. It has already been tested with English and Spanish texts and it could be easily ported to other languages. It works by processing the student's and teacher's answers according to several or all of the following NLP techniques, using the `wraetlic` tools (Alfonseca 03)[1]:

- **Stemming**: To be able to match inflected nouns or verbs.
- **Removal of closed-class words**: To be able to ignore them.
- **Word Sense Disambiguation**: To identify the sense intended by both the teacher and the student.

Then, the processed answers enter in the comparison module (ERB) that calculates the student's score and generates the student's feedback. This module is based on a modification of the n-gram co-occurrence scoring BLEU algorithm (Papineni *et al.* 01). The modification is necessary to take into account not only the precision but also the recall (Alfonseca & Pérez 04). The pseudocode of ERB is as follows:

1. For each value of N (typically from 1 to 3), calculate the Modified Unified Precision (MUP_N) as the percentage of N-grams from the candidate answer that appears in any of the reference texts. It will be clipped by the maximum frequency with which it appears in any of the references.
2. Calculate the weighted linear average of MUP_N obtained for each value of N. Store it in $combMUP$.
3. Calculate the Modified Brevity Penalty (MBP) factor, which is intended to penalise answers with a very high precision, but which are too short, to measure the recall:
   (a) For N from a maximum value (e.g. 10) down to 1, look whether each N-gram from the candidate text appears in any reference. In that case, mark the words from the found N-gram, both in the candidate and in the reference.
   (b) For each reference text, count the number of words that are marked, and calculate the percentage of the reference that has been found in the student's answer.
   (c) The MBP factor is the sum of all those percentage values.
4. The final score is the result of multiplying the MBP factor by $e^{combMUP}$.

The answer will be returned to the student, together with a score and a feedback based on a colour code, in which the parts of the student's answer which appear in the references are marked with a darker background (see Figure 1).

3.2 RARE

RARE (Robust Anaphora Resolution Engine) allows the design, implementation and evaluation of different multilingual anaphora resolution models

[1] www.ii.uam.es/~ealfon/eng/research/wraetlic.html

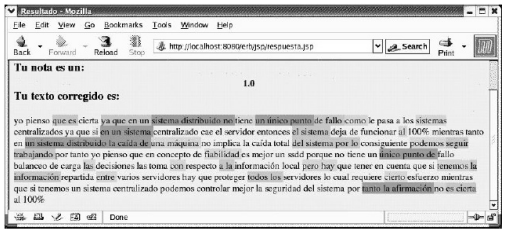


Figure 1: Feedback for the student, and score.

on free texts. The engine (Cristea *et al.* 02; Postolache & Forascu 04) has successfully been integrated into a discourse parser (Cristea *et al.* 05) and a time tracking approach (Puscasu 04). It allows postponed resolution and deals with several varieties of anaphora from only pronominal anaphora to more complex types such as bridging anaphora. The information is organised in RARE on three layers:

1. **The text layer**: It is composed by the words that form the discourse and it is populated with the referential expressions (REs). For example, in the candidate answer "Unix is an operating system. It is multiuser", "Unix", "operating system" and "it" are the REs.
2. **The projection layer**: This layer stores information about the found REs in feature structures called projection structures (PSs) to help in determining which ones are coreferential.
3. **The semantic layer**: The REs represent entities from the real world. The underlying meaning of the REs is treated in the semantic layer on the form of Discourse Entities (DEs).

It is said that a PS is projected from an RE and a DE is proposed or evoked by a PS. The process should be done from left to right in languages that are read in that way and vice versa from those read from right to left. Irrespectively of the language, the necessary features for any AR model to be used in RARE are (Cristea & Dima 01):

- **A set of primary attributes**: indicating, for example, morphological, syntactic, semantic or positional information.
- **A set of knowledge resources**: such as a part-of-speech tagger and an NP extractor to fill in the primary attributes to be stored in the PSs.
- **A set of heuristics or rules**: for each RE they decide if it refers to a new DE or to an already existing one.

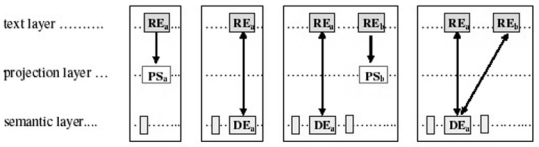


Figure 2: RARE layers.

- **A domain of referentiality**: it says where, how many and the order in which the DEs have to be checked.

The phases in the processing done by RARE are as follows (see Figure 2):

1. A referential expression RE_a is projected from the text layer into a feature structure PS_a on the projection layer. At this moment, the engine searches the space of existing discourse entities in order to recognise one against which the newly projected structure matches the best.
2. If no such DE is found, the projected structure PS_b is transformed in a new discourse entity DE_a, on the semantic layer, and disregarded from the projection layer. As the text unfolds, a new referential expression RE_b can be found on the text layer and, in its turn, projected as PS_b.
3. If PS_b matches an already existing discourse entity DE_a, with the meaning that their respective referential expressions, RE_a and RE_b, are coreferential. If this happens, PS_b is combined with DE_a and, subsequently, is disregarded from the projected layer.
4. Finally, chains of coreferential expressions are linked to the same object of the semantic layer, signifying that a unique discourse entity is evoked by all REs of the chain.

3.3 Techniques to use RARE in Atenea

The use of RARE as a new NLP module in Atenea requires the introduction of a new pre-initial phase to perform the pre-processing necessary to RARE. This phase includes a Functional Dependency Grammar (FDG) parsing of the text and the transformation of its result into an intermediate format understandable by RARE and Atenea. This format is a table in which each row represents a chain and, for each row, there are as many cells as NPs are in the chain. For the example candidate text (1) from Section 2, the equivalence table would have just one row (as it only has one chain) and it would be: *[Unix, an*

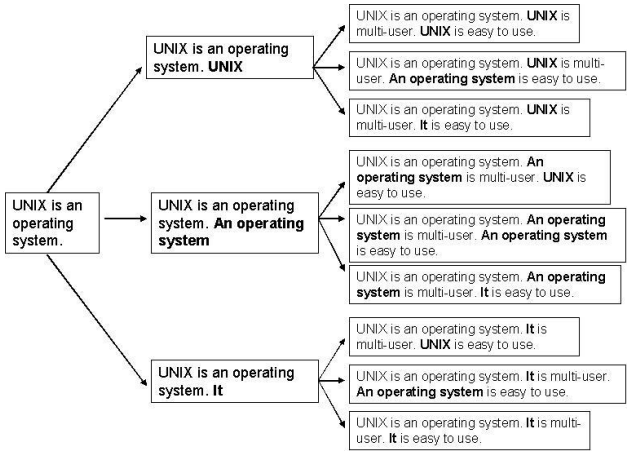


Figure 3: Example of the generation of new references from the original text "Unix is an operating system. It is multi-user. It is easy to use".

operating system, it].

The next step varies according to the method chosen. If it is is **First-NP** then each NP found in a row of the equivalence table is replaced by the first NP which is not an "it" in the chain. For **All-NPs** each NP found in a row of the equivalence table is replaced by the whole chain as a set. Finally, for **Only-it** each non-pleonastic "it" found in a row of the equivalence table is replaced by the first NP which is not an "it" in the coreferential chain.

Secondly, to implement the procedure for automatically generating *new paraphrases of the reference texts*, the following pseudocode has been used. It starts with one reference text that has been written by hand by a teacher.

1. Initialise an empty array $genRefTexts$ with the reference text.
2. Look for the next non-pleonastic "it". If none is found, stop.
3. Identify the row of the table that contains the coreferential chain which includes the "it" pronoun found.
4. Create as many copies of all the references in $genRefTexts$ as NPs exist in the coreferential chain. For each of the copies, the last "it" found has been replaced by each possible RE.
5. Go back to the second step.

Figure 3 shows an execution example.

3.4 Evaluation

For evaluation purposes, we have used a corpus composed of four sets of answers written by Spanish students in real exams about Operating Systems and a set of definitions of *Operating System*, retrieved from *Google glossary* in English.

N	NC	MC	NR	MR	Type
1	79	51	3	42	Def.
2	143	48	7	27	A/D
3	295	56	8	55	A/D
4	117	127	5	71	Y/N
5	38	67	4	130	Def.
M	134.4	69.4	5.4	65	-

Table 1: Answer sets used in the evaluation. Columns indicate: set number; number of candidate texts, mean length of the candidate texts (no. of words), number of references, mean length of the references, question type (Def.=definitions; A/D=advantages/disadvantages; Y/N=justified Yes/No).

N	ERB	S	C	S+C	W	W+C
1	0.5323	0.4337	0.5479	0.5310	0.4176	0.4841
2	0.6442	0.6899	0.6066	0.7567	0.6998	0.7655
3	0.2201	0.2426	0.3213	0.3459	0.2358	0.3282
4	0.3121	0.3326	0.3450	0.3754	0.3150	0.3586
5	0.5868	0.6007	0.5663	0.5702	0.6194	0.5919
M	0.4591	0.4599	0.4774	0.5158	0.4575	0.5057

Table 2: Results of Atenea without RARE, with (ERB) the statistical module, (S) stemming, (C), closed-class words removal, (W) word-sense disambiguation, and a combination of the previous procedures.

Given that RARE only works in English, we have been forced to translate the first four datasets into English. The translation has been done with Altavista Babelfish[2]. In previous work (Pérez *et al.* 05), we have observed that the variation in accuracy of Atenea is not statistically significant when Babelfish is used to translate the texts (Pérez *et al.* 05), as the correlations got are very similar to the correlations when evaluating with the original texts.

The five data sets are described in Table 1. Table 2 shows the results, measured as the Pearson correlation between Atenea's scores and the teachers' scores, for several of Atenea's configurations without using RARE.

Reduction of paraphrasing The first experiment explores the impact of the reduction of paraphrasing both in the candidate answers and the references. The correlation between the teachers' and the system's scores has been calculated using the different settings of the system. The FDG-parsing of these data sets was done with the on-

[2]http://world.altavista.com/

N	FNP	ANP	It	NGR	ERB	S	C	S+C	W	W+C
1	0.5217	0.2506	0.5176	3	0.5212	**0.4688**	**0.5824**	**0.5501**	**0.4405**	**0.4951**
2	0.5984	0.5107	0.6337	8	0.6442	0.6355	**0.6667**	0.7094	0.6537	0.7199
3	0.1731	0.0209	0.1529	17	**0.2218**	0.2370	0.3083	0.3390	0.2255	0.3238
4	0.2102	0.1878	0.2222	13	0.2918	0.2853	**0.3806**	**0.4233**	0.2745	**0.4182**
5	0.5799	0.0239	**0.5941**	36	**0.5964**	**0.6141**	0.5607	**0.5903**	**0.6208**	**0.6054**
M	0.4167	0.1968	0.4241	15.4	0.4551	0.4481	**0.4997**	**0.5224**	0.443	**0.5125**
1m	**0.5806**	*0.4655*	**0.5498**	*3*	**0.5736**	**0.5373**	**0.5727**	**0.5597**	**0.5270**	**0.5608**

Table 3: Results achieved using Atenea with RARE. The first three columns show the results for reducing paraphrasing, using just the statistical ERB module: *First-NP* (FNP), *all-NPs*(ANP), and *only-It* (It). The other columns show the results when creating new references, tested with all of Atenea's configurations. Columns indicate the Number of Generated References (NGR), and the results with ERB, stemming (S), closed-class words removal (C), Word Sense Disambiguation (W) and several combinations between them. The last row, called *1m*, shows the results working with a manual translation of set 1 rather than with Babelfish's output

line demo of Connexor[3].

Table 3 (first three columns) shows the correlation values for different configurations of Atenea using RARE. The columns contain the results for each of the three heuristics. The bold font figure indicates the case in which using RARE has improved the result over the original ERB.

Contrary to our intuition, the results show that there is no significant improvement in using RARE and, in some cases, such as in the *all-NPs* method, the correlations decrease for all data sets. Therefore, our conclusion is that AR is not useful to improve the results of n-gram co-occurrence similarity metrics. However, as can be seen in row labelled *1m*, the correlations of all three strategies greatly improved when we work with a set of manual translations, and in two of them we obtain a higher correlation than when we worked without RARE (Table 2).

Creation of new references RARE has also been used to create new references by substituting the non-pleonastic it pronouns with all its Referential Expressions. Table 3, in its last seven columns, shows the results for several of Atenea's configurations. It can be seen that the use of RARE has improved three of the five configurations under test (C, S+C and W+C). Using RARE, the best configuration is to combine stemming and closed-class word removal.

Concerning the use of Set 1 translated manually, it can be seen that it also improves several of the configurations; however, the best results for set 1 are obtained when using the automatic translation and closed-class word removal, with a correlation that also exceeds that of the experiments on reduction of paraphrasing.

[3]http://www.connexor.com/

N	S	C	S+C	W	W+C
1	**0.4453**	**0.5677**	**0.4901**	**0.4195**	0.4356
2	0.6563	**0.6277**	0.6906	0.6756	0.7059
3	0.2288	0.2735	0.3192	0.2031	0.2746
4	**0.3449**	0.3126	0.3025	**0.3261**	0.2827
5	**0.6332**	0.5643	**0.5959**	**0.6529**	**0.6078**
M	**0.4617**	0.4692	0.4797	0.4554	0.4613

Table 4: Results achieved by Atenea using several NLP modules and the method of manually generating new references.

Finally, in order to study the effect of using RARE rather than any other Anaphora Resolution module, a last experiment has been performed by annotating the co-referential chains by hand. Table 4 shows that the results do not have a dramatic improvement, and even in some cases the correlation decrease when compared with the results using RARE. The reason is probably that RARE is probably more consistent in its answers (either correct or wrong) than a human annotator.

4 Conclusions and future work

In this paper, Anaphora Resolution has been applied to the task of automatically assessing students' free-text answers. In particular, the AR-engine RARE has been integrated into Atenea, to test four proposed methods: **first-NP**, in which the NPs are replaced by the first RE which is not the "it" pronoun; **all-NPs**, in which the NPs in the candidate and reference's texts are replaced by the whole coreferential chain; **only-it**, in which

only the "it" pronouns are replaced by the first RE; and the **automatic generation of variable references** from the original reference text, to automatically obtain new variants by replacing each non-pleonastic "it" with all the possible NPs in its coreferential chain.

From the results obtained, we can draw several interesting conclusions:

1. Previous findings indicated that BLEU-like algorithms produced consistent results on data that had been processed by MT engines (Pérez *et al.* 05). That was specially useful, in our case, in order to provide adaptation to the student's language without any intervention by the teacher. However, if we want to incorporate more sophisticated NLP steps, such as the reduction of redundancies using Anaphora Resolution, MT may not be adequate.

   On the other hand, MT is still acceptable using the procedure of automatic generation of references, in which the results increased with the use of RARE, and the best results have been obtained with the output of the MT engine.

2. The worst results have been obtained in the All-NPs configuration. We believe that it is due to the fact that the number of times that the candidate and the reference matches may be artificially inflated when the referential NPs are substituted by their REs. This is specially evident in the all-NPs experiment. We believe that there has not been much improvement because of the characteristics of the n-gram co-occurrence metric used.

3. Concerning the generation of new references, the results are slightly better, and the average correlation increases up to 52%. Furthermore, this method opens a promising line of future work that could be further exploited to automatically generate new references (for instance, with synonyms of the words in the references).

Other lines of future work are the following: to improve the AR model with features specific to the types of answers to be processed, to finish the development of the Spanish anaphora resolution model for RARE, and to test more possibilities for using RARE with Atenea.

References

(Alfonseca & Pérez 04) E. Alfonseca and D. Pérez. Automatic assessment of short questions with a BLEU-inspired algorithm and shallow nlp. In *Advances in Natural Language Processing*, volume 3230 of *Lecture Notes in Computer Science*, pages 25–35. Springer Verlag, 2004.

(Alfonseca 03) E. Alfonseca. Wraetlic user guide version 1.0, 2003.

(Burstein *et al.* 98) J. Burstein, K. Kukich, S. Wolff, C. Lu, M. Chodorow, L. Bradenharder, and M. Dee Harris. Automated scoring using a hybrid feature identification technique. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics*, 1998.

(Burstein *et al.* 01) J. Burstein, C. Leacock, and R. Swartz. Automated evaluation of essays and short answers. In *Proceedings of the International CAA Conference*, 2001.

(Cristea & Dima 01) D. Cristea and G.E. Dima. An integrating framework for anaphora resolution. *Information Science and Technology*, 4(3), 2001.

(Cristea *et al.* 02) D. Cristea, O. Postolache, G.E. Dima, and C. Barbu. Ar-engine - a framework for unrestricted co-reference resolution. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC)*, 2002.

(Cristea *et al.* 05) D. Cristea, O. Postolache, and I. Pistol. Summarisation through discourse parsing. In *Proceedings of CICLING 2005*, 2005.

(Hirschman *et al.* 97) Hirschman, Lynette, and Chinchor. Muc-7 coreference task definition, version 3.0. In *MUC-7 Proceedings*, 1997. See also: http://www.muc.saic.co.

(Landauer *et al.* 01) T.K. Landauer, D. Laham, and P.W. Foltz. The intelligent essay assesor: putting knowledge to the test. In *Proceedings of the Association of Test Publishers Computer-Based Testing: Emerging Technologies and Opportunities for Diverse Applications conference*, 2001.

(Larkey 98) L. S. Larkey. Automatic essay grading using text categorization techniques. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 90–95, 1998.

(Ming *et al.* 00) Y. Ming, A. Mikhailov, and T.L. Kuan. Intelligent essay marking system. *Learners Together*, 2000.

(Mitchell *et al.* 02) T. Mitchell, T. Russell, P. Broomhead, and N. Aldridge. Towards robust computerised marking of free-text responses, 2002.

(Papineni *et al.* 01) K. Papineni, S. Roukos, T. Ward, and W. Zhu. BLEU: a method for automatic evaluation of machine translation. Research report, IBM, 2001.

(Perez 04) D. Perez. Automatic evaluation of users' short essays by using statistical and shallow natural language processing techniques. Advanced Studies Diploma (Escuela Politécnica Superior, Universidad Autónoma de Madrid), 2004.

(Pérez *et al.* 05) D. Pérez, E. Alfonseca, and P. Rodríguez. Adapting the automatic assessment of free-text answers to the students profiles. In *Proceedings of the CAA conference*, Loughborough, U.K., 2005.

(Postolache & Forascu 04) O. Postolache and C. Forascu. A coreference model on excerpt from a novel. In *Proceeding of The European Summer School in Logic Language and Information - ESSLLI'2004*, Nancy, France, 2004.

(Puscasu 04) G. Puscasu. A framework for temporal resolution. In *Proceedings of the Language Resources and Evaluation Conference (LREC-2004)*, 2004.

(Valenti *et al.* 03) S. Valenti, F. Neri, and A. Cucchiarelli. An overview of current research on automated essay grading. *Journal of Information Technology Education*, 2:319–330, 2003.

(Vicedo & Ferrández 00) J.L. Vicedo and A. Ferrández. Importance of pronominal anaphora resolution to question answering systems. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 555–562, 2000.

Evaluating corpus query systems on functionality and speed: TIGERSearch and Emdros

Ulrik Petersen
Department of Communication, University of Aalborg
Kroghstræde 3
9220 Aalborg East, Denmark
ulrikp@hum.aau.dk
http://emdros.org/

Abstract

In this paper, we evaluate two corpus query systems with respect to search functionality and query speed. One corpus query system is TIGERSearch from IMS Stuttgart and the other is our own Emdros corpus query system. First, we show how the database model underlying TIGERSearch can be mapped into the database model of Emdros. Second, the comparison is made based on a set of standard linguistic queries culled from the literature. We show that by mapping a TIGERSearch corpus into the Emdros database model, new query possibilities arise.

1 Introduction

The last decade has seen a growth in the number of available corpus query systems. Some query systems which have seen their debut since the mid-1990ies include MATE Q4M (Mengel 99), the Emu query language (Cassidy & Bird 00), the Annotation Graph query language (Bird *et al.* 00), TGrep2 (Rohde 04), TIGERSearch (Lezius 02b), NXT Search (Heid *et al.* 04), Emdros (Petersen 04), and LPath (Bird *et al.* 05). In this paper, we have chosen to evaluate and compare two of these, namely TIGERSearch and Emdros.

TIGERSearch is a corpus query system made at the Institut für Maschinelle Sprachverarbeitung at the University of Stuttgart (Lezius 02a; Lezius 02b). It is a general corpus query system over so-called *syntax graphs* (König & Lezius 03), utilizing the TIGER-XML format for import (Mengel & Lezius 00). Converters have been implemented for the Penn Treebank, NeGRA, Susanne, and Christine formats, among others. It is available free of charge for research purposes.[1]

Emdros is also a general corpus query system, developed at the University of Aalborg, Denmark. It is applicable to a wide variety of linguistic corpora supporting a wide variety of linguistic theories, and is not limited to treebanks. It implements the EMdF model and the MQL query language described in (Petersen 04). Importers for the TIGER-XML and other corpus formats have been implemented, and more are under development. It is available free of charge as Open Source software from the address specified at the beginning of the paper.

The layout of the rest of the paper is as follows. First, we briefly introduce the EMdF database model underlying Emdros. Second, we introduce the database model underlying TIGERSearch. Next, we show how to map the TIGERSearch database model into the EMdF model. The next section explores how the TIGERCorpus (Brants & Hansen 02), now in Emdros format, can be queried with – in some instances – greater functionality and speed by Emdros than by TIGERSearch. Finally, we conclude the paper.

[1] See http://www.tigersearch.de/

2 The EMdF model of Emdros

The EMdF text database model underlying Emdros is a descendant of the MdF model described in (Doedens 94). At the backbone of an EMdF database is a string of *monads*. A monad is simply an integer. The sequence of the integers dictates the logical reading sequence of the text. An *object* is an arbitrary (possibly discontiguous) set of monads which belongs to exactly one *object type*. An object type (e.g., Word, Phrase, Clause, Sentence, Paragraph, Article, Line, etc.) determines what *features* an object has. That is, a set of attribute-value pairs are associated with each object, and the attributes are determined by the object type of the object. All attributes are strongly typed. Every object has a database-widely unique ID called its *id_d*, and the feature *self* of an object denotes its id_d. The notation $O.f$ is used to denote the value of feature f on an object O. Thus, for example, $O_1.self$ denotes the id_d of object O_1. An id_d feature can have the value NIL, meaning it points to no object. No object can have NIL as its id_d.

The sample tree in Figure 1 shows a discontiguous element, and is adapted from (McCawley 82, p. 95). The tree can be visualized as an EMdF database as in Figure 2. This figure exemplifies a useful technique used for representing tree-structures in Emdros: Since, in a tree, a child node always has at most one parent, we can represent the tree by means of id_d features pointing upwards from the child to its parent. If a node has no parent (i.e., is a root node), we can represent this with the value NIL. This technique will be used later when describing the mapping from TIGERSearch to EMdF.

3 The TIGERSearch database model

The database model underlying TIGERSearch has been formally described in (Lezius 02a) and (König & Lezius 03). The following description has been adapted from the former, and is a slight reformalization of the database model with respect to edge-labels.

Definition 1 A *feature record* F is a relation over $FN \times C$ where FN is a set of feature-names and C is a set of

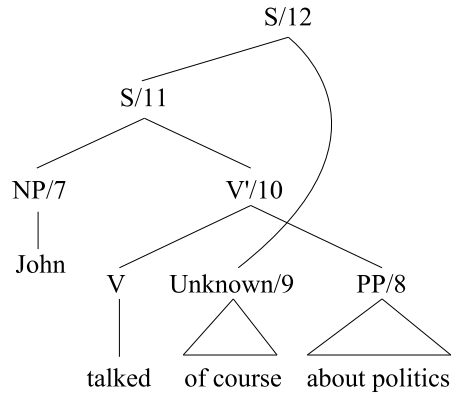


Figure 1: A tree with a discontiguous clause, adapted from (McCawley 82, p. 95).

	1	2	3	4	5	6
Word	id_d: 1 surf.: John pos: NProp parent: 7	id_d: 2 surf.: talked pos: V parent: 10	id_d: 3 surf.: of pos: P parent: 9	id_d: 4 surf.: course pos: N parent: 9	id_d: 5 surf.: about pos: P parent: 8	id_d: 6 surf.: politics pos: N parent: 8
Phrase	id_d: 7 type: NP parent: 11		id_d: 9 type: Unknown parent: 12		id_d: 8 type: PP parent: 10	
Phrase		id_d: 10 type: V' parent: 11			id_d: 10 type: V' parent: 11	
Clause	id_d: 11 type=S parent: 12				id_d: 11 type=S parent: 12	
Clause	id_d: 12 type=S					

Figure 2: An EMdF representation of the tree in Figure 1.

constants. The relation is defined such that for any $l_i = \langle f_i, c_i \rangle$ and any $l_j = \langle f_j, c_j \rangle$, $l_i \neq l_j \Rightarrow f_i \neq f_j$. That is, all f_i within a feature-record are distinct. The set of all feature-records over FN and C is denoted $\mathcal{F}$.

Definition 2 The set of all node ids is called ID and the relation $ID \subset C$ holds.

Definition 3 A *node* is a two-tuple $v \in ID \times \mathcal{F}$. That is, a node consists of a node id ν and a feature-record F.

Definition 4 A *syntax graph* G in the universe of graphs $\mathcal{G}$ is a six-tuple $G = (V_{NT}, V_T, L_G, E_G, O_G, R_G)$ with the following properties:

1. V_{NT} is the (possibly empty) set of non-terminals.
2. V_T is the non-empty set of terminals.
3. L_G is a set of edge labels where $L_G \subset C$.[2]
4. E_G is the set of labeled, directed edges of G. E_G is a set of two-tuples from $V_{NT} \times (V_{NT} \cup V_T)$. If L_G is non-empty, there exists an assignment of edge-labels el which is a total function $el : E_G \rightarrow L_G$ which need be neither surjective nor injective.[3]
5. O_G is a bijective function $O_G : V_T \rightarrow \{1, 2, \ldots, |V_T|\}$ which orders the terminal nodes. That the function is bijective guarantees that all terminal nodes can be ordered totally by O_G.
6. $R_G \in V_{NT}$ is the single root node of G, and has no incoming edges.

G is a graph with the following characteristics:

G1: G is a DAG with exactly one root node R_G.

G2: All nodes $v \in ((V_{NT} \cup V_T) \setminus R_G)$ have exactly one incoming edge in E_G.

G3: All nonterminals $v \in V_{NT}$ must have at least one outgoing edge. That is, $\forall v \in V_{NT} \exists v' \in (V_{NT} \cup V_T) : \langle v, v' \rangle \in E_G$.[4]

Thus syntax graphs are not strict trees in the traditional sense, since crossing edges are not prohibited. Nevertheless, syntax graphs are not arbitrary DAGs, since by **G2**, every node has at most one parent, and in this respect they do resemble trees.

This brief reformulation does not do justice to the full description available in (Lezius 02a) and (König & Lezius 03). For more information on the syntax graph formalism, see the cited publications.

4 Mapping syntax graphs to EMdF

TIGERSearch was developed specifically for use with the TIGERCorpus (Brants & Hansen 02), though it is applicable to other corpora as well (Lezius 02a, p. 136). In order to compare TIGERSearch with Emdros, we had to import a corpus available for TIGERSearch into Emdros. The TIGERCorpus was chosen because it represents the primary example of a TIGERSearch database, and because it has a reasonably large size, furnishing a basis for speed-comparisons.

We have developed an algorithm to transform any database encoded in the syntax graph formalism into an EMdF database. This section describes the algorithm. First, we give some definitions, after which we show the four algorithms involved.

Definition A1: For any syntax graph G, Obj_G is the set of EMdF objects which G gives rise to, and IDD_G is the set of id_d's of the objects in Obj_G. Note, however, that IDD_G may be defined before Obj_G, since there is no causality in the direction from Obj_G to IDD_G; in fact it is the other way around in the algorithms below.

[2] The latter restriction is not mentioned by (Lezius 02a) directly on page 103 where this is defined, but is inferred from the rest of the dissertation.

[3] This is where our reformulation differs in meaning from (Lezius 02a). We think our formalization is slightly clearer than Lezius', but we may, of course, have misunderstood something.

[4] Again, my reformulation differs slightly from Lezius' formulation, due to my reinterpretation of E_G.

Definition A2: For any syntax graph G, NOB_G is a bijective function from syntax graph nodes in G to Obj_G. That is, $NOB_G : (V_{NT} \cup V_T) \rightarrow Obj_G$.

Definition A3: For any syntax graph G and $v \in (V_{NT} \cup V_T)$, $parent(v)$ is the parent node of v if v is not R_G, or $\emptyset$ if v is R_G.

Definition A4: For any syntax graph G and its concomitant Obj_G, id_d_G is a bijective function $id\_d_G : (V_{NT} \cup V_T) \rightarrow IDD_G$ with the definition $id\_d(v) ::= NOB_G(v).\text{self}$. Note, however, that this definition only holds *after* the algorithms have all been applied; in fact id_d_G is defined by construction rather than by the given intensional, after-the-fact definition.

With this apparatus, we can define four algorithms which use each other. Algorithm 0 merely creates an empty object with a unique EMdF id_d corresponding to each node in a syntax graph G. Algorithm 1 adds monads to all objects corresponding to a nonterminal (i.e., all syntax-level nodes). Algorithm 2 constructs a set of EMdF objects for a given syntax graph G, and uses Algorithm 0 and 1. Algorithm 3 constructs an EMdF database from a set $\mathcal{G}$ of syntax graphs, and uses Algorithm 2

Algorithm 0: *Purpose*: Create empty objects in Obj_G and assign id_ds to each object and to the id_d_G function and IDD_G.
Input: A syntax graph G and a starting id_d d.
Output: A four-tuple consisting of the function id_d_G, the set IDD_G, the set Obj_G, the set NOB_G and an ending id_d d_e.

1. let $id\_d_G := \emptyset$, and let $Obj_G := \emptyset$
2. For all nodes $v \in (V_{NT} \cup V_T)$ (the ordering does not matter, so long as each node is treated only once):
   (a) let $id\_d_G(v) := d$
   (b) Create an EMdF object O_d being an empty set of monads and let O_d.self := d
   (c) let $Obj_G := Obj_G \cup \{O_d\}$
   (d) let $IDD_G := IDD_G \cup \{d\}$
   (e) let $NOB_G := NOB_G \cup \langle v, O_d \rangle$
   (f) let $d := d + 1$
3. Return $\langle id\_d_G, IDD_G, Obj_G, NOB_G, d \rangle$.

Algorithm 1: *Purpose*: To add monads to all objects corresponding to a non-terminal.
Input: A non-terminal p, the set IDD_G, and the set Obj_G.
Output: Nothing, but Obj_G is changed. (Obj_G is call-by-value here, so it is changed as a side-effect and not returned.)

1. Let $Ch := \{c | parent(c) = p\}$ (all immediate children of p.
2. For all $c \in Ch$:
   (a) If $c \in V_T$: Let $IDD_G(parent(c)) := IDD_G(parent(c)) \cup IDD_G(c)$ (Add terminals' monad-set to parent.)
   (b) Else:
      i. Call ourselves recursively with the parameters $langlec, IDD_G, Obj_G\rangle$.
      ii. Let $IDD_G(parent(c)) := IDD_G(parent(c)) \cup IDD_G(c)$ (Add c's monad-set to parent.)

Algorithm 2: *Purpose*: To construct a set of EMdF objects from a syntax graph G.
Input: A syntax graph G, a starting id_d d, and a starting monad m.
Output: A three-tuple consisting of a set of EMdF objects Obj_G, an incremented id_d d_e and an ending monad m_e.

1. Call Algorithm 0 on $\langle G, d \rangle$ to obtain $\langle id\_d_G, IDD_G, Obj_G, NOB_G, d_e \rangle$.
2. For all terminals $t \in V_T$:
   (a) let $O_t := NOB_G(t) \cup \{m_t\}$ where $m_t = O_G(t) + m - 1$. (Remember that an object is a set of monads, so we are adding a singleton monad set here.)
   (b) Let O_t.parent $:= id\_d_G(parent(t))$ if t is not R_G, and NIL if t is R_G.
   (c) Assign other features of O_t according to the feature-record F in $t = \langle \nu, F \rangle$.[5]
   (d) if L_G is non-empty, let O_t.edge $:= el(\langle parent(t), t \rangle)$
3. Call Algorithm 1 with the parameters $\langle R_G, IDD_G, Obj_G \rangle$. This assigns monad sets to all objects.
4. For all v in V_{NT}:
   (a) Let $O_v := Obj_G(v)$.
   (b) Let O_v.parent $:= id\_d_G(parent(v_v))$ if v is not R_G, and NIL if v is R_G.
   (c) Assign other features of O_v according to the feature-record F in $v = \langle \nu, F \rangle$.
   (d) if L_G is non-empty, let O_t.edge $:= el(\langle parent(t), t \rangle)$
5. Return $\langle Obj_G, d, m_t \rangle$ where $m_t \equiv O_G(v_t) + m - 1$ where v_t is the rightmost terminal node, i.e., $\exists v_t \in V_T : \forall v_j \in V_T : v_j \neq v_t \Rightarrow O_G(v_t) > O_G(v_j)$

Algorithm 3: *Purpose*: To construct a set of EMdF objects from a universe of syntax graphs $\mathcal{G}$.
Input: A set of syntax graphs $\mathcal{G}$, a starting id_d d, and a starting monad m.
Output: A two-tuple consisting of an incremented id_d d_e and an ending monad m_e.

[5] It is assumed, though the formalisation does not say so, that the feature-records of all V_T in all $G \in \mathcal{G}$ have the same "signature", i.e., have the same set of feature-names that are assigned a value in each F in each $v \in V_T$. A similar assumption is made for the signatures of all feature-records of all V_{NT}. This is certainly the case with the TIGERCorpus. Therefore, the object type `Terminal` is well-defined with respect to its features. Similarly for the object type `Nonterminal` used below.

```
Q1. Find sentences that include the word 'saw'.
Q2. Find sentences that do not include the word 'saw'.
Q3. Find noun phrases whose rightmost child is a noun.
Q4. Find verb phrases that contain a verb immediately
    followed by a noun phrase that is immediately
    followed by a prepositional phrase.
Q5. Find the first common ancestor of sequences of a
    noun phrase followed by a verb phrase.
Q6. Not relevant to TIGER Corpus.
Q7. Find a noun phrase dominated by a verb phrase.
    Return the subtree dominated by that noun phrase.
```

Figure 3: The test queries from (Lai & Bird 04), Fig. 1.

```
Q1  #s:[cat="S"] & #l:[word="sehen"] & #s >* #l
Q2* #s:[cat="S"] & #l:[word="sehen"] & #s !>* #l
Q3  #n1:[cat="NP"] & #n2:[pos="NN"]  & (#n1 >@r #n2)
Q4  #vp:[cat="VP"] & #v:[pos="VVFIN"] & #np:[cat="NP"]
    & #pp:[cat="PP"]& #vp >* #v & #vp >* #np
    & #vp >* #pp & #v >@r #vr & #np >@l #npl
    & #vr .1 #npl & #np >@r #npr & #pp >@l #ppl
    & #npr .1 #ppl
Q5* #vp:[cat="VP"] & #np:[cat="NP"] & (#np .* #vp)
    & (#x >* #vp) & (#x >* #np)
Q7* #vp:[cat="VP"] & #np:[cat="NP"] & (#vp >* #np)
```

Figure 4: The test queries of Figure 3 attempted implemented in TIGERSearch. Adapted from (Lai & Bird 04), Fig. 4. The queries marked with a * may not produce the correct results.

1. For all graphs G in $\mathcal{G}$ (if an ordering is intended, i.e., this is not a quotation corpus, then that order should be applied; otherwise, the order is undefined):
   (a) Let $\langle Obj_G, d_e, m_e \rangle$ be the result of calling Algorithm 2 on $\langle G, d, m \rangle$
   (b) Add Obj_G to the EMdF database.
   (c) Let $d := d_e$ and let $m := m_e + 1$
2. Return $\langle d, m \rangle$

5 Comparing TIGERSearch and Emdros

Using a variant of this algorithm, we have imported the TIGERCorpus into Emdros. This gives us a common basis for comparing TIGERSearch and Emdros.

The paper (Lai & Bird 04) sets out to specify some requirements on corpus query systems for treebanks that the authors perceive to be essential. Among other criteria, Lai and Bird set up a set of standard queries which are reproduced in Figure 3.

Lai and Bird show how some of the queries can be expressed in TIGERSearch, though they find that not all queries can be expressed. I have attempted to reformulate Lai and Bird's TIGERSearch queries in therms of the TIGERCorpus (see Figure 4).

Query Q2 cannot be formulated correctly in TIGERSearch. This is because what is being negated is the *existence* of the word "sehen", and in TIGERSearch, all nodes are implicitly existentially quantified. Negated existence would require a forall-quantification, as mentioned e.g. in (König & Lezius 03).

Query Q5 is probably not expressible in TIGERSearch, and the given query fails to find the *first* common ancestor only. The currect syntax graphs are returned, but with a number of subgraphs which are not rooted in the first common ancestor.

```
Q1  [Sentence [Word surface="sehen"] ]
Q2  [Sentence NOTEXIST [Word surface="sehen"] ]
Q3  [Phrase tag="NP" [Word last postag="NN"] ]
Q4  [Phrase tag="VP"
       [Word postag="VVFIN"]!
       [Phrase tag="NP"]!
       [Phrase tag="PP"]
    ]
Q5* [Phrase
       [Phrase tag="NP"][Phrase tag="VP"]
    ]
Q7* [Phrase tag="VP"  [Phrase tag="NP"]  ]
```

Figure 5: Emdros queries for Q1-Q7

```
Find all NPs which is a subject, inside of which there
is a relative clause whose parent is the NP.  Inside
the relative clause, there must be a phrase p2, inside
of which there must be a word which is a cardinal.  At
the end of the relative clause must be a finite verb
whose parent is the same as that of p2.  No PP may
intervene between p2 and the verb.

[Phrase as p1 tag="NP" AND edge="SB"
   [Phrase edge="RC" and parent=p1.self
      [Phrase as p2  [Word postag="CARD"] ]
      ..
      NOTEXIST [Phrase tag="PP"]
      ..
      [Word last postag="VVFIN"
            AND parent=p2.parent]
   ]
]
```

Figure 6: Emdros query for Q8

Query Q7 again finds the correct syntax graphs, but fails to retrieve exactly the subtree dominated by the NP. In TIGERSearch, what parts of a matched syntax-graph to retrieve is, in a sense, an irrelevant question, since the main result is the syntax graph itself. Thus the assumption of Lai and Bird that only parts of the matched tree is returned does not hold for TIGERSearch.

Emdros fares slightly better as regards functionality, as can be seen in Figure 5. Query Q2 is correctly expressed in Emdros using the NOTEXIST operator at object-level, which gives Emdros a slight edge over TIGERSearch in this comparison. However, queries Q5 and Q7 fail to give correct results on Emdros as they did on TIGERSearch. Query Q5 fails because, while it returns the correct syntax graphs, it fails to find only the first common ancestor. This is the same situation as with TIGERSearch. As in TIGERSearch, the requirement to find the "first common ancestor" is difficult to express in Emdros. Query Q7 fails because Emdros, like TIGERSearch, was not designed to retrieve subgraphs as part of the query results – subgraphs are to be retrieved later, e.g., for viewing purposes. Like TIGERSearch, Emdros returns the correct syntax graphs, and thus works as designed.

Query Q8 can be seen in Figure 6 along with the Emdros equivalent. It cannot be expressed in TIGERSearch because of the negated existence-operator on the intervening PP.

The queries were all timed, except for Q2 and Q6, which were not expressible in either or both of the corpos query systems. The hardware was an AMD Athlon 64 3200+ with

Query	Emdros	TIGERSearch
Q1	0.199; 0.202; 0.179	0.5; 0.3; 0.3
Q3	1.575; 1.584; 1.527	10.1; 9.9; 9.9
Q4	1.604; 1.585; 1.615	9.9; 9.9; 9.9
Q5	3.449; 3.319; 3.494	5.5; 6.6; 5.5
Q7	0.856; 0.932; 0.862	1.1; 1.1; 1.1
Q8	3.877; 3.934; 4.022	N/A

Table 1: Execution times in seconds

1GB of RAM and a 7200RPM harddrive running Linux Fedora Core 4. Three measurements were taken for each query. In the case of TIGERSearch, the timings reported by the program's status bar were used. For Emdros, the standard Unix command `time` was used. The results can be seen in Table 1.

As can be seen, Emdros is faster than TIGERSearch on every query that they can both handle. (Lezius 02a) mentions that the complexity is exponential in the number of query terms. It is very difficult to assess the complexity of an Emdros query, since it depends on a handful of factors such as the number of query items, the number of objects that match each query item, and the number of possible combinations of these.

Probably Emdros is faster in part because it takes a different algorithmic approach to query resolution that TIGERSearch: Instead of using proof-theory, it uses a more linear approach of first retrieving all possible object-"hits", then iteratively walking the query, combining the objects in monad-order as appropriate. Part of the speed increase may stem from its being written in C++ rather than Java, but for queries such as Q3 and Q4, the algorithm rather than the language seems to be the decisive factor, since such a large difference in execution time, relative to the other increases, cannot be accounted for by language differences alone.

6 Conclusion

In this paper, we have compared two corpus query systems, namely TIGERSearch on the one hand and our own Emdros on the other. We have briefly introduced the EMdF model underlying Emdros. The EMdF model is based on the MdF model described in (Doedens 94). We have also given a reformalization of the syntax graph formalism underlying TIGERSearch, based on the presentation given in (Lezius 02a). We have then presented an algorithm for converting the syntax graph formalism into the EMdF model.

Having done this, we have compared the two corpus query systems with respect to query functionality and speed. The queries were mostly culled from the literature. It was found that Emdros was able to handle all the test queries that TIGERSearch was able to handle, in addition to a few that TIGERSearch was not able to express. The latter involved the negation of the existence of an object; it is a limitation in the current TIGERSearch that all objects are implicitly existentially quantified, which means that negating the existence of an object is not possible. Negation at the feature-level is, however, possible in both corpus query systems. In both systems, the semantics of feature-level negation is the same as the $\neg$ operator in First Order Logic.

Finally, the test queries which both systems were able to handle were executed on the same machine over the same corpus, namely the TIGERCorpus, and it was found that Emdros was faster than TIGERSearch on every query, and that the algorithm of Emdros seems to scale better than that of TIGERSearch.

References

(Bird *et al.* 00) Steven Bird, Peter Buneman, and Tan Wang-Chiew. Towards a query language for annotation graphs. In *Proceedings of the Second International Conference on Language Resources and Evaluation*, pages 807–814. European Language Resources Association, Paris, 2000. http://arxiv.org/abs/cs/0007023 Access Online August 2004.

(Bird *et al.* 05) Steven Bird, Yi Chen, Susan Davidson, Haejoong Lee, and Yifeng Zheng. Extending XPath to support linguistic queries. In *Proceedings of Programming Language Technologies for XML (PLANX) Long Beach, California. January 2005.*, pages 35–46, 2005.

(Brants & Hansen 02) Sabine Brants and Silvia Hansen. Developments in the TIGER annotation scheme and their realization in the corpus. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002), Las Palmas, Spain, May 2002*, pages 1643–1649, 2002. http://www.ims.uni-stuttgart.de/projekte/TIGER/paper/lrec2002-brants-hansen.pdf Access Online August 2004.

(Cassidy & Bird 00) Steve Cassidy and Steven Bird. Querying databases of annotated speech. In M.E. Orlowska, editor, *Database Technologies: Proceedings of the Eleventh Australasian Database Conference, volume 22 of Australian Computer Science Communications, Canberra, Australia*, pages 12–20. IEEE Computer Society, 2000. http://arxiv.org/abs/cs/0204026, Access Online August 2004.

(Doedens 94) Christianus Franciscus Joannes Doedens. *Text Databases: One Database Model and Several Retrieval Languages*. Number 14 in Language and Computers. Editions Rodopi, Amsterdam and Atlanta, GA., 1994.

(Heid *et al.* 04) U. Heid, H. Voormann, J-T Milde, U. Gut, K. Erk, and S. Pado. Querying both time-aligned and hierarchical corpora with NXT Search. In *Fourth Language Resources and Evaluation Conference, Lisbon, Portugal, May 2004*, 2004.

(König & Lezius 03) Esther König and Wolfgang Lezius. The TIGER language. a description language for syntax graphs. formal definition. Technical report, Institut für Maschinelle Sprachverarbeitung (IMS), University of Stuttgart, Germany, April 22 2003.

(Lai & Bird 04) Catherine Lai and Steven Bird. Querying and updating treebanks: A critical survey and requirements analysis. In *Proceedings of the Australasian Language Technology Workshop, December 2004*, pages 139–146, 2004.

(Lezius 02a) Wolfgang Lezius. *Ein Suchwerkzeug für syntaktisch annotierte Textkorpora*. Unpublished PhD thesis, Institut für Maschinelle Sprachverarbeitung, University of Stuttgart, December 2002. Arbeitspapiere des Instituts für Maschinelle Sprachverarbeitung (AIMS), volume 8, number 4. http://www.ims.uni-stuttgart.de/projekte/corplex/paper/lezius/diss/, Access Online August 2004.

(Lezius 02b) Wolfgang. Lezius. TIGERSearch – ein Suchwerkzeug für Baumbanken. In Stephan Busemann, editor, *Proceedings der 6. Konferenz zur Verarbeitung natürlicher Sprache (KONVENS 2002), Saarbrücken*, pages 107–114, 2002.

(McCawley 82) James D. McCawley. Parentheticals and discontinuous constituent structure. *Linguistic Inquiry*, 13(1):91–106, 1982.

(Mengel & Lezius 00) Andreas Mengel and Wolfgang Lezius. An XML-based encoding format for syntactically analyzed corpora. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC 2000), Athens, Greece, 31 May – 2 June 2000*, pages 121–126, 2000.

(Mengel 99) Andreas Mengel. MATE deliverable D3.1 – specification of coding workbench: 3.8 improved query language (Q4M). Technical report, Institut für Maschinelle Sprachverarbeitung, Stuttgart, 18. November, 1999. http://www.ims.uni-stuttgart.de/projekte/mate/q4m/.

(Petersen 04) Ulrik Petersen. Emdros — a text database engine for analyzed or annotated text. In *Proceedings of COLING 2004, held August 23-27 in Geneva*. International Commitee on Computational Linguistics, 2004. http://www.hum.aau.dk/~ulrikp/pdf/petersen-emdros-COLING-2004.pdf, Access online August 2004.

(Rohde 04) Douglas L. T. Rohde. Tgrep2 user manual, version 1.12. Available online http://tedlab.mit.edu/~dr/Tgrep2/tgrep2.pdf. Access Online April 2005, 2004.

(Voormann & Lezius 02) Holger Voormann and Wolfgang Lezius. TIGERin - Grafische Eingabe von Benutzeranfragen für ein Baumbank-Anfragewerkzeug. In Stephan Busemann, editor, *Proceedings der 6. Konferenz zur Verarbeitung natürlicher Sprache (KONVENS 2002)*, pages 231–234, Saarbrücken, 2002.

Spelling Correction in Context

Guillaume Pinot and **Chantal Enguehard**
Laboratoire d'Informatique de Nantes Atlantique (LINA)
Université de Nantes — 2, rue de la Houssinière — BP 92208
44322 Nantes Cedex 03 — FRANCE
http://www.sciences.univ-nantes.fr/lina/
guillaume.pinot@lina.univ-nantes.fr chantal.enguehard@univ-nantes.fr

Abstract

Spelling checkers, frequently used nowadays, do not allow to correct real-word errors. Thus, the erroneous replacement of *dessert* by *desert* is not detected. We propose in this article an algorithm based on the examination of the context of words to correct this kind of spelling errors. This algorithm uses a training on a raw corpus.

1 Introduction

Spell checkers distributed with text processing such as MS Word or OpenOffice are based on the use of a dictionary. The text is analyzed word by word: each word which does not appear in the dictionary is supposed to be erroneous so corrections are proposed to the user. Paradoxically, the performances of these checkers in error detection are degraded with the increase in the size of the dictionary because they are unable to detect real-word errors.

These real-word errors occur when one or more modifications of a word transform it into another word which is present in the dictionary.

> **example :** *This chocolate cake is a famous desert.*

The omission of an *s* in *dessert* reveals the word *desert*. This error is not detected because *desert* is present in the dictionary.

This problem was tackled during the second half of the 90's, in particular by Andrew R. Golding in (Golding 95) and (Golding & Schabes 96). He defines confusing sets (like {desert , dessert} for example) and then determines by examining the text which of these words is the best candidate. This method was used in other papers like (Jones & Martin 97) and (Mangu & Brill 97).

First, we will explain our algorithm and then, we will compare it with the method named *context word* by Andrew R. Golding (Golding 95).

2 Simultaneous Detection and Correction

Our algorithm detects and corrects the errors simultaneously.

During the examination of a word m, the algorithm compares its probability of appearing in its context with the probability that another word m' appears in the same context, m' being close to m in the sense of an arbitrary distance.

The context of a word is defined by the set of the words present in a vicinity of fixed size in number of words. Considering that it is the semantic aspect of a word which will guide the correction, we make the assumption that the order of these words is not important.

The probabilities are collected during the training part.

We now present the two distinct parts of our algorithm: the computation of the contextual probabilities and the error detection/correction process.

3 Training

The training is made on a raw corpus. This algorithm is parameterized by k: the number of words around a word that constitute its context.

3.1 Reading the Corpus

The corpus is parsed word by word. Let w_c be the current word.

3.1.1 Constitution of the Dictionary

The goal is to index all the words appearing in the corpus with their frequency.

Let D be the dictionary, composed of a set of pairs $D_i = (w_i, c_i)$, w_i being a word. Each w_i is unique. c_i is the number of occurrences of the word w_i.

The constitution of the dictionary is processed as follows:

- if D_c exists, c_c is incremented.

- else, $D_c = (w_c, 1)$ is added to D.

Thus, we obtain the number of appearances of each word in the corpus. This information will allow to calculate various probabilities thereafter.

3.1.2 Context Dictionary C

Definition The context dictionary named C gathers the co-occurrences of the words w_i and w_j, the distance between these words being lower or equal to k words. The word order is not taken into account. Each co-occurrence is supplied with its frequency $f_{i,j}$:

$$C = \{C_{i,j} \quad / \quad C_{i,j} = (\{w_i, w_j\}, f_{i,j})\}$$

Algorithm The corpus is parsed word by word. During the treatment of a word w_c, the $C_{c,j}$ with $j \in [c-k, c-1]$ are calculated. They are the k co-occurrences generated while combining w_c with the words appearing in a window of width k preceding w_c (see figure 1):

- if $C_{c,j}$ exists, $c_{c,j}$ is incremented.
- else, $C_{c,j} = (\{w_c, w_j\}, 1)$ is added to C.

We thus obtain all the 2-word sets located at a distance lower or equal to k, and their frequencies.

Complexity Complexity in space is $O(nk)$ with n the size of the corpus in number of words. In practice, it should be lower because of the redundancy of words and co-occurrences.

Let $O(f(x))$ be the complexity of the search, with x the size of the database in which the search is processed. As the size of this base can be raised by nk, complexity in time is $O(nkf(n))$.

3.2 Calculating the Probabilities

We use the data gathered during the parsing of the corpus to calculate the probabilities of the contexts kept in C.

Let B be the dictionary of the pairs of words associated with their probability. We thus have:

$$B_{i,j} = ((w_i, w_j), P(w_i|w_j))$$

$B_{i,j}$ and $B_{j,i}$ are defined for each $C_{i,j}$. The probability is calculated as follows:

$$P(w_i|w_j) = \frac{c_{i,j}}{c_j}$$

4 Detection and Correction

4.1 Similarity Between Two Words

Let $\text{edist}(w_i, w_j)$ be a function comparing two strings and returning a positive number with the condition

$$\text{edist}(w_i, w_j) = 0 \Leftrightarrow w_i = w_j$$

The largest $\text{edist}(w_i, w_j)$ is, the most distant w_i is from w_j.

The Aspell (Atkinson 05) distance function takes in account the phonetic of words so it needs a linguistic knowledge and depends on the target language. In this first version, we choose to use the minimal edit distance (Wagner & Fischer 74) which is totally independent of the target language. However, we slightly modified this function to reduce the cost of the inversion of letters.

To determine if a word is a plausible correction of the word to be corrected, we use a $\text{sim}(w_i, w_j)$ function, which takes in arguments 2 words and returns true if the 2 words are similar and false if not.

Let ϵ be the empty string, we can define sim as in figure 2.

In practice, we will take $\gamma = 8$ and $c = \text{edist}(\texttt{"a"}, \epsilon)$. These values have been determined after several experimentations.

4.2 Detection and Correction Algorithm

Let K_c be the context of a word w_c. K_c is the set of the $2k$ words located around w_c, that is to say the k words located before w_c and the k words located after w_c (see figure 3).

Let w_c be the word to correct. For each $w_j \in K_c$, we constitute the set F_j such that

$$\begin{aligned} F_j = \{B_{i,j} \in B| \\ \text{sim}(w_c, w_i) = \text{true}, \\ P(w_i|w_j) > P(w_c|w_j)\} \end{aligned}$$

We thus obtain $2k$ sets of propositions. The set of the possible propositions, F, is the union of the sets F_j (see example figure 4).

We now need a heuristic to give a score to each proposition in order to have them in a pertinent order. We here propose a first heuristic, but we are also elaborating tests to refine it.

Let G_j be the subset of F such that

$$G_j = \{B_{i,j}, B_{i,j} \in F\}$$

Let $k = 3$ and $c = 6$

We	can	have	a	lot	of	money.
w_1	w_2	w_3	w_4	w_5	w_6	w_7

w_3, w_4 and w_5 are in the context of w_6, that implies the sets $\{w_3, w_6\}$, $\{w_4, w_6\}$ and $\{w_5, w_6\}$.

Figure 1: Example of a Context During a Corpus Parsing

$$\text{sim}(w_i, w_j) = \begin{cases} \text{true if } \text{edist}(w_i, w_j) \leq \frac{\text{edist}(w_i,\epsilon)+\text{edist}(w_j,\epsilon)}{k} + c \\ \text{false else} \end{cases}$$

Figure 2: the $\text{sim}(w_i, w_j)$ definition

Let $k = 3$ and $c = 6$

We	can	see	that	this	is	a	very	good	dog.
w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}

We here have w_3, w_4, w_5, w_7, w_8 and w_9 in the context of w_6, that gives

$$K_6 = \{w_3, w_4, w_5, w_7, w_8, w_9\}$$

Figure 3: Example of a Context During the Correction

Let $k = 3$ and $w_c =$ game

And	so	my	mind	game	round	to	the	business.
	0.03086	0.01207		came	0.07317	0.01620	0.01571	
				same			0.00523	
				gate		0.00324	0.00305	
		0.00966		gave		0.00324	0.00174	
	0.00617			name			0.00087	
				game			0.00043	

The numbers are $P(w_i|w_j)$. For example $P(\text{round}|\text{came}) = 0.07317$

$$G_1 = \{((\text{came}, \text{so}), 0.03086), ((\text{came}, \text{my}), 0.01207), ((\text{came}, \text{round}), 0.07317), ((\text{came}, \text{to}), 0.01620), ((\text{came}, \text{the}), 0.01571)\}$$

We do not detail $G_2, \ldots, G_6$ which are built in the same way.
The heuristic gives the highest score to the first proposition *came* because it appears 5 times.

For this example, we trained our algorithm on the novel *The War of the Worlds* by H. G. Wells. The sentence came from *The Time Machine* by H. G. Wells (Chapter 3) with the error added.

Figure 4: Propositions

One can then define the following H_j heuristic:

$$H_j = |G_j| + \prod_{B_{i,j} \in G_j} P(w_i|w_j)$$

This heuristic favors the propositions appearing in several sets of proposals F_j, then the strongest probabilities (see example in figure 4).

5 Comparison with Other Works

This algorithm is close to the *Context Words Method* by Andrew R. Golding (Golding 95).

5.1 The Context Words Method

In (Golding 95) like in other articles based on it ((Jones & Martin 97), (Golding & Schabes 96) and (Mangu & Brill 97)), confusion sets are used to correct real-word errors. A confusion set is a set of words which can be confused among each other, because of their close spellings ($\{\text{dessert}, \text{desert}\}$) or because they are often confused ($\{\text{between}, \text{among}\}$).

During the training, a set of $(w_c, w_i, P(w_c|w_i))$ is created (with w_c a word belonging to at least one confusion set, w_i any word). To correct a word, the probabilities of all the words of the corresponding confusion sets are computed, the highest probability being proposed in correction.

5.2 Comparison of the Two Methods

These two methods are based on the same idea: using the words present around the word to correct. They differ especially in the way of establishing sets and also in the nature of them:

- Golding supposes that its confusion sets are preestablished.

  We automatically determine words which can be confused during the correction process. This selection is based on the corpus itself and on our similarity function.

- The Golding's confusion sets are disjoint (their intersections are empty).

  This is not the case in our method: for each word w_i, we determine automatically a list of words that are similar to w_i and which occur in the same context of the words cooccurring with w_i. The list established for w_i and the list established for w_j ($w_i \neq w_j$) can encounter some words in commom.

6 Experimentations

6.1 Corpus

We would have liked to experiment our method on the same corpus as Golding in order to compare fruitfully our results with those he has obtained. Unfortunately, the Brown corpus used by Golding is not free, so we could not perform our algorithm on it.

So, our experiments on this spelling checker use the novel *les Misérables* by Victor Hugo. This corpus is divided into two parts: the training part (480588 words) and the part to be corrected (53405 words) in which errors have been added.

6.2 Experimental Method

We performe the correction and then generate the precision and the recall on the detection of error as well as the precision on correction (see figure 5 for the formulas).

6.3 Adding Errors

Real-word errors are previously added automatically without using any external resource.

This introduction is done in two steps: first the generation of the possible errors, then the introduction of these errors in the text.

6.3.1 Generation of the Possible Errors

Let D be a simple dictionary (a set of word). For each word $w_i \in D$, we associate a set of words included in D and close to w_i (in the sense of our similarity function). This method generates a base which can be used to generate real-word errors.

In practice, we use as dictionary the words appearing more than ten times in *les Misérables* to select errors using words whose context is known by our corrector. Errors on low frequency words (like apax) could not be detected in such experimentations because their context is completely unknown.

6.3.2 Adding Real-word Errors

Two parameters control the introduction of real-word errors: the density of inserted errors and the previously determined possible errors.

Errors are located using a XML tag which keeps the original word (this is the correction we wish to find).

Example: we introduce the word "game" instead of "came" in the text "And so my mind came round to the business.":

$$\text{Precision on detection} = \frac{\text{Number of words rightly detected as being erroneous}}{\text{Number of words detected as being erroneous}}$$

$$\text{Recall on detection} = \frac{\text{Number of words rightly detected as being erroneous}}{\text{Number of words rightly detected}}$$

$$\text{Precision on correction} = \frac{\text{Number of correctly corrected words}}{\text{Number of words rightly detected as being erroneous}}$$

Figure 5: Formulas of the precision and the recall

```
And so my mind
<error correction="came">game</error>
round to the business.
```

Each word of the corpus is affected or not by an error according to the probability of error fixed by the wished density.

6.4 Results

A summary of the results is given in the table 1.

Density	Precision	Recall	P. on correction
10%	0.1081926	0.9363030	0.9622054
1%	0.0206164	0.9615384	0.9500000

Table 1: Results

We note that the precision on detection is very bad. This overdetection of our algorithm is problematic.

On the other hand, we obtain a very good recall on detection and the precision of the correction is more than 95%. The correction in itself is thus very efficient.

7 Conclusion

We have presented here an algorithm that uses non-ordered contexts to detect and correct real-word errors.

The advantages of this algorithm are:

- simplicity;
- independence from any linguistic information;
- use of a raw corpus for the training;
- few parameters have to be regulated.

The disadvantages are:

- the significant size of the data generated by the training.
- the low precision on detection: the algorithm proposes corrections for a lot of correct words.

Our algorithm is intended to be used during the interactive correction of a text so its speed should be sufficient. On the other hand, the overdetection of errors constitutes a real problem.

We thus direct our research towards the definition of better heuristics of scheduling of the propositions. The size of the training corpus may also influence the quality of the results, its influence should be observed. We also plan to define ordered contexts to use the syntax in addition to semantics.

These various methods will be precisely evaluated on the same corpus to analyze their relevance.

References

(Atkinson 05) Kevin Atkinson. GNU Aspell. `http://aspell.net/`, 2005.

(Golding & Schabes 96) Andrew R. Golding and Yves Schabes. Combining trigram-based and feature-based methods for context-sensitive spelling correction. In *Proceedings of the 34th conference on Association for Computational Linguistics*, pages 71–78. Association for Computational Linguistics, 1996.

(Golding 95) Andrew R. Golding. A bayesian hybrid method for context-sensitive spelling correction. *CoRR*, cmp-lg/9606001, 1995.

(Jones & Martin 97) Michael P. Jones and James H. Martin. Contextual spelling correction using latent semantic analysis. In *Proceedings of the fifth conference on Applied natural language processing*, pages 166–173. Morgan Kaufmann Publishers Inc., 1997.

(Mangu & Brill 97) Lidia Mangu and Eric Brill. Automatic rule acquisition for spelling correction. In *ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning*, pages 187–194. Morgan Kaufmann Publishers Inc., 1997.

(Wagner & Fischer 74) Robert A. Wagner and Michael J. Fischer. The string-to-string correction problem. *J. ACM*, 21(1):168–173, 1974.

Lexical Transfer Selection Using Annotated Parallel Corpora

Stelios Piperidis, Panagiotis Dimitrakis and **Irene Balta**
Institute for Language and Speech Processing
6 Artemidos & Epidavrou, 151 25 Marousi, Athens, Greece
spip@ilsp.gr

Abstract

This paper addresses the problem of bilingual lexicon extraction and lexical transfer selection, in the framework of computer-aided and machine translation. The method relies on parallel corpora, annotated at part of speech and lemma level. We first extract a bilingual lexicon using unsupervised statistical techniques. For each word with more than one translation candidates we build context vectors, based on the annotated parallel corpus information, in order to aid the selection of the contextually correct translation equivalent. The method achieves an overall precision of ca. 85% while the maximum recall reaches 75%.

1 Introduction

The emergence of parallel corpora has evoked the appearance of many methods that attempt to deal with different aspects of computational linguistics (Véronis 00). Of special significance in the field of lexicography, terminology and machine translation is the impact of "bitexts"; a pair of texts in two languages, where each text is a translation of the other (Melamed 97). Such texts are necessary for providing evidences of use, directly deployable in statistical-based methodologies and enhance the automatic elicitation of the otherwise sparse linguistic resources.

This paper describes the design and development of a method for automatic bilingual lexicon extraction from a parallel bilingual corpus. Of particular importance is the integration of a lexical transfer selection strategy, which enables the rendering of the contextually correct translation for a given word. In this framework, we explore the relationship between word-senses and word-uses in a bilingual environment. We also analyse the way they can be represented in a context vector model for word translation prediction.

In the next section we give a brief overview of previous work in the field of automatic lexicon extraction from parallel corpora. In section 3 we present the proposed method, which aims at formulating and applying a context vector representation towards a context-based solution of translational ambiguities. In section 4 we analyse the evaluation process and the current results, while in sections 5 and 6 we discuss the conclusions drawn as well as possible applications and future enhancements.

2 Background

Recent developments in computer-aided and machine translation have moved towards the use of parallel corpora, aiming at two primary objectives: (i) to overcome the sparseness of the necessary resources and (ii) to avoid the burden of producing them manually. Furthermore, parallel corpora have proven rather useful for automatic dictionary extraction, which offers the advantages of a lexicon capturing the corpus specific translational equivalences, as Brown has pointed in (Brown 97; Piperidis *et al.* 00). Extending this approach, we investigate the possibility to use bilingual corpora in order to extract translational correspondences coupled with information about the word-senses and contextual use. In particular, we focus on polysemous words with multiple translational equivalences.

The relation between word and word-usage, as compared to the relation of word and word-sense has been thoroughly addressed (Gale *et al.* 92a; Yarowsky 93; Kilgarriff 97). In this scope, we argue that through the exploitation of parallel corpora and without other external linguistic resources, we can adequately resolve the task of target word selection in computer-aided and machine translation. Along this line, it has been argued that the accumulative information added by a second language could be very important in lexical ambiguity resolution in the first language (Dagan *et al.* 91), while tools have been implemented for translation prediction, by using context information extracted from a parallel corpus (Tiedemann 01).

In addition, research on word sense disambiguation is significant in the design of a methodology for automatic lexical transfer selection. Although monolingual word sense disambiguation and translation are perceived as different problems (Gale *et al.* 93), we examine whether certain conclusions, which are extracted during the process of word sense disambigua-

tion, could be useful for translation prediction.

The role of context, as the only means to identify the meaning of a polysemous word (Ide & Véronis 98), is of primary importance in various statistical approaches. Brown in (Brown *et al.* 91) and Gale in (Gale *et al.* 92b; Gale *et al.* 93), use both the context of a polysemous word and the information extracted from bilingual aligned texts, in order to assign the correct sense to the word. Yarowsky explores the significance of context in creating clusters of senses (Yarowsky 95), while Schütze in (Shütze 98), addresses the sub-problem of word sense discrimination through the context-based creation of three cascading types of vectors.

We examine the impact of context upon translation equivalent selection, through an "inverted" word sense discrimination experiment. Given the possible translational candidates, which are extracted from the statistical lexicon, we investigate the discriminant capacity of the context vectors, which we build separately for each of the senses of the polysemous word.

3 Proposed method

The basic idea underlying the proposed method is the use of context vectors, for each of the word usages of a polysemous word, in order to resolve the problem of lexical transfer selection. The method consists of three stages:

- Bilingual Lexicon Extraction
- Context Vectors Creation
- Lexical Transfer Selection

The first stage could be omitted, if a bilingual lexicon is already available. Figure 1 depicts an overall view of the system's architecture.

3.1 Lexicon Building

The first goal is to build a bilingual lexicon. Parallel corpora are sentence-aligned using a Gale & Church-like algorithm (Gale *et al.* 91) and annotated on both language sides for part-of-speech and lemma. Focusing on the semantic load bearing words, we filter the pos tagged corpus and retain only nouns, adjectives and verbs. The corpus-specific lexicon is extracted using unsupervised statistical methods, based on two basic principles:

- No language-pair specific assumptions are made about the correspondences between grammatical categories. In this way, all possible correspondence combinations are produced, as this is possible during the translation process from a source language to the target language.
- For each aligned sentence-pair, each word of the target sentence is a candidate translation for each word of the aligned source sentence.

Following the above principles we compute: the absolute frequency (the number of occurrences) of each word and the frequency of each word pair, in a sentence pair. Using these frequencies, we extract lexical equivalences, based on the following criteria:

1. The frequency of the word pair must be greater than threshold Thr_1.
2. Each of the conditional probabilities $P(W_t|W_s)$ and $P(W_s|W_t)$ has to be greater than threshold Thr_2.
3. The product $P(W_t|W_s) \cdot P(W_s|W_t)$ must be greater than threshold Thr_3. This product is indeed the score of the translation.

After experimentation and examination of the results, $\{Thr_1, Thr_2, Thr_3\}$ were set to $\{5, 0.25, 0.15\}$. More specifically, experiments with Thr_1=1

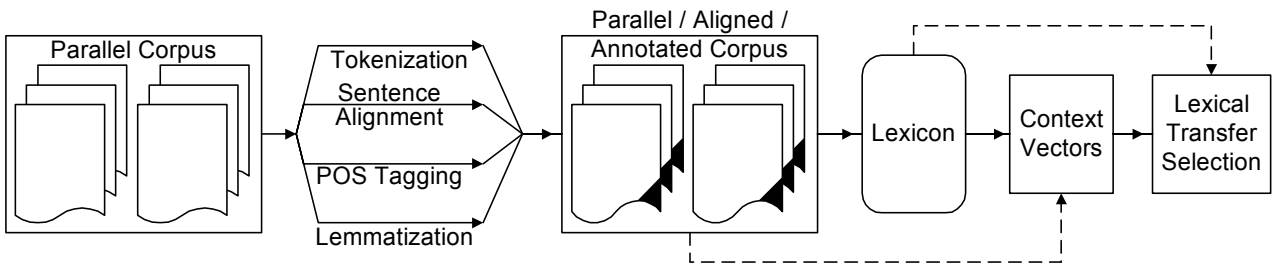


Figure 1: Lexicon building and transfer selection architecture

and Thr_1=10 revealed that the former resulted in high recall with significantly low precision, i.e. a fairly large but low quality lexicon, while the latter resulted in relatively high precision, whilst recall was radically reduced, i.e. a fairly small high-quality lexicon. We empirically decided to fix Thr_1 in the middle of the experimentation range. Thr_2 was set to 0.25 to account for a maximum number of 4 possible translations, which a presumable polysemous word could have, according to the corpus data. Thr_3 was set to 0.15 to account for the lower bound of the product of conditional probabilities $P(W_t|W_s)$ and $P(W_s|W_t)$, taking into consideration the threshold of the individual conditional probabilities, i.e. we empirically set the lower bounds to the conditional probabilities as $\{0.25, 0.6\}$.

The extracted bilingual lexical equivalences account for : words with one translation (90% of total) and words with multiple translations (10%). Words with multiple extracted equivalents are further distinguished in: (i) words that have multiple, different in sense, translational equivalents, (ii) words that have multiple, synonymous translations and (iii) words that have multiple translations, which are in fact wrong due to statistical errors of the method. In the following, we focus on (i), that is polysemous words, with multiple translational equivalents, which cannot mutually replace each other in the same context.

3.2 Context Vectors Creation

For each word, in the set of the treated grammatical categories, in the corpus, a context vector is created based on: (i) the extracted lexicon, in order to retrieve the possible translations of a word and (ii) the parallel aligned sentences to retrieve those words that systematically co-occur with that word (thus contributing to the definition of its meaning). The process is described below:

Step 1.1: For "univocal" words, words with only one translation, we assume that the translation is also its "sense". For the untranslated words we make no assumption, though they also participate in the created vectors. Both these categories of words are denoted by W_u.

Step 1.2: For the words with multiple translations in the extracted lexicon, we cannot automatically pick out the polysemous words. Therefore for each of these W_p, we make the following assumption: *Each word, which has more than one translation in the lexicon, could potentially be a polysemous one.* We suppose W_p is one of those words and let T_1 and T_2 be two possible translations. When W_p is found in a source sentence, we search for the words T_1 and T_2 in the target sentence. If one and only one is matched, e.g. T_1, we conclude that this is the correct translation and W_p is replaced in the source sentence by $W_p\_T_1$. This is repeated for each word with multiple translations. In the case of erroneous multiple translations (caused by statistical errors), none of T_1 or T_2 are assigned as a sense, due to the simultaneous appearance of more than one in the target sentence. In the end, we have a new corpus in which some words appear as before and some have been labeled by their "local senses".

Step 2: In order to build the context vectors, we address the words $W_p\_T_1$ and $W_p\_T_2$ as being different. Then we isolate those source sentences where either "word-sense" $W_p\_T_1$ or $W_p\_T_2$ appear exclusively. In each different set of sentences we examine the context of $W_p\_T_i$ in a window of certain length centered to the word of interest $W_p\_T_i$. The size of the window is defined as:

$$window\_size = 2n, \tag{1}$$

where n denotes the number of word tokens on either side of the word of interest.

Inside this window of token-words, we look for words, W_x, which belong to the selected grammatical categories. Each of W_x is added to the vector, along with the number of times this word has appeared in the context of the word $W_p\_T_i$. We follow a similar procedure for each word W_u.

Step 3: The final formation of the context vectors is based on the following equations:

$$N_{W_x W_p\_T_i} \geq k \tag{2}$$

$$P(W_x|W_p\_T_i) \geq a_1, \tag{3}$$

where $N_{W_x W_p\_T_i}$ is the number of the total co-occurrences of words W_x in the window of $W_p\_T_i$, k is the minimum co-occurrences that a word W_x must have in order to participate in the context vector which describes the word $W_p\_T_i$, $P(W_x|W_p\_T_i)$ is the conditional probability of the word W_x given the appearance of $W_p\_T_i$ and a_1 is a threshold, which the probability $P(W_x|W_p\_T_i)$ must exceed. $P(W_x|W_p\_T_i)$ is also the score of W_x in the context vector of $W_p\_T_i$.

In the case of a word W_u, with only one translation or no translation in the lexicon, similar equations are used:

$$N_{W_x W_u} \geq k \tag{4}$$

$$P(W_x|W_u) \geq a_2, \tag{5}$$

where $N_{W_xW_u}$ is the number of the total co-occurrences of words W_x in the window of W_u , k is defined as the minimum co-occurrences of W_x and W_u, $P(W_x|W_u)$ is the conditional probability of the word W_x given the appearance of W_u and a_2 is a threshold, which the probability $P(W_x|W_u)$ must exceed. $P(W_x|W_u)$ is the score of W_x in the context vector of W_u.

Whenever at least one of these criteria in each set of equations is not met, the word W_x is deleted from the vector. In (3) and (5) we use two distinct thresholds a_1 and a_2 with

$$a_1 < a_2, \tag{6}$$

as we would like polysemous words to have greater vectors. The parameters a_1 and a_2 are set, after experimentation, to 0.05 and 0.1 respectively.

Thus, we have constructed, a context vector for each word in the set of the specified grammatical categories. The vector consists of words that systematically co-occur with the word of interest and their context-vector scores. In Figure 2 we present the flow diagram for the creation of context vectors. For each new word the process is iterated over the words of the corpus.

3.3 Lexical Transfer Selection

Based on the lexicon and the context vectors, the algorithm can disambiguate an ambiguous word, when appearing in a certain context, by comparing this context, with the previously created vectors of its "senses". The process includes the following steps:

Step I: Let W_p be an ambiguous word, with T_i translational equivalents extracted from the lexicon. A sentence is fed to the system and W_p is one of the words. Each T_i of the "senses" $W_p\_T_i$ are considered to be translation candidates for the sentence at hand.

Step II: For each of $W_p\_T_i$ an extended vector V_{xyzw} is produced. The main characteristic of the expanded vector is its depth d_i. The depth d_i denotes the number of "co-occurrence connections" between words, which we use in order to "meaningfully connect" the word $W_p\_T_i$ with any word W_x. In our methodology:

$$d_i = 4 \tag{7}$$

We believe that a greater value for d_i would capture the spurious co-occurrences of words, thus it would not represent a logical and linguistically expected "sense-connectivity". The vector of $W_p\_T_1$ consists of the words W_x that appear in the context vector V_1

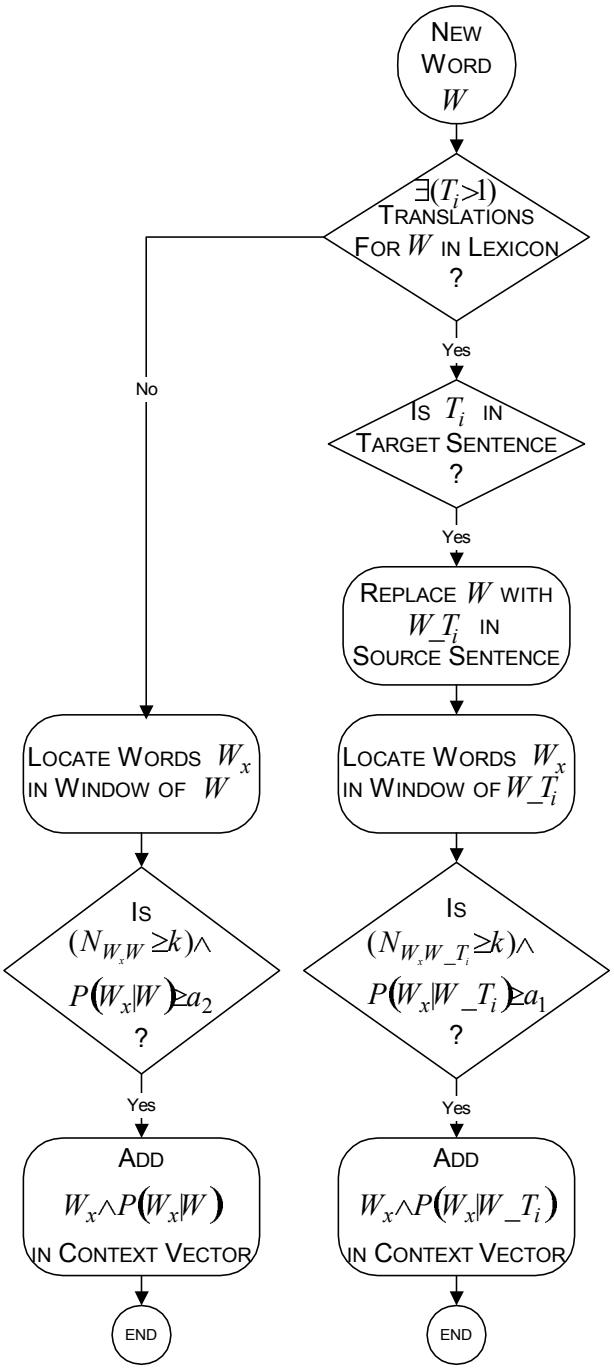


Figure 2: Context-Vector Creation Procedure

of $W_p\_T_1$, the words that appear in the context vectors V_{1i} of each word in V_1, and so on until depth = 4 (V_1 words are in depth 1, V_{1i} words are in depth 2 etc). Figure 3 in the next page shows a diagram for the created vector.

Step III: Each of the word W_x that participates in the enlarged vector is assigned an extended-vector score $EVScore_{x_{W_p}}$ or $EVScore_{x_{W_u}}$, depending on

the type of the word with which it co-occurs:

$$EVScore_{x_{W_p\_T_i}} = \frac{P(W_x|W_p\_T_i)}{2^{1-d_x}} \quad (8)$$

$$EVScore_{x_{W_u}} = \frac{P(W_x|W_u)}{2^{1-d_x}}, \quad (9)$$

where $P(W_x|W_p\_T_i)$ and $P(W_x|W_u)$ are defined in (3) and (5) and d_x is the depth in which W_x was found. In case of multiple appearances of W_x in the extended vector, we choose the one in the lowest depth, as it is the most significant in the process of defining the sense of W_p.

Step IV: The final lexical transfer selection procedure examines each extended vector of $W_p\_T_i$ separately. We compare the words inside the $\pm n$ window of word W_p of the sentence under examination with those included in the vector. For each matched word we compute the appropriate score, using (8) and (9). By adding the scores of the matched words, we assign to each possible translational equivalent T_i a total score, depending on the associated extended vector. Finally, for the lexical transfer selection, we choose the word-sense $W_p\_T_i$ with the highest score. If both scores are equal, the algorithm does not choose randomly and can output both as candidate translations. A feedback mechanism could be foreseen to minimize these cases, if appropriate, in a subsequent transfer selection round.

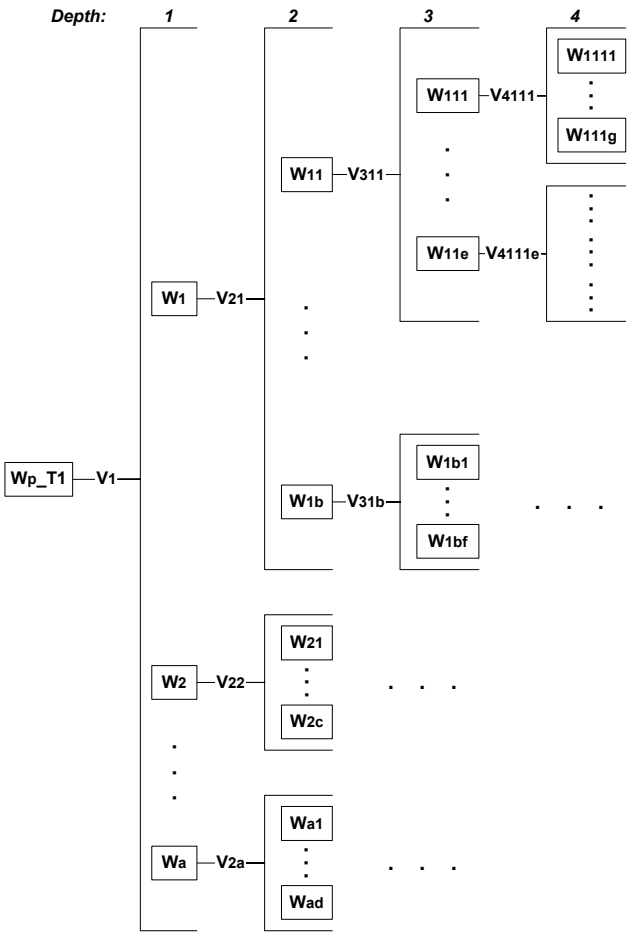


Figure 3: Context Vector Layout

4 Results - Evaluation

The corpus used was the INTERA parallel corpus (Gavrilidou *et al.* 04) consisting of official EU documents in English and Greek from five different domains; education, environment, health, law and tourism. The corpus comprises $100,000$ aligned sentences, containing on average $830,000$ tokens of the selected grammatical categories (nouns, verbs and adjectives) in either language. The corresponding lemmas are $20,000$. The complete bilingual lexicon comprises 5280 records (where multiple translational equivalences of a word are counted as one record).

Evaluation was focused on the ability to resolve truly ambiguous words, leaving aside words with synonymous translations, or erroneous translational candidates. For this purpose, a set of ambiguous words in English, the contextually correct translation equivalent of which is "univocal" in Greek, were manually selected. The selected set was {*active, floor, seal, settlement, solution, square, vision*}.

For the above words, we extracted the sentences, in the parallel corpus, that contain them. We adopted the 10-fold cross validation technique for evaluation, computing the average results over the 10 iterations of the algorithm. The possible answers, given by the algorithm were:

- *Correct*, when only the selected translational equivalent was present in the target sentence.
- *Wrong*, when the selected translational equivalent was different from the one appearing in the target sentence.
- *No answer*, when the translational equivalents were assigned the same score.

Precision was calculated as the ratio of the correct answers to the sum of correct and wrong answers. Recall was calculated as the ratio of the correct answers to the possible correct answers. The experiment was first performed with three sets of k, n: (i) k=3 and a window of n=5, (ii) k=3 and a window of n=7, and (iii) k=3 and a window of n=15 (k referring to (2) and (4)). The averaged results over the 10 iterations are shown in Table 1.

In order to simulate a larger corpus, we enlarged the produced context vectors. We conducted the experiment again, with k=1 and the three variants for the size of the windows defined as previously. The results are shown in Table 2.

	$k=3$ $n=\pm5$	$k=3$ $n=\pm7$	$k=3$ $n=\pm15$
Correct	85.5	90.8	91.8
Wrong	9.6	14.5	26.3
No answer	25.1	15.2	2.4
Precision	89.9%	86.2%	77.7%
Recall	71.1%	75.3%	76.2%
Answered	79.1%	87.4%	98.0%

Table 1: Results for k=3

	$k=1$ $n=\pm5$	$k=1$ $n=\pm7$	$k=1$ $n=\pm15$
Correct	88.1	92.9	87.4
Wrong	11.7	17.0	31.7
No answer	20.7	10.6	1.4
Precision	88.2%	84.5%	73.3%
Recall	73.1%	77.1%	72.5%
Answered	82.8%	91.2%	98.8%

Table 2: Results for k=1

In Figure 4 we present the created vector V_1 for the polysemous word "solution" and in Figures 5 and 6 we show two examples of our system's behavior.

As expected, the wider the window, the more likely that the system gives an answer, although the precision decreases. Especially for a window size n=15, which in most cases in the given corpus contains all the tokens in a sentence, we notice that the system's performance declines disproportionately. This is due to multiple erroneous statistical co-occurrences that are semantically irrelevant is such wide windows.

In the second experiment, the percentage of the answered cases and recall increased, compared to the first experiment, while precision slightly decreased. The results also indicate that although a smaller window and a higher absolute appearance threshold k would lead to a lower number of answers, the accuracy increases.

To evaluate the performance of the method taking into consideration the special characteristics of our corpus, we computed a "baseline" performance (Gale *et al.* 92c). We assign to each polysemous word, found in the test set, the most frequent of its possible senses. The estimated baseline performance was 55% on average due to the almost equal distribution of the different senses of the words. Thus, the employment of context vectors method lead to an increase in recall of almost 20%.

Polysemous Word Wp	**Words in First Level Vector V1 of Wp**		
Wp_T1 = solution_διάλυμα (solution, as a homogeneous liquid)	**agenerase** **aqueous** **be** **capsule** **child** **clear** **colourless**	**concentrate** **contain** **contraindicated** **fill** **infusion** **injection** **insulin**	**IU** **mg** **ml** **oral** **patients** **pen** **vial**
Wp_T2 = solution_λύση (solution, as answer, decision)	**adopt** **be** **find** **have** **possible** **problem**		

Figure 4: First level vector for "solution"

Test Sentence
(dots indicate tokens not from the selected POS types)
. clear . . colourless . solution . . visible . particle . . be . use

Polysemous word Wp = solution
Wp_T1 = solution_διάλυμα (solution, as a homogeneous liquid)
Wp_T2 = solution_λύση (solution, as answer, decision)

Context Vectors Comparison

Wp_T1 = solution_διάλυμα
Wi found in context vector
be: score=0.3
clear: score=0.11
colourless: score=0.08
visible: score=0.025
use: score=0.0475

Wp_T2 = solution_λύση
Wi found in context vector
be: score=0.44

Lexical Transfer Selection
Score(Wp_T1=solution_διάλυμα)=0.5625
Score(Wp_T2=solution_λύση)=0.44
Correct Selected: Wp_T1 = solution_διάλυμα

Figure 5: First example of Lexical Transfer Selection

Test Sentence
(dots indicate tokens not from the selected POS types)
crossborder . EURES . adviser . help . . find . practical . solution . . . problem . . customise . . service . . . need . . regional . customer

Polysemous word Wp = solution
Wp_T1 = solution_διάλυμα (solution, as a homogeneous liquid)
Wp_T2 = solution_λύση (solution, as answer, decision)

Context Vectors Comparison

Wp_T1 = solution_διάλυμα
Wi found in context vector
service: score=0.0825

Wp_T2 = solution_λύση
Wi found in context vector
find: score=0.09
problem:score=0.07

Lexical Transfer Selection
Score(Wp_T1=solution_διάλυμα)=0.0825
Score(Wp_T2=solution_λύση)=0.16
Correct Selected: Wp_T1 = solution_λύση

Figure 6: Second example of Lexical Transfer Selection

5 Applications and Future work

The proposed method can be used as a translational tool, for computer-aided and machine translation. especially as it concerns translation customization processes. Furthermore the method can be used as feedback mechanism for a refinement in statistical lexicon extraction. As a validation, we conducted a second experiment, over all the words in the lexicon, which had multiple translations (although not always correct). The results were similar to the ones presented in Table 1 and Table 2. Thus, such methods can be of utmost importance for bootstrapping the development of multilingual lexica with semantic constraints on the potential cross-lingual equivalences. Forthcoming experiments will include tests on larger corpora and use of linguistically principled window selection.

6 Conclusions

We presented a statistical method for lexical transfer selection, with special attention to polysemous words. Our technique relies on a bilingual parallel, aligned and annotated corpus, without resorting to other external linguistic resource. The method is language independent and suitable for translation prediction for any language pair.

Based on the aligned sentences, we extracted a statistical bilingual lexicon, from which we identified words with multiple translational equivalents. We then extracted context vectors, representing the impact of adjacent words to the sense of an ambiguous word. Finally, we merge the information derived from the context and the lexicon to obtain the selection of the contextually correct translational equivalent. Evaluation shows a promising overall performance, as compared to the evaluated baseline performance.

References

[Brown *et al.* 91]: Peter F. Brown, Stephen Della Pietra, Vincent J. Della Pietra, and Robert L Mercer, *Word-Sense Disambiguation Using Statistical Methods*, Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics, Berkeley, California, June 1991, pp. 264-270.

[Brown 97]: Ralf D. Brown, *Automated Dictionary Extraction for "Knowledge-Free" Example-Based Translation*, Proceedings of the 7th International Conference on Theoretical and Methodological Issues in Machine Translation, Santa Fe, New Mexico, July 1997, pp. 111-118.

[Dagan *et al.* 91]: Ido Dagan, Alon Itai, Ulrike Schwall, *Two languages are more informative than one*, Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics, Berkeley, California, June 1991, pp. 130-137.

[Gale *et al.* 91]: William A. Gale, Kenneth W. Church *A program for aligning sentences in parallel corpora*, Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics, Berkeley, California, June 1991, pp. 177-184.

[Gale *et al.* 92a]: William A. Gale, Kenneth W. Church and David Yarowsky *One sense per discourse*, Proceedings of the Speech and Natural Language Workshop, San Francisco, Morgan Kaufmann, pp. 233-237.

[Gale *et al.* 92b]: William A. Gale, Kenneth W. Church and David Yarowsky *Using Bilingual Materials to Develop Word Sense Disambiguation Methods*, Proceedings of the International Conference on Theoretical and Methodological Issues in Machine Translation, pp. 101-112.

[Gale *et al.* 92c]: William A. Gale, Kenneth W. Church and David Yarowsky *Estimating Upper and Lower Bounds on the Performance of Word-Sense Disambiguation Programs*, Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics, University of Delaware, Newark, Delaware, June-July 1992, pp. 249-256.

[Gale *et al.* 93]: William A. Gale, Kenneth W. Church and David Yarowsky *A Method for Disambiguating Word Senses in a Large Corpus*, Computers and the Humanities, Volume 26, pp. 249-256.

[Gavrilidou *et al.* 04]: M. Gavrilidou, P. Labropoulou, E. Desipri, V. Giouli, V. Antonopoulos, S. Piperidis *Building parallel corpora for eContent professionals*, MLR2004: PostCOLING Workshop on Multilingual Linguistic Resources, Geneva, Switzerland, August 2004.

[Ide & Véronis 98]: Nancy Ide, Jean Véronis, *Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art*, Computational Linguistics, Volume 24 (1), pp. 1-40.

[Kilgarriff 97]: Adam Kilgarriff, *"I don't believe in word senses"*, Computers and the Humanities, Volume 31 (2), pp. 91-113.

[Melamed 97]: I. Dan Melamed, *A Word-to-Word Model of Translationan Equivalence*, Proceedings of the 35th Conference of the Association for Computational Linguistics, Madrid, Spain, July 1997, pp. 490-497.

[Piperidis *et al.* 00]: Stelios Piperidis, Harris Papageorgiou and Sotiris Boutsis *From sentences to words and clauses*, Parallel Text Processing, Jean Véronis (editor), Kluwer Academic Publishers, pp. 117-138.

[Shütze 98]: Hinrich Shütze, *Automatic Word Sense Discrimination*, Computational Linguistics, Volume 24 (1), pp. 97-123.

[Tiedemann 01]: Jörg Tiedemann, *Predicting Translations in Context*, Proceedings of the Conference on Recent Advances in Natural Language Processing (RANLP), Tzigov Chark, Bulgaria, September 2001, pp. 240-244.

[Véronis 00]: Jean Véronis, *From the Rosetta stone to the information society: A survey of parallel text processing*, Parallel Text Processing, Jean Véronis (editor), Kluwer Academic Publishers, pp. 1-25.

[Yarowsky 93]: David Yarowsky, *One Sense Per Collocation*, Proceeding of ARPA Human Language Technology Workshop, Princeton, New Jersey, pp. 266-271.

[Yarowsky 95]: David Yarowsky, *Unsupervised Word Sense Disambiguation Rivaling Supervised Methods*, Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics, Cambridge, Massachusetts, June 1995, pp. 189-196.

Enough is Enough! — Estimating Upper Bounds of the Size of Training Corpora for Unsupervised PP Attachment Disambiguation

Michael Poprat [a,b] **Udo Hahn** [a]

[a] Jena University Language and Information Engineering (JULIE) Lab, D-07743 Jena, Germany
http://www.coling.uni-jena.de

[b] Freiburg University Hospital, Medical Informatics Department, D-79104 Freiburg, Germany
http://www.imbi.uni-freiburg.de/medinf

Abstract

The use of ever larger corpora for NLP research seems to reflect the folk theorem that increasing sizes of training data for supervised, and definitively for unsupervised, machine learning approaches will (always) lead to improving the quality of the learning results for various NLP tasks. We challenge this general assumption in the light of empirical counter-evidence. Following up on work in machine translation and word sense disambiguation, we wanted to estimate the necessary and sufficient, and hence fully adequate, size of underlying training corpora. We conducted various experimental studies on the unsupervised disambiguation of ambiguous prepositional phrase attachments for the English and German language. Based on this evidence, we are able to estimate reasonable upper bounds of the sufficient size of a proper training corpus, for this task at least.

1 Introduction

Machine learning approaches for natural language processing can roughly be divided into two classes, *viz.* supervised and unsupervised ones. Supervised learning implies training on (often manually) annotated corpora, whereas unsupervised methods extract training instances from non-annotated texts. In general, supervised approaches outperform unsupervised ones in terms of accuracy, but their superiority largely depends on the amount and the quality of the annotated corpora. As corpus annotation is a very time- and cost-consuming task, the amount of training data can hardly be increased and their size is more or less fixed. Therefore, in domains which lack annotated corpora unsupervised learning methods are not really a matter of choice but rather an inevitable necessity.

One of the areas where unsupervised learning has already been applied is the disambiguation of alternative prepositional phrase (PP) attachments (Ratnaparkhi 98; Pantel & Lin 00; Volk 02; Schwartz *et al.* 03). Such a PP ambiguity arises when there is more than one constituent as a possible attachment site for the prepositional phrase.

We will here investigate the influence of the size of the training corpus on the disambiguation results. By doing so, we will also question the folk theorem that increasing sizes of training corpora for unsupervised learning will (always) lead to improving the quality of the learning results, for this task at least. Using a statistical learning model based on n-grams (bigrams and trigrams) we will collect empirical evidence that, beyond a certain level, increasing the size of corpora will only marginally raise the quantity of resolvable ambiguous PPs though not improve the quality of the disambiguation decisions for PP attachments.

2 Related Work and Purpose

Size of the underlying corpora has always been an issue in many different application fields of NLP. Studies dealing with machine translation, e.g., (Callison-Burch & Osborne 03), report a positive impact of increasing size of parallel corpora on the translation quality. Also for the problem of confusion sets (like *weather* vs. *whether*), there is ample benefit of enlarging corpora as far as disambiguation performance is concerned (cf., e.g., (Banko & Brill 01)). Supervised methods for the disambiguation of PP attachment ambiguities as well report the advantageous influence of increasing corpus size (Brill & Resnik 94).

Although the disambiguation approaches in these studies are related to our PP attachment disambiguation problem, we have reasons to believe that these conclusions do not carry over to *unsupervised* PP disambiguation. Running through various important PP disambiguation studies, we encounter a broad range of different corpus sizes: (Hindle & Rooth 93) use a corpus composed of roughly about 13 million tokens; (Ratnaparkhi 98) deals with 970,000 sentences from that part of the *Wall Street Journal* which has not been annotated so far; (Pantel & Lin 00) employ a 125 million token-sized corpus and exploit contextually similar words for PP attachment disambiguation; while (Volk 02) achieved his disambiguation results on a 5.5 million tokens training corpus. Virtually none of these studies has scrutinized the correlation between the quality of the experimental results and the corresponding corpus size for its task.

There are, however, two exceptions to this rule: (Brill & Resnik 94) observe for their rule-based and

error-driven learning approach that a growing training corpus can lead to improved accuracy values. Nevertheless, their learning approach is supervised and therefore not comparable to our one. Using an unsupervised learning approach, (Volk 01) exploits the world's largest text corpus, the Web. He reports on a reduction of the sparse data problem using trigrams for disambiguation. Morphological inflection and the choice of reasonable query operators, however, are recognized to be problematic. Still, all of these studies fail to give any concrete estimate of the required size of the corpus to solve the problem at hand.

By contrast, our goal is to determine a reasonable order of magnitude for the size of the training data needed for acceptable disambiguation results. Such an estimate is needed for two reasons: First, given a potentially limited amount of training data, does it make sense at all to apply an unsupervised approach? Second, given a very large amount of available training data, what is an optimal upper bound for the size of corpora? For our purpose, 'optimality' will be defined as a reasonable upper bound of the size of the training corpus, which still produces a substantial gain in disambiguation performance.

3 Learning to Disambiguate

3.1 Corpora for Training and Test Set

The reason for applying an unsupervised learning technique to the PP attachment problem is due to the fact that for the languages and the domain we are working in, *viz.* biomedical documents in English and German, no suitable annotated corpora are available: For German in general, there is only the NEGRA treebank (Brants *et al.* 03) based on newspaper articles (about 355,000 tokens in 20,600 sentences), while for the sublanguage used in English biomedical texts, we would be restricted to the beta version of the GENIA treebank,[1] with only 200 documents (about 42,000 tokens). The size of both corpora is probably too small-scaled for a reasonable supervised training of statistical models for this task (see (Brill & Resnik 94) on this issue). However, both annotated corpora are valuable resources for the creation of a test set with which we can measure the performance of the unsupervised statistical method (used in this study) and also examine the effect of the size of the training corpora.

Our training corpus for the biomedical domain consists of approximately 500,000 MEDLINE[2] abstracts, which were extracted using the following query terms: *transcription factors*, *blood cells* and *human*. Thus, we ensured a thematical overlap without having an intersection of documents in the training and test corpus. All in all, our training corpus consists of roughly 104 million text tokens. We then annotated it with the GENIA part-of-speech tagger[3] and identified text chunks with the YAMCHA chunker (Kudo & Matsumoto 01).

For the German newspaper domain, we obtained the online version of a German newspaper. This resulted in a training corpus which also amounted to roughly 114 million text tokens. We tagged the corpus with the TNT tagger (Brants 00) and chunked it with a home-grown phrase chunker.

3.2 Extracting Possible PP Attachments as Training Data

To obtain training instances for PP attachments, we extracted head-word tuples of the form $[\{v, n, a\}, p, n_2]$ from our part-of-speech tagged and chunked training corpora, where v is a verb, n is the head noun of a noun phrase, a an adjective, p a preposition, and n_2 the head noun of the noun phrase directly following the preposition (i.e., the PP). This resulted in 9,700,000 unique (out of 23,000,000) $[\{v, n, a\}, p, n_2]$ tuples from the English MEDLINE corpus and 10,900,000 (out of 14,700,000) tuples from the German newspaper corpus, together with their frequencies.

The challenge with this extraction heuristic is to identify *all* possible nouns, verbs and adjectives to which the prepositional phrase can be attached. Although we cannot predict the correct attachment point, we rely for extraction on the information provided by the tagger and the chunker solely: For the English language, within a sentence a PP can potentially be related to each preceding NP from which we extracted its nominal head. Furthermore, a PP can also be related to a preceding verb[4] or a preceding predicative adjective not embedded in an NP. Within a sentence, verbs and adjectives constitute boundaries such that possible attachment points preceding the verb or adjective are precluded. For German, we additionally have to consider two particularities: First, PPs in subordinate clauses can be related to main verbs occur-

[1] http://www-tsujii.is.s.u-tokyo.ac.jp/~genia/topics/Corpus/GTB.html

[2] MEDLINE is the largest bibliographic database for biomedicine (http://www.ncbi.nlm.nih.gov/entrez/query.fcgi)

[3] http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/postagger/

[4] We excluded auxiliary and modal verbs, since they hardly function as PP attachment sites.

ring at the very end of the sentence. Second, some German inflected verbs separate their prefix that must then be re-attached to the verb in order not to change the semantics and the valency of the verb.

It is obvious that many relations extracted this way can be false and might add a lot of noise to the training data. Still, the underlying assumption is that its sheer size (cf. the numbers above) provides plenty of useful evidence for an unsupervised statistical classifier.

3.3 Creating the Test Set

For our test set, we used the beta version of the syntactically annotated GENIA treebank for English and the NEGRA treebank for German, in both of which an attachment decision is made for every occurring PP. From such a gold standard, a test set for ambiguous cases can easily be created.

Many studies (e.g., (Hindle & Rooth 93), (Ratnaparkhi 98) or (Volk 02)) only consider PP attachment ambiguities of the form $[v, n, p, n_2]$ and thus limit the decision to relate the PP either to the verb or to the noun. Such a format, however, only covers a subset of potential ambiguities when we determine the relations between entities in a sentence. To create test cases for all possibly arising ambiguities, we assumed that the GENIA and the NEGRA treebank were analyzed by a shallow parser which does not return any attachments. This is a pretty realistic scenario given the complexity of the language patterns encountered and the size of the document collections to be dealt with.

When a PP ambiguity occurs, the correct anchor point for attachment can only be determined on the basis of the information returned by the shallow parser. The only possible restrictions are again linguistically motivated and similar to the language specific heuristics used for constructing the training set.

It is obvious that when we create a test set under such conditions, the resulting test cases can be more complex (but also more realistic) than those typically examined in the NLP literature. Given these considerations, we extracted 2,411 test cases from the GENIA treebank and 5,360 from the NEGRA treebank.

3.4 The Statistical Learning Model

We built on the statistical learning model proposed by (Ratnaparkhi 98):

$$\begin{aligned} P(v,n,p,att) &= P(v) \times P(n) \\ &\quad \times P(att|v,n) \\ &\quad \textit{//overall attachment preference} \\ &\quad \times P(p|att,v,n) \\ &\quad \textit{//particular attachment preference} \end{aligned}$$

where v, n, and p denote a verb, noun and preposition, respectively, and att the random variable, whether there is an attachment or not. The product is composed of the following factors: The probability that a particular verb $P(v)$ or noun $P(n)$ occurs in the training data is constant and can thus be ignored. The overall attachment preference indicates that there are PP attachments to verbs ($P(att_V|v,n)$) or to nouns ($P(att_N|v,n)$). Informally speaking[5], the overall attachment preference can be approximated by calculating the fraction of the frequency of a particular verb or noun in the extracted $[\{v,n\},p,n_2]$ tuples and the occurrence of the verb or noun in the whole training corpus. The particular attachment preference ($P(p|att,v,n)$) is the frequency of a $[\{v,n\},p]$ pattern divided by the occurrence of a particular verb or noun in the extracted $[\{v,n\},p,n_2]$ tuples.

In our experiments, we cover both a wider space of ambiguities (including predicative adjectives) and more complex ambiguities. In addition, we do not only consider bigrams ($[\{v,n,a\},p]$), but also trigrams ($[\{v,n,a\},p,n_2]$) because their inclusion promises better disambiguation results. To accommodate these changes we had to adapt the model by including adjectives as possible anchor points and the calculation of trigrams according to the particular attachment preference. The probabilities of all possible attachment points in an ambiguous case are computed and the attachment point with the maximum probability will be chosen for the disambiguation. There is no disambiguation decision if all probability values are equal or zero.

3.5 Morphological Normalization

Through inflection, verbs, nouns or adjectives surface as different tokens, although their meaning remains unchanged. As our statistical model is only based on the occurrences of words in their surface form, such tokens will be counted as different entities. Morphological normalization of the tokens in the training and the test sets was achieved for the English biomedical texts using a comprehensive biomedical lexicon (the UMLS SPECIALIST lexicon (Browne *et al.* 98)), which contains about 674,000 fully inflected word forms. For the German newspaper texts, we used the morphological analyzer from the MORPHOSAURUS text retrieval system (Hahn *et al.* 04) with a domain-adapted lexicon.

[5] See (Ratnaparkhi 98) for the formal details. He suggests two techniques to calculate the particular attachment preference: interpolation and n-gram count (bigrams in his work). We adopted the latter one.

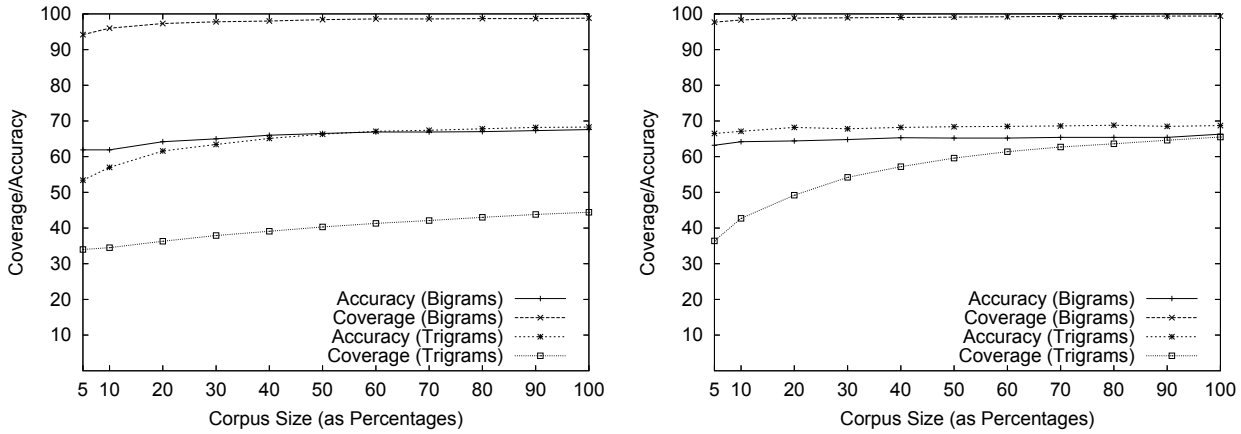


Figure 1: Coverage/Accuracy Values and Corpus Size for NEGRA: original (left) *vs.* morpho. normalized (right)

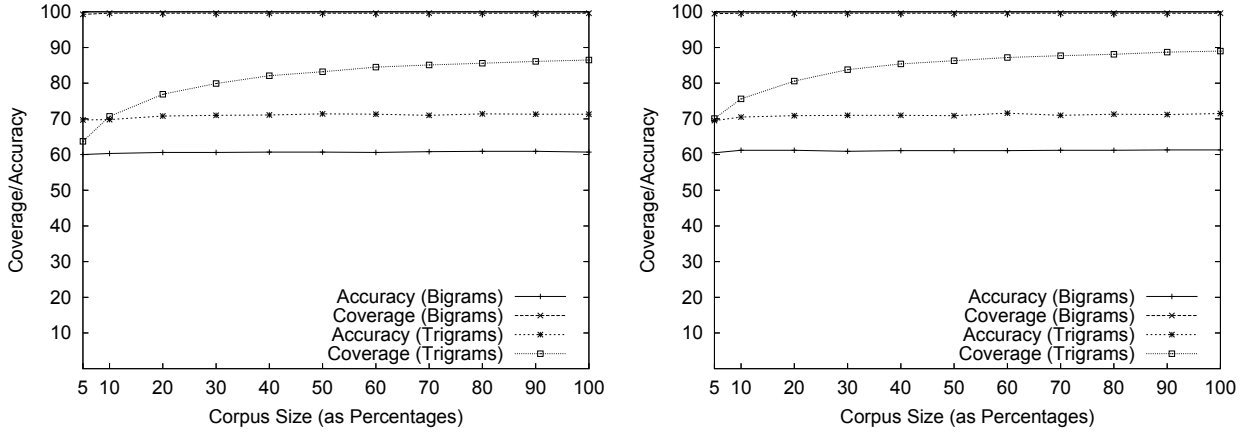


Figure 2: Coverage/Accuracy Values and Corpus Size for GENIA: original (left) *vs.* morpho. normalized (right)

4 Experimental Results

In the following, we report on the experimental results for PP disambiguation using the training methods, test corpora and the statistical model described above. We limit our observations to coverage and accuracy which we define as follows:

$$Coverage = \frac{\text{\# decidable ambiguities}}{\text{\# ambiguities in the test set}}$$

$$Accuracy = \frac{\text{\# correctly disambiguated ambiguities}}{\text{\# decidable ambiguities}}$$

The number of PP attachment ambiguities in both test sets (cf. Section 3.3) are related to those decidable ones for which Ratnaparkhi's learning model (cf. Section 3.4) is able to generate a disambiguation decision (*coverage*), while we also relate the decidable ambiguities to the ones which can correctly be resolved (*accuracy*). We focus here on the change of these values when we vary the size of the training corpus, and, along with this, the number of extracted potential PP attachments (see Subsection 3.2). In order to achieve this goal we downsized our test corpora in steps of 10% (starting from 100%), which means 10.4 million text tokens for the MEDLINE corpus and 11.4 million tokens for the newspaper corpus. We additionally created a corpus with 5% of the original corpus size. For each experimental setting, we also measured the impact of the morphological normalization. Figures 1 and 2 show the learning behavior for both the German and English corpus.

4.1 General Results

Using bigrams as training instances extracted from the original German corpus (see Figure 1, left diagram) in its full size (100%), the accuracy value is about 67.6%. This value changes only marginally when we cut the training corpus in halves (66.5%). Concerning the coverage values of decidable ambiguities with bigrams, a similar behavior can be observed. Although the maximum value of 98.8% is reached with the full training corpus, the increase is only minimal (1 percentage point) from a 30% corpus size onwards.

When we compute the PP disambiguation rate for trigrams, we obtain different results: As expected, the number of decidable ambiguities is considerably smaller compared to those covered by bigrams: 44.4% *vs.* 98.8% for the whole training corpus. This coverage directly depends on the corpus size – it increases from 34% with the 5% training corpus to 44.4% when 100% of the corpus are used. But also under these disambiguation conditions, above 40% of the original corpus size, no substantial improvement concerning the disambiguation accuracy (65.1% to 68.3%) can be attested.

For the morphologically normalized German corpus (see Figure 1, right diagram), the already high coverage of decidable ambiguities for bigrams as the basis of the computation will hardly be influenced. But we found a change of coverage when applying morphologically normalized trigrams. Although the proportion of decidable ambiguities differs barely with the 5% corpus (34,0% original *vs.* 36.4% morphologically normalized), a noticeable difference of about 20 percentage points (39.1% *vs.* 57.2%) emerges when the corpus is expanded to 40% of its original size, with this interval remaining constant (44.4% *vs.* 65.5% with the 100% corpus). Thus a flat, yet constant increase of the coverage values can be noted between for the 40% and the 100% portion of the training corpus. The accuracy values for the extracted bi- and trigrams from the morphologically normalized corpus are hardly influenced by the size of the training corpus.

Turning to the test scenario for English biomedical documents (see Figure 2), we made similar observations, though in an even more clear-cut manner. The coverage values for decidable ambiguities based on original and morphologically normalized bigrams are nearly 100%, even with the minimal 5%-sized corpus. Using trigrams for disambiguation, we witness a strong boost expanding the 5% corpus to about 30% of the original size. From this level on, there is a constantly but slowly growing coverage rate. Here, the coverage of decidable ambiguities for trigrams is much higher than for the corresponding proportion in German newspaper texts, which may be explained by the domain-specific nature of the sublanguage in biomedicine. The accuracy values for the PP disambiguation with bi- and trigrams, both original and morphologically normalized, are not influenced by the corpus size. Compared to the results from the German test scenario, a clear advantage of using trigrams over bigrams (about 10 percentage points) is found.

4.2 The Delta Factor

The results just discussed do not take into account that the maximum probability value might differ only minimally from the other probabilities. In the following, we introduce a delta factor δ that only allows a disambiguation decision, if there is a reasonable difference between the maximum value p_{max1} and the second highest value p_{max2}. Otherwise, the attachment ambiguity cannot be resolved, resulting in a degression of the coverage value (see the Algorithm below).

```
1: if 1 − (p_max2/p_max1) > δ then
2:     decide disambiguation
3: else
4:     no disambiguation
5: end if
```

We experimented with δ values between 0% and 90% in steps of 10 percentage points and observed that with increasing δ, the number of resolvable ambiguities decreases, but the overall disambiguation accuracy increases. We also examined the influence of the size of the training corpus (see Figure 3 for the NEGRA and Figure 4 for the GENIA test corpus, both morphologically normalized). In these 3-dimensional figures, the darker planes represent the accuracy values, whereas the brighter planes stand for the coverage values when varying the corpus size (x-axis) and the δ value (z-axis). For bigrams (left diagrams in Figure 3 and 4), for both NEGRA and GENIA we get the characteristic picture of a strong increase of accuracy for growing δ (from 66.3% with δ=0% to 86.5% with δ=90% (NEGRA) and 61.5% with δ=0% to 81.3% with δ=90% (GENIA)) and an even stronger decrease of coverage (for NEGRA and GENIA from about 99% with δ=0% to 10% with δ=90%). Beyond a corpus size of 40%, the accuracy value only differ minimally (about 1-2 percentage points) for all δ values considered.

For trigrams (right diagrams in Figure 3 and 4), we obtain a different picture. The planes reflecting the accuracy values are more even. This reveals that the influence of both the corpus size and the δ value become weaker. The bending of the coverage planes differs substantially from those discussed above. On the one hand, we see a decline of coverage evoked by the augmentation of δ, but only with a maximum difference of 44 percentage points (compared to nearly 90 percentage point under bigram conditions). On the other hand, for all δ values, we recognize the strong impact of an extending corpus size up to a level of

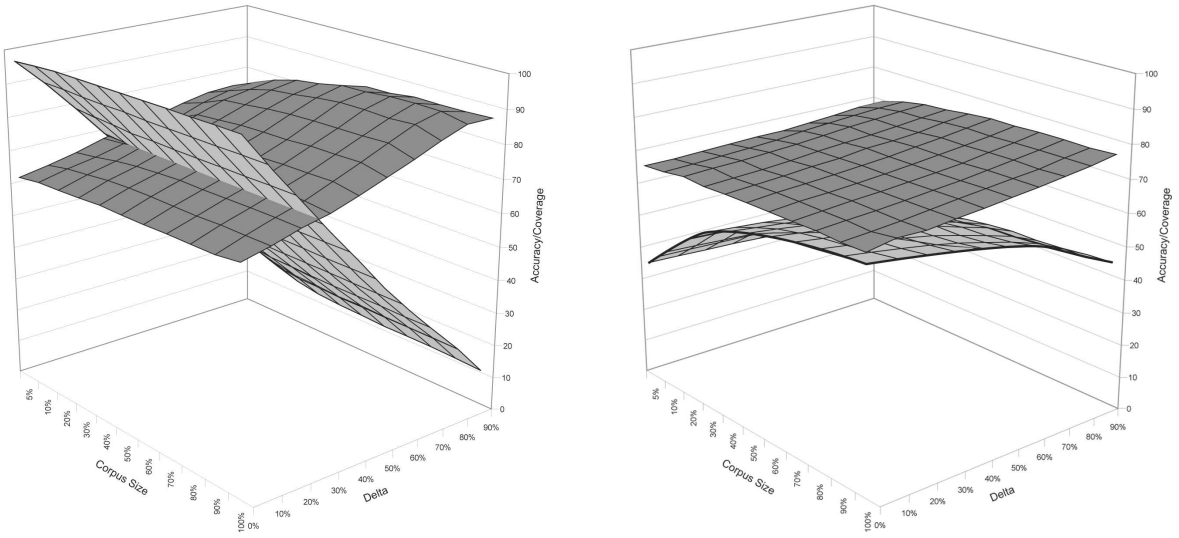


Figure 3: Coverage and Accuracy for bigrams (left) and trigrams (right) (NEGRA)

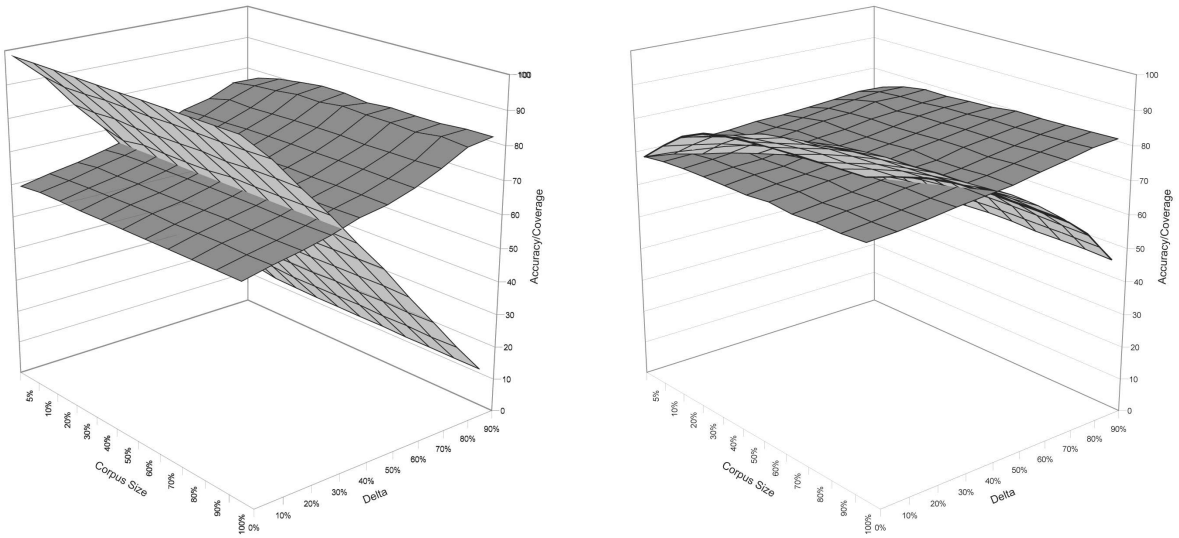


Figure 4: Coverage and Accuracy for bigrams (left) and trigrams (right) (GENIA)

40%. Beyond this barrier, the gain of coverage is fading. Thus even varying assignments of δ values have only marginal effects on the disambiguation performance, once a certain size level has been passed.

5 Discussion

We conclude from our experiments that increasing the size of the training corpus beyond a certain level hardly influences the accuracy of PP disambiguation, under almost all test conditions (German *vs.* English, bigrams *vs.* trigrams, original *vs.* morphologically normalized tokens). Considering the coverage of decidable ambiguities for bigrams, we can already observe a remarkable rate for small corpora, both for original and morphologically normalized tokens (over 90% with a 5-6 million token-sized training corpus). Taking a morphologically normalized corpus and letting its size grow up to 20 million text tokens, no substantial coverage boost is found.

In particular for the biomedical sublanguage, the computation of PP disambiguation on the base of trigrams leads to remarkable positive effects on accuracy. Under these conditions, an increasing corpus size up to a level of about 30-40 million tokens has a positive impact on the coverage rate. Beyond this level, however, the coverage curves do not show an asymptotic expansion, but their gradients are very low. An increase of 1 percentage point already requires a corpus expansion of 10-20 million tokens. This overall picture is further emphasized when the δ factor

comes into play. Although a positive impact on accuracy with increasing δ can be observed, we cannot counter-balance the sometimes drastic decline of coverage by expanding the training corpus.

6 Conclusion

In this paper, we have focussed on the influence of the size of training corpora on coverage and accuracy of PP disambiguation. We have not tried to improve the disambiguation results other than by the variation of corpora size and the introduction of a δ factor, although there are methods that can be applied immediately (e.g., combining the bigram and trigram approach in a back-off model). Furthermore, we tried to strengthen our results not only by considering different languages but also by applying corpora from different domains. We have observed a similar disambiguation behavior for the English and German language, and also for different domains, which might indicate a more general validity of our results.

A natural challenge is now to come up with concrete values of reasonable corpus sizes. Under cautious interpretation of our data, we would set the upper bound for bigram-based PP disambiguation at a level of 20 million morphologically normalized text tokens, for trigrams at an upper level of 40 million text tokens. Of course, especially when dealing with trigrams from the German newspaper training corpus, we observe an increase of coverage from about 56% (40 million tokens) to 65.5% (114 million tokens). But this requires a nearly three times larger training corpus. Although larger corpora do not lead to negative effects on accuracy, we have to keep in mind that linguistic preprocessing of large-sized text (tagging, chunking, shallow parsing, morphological analysis) and the heuristics-based extraction of bi- and trigrams is time- and space-consuming.

Therefore an obvious question arises: How can we substantially increase the number of decidable ambiguities without loosing accuracy, in particular when applying a trigram-based approach or a delta factor, without an enormous extension of the training corpus? In our view, the answer lies in corpus-internal computations, like morphological analysis, that have a positive impact on coverage (a plus of about 20 percentage points on a 40 million German newspaper corpus). We can extend this idea by mapping the head noun of the PP-internal NP to a concept and generalize this concept by exploiting a hierarchically organized concept structure, such as a thesaurus or an ontology, an approach particularly apt for the biomedical domain.

7 Acknowledgements

This work was partly supported by Deutsche Forschungsgemeinschaft (DFG), grant KL 640/5-2, and the European Network of Excellence ”Semantic Mining” (NoE 507505).

References

(Banko & Brill 01) Michele Banko and Eric Brill. Scaling to very very large corpora for natural language disambiguation. In *ACL'01/EACL'01 – Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics and the 10th Conference of the European Chapter of the Association for Computational Linguistics*, pages 26–33. Toulouse, France, July 9-11, 2001. San Francisco, CA: Morgan Kaufmann, 2001.

(Brants 00) Thorsten Brants. TnT: A statistical part-of-speech tagger. In *ANLP 2000 – Proceedings of the 6th Conference on Applied Natural Language Processing*, pages 224–231. Seattle, Washington, USA, April 29 - May 4, 2000. San Francisco, CA: Morgan Kaufmann, 2000.

(Brants *et al.* 03) Thorsten Brants, Wojciech Skut, and Hans Uszkoreit. Syntactic annotation of a German newspaper corpus. In Anne Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*, volume 20 of *Text, Speech and Language Technology*, pages 73–87. Dordrecht: Kluwer Academic Publishers, 2003.

(Brill & Resnik 94) Eric Brill and Philip Resnik. A rule-based approach to prepositional phrase attachment disambiguation. In *COLING'94 – Proceedings of the 15th International Conference on Computational Linguistics*, volume 2, pages 1198–1204. Kyoto, Japan, August 5-9, 1994, 1994.

(Browne *et al.* 98) Allen C. Browne, Guy Divita, Van Nguyen, and Vincent C. Cheng. Modular text processing system based on the Specialist lexicon and lexical tools. In C. G. Chute, editor, *AMIA'98 – Proceedings of the 1998 AMIA Annual Fall Symposium. A Paradigm Shift in Health Care Information Systems: Clinical Infrastructures for the 21st Century*, page 982. Orlando, FL, November 7-11, 1998. Philadelphia, PA: Hanley & Belfus, 1998.

(Callison-Burch & Osborne 03) Chris Callison-Burch and Miles Osborne. Bootstrapping parallel corpora. In Rada Mihalcea and Ted Pederson, editors, *Proceedings of the HLT-NAACL 2003 Workshop 'Building and Using Parallel Texts: Data Driven Machine Translation and Beyond'*, pages 44–49. Edmonton, Alberta, Canada, May 31, 2003. New Brunswick, NJ: Association for Computational Linguistics, 2003.

(Hahn *et al.* 04) Udo Hahn, Kornél Markó, Michael Poprat, Stefan Schulz, Joachim Wermter, and Percy Nohama. Crossing languages in text retrieval via an interlingua. In *RIAO 2004 – Conference Proceedings: Coupling Approaches, Coupling Media and Coupling Languages for Information Retrieval*, pages 100–115. Avignon, France, 26-28 April 2004. Paris: Centre de Hautes Etudes Internationales d'Informatique Documentaire (CID), 2004.

(Hindle & Rooth 93) Donald Hindle and Mats Rooth. Structural ambiguity and lexical relations. *Computational Linguistics*, 19(1):103–120, 1993.

(Kudo & Matsumoto 01) Taku Kudo and Yuji Matsumoto. Chunking with support vector machines. In *NAACL'01, Language Technologies 2001 – Proceedings of the 2nd Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 192–199. Pittsburgh, PA, USA, June 2-7, 2001. San Francisco, CA: Morgan Kaufmann, 2001.

(Pantel & Lin 00) Patrick Pantel and Dekang Lin. An unsupervised approach to prepositional phrase attachment using contextually similar words. In *ACL'00 – Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 101–108. Hong Kong, China, 1-8 August 2000. San Francisco, CA: Morgan Kaufmann, 2000.

(Ratnaparkhi 98) Adwait Ratnaparkhi. Statistical models for unsupervised prepositional phrase attachment. In *COLING/ACL'98 – Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics & 17th International Conference on Computational Linguistics*, volume 2, pages 1079–1085. Montréal, Quebec, Canada, August 10-14, 1998. San Francisco, CA: Morgan Kaufmann, 1998.

(Schwartz *et al.* 03) Lee Schwartz, Aikawa Takako, and Chris Quirk. Disambiguation of English PP attachment using multilingual aligned data. In *MT Summit IX – Proceedings of the 9th Machine Translation Summit of the International Association for Machine Translation*, pages 330–337. New Orleans, Louisiana, USA, September 23-27, 2003s, 2003.

(Volk 01) Martin Volk. Exploiting the WWW as a corpus to resolve PP attachment ambiguities. In Paul Rayson, Andrew Wilson, Tony McEnery, Andrew Hardie, and Shereen Khoja, editors, *Proceedings of the Corpus Linguistics 2001 Conference*, pages 601–606. Lancaster, UK, 29 March - 2 April 2001. Lancaster University, University Centre for Computer Corpus Research on Language, 2001.

(Volk 02) Martin Volk. Combining unsupervised and supervised methods for PP attachment disambiguation. In *COLING 2002 – Proceedings of the 19th International Conference on Computational Linguistics*. Taipei, Taiwan, August 24 - September 1, 2002. Association for Computational Linguistics, 2002.

Persian word order is free but not (quite) discontinuous

Allan Ramsay
School of Informatics
Manchester M60 1QD, UK
Allan.Ramsay@manchester.ac.uk

Najmeh Ahmed, Vahid Mirzaiean
Technical University Tehran
Tehran, Iran

Abstract

Persian word order is fairly free. In particular, the major constituents of a sentence may be permuted into a variety of orders, some of which are more marked than others but all of which are permissible. To make matters worse, subjects are often left implicit, as there is generally enough information in the agreement marker of the verb to indicate who or what the subject is. To make matters worse still, relative clauses in Persian often, but not always, include a 'resumptive pronoun'.

Taken together, these phenomena make it plausible that Persian sentences would be wildly ambiguous. It turns out that in many cases there is a surface form which encodes the intended meaning unambiguously. We outline an implemented grammar of Persian which produces unique analyses of a range of surface forms, simply by exploiting one rather uncontroversial constraint.

1 Introduction

Persian is a language where even quite simple sentences seem likely to produce multiple analyses. The constituents of a sentence can be freely permuted (Lazard 92), subjects can be freely omitted (Farid 97), there is generally no distinction between nouns and NPs (Amtrup & Rad 00), relative clauses may or may not contain resumptive pronouns (Taghvaeipour 03), ... There is, apparently, scope for considerable ambiguity.

And yet Persian speakers have no more difficulty with understanding Persian than English speakers have with understanding English, or German speakers with understanding German. It seems that there are enough constraints on what relations actually hold within a sentence to allow people to find a small number of analyses of an input string, just as there are in general for other languages. In the current paper we will outline a description of Persian within a framework where the default assumption is that word-order is completely free, with local constraints being imposed to reflect the facts about specific languages. This framework is supported by an appropriate parsing algorithm which we have used elsewhere to give an account of a number of other languages ((Ramsay 99; Ramsay & Seville 00; Ramsay & Mansour 04)). We include the output of the parser, with timings, to illustrate the effects of specific constraints.

The general framework is as follows:

1. Lexical entries for verbs specify the syntactic category and thematic roles of their arguments (Mirzaeian 03). The vast majority of Persian verbs are 'light' – they involve some fairly generic verb, meaning something like *'do'* or *'make'*, and a noun which specialises the meaning (Butt 02; Mohammad & Karimi 92)[1].

2. These arguments compete for the surface roles of subject, first object and second object. The winners of the competition acquire appropriate constraints. For Persian this means that the subject is required to share number and person agreement with the verb and to be potentially case-marked as nominative, and that the first object is marked by the postposition را (rā) (Khanlari 73; Lazard 92). The light argument, if there is one, normally comes immediately before the verb (Dabir-Moghaddam 89; Mohammad & Karimi 92).

3. The canonical order of the arguments is determined. For Persian we take this to be SUBJ-OBJ-V for simple transitive verbs, and SUBJ-V-COMP for verbs that take sentential complements (Farid 97).

4. Constraints on where the arguments can actually appear are specified. For Persian we initially assume that there are no constraints

[1]There is no obvious surface difference between bare nouns and definite NPs in Persian: we assume that the light argument is a noun, with the combination making a phrasal verb, but nothing in the remainder of this paper depends on this assumption.

on argument order (we will revise this later).

5. Discontinuous non-WH clauses are not allowed.

Given this basic framework, all the sentences in (1) produce the same dependency tree, namely the one given in Fig. 1[2].

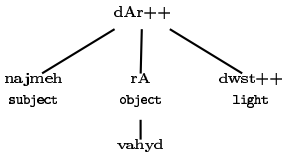


Figure 1: نجمه وحید را دوست داشت. (najmeh vaḥyd rā dvst dāšt.) (0.02 sec)

It is clear in each case that وحید را (vaḥyd rā) is the object, since it carries the object marker, which means that نجمه (najmeh) must be the subject. In other words, the case marking is enough to determine the roles of the two NPs, and hence the order is irrelevant, at least as far as assigning thematic roles is concerned.

(1) a. نجمه وحید را دوست داشت. (najmeh vahyd obj-marker friend has. ≈ Najmeh likes Vahid – دوست داشت is a light verb combination)

b. وحید را نجمه دوست داشت. (vahyd obj-marker najmeh friend has.)

c. نجمه دوست داشت وحید را. (najmeh friend has vahyd obj-marker.)

d. وحید را دوست داشت نجمه. (vahyd obj-marker friend has najmeh.)

e. دوست داشت وحید را نجمه. (friend has vahyd obj-marker najmeh.)

f. دوست داشت نجمه وحید را. (friend has najmeh vahyd obj-marker.)

Some of the orders in (1) are rather marked, and would only occur when discourse or stylistic factors demanded them, but they are all at least comprehensible, and are acceptable in appropriate contexts. We assign a penalty to marked orders, and in general we assume that the lower scoring analyses are preferred. Note that although Persian word order is potentially very free, it is *very* unusual for the components of a light verb to be split up, and we do not allow analyses where this happens – دوست (dvst) must occur immediately before داشت (dāšt).

[2]This analysis includes information about morphological structure, so that the head verb shown as dAr++, consisting of the stem دار (dār) and two empty inflectional affixes, rather than as the surface form داشت (dāšt)

2 Sentential complements

The situation gets a bit more complicated when we consider verbs that take sentential complements. If there were no constraints at all on word order then a sentence like نجمه الن وحید را دوست دارد میداند. (najmeh ālan vaḥyd rā dvst dārd mydānd.) (najmeh Alan vahyd obj-marker friend has knows. ≈ Najmeh knows Allan likes Vahyd) would have two analyses, since either نجمه (najmeh) or الن (ālan) could be the subject of دوست دارد (dvst dārd), with the other one acting as the subject of میداند (mydānd). Worse than that, نجمه الن وحید را میداند دوست دارد. (najmeh ālan vaḥyd rā mydānd dvst dārd.) would be accepted, with the same two analyses, since all we would have to do is to find roles for each of the NPs, and that could be done perfectly easily.

We start by considering a sentence in canonical form:

(2) نجمه میداند الن وحید را دوست دارد. (najmeh mydānd ālan vaḥyd rā dvst dārd.)

If there are no constraints at all on word order, (2) produces the two analyses in Fig. 2.

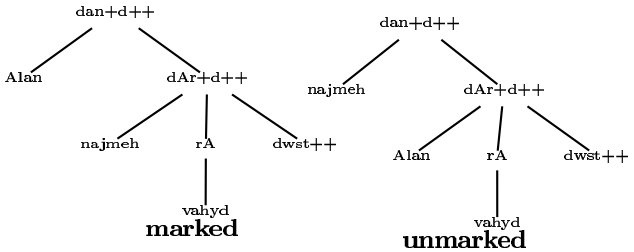


Figure 2: نجمه میداند الن وحید را دوست دارد. (najmeh mydānd ālan vaḥyd rā dvst dārd.) (0.09 sec)

One of these is reasonable, the second is rather less so. To obtain the implausible one we have to assume that the subject of میداند (mydānd) has been right-shifted, that the subject of دوست دارد (dvst dārd) has been left-shifted, with the subject of دوست دارد (dvst dārd) being shifted *beyond* the main verb, and that the sentential complement has also been right-shifted and is non-compact.

This second analysis is clearly extremely marked – various items have been shifted around, and the sentential complement is discontinuous but is not WH-marked. If we ban items of this kind then the second analysis is eliminated.

The freedom of word order allowed in Persian means that quite a number of permutations of the constituents of (2) can occur. In most cases the requirement that all non-WH clauses are contin-

uous means that at most one reading is obtained, but in (3) we get the two interpretations in Fig. 3.

(3) نجمه وحید را دوست دارد الن میداند. (naǰmeh vaḥyd rā dvst dārd ālan mydānd.)

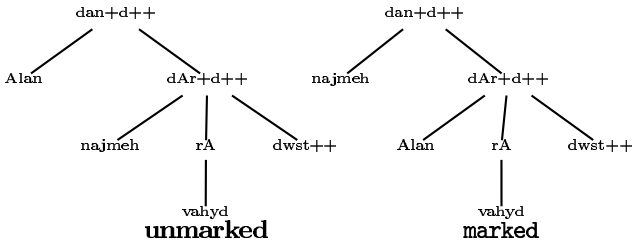


Figure 3: Two readings of نجمه وحید را دوست دارد الن میداند. (naǰmeh vaḥyd rā dvst dārd ālan mydānd.) (0.08 sec)

The second reading here is extremely marked, since it involves treating الن (ālan) as the right-shifted subject of دوست دارد (dvst dārd), but it is allowed if the only constraint we exploit involves banning discontinuous phrases.

We can force a choice between the two readings of (3) by using the postposition را (rā) to mark the end of the embedded clause:

(4) a. نجمه وحید را دوست دارد را الن میداند. (naǰmeh vaḥyd rā dvst dārd rā ālan mydānd.)

b. نجمه وحید را دوست دارد الن را میداند. (naǰmeh vaḥyd rā dvst dārd ālan rā mydānd.)

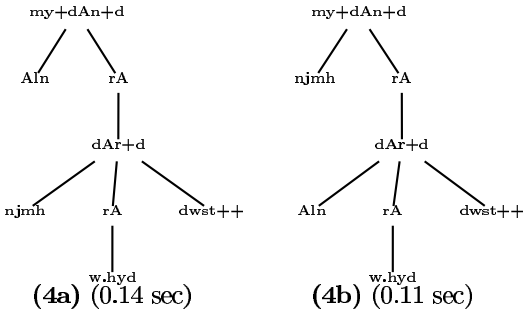


Figure 4: Using را (rā) to mark the boundaries forces unique readings

A similar effect is produced by the complementisers اینکه (īnkh) and که (kh). We assume that اینکه (īnkh) is used to mark left-shifted sentential complements and که (kh) to mark complements in canonical position (Amtrup & Rad 00).

(5) a. نجمه وحید را دوست دارد الن میداند. اینکه (īnkh naǰmeh vaḥyd rā dvst dārd ālan mydānd.)

b. اینکه وحید را دوست دارد الن میداند. نجمه (naǰmeh īnkh vaḥyd rā dvst dārd ālan mydānd.)

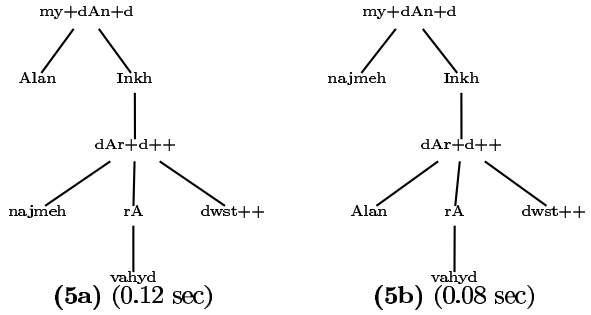


Figure 5: Using اینکه (īnkh) to mark the complement

(5b) cannot have نجمه (naǰmeh) as the subject of دوست دارد (dvst dārd), since it is outside the scope of اینکه (īnkh). دوست دارد (dvst dārd) therefore has to take الن (ālan) as its subject, despite its marked position, forcing نجمه (naǰmeh) to be the subject of میداند (mydānd), since that is the only vacant role.

You can often use both اینکه (īnkh) and را (rā) to completely delineate the boundary of the sentential complement. The exact circumstances under which you need one or the other are too detailed to discuss at length here.

3 Zero subjects

The situation is made more complicated by the fact that Persian, like many other languages with reasonably rich inflectional morphology, allows the subject to be omitted, on the grounds that the morphology provides enough information to determine who it is (Mirzaeian 03; Khanlari 73; Lazard 92). Consider (6):

(6) a. نجمه وحید را دوست دارم را میداند. (naǰmeh vaḥyd rā dvst dārm rā mydānd.)

b. نجمه وحید را دوست دارد را میدانم. (naǰmeh vaḥyd rā dvst dārd rā mydānm.)

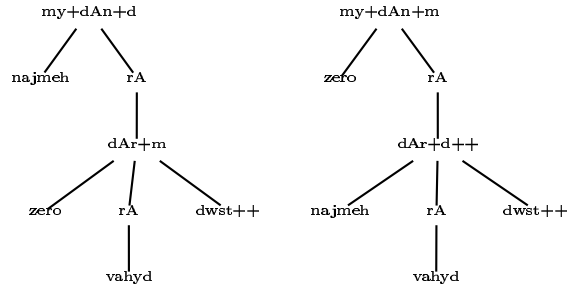


Figure 6: The agreement chooses the interpretation

In (6a), the first-person marker on دارم (dā-rm) means that نجمه (naǰmeh) cannot be its subject. So the only thing نجمه (naǰmeh) can be is

the subject of میداند (mydānd), and hence the subject of دارم (dārm) must be implicit. likewise نجمه (najmeh) cannot be the subject of the first-person verb میدانم (mydānm) in (6b), so it must the subject of دارد (dārd) and the hence the subject of میدانم (mydānm) must be implicit.

In (7) these constraints do not apply, and hence we get the two analyses in Fig. 7:

(7) نجمه وحید را دوست دارد میداند. (najmeh vaḥyd rā dvst dārd mydānd.)

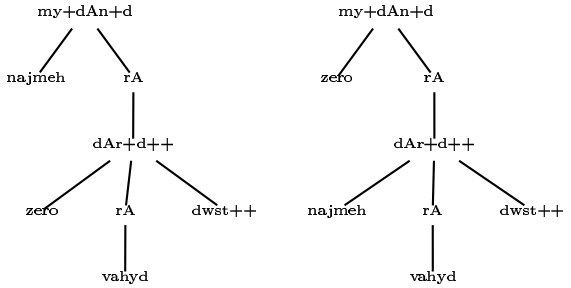
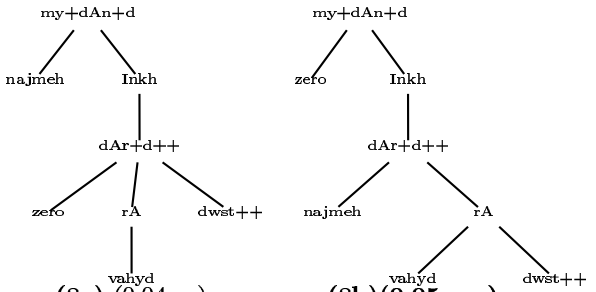


Figure 7: The agreement doesn't help (0.06 sec to get both readings)

At this point, however, including the complementiser at the appropriate point makes one interpretation substantially less acceptable than the other, as before.

(8) a. نجمه اینکه وحید را دوست دارد میداند. (najmeh īnkh vaḥyd rā dvst dārd mydānd.)

b. اینکه نجمه وحید را دوست دارد میداند. (īnkh najmeh vaḥyd rā dvst dārd mydānd.)



(8a) (0.04sec) **(8b)(0.05 sec)**

Figure 8: اینکه (īnkh) saves the day again.

Note that simply using را (rā) after the embedded clause, نجمه وحید را دوست دارد را میداند (najmeh vaḥyd rā dvst dārd rā mydānd), does not do the job this time, since there is then no way of telling whether نجمه (najmeh) is the subject of دوست دارد (dvst dārd) or of میداند (mydānd), and hence of telling which one should have a zero subject. If we included both اینکه (īnkh) and را (rā), of course, then the position of اینکه (īnkh) would sort it out as in Fig. 8.

The examples above show that although Persian allows a great deal of freedom of word order, which can be compounded by the possibility of leaving the subject implicit, including appropriate particles (complementisers and the postposition را (rā)) can distinguish between different readings (Karimi 90; Hajati 76; Lazard 82). All we need is a ban on discontinuous non-WH clauses.

4 Relative clauses

The situation gets more complex when we consider relative clauses. There are two ways of making relative clauses in Persian.

1. The word که (kh) can function as a pronoun which marks the clause that contains it as a relative clause, in much the same way as the English relative pronoun *'who'* (Amtrup & Rad 00). Like *'who'* it has to appear as the first item in the relative clause. Unlike the English counterparts, که (kh) is not case-marked and can denote either animate or inanimate entities (though when it is used in questions it can only denote animate entities).

   When که (kh) is the object of the verb, the object marker را (rā) is optional, but if it is present then it must precede the که (kh) (Taghvaeipour 03).

2. The same word که (kh) can do very much the same job, but the relative clause contains a 'resumptive pronoun'. We will return to examples like this in Section 4.2.

(9) illustrates a very simple NP including a relative clause. The analysis is given in Fig. 4. Note the suffix ی (-ī) on the head noun. This item has a rather curious status. It can appear on a simple noun, in which case it marks it as being indefinite, but it is obligatory if the noun is modified by a relative clause, in which case the entire NP is definite (Amtrup & Rad 00). We deal with this by letting the relative clause specify that its target should be indefinite but that the result of adding the relative clause to the target is definite.

(9) مردی را که نجمه دوست دارد (mardī rā kh najmeh dvst dārd) (man obj-marker who Najmeh friend has)

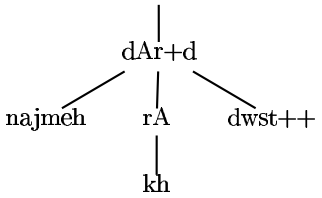


Figure 9: the man who Najmeh likes (0.05 sec)

The existence of two ways of using که (kh) to make relative clauses seems likely to increase the ambiguity of such constructions (Taghvaeipour 03). It turns out that in many cases they make it easier to produce *unambiguous* forms.

4.1 Relative clauses with zero subjects

The first such case involves relative clauses where the WH-marked item is the subject. We start with a marginally acceptable example:

(10) مارديِ که میداند وحید را دوست دارد (mārdī kh mydānd vaḥyd rā dvst dārd)

Neither دوست دارد (dvst dārd) nor میداند (mydānd) has an explicit subject. There are therefore two conceivable interpretations of this phrase, one where که (kh) is the subject of میداند (mydānd) and دوست دارد (dvst dārd) has a zero subject (something like *'man who knows (she) likes Vahid'*), and one where it is میداند (mydānd) which has the zero subject and دوست دارد (dvst dārd) which has که (kh) as subject (*'man who (she) knows likes Vahid'*).

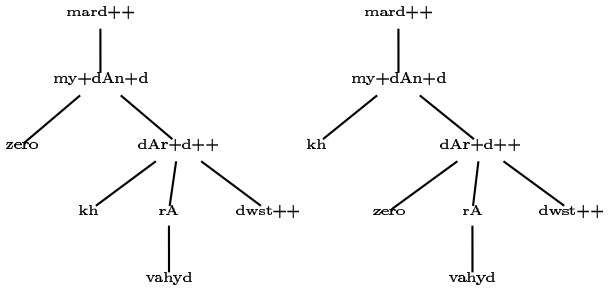


Figure 10: Which one has the zero subject?

The interpretation that makes که (kh) the subject of دوست دارد (dvst dārd) is much more marked than the one where it is the subject of میداند (mydānd), but the fact that it is at least conceivable is shown by میدانم وحید را دوست دارد مردیِ که (mardī kh mydānm vaḥyd rā dvst dārd). Here, as in Fig. 3, the agreement marked on میدانم (mydānm) rules out mکه (kh) as its subject. It therefore becomes the subject of دوست دارد (dvst dārd), leaving the subject of میدانم (mydānm) implicit.

The fact that this reading is possible when agreement rules out the other one must mean that it is at least theoretically possible to get the marked reading of (10). The other reading is strongly preferred, however, partly because of the extreme markedness of having the subject shifted so far, but also because there is a form of words that encodes this interpretation uniquely:

(11) مردیِ که او میداند وحید را دوست دارد (mardī kh ū mydānd vaḥyd rā dvst dārd)

In (11) the pronoun او (ū) cannot be part of a که او (kh ū) resumptive pronoun construction because such constructions cannot be subjects. So او (ū) itself must be the subject of میداند (mydānd), in which case the only role free for که (kh) is as the subject of دوست دارد (dvst dārd), as shown in Fig. 11.

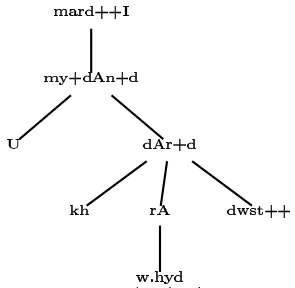


Figure 11: (11) (0.36 sec)

The key point here is that there is a form which expresses the *'man who she knows likes Vahid'* reading, and which furthermore does so unambiguously. This will tend to support the tendency to read (10) as meaning *'man who knows she likes Vahid'*: since there is a way of expressing the more marked interpretation of (10), there is no particular reason to read (10) this way.

4.2 Relative clauses with resumptive pronouns

Relative clauses can also be constructed with که (kh) as the initial item but also including a 'resumptive pronoun' (Taghvaeipour 03; Ahangar 03). We have seen two situations in which که (kh) can be the first item in a clause:

- It could be a complementiser, in which case the clause should contain all the other elements that are required by the main verb.
- It could be a relative pronoun, in which case one of the other items that are normally required by the verb should be missing (since the pronoun will be playing this role).

A relative clause which includes a resumptive pronoun is likely to be a variant of one of these. But which?

We choose to say that که (kh) and the resumptive pronoun make a single discontinuous NP, which plays some syntactic role within the clause. The advantage of this is that we get an automatic check for the presence of the resumptive pronoun, and that we also automatically spot that it is indeed the relativised item. The disadvantage is, clearly, that it makes parsing a bit more complex, since we have to spot the presence of such discontinuous items and then we have to allow them to play their normal syntactic roles.

Fig. 12 shows the analysis that we obtain for (12) if we take this approach.

(12) مردي که وحيد او را ديد (mardī kh vaḥyd ū rā dyd)

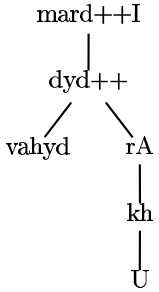


Figure 12: NP containing a relative clause with a resumptive pronoun (0.04 sec)

The key point to note about the analysis in Fig. 12 is that که (kh) and او (ū) form a single discontinuous relative pronoun. This makes it easy to see from the outside which component of the clause has been relatavised, since the relative pronoun is in fact the head of the relativised pronoun.

Resumptive pronouns are obligatory in some situations, optional in some, and illegal in some. They are obligatory, or at least very strongly preferred, in the following cases:

- when the relativised item is part of a prepositional phrase.
- when the relativised item is a possessive marker
- when the relativised item is part of an embedded clause

What these have in common is that the relativised item is not an argument of the main verb (noted that this also applies to (12), where the relative pronoun is dominated by the postposition را (rā), and hence is not directly dominated by the main verb. This contrasts with examples like (9), where the extraposed item را که (rā kh) is complete constituent and hence does not require a resumptive pronoun).

We can capture this by saying that the non-resumptive form can only be used in cases where the relativised item is an argument of the main verb of the clause, and the resumptive form can only be used when the relativised item is *not* an argument of the main clause. Testing for whether or not there is a resumptive pronoun can be done by direct inspection of the relativised item, since this will be a complex discontinuous phrase containing the resumptive pronoun if one is indeed present, and hence can be done locally.

Note that this allows both (13a) and (13b), since in (13a) the whole relativised item که را (rā kh) is an argument of دید (dyd), whereas in (13b) the discontinuous relative pronoun که+او (kh+ū) is an argument of را (rā) rather than being a direct argument of دید (dyd).

(13) a. مردي را که نجمه دید (mardī rā kh najmeh dyd)

b. مردي که نجمه را او دید (mardī kh najmeh rā ū dyd)

This treatment of relative clauses containing resumptive pronouns allows us to obtain the analyses in Fig. 13 for (14a) and (14b).

(14) a. مردي که نجمه میداند وحید او را دید (mardī kh najmeh mydānd vaḥyd ū rā dyd)

b. مردي که نجمه میداند او وحید را دید (mardī kh najmeh mydānd ū vaḥyd rā dyd)

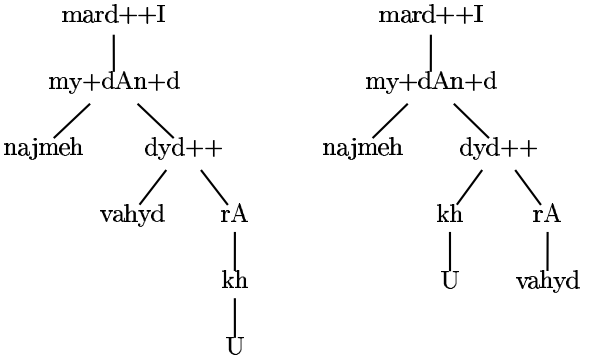


Figure 13: Relative clauses with sentential complements

Because Persian allows zero subjects, if resumptive pronouns were not required in contexts like

(14) (particularly (14b) then there would potentially be considerable ambiguity about whether the subject of such a clause was implicit or was the relativised item. Thus although the fact that که (kh) may or not require a resumptive pronoun leads to a certain degree of local ambiguity, it does in fact help reduce the number of globally acceptable interpretations (the analyses in Fig. 13 are the *only* analyses we obtain for the examples in (14)).

5 Conclusions

The analyses given above illustrate how the apparent freedom of Persian word order is constrained by interactions between sets of local constraints. Where simple constraints such as the need for the subject and the verb to agree, or the requirement that the object be followed by را (rā-), are available then a wide variety of orders are permitted. In many cases, however, simple constraints of this kind are not available, or do not suffice for clear disambiguation of the various options. This becomes particularly significant when there is a sentential complement; when there is a zero subject, either for the main verb or for the verb of the embedded clause; and where there is a relative clause.

To some degree you can select among the various choices by imposing a penalty on non-canonical orders, as in Fig. 2, but this is not always enough to make the situation entirely clear. In many cases bracketing the embedded clause with either اینکه (īnkh) or را (rā) constrains the allocation of arguments to verbs, so that it is, for instance, possible to force the subject of an embedded clause to follow its verb simply by following it with را (rā) (3)(b). In this example را (rā) cannot be an object marker for the preceding noun, because then the other را (rā) marked NP in the sentence would have no role. The second را (rā) in this example, then, must be a marker on the embedded sentence as a whole, and as such forms a boundary which forces الن (āln) to be constituent of the embedded sentence (and hence, in fact, to be its subject).

In general, Persian sentences contain just enough markers to constrain the allocation of arguments to verbs. This can be seen even more clearly in the case of relative clauses, where the combination of zero subjects and the ambiguity between using که (kh) as a complementiser and using it as a relative pronoun could lead to considerable amounts of ambiguity. By treating the combination of the relative pronoun که (kh) and a resumptive pronoun like او (ū) as a single *discontinuous* item, and placing fine-grained constraints on whether the WH-marker for the relative clause is the single atomic item که (kh) or is a discontinuous compound of this kind, we again find that there is just enough information to derive unique interpretations in cases which would otherwise be highly underdetermined. The moral of the work described here is that the potential for a wide range of non-canonical orders in Persian *could* lead to a very high degree of ambiguity, but the language provides just enough local constraints to avoid this happening in the vast majority of cases.

References

(Ahangar 03) A A Ahangar. Persian relative clause derivation based on move-alpha. In *XVII International Congress of Linguists*, Prague, Czech Republic, 2003.

(Amtrup & Rad 00) J W Amtrup and H M Rad. Persian-English machine translation: An overview of the Shiraz project. Technical report, Computing Research Laboratory New Mexico State University, Las Cruces, New Mexico, 2000.

(Butt 02) M Butt. The light verb jungle. In *Workshop on light verbs*, Harvard, 2002.

(Dabir-Moghaddam 89) M Dabir-Moghaddam. Piramune 'ra' dar zabane farsi. *Majalleye Zabanshenasi*, 7(1):2–60, 1989.

(Farid 97) K A Farid. *A Discourse-Pragmatic Description of Marked Constructions in Persian.* Unpublished PhD thesis, Department of Language Engineering, Manchester, UMIST, 1997.

(Hajati 76) A Hajati. Fe'le lazem va ra dar zabane Farsi. *Majalleye Daneshkadeye Adabiyat va Olume Ensaniye Tarbiyat Mo'allem*, 5:185–211, 1976.

(Karimi 90) S Karimi. Obliqueness, specificity and discourse functions: Ra in Persian. *Linguistic Analysis*, 20:139–190, 1990.

(Khanlari 73) P Khanlari. *Tarix-e Zaban-e Farsi, Vol 2.* Bonyad-e Farhang, Tehran, 1973.

(Lazard 82) G Lazard. Le morpheme ra en Persan et les relations actancielles. *Bulletin de la Societe de Linguistique de Paris*, 73:177–208, 1982.

(Lazard 92) G Lazard. *A Grammar of Contemporary Persian.* Mazda Publishers in association with Bibliotheca Persica, Costa Mesa, Calif., 1992.

(Mirzaeian 03) V R Mirzaeian. *Computational Content-Based Support for Persian Learners of English.* Unpublished PhD thesis, Department of Computation, Manchester, UMIST, 2003.

(Mohammad & Karimi 92) J Mohammad and S Karimi. Light verbs are taking over: Complex verbs in Persian. In *Proceedings of the Western Conference on Linguistics*, 1992.

(Ramsay & Mansour 04) A M Ramsay and H Mansour. The parser from an arabic text-to-speech system. In *Traitement automatique du language naturel (TALN'04)*, Fès, Morroco, 2004.

(Ramsay & Seville 00) A M Ramsay and H Seville. Unscrambling English word order. In M Kay, editor, *Proceedings of the 18th International Conference on Computational Linguistics (COLING-2000)*, pages 656–662, Universität des Saarlandes, July 2000.

(Ramsay 99) A M Ramsay. Direct parsing with discontinuous phrases. *Natural Language Engineering*, 5(3):271–300, 1999.

(Taghvaeipour 03) M Taghvaeipour. Persian relative clauses in HPSG. In *Camling*, University of Cambridge, 2003.

An Algorithm Detecting and Tracing Errors in ICALL Systems

Veit Reuer and **Kai-Uwe Kühnberger**
Institute of Cognitive Science
University of Osnabrück
Katharinenstr. 24
49069 Osnabrück, Germany
{vreuer,kkuehnbe}@uos.de

Abstract

If a language learner generates grammatical errors, a classical problem for unification-based ICALL systems is the occurrence of inconsistencies with the grammar formalism. To solve this problem, we propose an account that codes errors of learners explicitly. Using a particular extension of classical feature logic (by introducing a designated error feature) it is possible to code errors of the language learner. We will present an algorithm that computes the unification of two feature structures for such an extension. Furthermore some properties of the algorithm and an evaluation will be presented.

1 Introduction

A classical problem for ICALL systems (Intelligent Computer-Assisted Language Learning systems) is the balance between grammar formalisms requiring consistent data structures, and errors of language learners that are inconsistent with the grammar. In particular, for unification-based grammar formalisms – e.g. HPSG (Pollard & Sag 94) or LFG (Bresnan 01) – it is hard to handle inconsistencies, because unification requires unifiable, hence consistent, data structures.

In order to solve this problem, different parsing methods and strategies to identify errors were developed. With respect to parsing methods, so-called *robust* parsing tries to continue parsing past a position that cannot be handled by the grammar without considering the type and exact location of the error (Jensen et al. 83). Second, *sensitive* strategies are being developed specifically for locating and analyzing errors in the input. The parsing process continues across the error position and yields a complete description of the input, usually including the position and the type of error. In a system, aimed at both determining the grammaticality *and* providing as much feedback about errors as possible, only the second type of parsing method can be adopted. Concerning possible strategies for identifying errors, we can distinguish between *anticipation-based* parsing and *anticipation-less* parsing. The first strategy tries to extend the grammar with additional rules covering the various cases of erroneous input. The second strategy modifies the parsing algorithm itself to allow for error recognition (Menzel 92). In the present account, we use an anticipation-less parsing strategy, because the most efficient parsing algorithms can be chosen and it is possible to import linguistic data from other computational linguistic applications.

There is a certain tradition of related work in the field of modeling clashing feature values in unification-based approaches.[1] In (Schwind 88), plain disjunctive features are used to keep the clashing information, but additional principles are needed to select the most sensible resulting structure of several alternatives. In the approaches (Carpenter 93) using default unification and (Sågvall Hein 98) applying grammar checking, it is impossible to generate precise feedback, because only one of the clashing features is coded in the resulting feature structure, i.e. information gets lost. Furthermore the approaches (Vogel & Cooper 95) and (Fouvry 03) raise the question how clashing values can be coded in the feature structure. All these approaches show disadvantages that do not arise in the present approach.

The paper is structured as follows: In Section 2, we will roughly sketch an extension of classical feature logic. Section 3 describes the parsing algorithm that is used in the implemented ICALL system. Section 4 presents some properties of this algorithm together with an evaluation and Section 5 adds some ideas for future work.

2 An Extension of Feature Logic

2.1 An Introductory Example

Consider the sentence "Ich habe jetzt eine Unfall gesehen" ("I have seen an accident now."). There is a clash in agreement between the determiner and the noun in the object NP: "gese-

[1] A good overview of existing proposals is provided by (Vandeventer Faltin 03).

hen" ("seen") calls for an accusative object, however "Unfall" ("accident") is accusative masculin whereas "eine" ("a") is accusative feminin. Figure 1 shows the (simplified) corresponding lexical information for "eine" and "Unfall".

"eine"

$$\begin{bmatrix} \text{def} & : - \\ \text{gen} & : \text{f} \\ \text{num} & : \text{sg} \\ \text{case} & : \text{acc} \end{bmatrix}$$

"Unfall"

$$\begin{bmatrix} \text{pred} & : \text{'ACCIDENT'} \\ \text{gen} & : \text{m} \\ \text{num} & : \text{sg} \\ \text{case} & : \text{acc} \end{bmatrix}$$

Figure 1: Lexical entries for "eine" and "Unfall".

The unification of these two feature structures should fail because the values of the "gen"-feature clash. In our approach, the clash of these values leads to the insertion of a designated feature "*err*" containing the relevant error information (Figure 2). The resulting feature structure is the description of the phrase "eine Unfall" including the information about the clash of values.

$$\begin{bmatrix} \text{def} & : - \\ \text{gen} & : \text{f} \\ \text{num} & : \text{sg} \\ \text{case} & : \text{acc} \end{bmatrix} \sqcap_{err} \begin{bmatrix} \text{pred} & : \text{'ACCIDENT'} \\ \text{gen} & : \text{m} \\ \text{num} & : \text{sg} \\ \text{case} & : \text{acc} \end{bmatrix} =$$

$$\begin{bmatrix} \text{def} & : - \\ \text{gen} & : \text{f} \\ \text{num} & : \text{sg} \\ \text{case} & : \text{acc} \\ \text{pred} & : \text{'ACCIDENT'} \\ \hline err & : \left\{ [\ \text{gen} : \text{m}\ ] \right\} \end{bmatrix}$$

Figure 2: Example of a unification with mismatching values for the gender feature

2.2 The Logic of Extended Feature Structures

LFG-style representations are formally based on feature constraint logic. We will extend a classical feature constraint logic approach (Smolka 92) by a designated error feature *err* used to model occurring errors of language learners. We will not provide any proofs for the underlying facts and refer the interested reader to (Reuer & Kühnberger 05) for a development of the logical basis of this theory.

Definition 1 *A* feature algebra[2] $\mathfrak{A}$ *is a pair* $\langle D, I\rangle$ *where* D *is a non-empty set and* I *is an interpretation function defined on constants by* $I : Con \rightarrow D$ *and on features by* $I : Feat \rightarrow \mathfrak{P}(D \times D)$ *such that the following conditions hold:*

(i) $[\langle a, b\rangle \in I(f) \wedge \langle a, c\rangle \in I(f)] \rightarrow [b = c]$
(ii) $[I(a) = I(b)] \rightarrow [a = b]$
(iii) $\forall a \in Con \forall d \in D : \langle a, d\rangle \notin I(f)$

In other words, features are interpreted as (partial) functions, we assume the unique name assumption, and constants are considered as atomic.

Definition 2 *A* feature graph *is either a graph without edges, i.e. a pair* $\langle a, \emptyset\rangle$ *where* $a \in Con$*, or a graph* $\langle x, E\rangle$*, where* $x \in Var$ *is the root of the graph and* E *is a finite set of edges of the form* yfs *(for* $y \in Var$*,* $f \in Feat$*, and* $s \in Con \cup Var$*) such that the following three conditions are satisfied:*

(i) Edges are uniquely defined
(ii) The graph is connected
(iii) The graph is acyclic

It is well-known that a partial order relation $\leqslant$ can be introduced ordering the contained subgraphs of a matrix graph (Smolka 92). Furthermore a subsumption preorder $\preceq$ between feature graphs can be introduced by a homomorphic embedding $\psi : \langle x, E_G\rangle \mapsto \langle x', E_{G'}\rangle$ on feature graphs with the properties (Smolka 92):

- ψ maps x to x'
- $\psi(a) = a$ for all constants a
- Every edge $xfs \in E_G$ is mapped to the edge $\psi(x)f\psi(s) \in E_{G'}$

We will extend feature constraint logic by a designated error feature *err*. To get a subsumption relation on extended feature structures we define a substitution operation on feature-value pairs.

Definition 3 *Assume a feature graph* $G_\oplus$ *extended by a designated feature* err *is given where* err *is defined by the two nodes* x *and* $\{[f_1 : y_1], \ldots, [f_n : y_n]\}$*. A* substitution Θ *on* $G_\oplus$ *is a contravariant pair of substitution functions* $\Theta = \langle \Theta_i^\wedge, \Theta_i^\vee\rangle$ *such that the following holds:*

(i) $\Theta_i^\wedge$ *substitutes* x_i *by* y_i*, if* $i \in \{1, \ldots, n\}$*,* $xf_ix_i \in G_\oplus$*,* $x_i \in Con$*, and* $y_i \in Con$*.*
(ii) $\Theta_i^\vee$ *substitutes* y_i *by* x_i*, if* $i \in \{1, \ldots, n\}$*,* $xf_ix_i \in G_\oplus$*,* $x_i \in Con$*, and* $y_i \in Con$*.*

A substitution on a feature graph with a *non-empty* error feature *err* is a bidirectional substitution of the value of a feature f in *err* by a value of f occurring in the graph parallel to *err*.

Given the substitution operation Θ and the ordinary definition of subsumption $\preceq$ it was shown

[2] We assume that finite sets of features $Feat$ and constants Con and an infinite set of variables Var are given.

in (Reuer & Kühnberger 05) that an extension of $\preceq$ to feature structures containing an error feature *err* is possible: If two feature structures $G_\oplus = \langle x_{G_\oplus}, E_{G_\oplus}\rangle$ and $G'_\oplus = \langle x_{G'_\oplus}, E_{G'_\oplus}\rangle$ with error features err_1 and err_2 are given, then: $G_\oplus \preceq G'_\oplus$ iff there exists a mapping $\psi : Var_{G_\oplus} \cup Con_{G_\oplus} \rightarrow Var_{G'_\oplus} \cup Con_{G'_\oplus}$ such that the following properties hold:

- ψ maps $x_{G_\oplus}$ to $x_{G'_\oplus}$
- $\psi(a) = a$ for all constants a
- Every edge $xfs \in E_{G_\oplus}$ is mapped to the edge $\psi(x)f\psi(s) \in E_{G'_\oplus}$
- If it holds $xfs \in E_{G_\oplus}$, $\psi(x)fs' \in E_{G'_\oplus}$, and $\psi(x)f\psi(s) \notin E_{G'_\oplus}$, then there exists a substitution Θ on $G'_\oplus$ such that $\psi(x)f\Theta(\psi(s)) \in E_{G'_\oplus}$ and $\{y_1, \ldots, y_n\} \subseteq \Theta[\{y'_1, \ldots, y'_m\}]$

The extended feature structures can be used to define a subsumption relation which in turn can be used to define the unification process. We define an equivalence relation on feature graphs, collapsing feature graphs that subsume each other.

Definition 4 *Assume two feature graphs $G_\oplus$ and $G'_\oplus$ with error features are given. An* equivalence relation $\sim$ *is defined as follows:*

$$G_\oplus \sim G'_\oplus \quad \Leftrightarrow \quad G_\oplus \preceq G'_\oplus \ \wedge \ G'_\oplus \preceq G_\oplus$$

We denote the equivalence class of a feature graph $G_\oplus$ with $[G_\oplus]_\sim$. We can use Definition 4 to define a modified subsumption relation $\preceq_\sim$:

$$[G_\oplus] \preceq_\sim [G'_\oplus] \quad \Leftrightarrow \quad G'_\oplus \preceq G_\oplus$$

Feature graph algebras $\mathcal{F} = \langle [\mathbf{D}_\mathfrak{B}], I_\mathfrak{B}\rangle$ correspond to the collection of all equivalence classes of feature graphs containing error features. The interpretation $I_\mathfrak{B}$ is induced by the subgraph relation (Reuer & Kühnberger 05).

Fact 5 *A feature graph algebra $\mathcal{F} = \langle [\mathbf{D}_\mathfrak{B}], I_\mathfrak{B}\rangle$ with error feature* err *satisfies:*

(i) The subsumption relation $\preceq_\sim$ is reflexive, antisymmetric, and transitive.

(ii) For two feature graphs $[G_\oplus]$ and $[G'_\oplus]$ the greatest lower bound $[G_\oplus] \sqcap [G'_\oplus]$ exists.

Corollary 6 *$\langle [\mathbf{D}]_\mathfrak{B}, \preceq_\sim\rangle$ is a semilattice.*

The unification of two feature graphs can now be considered as the computation of the greatest lower bound of the two graphs.

Definition 7 *The* unification *of two feature graphs $[G_\oplus]$ and $[G'_\oplus]$ is the greatest lower bound $[G_\oplus] \sqcap [G'_\oplus]$ in the structure $\langle [\mathbf{D}]_\mathfrak{B}, \preceq_\sim\rangle$.*

3 The Unification Algorithm ESU-A

In Section 2, we discussed the logical foundations of an error-sensitive unification approach of feature structures. Unfortunately, the underlying logic of the presented account does not provide a specification of how an algorithm defined on extended feature structures can be realized. We will now specify the algorithm ESU-A (*Error-Sensitive Unification Algorithm*) in order to solve this problem.[3]

Consider Table 1: As input two feature structures G_1 and G_2 with (potentially empty) error features are given. The algorithm computes the unified feature structure $G^\star$ as output. Notice that any pair of feature structures can be unified due to Corollary 6. The unified feature structure $G^\star$ is initialized by the empty feature structure. Roughly speaking, ESU-A selects each feature-value pair from G_1 and searches for a corresponding feature-value pair in G_2 in order to check whether unification is possible. There are essentially three possible cases that can occur for each chosen pair $[f : x] \in G_1$:

- The algorithm can unify $[f : x] \in G_1$ with a pair $[f : y] \in G_2$, i.e. $x = y$.
- The values of the feature-value pairs $[f : x] \in G_1$ and $[f : y] \in G_2$ clash, i.e. $x \neq y$. The error list must be checked and a substitution Θ needs to be applied to G_2 in order to make unification possible.
- There is no corresponding feature-value pair in G_2, i.e. the error must be coded in the error list of the unified feature structure $G^\star$.

In the first case, the algorithm behaves like an ordinary unification algorithm. If there are clashing values (second case), a substitution $\langle \Theta^\wedge, \Theta^\vee\rangle$ must be applied to check whether a unification modulo substitution is possible. If such a pair $[f : y'] \in G_2$ can be found, the algorithm adds $[f : x]$ to $G^\star$ and $[f : y']$ to the error list of $G^\star$. Hence, no information gets lost. If no corresponding feature-value pair exists in the error list of G_2, the algorithm adds $[f : x]$ to $G^\star$ and $[f : y]$ to the error list of $G^\star$. In each case, corresponding pairs added to $G^\star$ are deleted in G_1 or G_2, respectively. In the third possible case, no $[f : y] \in G_2$ exists. Then the feature-value pair $[f : x]$ is added to $G^\star$. Finally, the error list $err^\star$ of $G^\star$ must be updated

[3] ESU-A is implemented in the ICALL system PromisD (Reuer 05).

Input: A pair of feature structures G_1 and G_2 with (potentially empty) error lists $err_1 \in G_1$ and $err_2 \in G_2$ for given sets of features $Feat$ and constants Con
Output: A representative $G^\star$ of the equivalence class $[G^\star]$ of unified feature structures with (potentially empty) error list $err^\star \in G^\star$

G_1, G_2 two input feature structures
$G^\star$ = empty feature structure
FOR each feature-value pair $[f : x] \in G_1$
  SELECT $[f : y] \in G_2$
    IF $x = y$
      ADD $[f : x]$ to $G^\star$
      $G_1 = G_1 \setminus [f : x]$
      $G_2 = G_2 \setminus [f : y]$
    ELSE FAIL
    IF $x \neq y$
      IF $x, y \in Con$
        SELECT $[f : y'] \in err_2$
          IF $x = y'$
            $G_2 = \langle \Theta^\wedge, \Theta^\vee \rangle G_2$
            ADD $[f : x]$ to $G^\star$
            ADD $[f : y']$ to $err^\star$
            $G_1 = G_1 \setminus [f : x]$
            $err_2 = err_2 \setminus [f : y']$
          ELSE ADD $[f : x]$ to $G^\star$
          $G_1 = G_1 \setminus [f : x]$
        END SELECT
        ADD $[f : y]$ to $err^\star$
        $G_2 = G_2 \setminus [f : y]$
      ELSE FAIL
    ELSE $UNIFY([x], [y])$
    IF no $[f : y]$ exists
      ADD $[f : x]$ to $G^\star$
      $G_1 = G_1 \setminus [f : x]$
    ELSE FAIL
  END SELECT
  $err^\star = err^\star \cup err_1$
END FOR
FOR $[f : y] \in G_2$
  ADD $[f : y]$ to $G^\star$
END FOR
$err^\star = err^\star \cup err_2$

Table 1: The unification algorithm ESU-A on feature structures with error features.

using the error list of G_1 by a simple union operation on the error lists. Remaining pairs in G_2 are added to $G^\star$ as well as the error list of G_2, ensuring that no information is lost in the unification process. Notice that the algorithm needs to be defined recursively on the feature structures.

4 Some Properties of ESU-A

4.1 Some Properties of ESU-A

The algorithm ESU-A is practically realized using an Earley-type parsing algorithm. The particular choice of the parsing algorithm does not play an important role: ESU-A can be integrated in any type of parsing algorithm that can be used for LFG-style representations. It is obvious that ESU-A models the result of applying the infimum operation in Definition 7 (modulo the equivalence classes of feature structures).[4]

Fact 8 *Assume the structure* $\langle [\mathbf{D}]_{\mathfrak{B}}, \preceq_\sim \rangle$ *is given. Then it holds:* $\forall [G_1] \in [\mathbf{D}]_{\mathfrak{B}} \forall [G_2] \in [\mathbf{D}]_{\mathfrak{B}} : [G_1] \sqcap [G_2] = [G^\star]$ *if and only if ESU-A is applied to* G_1 *and* G_2 *and yields* $G^\star$ *as output.*

It is well-known that the classical subumption preorder on feature structures relative to a given feature algebra $\langle D, I \rangle$ is linear-time decidable (Smolka 92). Concerning ESU-A we have a slightly more complicated situation, because of the additional substitutions and error lists. Nevertheless applying ESU-A is tractable:

Fact 9 *ESU-A has polynomial time complexity w.r.t. the number of features occurring in the input* G_1 *and* G_2 *and the depths of* G_1 *and* G_2*.*

Proof: Assume two feature structures G_1 and G_2 with (potentially empty) error lists are given. Assume further that $n = |Feat_{G_1}|$ denotes the number of features occurring in G_1, $m = |Feat_{G_2}|$ denotes the number of features occurring in G_2, d_1 denotes the depth of G_1, and d_2 denotes the depth of G_2. An upper bound of the time complexity of unifying G_1 and G_2 is $d_1 \cdot n(d_2 \cdot m + |err_2|)$. Assume $b = max\{n, m, d_1, d_2, |err_2|\}$. Then we get the following upper bound w.r.t. time complexity: $b^2(b \cdot b + b) = b^4 + b^3 = O(b^4)$. Hence the time complexity is at most polynomial. q.e.d.

Although ESU-A is defined on two given feature structures G_1 and G_2, it is straightforward to extend the unification process to finitely many features structures that should be unified.

4.2 Applying the Algorithm to a Corpus

In order to assess the value of the proposed mechanism, the annotated Heringer-Corpus (Heringer 95) containing 7107 sentences with errors produced by learners of German as a foreign language was analyzed with respect to frequently occurring error types. The analysis of error types revealed that around 35% of all errors should be dealt with inside the feature structure in a LFG-type analysis. In an evaluation, 75 sentences with morphosyntactic errors were randomly chosen from the Heringer-Corpus and another 75 were collected from trials with the mentioned ICALL-system. These sentences were manually error-tagged and contained 96 errors.

[4] Notice the usage of equivalence classes on the logical side and elements of equivalence classes on the algorithmic side.

Error-Type	Heringer + ICALL	Total
Agreement/Government		
with/in Subject	11 / 3 / 4 / –	18
with/in Object	17 / 6 / 3 / 1	27
with/in Prep.-Object	20 / 1 / 1 / 3	25
POS-error	– / – / 2 / 1	3
Verb form	11 / – / 2 / –	13
Auxiliar	9 / – / – / 1	10
Total	68 / 10 / 12 / 6	96
%	71 / 10 / 13 / 6	100

Table 2: Evaluation of errors: The first column represents correctly identified errors, the second column represents correctly identified errors among others, the third column shows false errors, and the fourth column represents no analysis. No false positives occurred.

As Table 2 shows, 68 errors were correctly identified by the proposed mechanism compared to the manual tagging with a very simple preference scheme preferring the result with less clashes. 10 errors were identified among other non preferred ones with an identical error measure. For 13 errors the resulting f-structure with the least clashes did not show the expected error and for 6 errors the analysis failed because of either an arbitrary threshold of 20,000 edges in the chart or because a final edge could not be generated. As the grammar was designed to cover the correct versions of the sentences following (Foster 04), no false positive occurred. In summary 71% of the errors were identified correctly and another 10% were identified among others which shows the value of our approach.

5 Conclusion and Future Research

In this paper, we presented an algorithmic approach to model detect and trace errors in an ICALL system. The logical basis for this theory is an extension of classical feature logic formally developed in (Reuer & Kühnberger 05). This paper adds the specification of the underlying algorithm and some of its formal properties. The algorithm implements a sensitive parsing procedure by coding and tracing errors explicitly without assuming knowledge about possible errors, making ESU-A different from ordinary unification algorithms.

Future research will be directed towards a careful evaluation of the presented algorithm in a larger application scenario. Additionally possible optimizations of the algorithm ESU-A needs to be examined. Finally an extension with additional properties (like structure sharing) will be implemented.

References

(Bresnan 01) J. Bresnan. *Lexical-Functional Grammar.* Oxford: Blackwell Publisher, 2001.

(Carpenter 93) B. Carpenter. *Skeptical and Credulous Default Unification with Applications to Templates and Inheritance.* In: Briscoe, T., Copestake, A., Paiva, V. (eds.): Inheritance, Defaults and the Lexicon, pp. 13–37, Cambridge University Press, Cambridge, 1993.

(Foster 04) J. Foster. *Parsing Ungrammatical Input: An Evaluation Procedure.* In: Proceedings of LREC 2004, pp. 2039–2042, Lisbon, 2004.

(Fouvry 03) F. Fouvry. *Constraint relaxation with weighted feature structures.* Papers from IWPT2003, 8th International Workshop of Parsing Technologies, Nancy, 2003.

(Heringer 95) H. J. Heringer. *Aus Fehlern lernen.* Augsburg – CD-ROM for Win9x/NT, 1995.

(Jensen et al. 83) K. Jensen, G.E. Heidorn, L.A. Miller and Y. Ravin. *Parse Fitting and Prose Fixing: Getting Hold on Illformdness.* In: Computational Linguistics 9(3-4), pp. 147–160, 1983.

(Menzel 92) W. Menzel. *Modellbasierte Fehlerdiagnose in Sprachlehrsystemen.* Niemeyer, Tübingen, 1992.

(Pollard & Sag 94) C. Pollard, I. Sag. *Head-Driven Phrase Structure Grammar.* University of Chicago Press, Chicago, 1994.

(Reuer 05) V. Reuer. *PromisD – Ein Analyseverfahren zur antizipationsfreien Erkennung und Erklärung von grammatischen Fehlern in Sprachlehrsystemen,* Library of the Humboldt University Berlin, http://edoc.hu-berlin.de/browsing/dissertationen, 2005.

(Reuer & Kühnberger 05) V. Reuer, K.-U. Kühnberger. *Feature Constraint Logic and Error Detection in ICALL Systems.* In: P. Blache & E. Stabler (eds.): Proceedings of the 5th International Conference on the Logical Aspects of Computational Linguistics (LACL 2005), LNAI 3492, pp. 255–270, Springer, Berlin Heidelberg, 2005.

(Sågvall Hein 98) A. Sågvall Hein. *A Grammar Checking Module for Swedish.* Uppsala University, Uppsala, 1998 – SCARRIE Deliverable 6.6.3.

(Schwind 88) C. Schwind. *Sensitive Parsing: Error Analysis and Explanation in an Intelligent Language Tutoring System.* In: Proceedings of the 12th International Conference on Computational Linguistics, 1988, pp. 608–613.

(Smolka 92) G. Smolka. *Feature Constraint Logics for Unification Grammars.* In: Journal of Logic Programming 12, pp. 51–87, 1992.

(Vandeventer Faltin 03) A. Vandeventer Faltin. *Syntactic Error Diagnosis in the context of Computer Assisted Language Learning.* Dissertation, University of Geneva.

(Vogel & Cooper 95) C. Vogel, R. Cooper. *Robust Chart Parsing with Mildly Inconsistent Feature Structures.* In: Schöter, A., Vogel, C. (eds.): Nonclassical Feature Systems Vol. 10, pp. 197–216, Edinburgh University, 1995.

Analysis and Visualization for Daily Newspaper Corpora

Matthias Richter
Leipzig University
Computer Science Institute, NLP Dept.
Augustusplatz 11, 04109 Leipzig, Germany
mrichter@informatik.uni-leipzig.de

Abstract

This paper describes an application of statistical NLP for the extraction of significant topics from time-dependent data. Text from daily national newspapers is taken for an example. Based on comparison with a large reference corpus a small number of terms is selected, categorized and clustered in order to describe characteristic topics from the analyzed texts by these terms. Statistical word co-occurrences are then used extensively to visualize the course of events. The result can be regarded as an enriched type of electronic press report and archive.

1 Introduction

> "Whatever we know about society, or indeed about the world in which we live, we know through the mass media." (Luhmann 00, p. 1)

Monitoring and analyzing mass media requires a lot of manpower. As of today, in qualitative analysis human judgment cannot be fully replaced by computers. Text Mining, however, can relieve the analyst from tedious acts and provide a set of tools allowing well founded judgments more efficiently. Moreover, means of statistical analysis combined with visualization may draw an analyst's attention to latent relations that might never have been uncovered otherwise.

The application described in this paper is used to analyze texts from newspapers on a daily basis. Comparison with a reference corpus is used to extract terms which are likely to be suitable for a very short description of the main events reported on in the texts. The extracted terms are grouped into very generic categories providing a rough overview at a first glance. Visualizations detailing information on statistical word co-occurrences and frequencies in the course of time can be accessed in a second step.

This paper gives an insight into some relevant influences in Section 2. The data basis will be described in Section 3 as will be in Section 4 the techniques currently used for analysis. In Section 5 the current means of visualization will be explained and examples for their usefulness will be given. After discussing further work in Section 6 conclusions will be drawn in Section 7.

2 Related Work

Diachronic corpora are usually built to reflect long-term changes in the use of language. For example (Ahmad & Musacchio 04) are using slices as wide as a quarter of a century to reconstruct the evolution of the language of nuclear physics in Italy. (Lüdeling *et al.* 04) describe the plan for a diachronic corpus of German from 800 AD through today. In this paper the focus is not on the change of language but on the change of topics in daily life driven by the events the press is reporting on. Thus for the analysis slices should be as thin as possible in order to be able to represent the sequence of events properly. It is, on the other hand, necessary to use a large-enough amount of input for statistical methods.

In *topic detection* and *topic tracking*, as for example summarized in the TDT Evaluation plan[1], the focus is on assigning stories to topics directly covered and a topic being defined as "... a seminal event or activity, along with all directly related events and activities." The present application needs to fulfill a similar task but has got a different interest of research: The focus is not on topics and their stories but on sequences and connections of events. This also makes up for a main difference to news aggregation sites such as Google News.

Certain aspects of *news value*, a term coined by (Galtung & Ruge 65) and enhanced by (Schulz 76), such as frequency, unambiguity, surprise etc. can be modeled by a set of statistics. This is not limited to counting words, but instead can be an application for arbitrarily complex models and algorithms for the mining of *multivariate time se-*

[1] http://www.nist.gov/speech/tests/tdt/tdt2004/TDT04.Eval.Plan.v1.2.pdf

ries data. A selection of these can be found in (Yang *et al.* 01). The current application does not yet take such sophisticated methods into account. During further development many of these techniques can be applied to the large time series database.

3 Data Basis

The data basis consists of two parts: A *comprehensive reference corpus* of German and a *daily collection* of news obtained from national media.

The first is described in (Biemann *et al.* 04) and consists of half a billion of running words in 35 million sentences originating from newspaper-texts of the 1990s and later. The data is processed by algorithms which will be briefly characterized in Section 4 and stored in a MySQL-database. It can be accessed by the public under the URI `http://wortschatz.uni-leipzig.de`[2] and several other sources. The corpus itself is not error-free but comprehensive. It is neither balanced according to a balancing standard, as for example the BNC is, but in (Richter 04) a comparison based on the lemma lists from German monolingual dictionaries has supported the assumption that it covers the current use of German language well enough to be a very useful resource even for the needs of lexicography. Nor is the corpus lemmatized, instead syntactical, semantical and pragmatical information on words is annotated and can can be accessed easily and quickly.

The daily data is retrieved from several online newspapers at portions of 10 000 to 20 000 sentences with approximately 250 000 running words per day. The same process that has been applied to the large corpus is applied to the daily data in a first step. Storage and processing works strictly on the basis of selected sentences. This breaks anaphoras etc. but is –as legal advice has shown– a viable way of meeting the requirements of German copyright. Links to the sources of the texts are kept and stored for later use and analysis.

4 Analysis

There are three steps of analysis applied to the data.

[2]For summer 2005 a replacement with 800 million running words is scheduled.

4.1 Statistical Text Analysis

The *first step* is described in (Biemann *et al.* 04) in more detail: Text is segmented into sentences and an index is built for the words and a supplied list of word groups. Using different window-sizes two types of co-occurrences are calculated: direct neighbors and co-occurrences in the sentence. The algorithm is based on the statistical *G-test* for Poisson distributions[3]. Let A and B be words with frequencies of occurrence a and b and k their number of co-occurrences, n the number of sentences in the corpus and $x = \frac{ab}{n}$. Then the value of the significance measure can be calculated as in Formula 1 with a threshold $sig(A, B) \geq 5$ in order for A and B to be significant co-occurrences:

$$sig(A, B) = x - k \log x + \log k! \qquad (1)$$

4.2 Selection of Significant Terms

The *second step* operates on two corpora: one *large reference corpus* and the *current daily news corpus*, with the former typically being at least several hundred times larger than the latter. The goal is to select a small number of terms being capable of representing as much information about what the most significant distinctions in the daily corpus are. The selection is based on a set of simple assumptions:

1. Nouns, especially noun-phrases, contain a high amount of information per unit[4] and are used as index terms consequently.

2. Very infrequent words from the reference corpus are likely to be unknown to the public, therefore a lower frequency boundary is assumed.

3. Very frequent words from the reference corpus are likely to be too generic for being a good index term, therefore they are omitted as stop words.

4. Items seen several times but only in the daily corpus are likely to carry important information if the occurrences stem from different articles.

[3]According to (Manning & Schütze 99) using mutual information would not be an appropriate choice here. (cf. p. 182)

[4]See (Witschel 05) for an evaluation of methods for terminology extraction taking nouns as the baseline.

5. Co-occurrences of less frequent terms found in the reference corpus carry information about common surroundings of terms.

6. A news report for a period of one day should contain a reasonably small number of items describing the main events of the day.

An average of 130 terms per day are proposed as candidates by these rules.

4.3 Presentation

In the *third step* applying an exclusion word list based on inflection and a small set of well known errors typically reduces the number of candidates to an average of 81 terms. These terms are being classified into a very generic set of subject areas such as POLITICS or SPORTS semi-automatically. They are presented on an overview-page which is aimed at the occasional reader. It is available on the web under the URI `http://wortschatz.uni-leipzig.de/wort-des-tages`. For each term a page is generated containing example sentences and visualizations as depicted in Figures 1 and 2, which will be described later in Section 5.

The simplicistic approach of the current implementation leads to certain common errors. Some of them have already been detected in (Quasthoff *et al.* 03) and will be addressed by a completely rewritten version in the near future:

1. Categorization into subject areas fails if a term is ambiguous.

2. Typically there is a delay of one to two days between the time an event taking place and its upcoming in the corpus.

3. The number of of papers examined is rather small.

4. More than one term can represent the same entity due to inflection.

5. Some aspects of self-references of the media system lead to a bias of the selection overrating persons, places and organizations connected to media.

6. Longer-term changes not yet being reflected properly in the corpus used as reference tend to select a small number of words, e.g. the relatively new European currency Euro, as candidate each day in error.

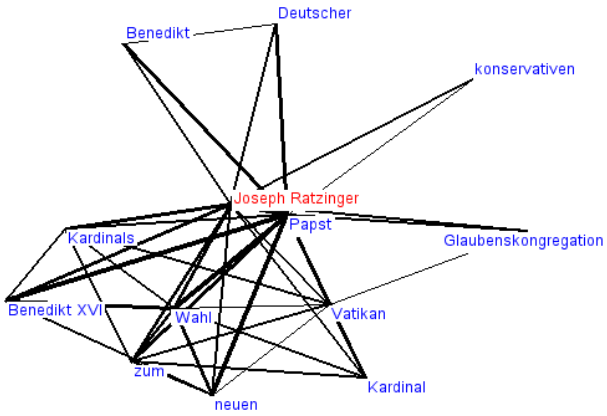


Figure 1: Fully automatic selection of co-occurrences for JOSEPH RATZINGER on April 21st, 2005.

5 Visualization

During the text analysis step three sets of data are calculated for each corpus: frequencies, significant neighbors and significant co-occurrences in the sentence. The data from the significant neighbors are only used to propose candidates for multi-word units. The frequency and the co-occurrences data sets are used for visualization. Currently three types of visualizations can be generated: a graph showing co-occurrences of a word on a specific day (see Section 5.1), a graph of frequencies in the course of time (see Section 5.2) and a graph of co-occurrences in the curse of time (see Section 5.3).

5.1 Co-occurrences for a word

Given an input term a set of interconnected co-occurrences for this term is used to select nodes and edges for a graph that is then laid out nicely using simulated annealing as described in (Davidson & Harel 96) and implemented in (Schmidt 99). This process is fully automatic. Figure 1 depicts an example of such a graph taking the prior name of the newly elected Pope as input and presenting the German origin, former occupation as prefect of the Congregation for the Doctrine of the Faith and new status as Pope Benedict XVI in the result. The data basis is the daily corpus from April 21st, 2005, i.e. the corpus with the texts covering the election in it. It is obvious that such a graphical representation is capable of reducing information to a small number key facts, thus helping the user to rapidly gain an overview.

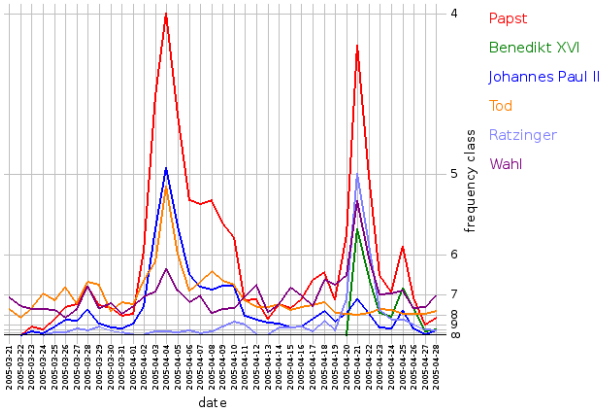


Figure 2: Logarithmic scaled frequencies of a selection of co-occurrences for PAPST (*pope*) for the period March 21st, 2005 through April 28th, 2005.

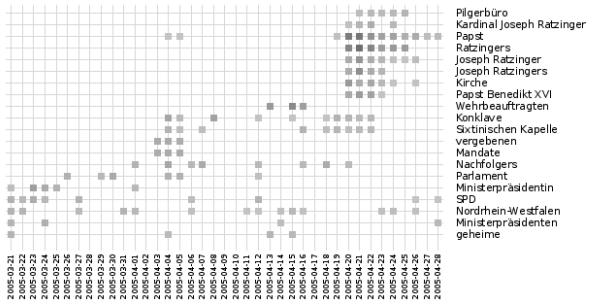


Figure 3: Fully automatic selection of co-occurrences for WAHL (*election*) for the period March 21st, 2005 through April 28th, 2005. The gray-value encodes the value of the significance measure for the co-occurrence of WAHL and *word* y on *day* x.

5.2 Frequencies in the course of time

Plotting frequencies of words in a graph is a task not worthwhile mentioning. Two features make the graphs such as the one depicted in Figure 2 specific: The ordinate of the graph shows the relative frequency of the terms and is scaled logarithmically[5], making up for the different sizes of the daily corpora. Secondly, for a given input term an automatic selection of other terms is made based on co-occurrences. The joint peaks in Figure 2 visualize the media-impact of the events DEATH OF POPE JOHN PAUL II. and ELECTION OF POPE BENEDICT XVI.

5.3 Co-occurrences in the course of time

Visualization of co-occurrences over the course of time can be done in different ways. For the sample graph depicted in Figure 3 several thresholds were applied, putting constraints on the selected terms in order to represent a certain degree of steady association and to exclude very common terms. A term has to be a co-occurrence of the input term with a minimum frequency f and a minimum significance s at least in n of the daily corpora. Upper limits for frequency and significance of co-occurrence are set, too. The result set is ordered from bottom to top by the first time a term fulfills the required value for s, thus grouping together events chronologically.

In the example for the input term WAHL (*election*) several events being connected to it can be seen in the period researched: One major event came up when Pope John Paul II. died and again when Pope Benedict XVI. was elected. A smaller event was the election of the new Wehrbeauftragter (*delegate for the army*) in the Bundestag (*German parliament*). Another major event were incidents preceding the election in the federal state of North Rhine-Westphalia.

Speaking in terms of complexity and speed of progression it seems likely that the parameters f, s and n are related to the extent of the events covered as well as to the resolution the user chooses to look at them. Therefore it does not seem too wise to calculate a set of *the best* values for the parameters. Instead moderately restrictive values are chosen in the beginning and further user interaction is suggested.

6 Further Work

The upcoming version of the application will address the issues already pointed out in Section 4.2:

1. Tools for disambiguation are available, e.g. as described in (Bordag 03) and will be integrated into the application.

2. Delay cannot be eliminated in general, but the base of sources is being broadened such that the amount of delay is reduced.

3. The list of sources is already about being broadened very extensively. Thus shortcomings in coverage and representativeness will be made up for.

[5]The *frequency classes* used for the labeling allow the comparison of (relative) frequencies of words in different corpora easily. The number is the rounded ratio of the logarithm (base 2) of the most frequent word *der* and the term.

4. Summarization of different word forms under the lemma has already been tried successfully and will be integrated.

5. Self-reference of the media can be compensated by special treatment for the usual or can be subsumed under the following point.

6. Permanent changes which are not reflected in the reference corpus can be compensated by using a *monitor corpus* made from a sensible number of preceding day's corpora.

Much further effort will be put into visualization. Techniques of data mining and time series analysis will be applied according to their aptitude for the specific task of supplying *interesting data*. A graphical representation roughly similar to the one in Figure 1 but being interactive and showing the actual flow of events can be integrated. Step by step the different visualizations are meant to be integrated into a tool journalists, social scientists, marketing and public relations etc. can use in their recherche or research. The current implementation has already been considered useful by such people and their feedback has got significant impact on the development of more mature versions.

Implementations for different languages are planned as the underlying techniques are mostly language-independent[6]. Comparing different languages' results can be of great value for empirical studies as research about varying ways of dealing with certain topics and aspects can be connected to reproducible facts – as required by, e.g. (Popper 72).

7 Conclusions

In this paper, a prototype of a tool-set for the chronological analysis of texts was introduced that uses corpora built from daily news sources and extracts relevant terms in order to represent the key events. Statistical word co-occurrences are used to group together related information which is then visualized in different ways focusing on clusters of terms and their changes in the course of time.

[6] As a rule of thumb, languages that can be segmented into *sentences* and *words* yield good results. It has already been tried (but needs yet to be published) to process arbitrarily chosen segmentations of textual data representations, such as genome sequences, with results rather similar in structure but less evident in meaning.

Some shortcomings of the current implementation have been addressed and reasonable solutions have been proposed. The application aims at subsequent use by media analysts and journalists, thus fulfilling their needs and requirements will be a crucial point. Further development is funded by a grant from the "Medienstiftung der Sparkasse Leipzig".

References

(Ahmad & Musacchio 04) Khurshid Ahmad and Maria Teresa Musacchio. Discovery of (New) Knowledge and the Analysis of Text Corpora. In M.T. Lino, editor, *Proc. of the 4th Int. Conf. on Language Resources and Evaluation*, volume V, pages 749–752, 2004.

(Biemann *et al.* 04) Christian Biemann, Stefan Bordag, Gerhard Heyer, Uwe Quasthoff, and Christian Wolff. Language-independent Methods for Compiling Monolingual Lexical Data. In *Proceedings of CicLING 2004*, volume 2945 of *Lecture Notes in Computer Science*, pages 215–228, Seoul, South Korea, 2004. Springer.

(Bordag 03) Stefan Bordag. Sentence Co-occurrences as Small-Woirld-Graphs: A solution to Automatic Lexical disambiguation. In Alexander F. Gelbukh, editor, *CicLING 2003*, volume 2588 of *Lecture Notes in Computer Science*, pages 329–333. Springer, 2003.

(Davidson & Harel 96) Ron Davidson and David Harel. Drawing graphs nicely using simulated annealing. *ACM Transactions on Graphics*, 15(4):301–331, 1996.

(Galtung & Ruge 65) Johan Galtung and Mari Holmboe Ruge. The Structure of Foreign News. The Presentation of the Congo, Cuba and Cyprus Crises in Four Foreign Newspapers. *Journal of Peace Research*, 2:64–91, 1965.

(Lüdeling *et al.* 04) Anke Lüdeling, Thorwald Poschenrieder, and Lukas Faulstich. DeutschDiachronDigital - ein diachrones Korpus des Deutschen. In Georg Braungart, Karl Eibl, and Fotis Jannidis, editors, *Jahrbuch für Computerphilologie (in print)*. Mentis, Paderborn, Germany, 2004.

(Luhmann 00) Niklas Luhmann. *The Reality of the Mass Media*. Polity Press, Cambridge, 2000.

(Manning & Schütze 99) Christopher Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

(Popper 72) Karl R. Popper. *Objective Knowledge*. The Clarendon Press, Oxford (UK), 1972.

(Quasthoff *et al.* 03) Uwe Quasthoff, Matthias Richter, and Christian Wolff. Medienanalyse und Visualisierung: Auswertung von Online-Pressetexten durch Text Mining. In Uta Seewald-Heeg, editor, *Sprachtechnologie für die multilinguale Kommunikation - Textproduktion, Recherche, Übersetzung, Lokalisierung - Beiträge der GLDV-Frühjahrstagung 2003*, volume 5 of *Sprachwissenschaft, Computerlinguistik, Neue Medien*, pages 442–459, 2003.

(Richter 04) Matthias Richter. Korpusbasierte Lemmaselektion. Unterstützung der Erstellung von Nachschlagewerken mit Mitteln der Automatischen Sprachverarbeitung. Magisterarbeit, Leipzig University, http://wortschatz.uni-leipzig.de/%7Emrichter/MR-Magisterarbeit.pdf, 2004.

(Schmidt 99) Fabian Schmidt. *Automatische Ermittlung semantischer Zusammenhänge lexikalischer Einheiten und deren graphische Darstellung*. Diplomarbeit, Leipzig University, http://lips.informatik.uni-leipzig.de/pub/1999-18, 1999.

(Schulz 76) Winfried Schulz. *Die Konstruktion von Realität in den Nachrichtenmedien*. Alber, Freiburg i. Br., Germany, 1976.

(Witschel 05) Hans Friedrich Witschel. Terminology Extraction and Automatic Indexing.Comparison and Qualitative Evaluation of Methods. In *Procedings of Terminology in Knowledge Engineering 2005*, 2005. to appear.

(Yang *et al.* 01) Jiong Yang, Wei Wang, and Philip S. Yu. Infominer: mining surprising periodic patterns. In *Knowledge Discovery and Data Mining*, pages 395–400, 2001.

Even very frequent function words do not distribute homogeneously

Anne De Roeck[1], Avik Sarkar[1], Paul H Garthwaite[2]
[1] Department of Computing, [2] Department of Statistics
The Open University
Milton Keynes, MK7 6AA, UK
{a.deroeck, a.sarkar, p.h.garthwaite}@open.ac.uk

Abstract

We have known for some time that content words have "bursty" distributions in text (eg Church 00). In contrast, much of the literature assumes that function words are uninformative because they distribute homogeneously (eg Katz 96). In this paper based on two sets of experiments, we show that assumptions of homogeneity do not hold, even for the distribution of extremely frequent function words. In the first experiment, we investigate the behaviour of very frequent function words in the TIPSTER collection by postulating a "homogeneity assumption", which we then defeat in a series of experiments based on the χ^2 test. Results show that it is statistically unreasonable to assume homogeneous term distributions within a corpus. We also found that document collections are not neutral with respect to the property of homogeneity, even for very frequent function words. In the second set of experiment, we model the gaps between successive occurrences of a particular term using a mixture of exponential distributions. Based on the "homogeneity assumption" these gaps should be uniformly distributed across the entire corpus. But, using the model we demonstrate that gaps are not uniformly distributed, and even very frequent terms do occur in bursts. Since the homogeneity assumption was defeated resoundingly for diverse collections, we propose that these homogeneity measures and the re-occurrence model are suitable candidates for corpus profiling.

1 Introduction

Some areas of statistical Natural Language Processing (NLP), and Information Retrieval (IR) adopt the "bag of words" model for text - i.e. they assume that terms in a document occur independently of each other. In spite of numerous drawbacks (Franz 97), this model has been used extensively, largely because it makes the application of standard mathematical and statistical techniques very convenient. At the same time, it is widely accepted that the term independence assumption is wrong, and that words do not occur independently of each other.

The actual extent to which the occurrence of terms depends on other terms is relatively unexploited. There is a growing literature which investigates term dependency between content words. For instance, Church (00) describes "burstiness" in the distribution of content words in documents - i.e. the fact that repeated occurrences of an informative word in a document tend to cluster together. In contrast, the distribution patterns of function words have received less attention. Typically, function words are assumed to distribute evenly throughout text. Katz (96), for instance, develops a model for bursty distributions of "concept" terms, and distinguishes these from function words on the basis that function words are distributed homogeneously.

This view of function words as general background noise is consistent with their removal through stop lists or frequency thresholds in many applications. More sophisticated approaches, however, show that stop word removal based on collection specific distribution patterns leads to improved performance in text categorization (Wilbur & Sirotkin 92; Yang & Wilbur 96). This constitutes some evidence that function words perhaps do not distribute quite that homogeneously throughout all text.

In short, the statistical NLP and IR literatures sustain a "homogeneity assumption" in two respects. First, it is adopted as a consequence of the "bag of words" model. Term independence is related to homogeneity in term distribution: terms that occur independently (randomly) distribute homogeneously. We know this is not the case for content words (Church 00). Second, the assumption is adopted indirectly in the treatment of function words, which are seen as uninformative precisely because they are taken to distribute homogeneously. This is the assumption this paper aims to contest.

2 Aims and Methods

In this paper, we aim to show that the homogeneity assumption does not generally hold, not just for content words, but also for the distribution of

very frequent function words. These results are significant in their own right because they demonstrate that it is statistically unreasonable to assume that function word distribution within a corpus is homogeneous. In addition, we show that data-sets and document collections display different homogeneity characteristics in the distribution of very frequent function words. The homogeneity assumption is defeated substantially for collections known to contain similar documents, and even more drastically for diverse collections.

We devise two sets of experiments to test this homogeneity assumption using the TIPSTER collection. In the first method, we start by postulating the homogeneity assumption: that very frequent function words distribute homogeneously in corpus text. These experiments extend an approach first introduced in the corpus literature, which casts homogeneity as a similarity measure of term frequency distributions between two halves of a collection (Kilgarriff 97) and which uses the χ^2 statistic as a homogeneity measure. In contrast, we use the χ^2 test (including the p-value) to relate a notion of homogeneity to a level of statistical significance. We also explore different ways of partitioning the datasets and measuring homogeneity.

In the second set of experiments, we study the gaps between successive occurrences of some very frequent function words. Here we examine two alternatives for modeling these gaps using exponential distribution. The first alternative is based on the "bag of words" assumption that very frequent terms are uniformly distributed and the gaps between successive occurrences of a particular term is generated from a single exponential distribution. This is in contrast to the second alternative, which assumes that terms occur in bursts and the gaps between successive occurrences of a term is generated from a mixture of two exponential distributions, the one with the larger mean reflecting the rate of occurrence of the term in the corpus and the one with the smaller mean reflecting the rate of re-occurrence after it has occurred recently (Sarkar *et al.* 05). Very frequent function words are believed to be distributed homogeneously and we investigate this belief based on the exponential mixture model.

3 Experimental Framework

3.1 χ^2 based Homogeneity

We adopted the methodology outlined in Kilgarriff (97) for measuring homogeneity in a corpus by measures of similarity. Specifically, he casts homogeneity as internal similarity of distributions, between two halves of a document collection as measured by the χ^2 statistic. His basic method involves the following steps:

(1) Split the corpus into two halves by randomly placing text in one of two sub-corpora.

(2) Produce a word frequency list for each half.

(3) Calculate the χ^2 statistic for the difference in term frequency distributions between the two sub-corpora.

(4) Normalize for corpus length.

We adopted this methodology as it is found to perform well in comparative experiments (Rose & Haddock 97; Cavagli 02) as long as certain conditions are met (Dunning 93). However, our aim of investigating homogeneity in frequent term distribution requires a more fine-grained tool than simple use of the χ^2 statistic as a measure. In this context, we clarify the relationship between the χ^2 test and the χ^2 statistic.

The χ^2 test is a standard method to test the null-hypothesis that two or more samples are homogeneous, i.e. that they are drawn from the same population at random. The χ^2 test is associated with three values[1]. It calculates the χ^2 statistic (first output) by the following formula:

$$\chi^2 = \sum \frac{(O-E)^2}{E}$$

where it tests the difference between expected (E) and observed (O) occurrences of events. It is calculated with (N-1) degrees of freedom (the second output), where N is the number of terms under consideration. The χ^2 statistic may be viewed as a similarity measure between corpora, provided the degrees of freedom (N-1) is kept constant. Where this is not the case, the statistic can be modified by dividing the value of the χ^2 statistic

[1]Experiments were conducted using the SPlus software under Linux

by the degrees of freedom (N-1). This measure (Chi-square By Degrees of Freedom, or CBDF) is the homogeneity measure used by Kilgarriff (97) and others (Rayson & Garside 00; Rose & Haddock 97). The third output is the p-value, a measure of whether the difference between the two samples is statistically significant. The p-value is a probability (value between 0 and 1) where a value close to 0 indicates that, based on sample size, the null hypothesis of similarity between two samples should be rejected.

In our experiments, we will differentiate results by showing not just the CBDF, but also the p-value. Given a null hypothesis (in our case, homogeneity), the p-value allows us to estimate the strength of the evidence offered by the data. A p-value < 0.1 is usually interpreted as constituting weak evidence against the hypothesis, a p-value < 0.01 as strong evidence against, and p-value < 0.001 as very strong evidence against the hypothesis. Normally, a p-value < 0.05 is considered significant - in our case it will mean that non-homogeneity is statistically significant. The CBDF relates to the text and indicates the level of heterogeneity. In the case of perfect similarity between the samples (i.e. in terms of homogeneity), one would expect the observed and expected occurrences to be close: a lower CBDF indicates greater similarity.

Compared to earlier work, our approach has some desirable properties. By reporting both the p-value and the CBDF, even a small departure from homogeneity can be detected if a sample's size is large enough. As the sample size increases, the p-value will get closer and closer to 0. CBDF provides a measure of homogeneity that is not affected greatly by sample size, so that corpora of different lengths can be compared. However, the similarity measure should be compatible with the test of homogeneity, so that if two corpora are of similar size, the one with the larger value on the similarity scale should also have the smaller p-value for the test of homogeneity. This is the case here.

Different partitions in a document set may affect the outcome of similarity based experiments. For instance, assigning one-word chunks to random halves would inject a high degree of randomness in the data and destroy all evidence of term dependence. In that case, we would expect our experiments to be unable to defeat the homogeneity assumption . On the other hand, repetition of Kilgarriff (97) and others (dissolving document boundaries and placing successive chunks of 5000 words in each partition) found resounding evidence of heterogeneity between the distributions. This leads to the following questions.
First, do very frequent function words distribute homogeneously across document boundaries?
Second, do very frequent function words distribute homogeneously throughout the same document?
We try to answer each of these questions by partitioning the collection in different ways (De Roeck *et al.* 04).
(1) Choose a document and assign it at random to either of two partitions (the **docDiv** experiment).
(2) Split each document in the middle, and randomly assign one half to either of the partitions, and the other half to the other partition (the **halfdocDiv** experiment).

3.2 Modeling gaps

The gaps between successive occurrences of a term is modeled based on a mixture of exponential distributions (Sarkar *et al.* 05). The model assumes that the term occurs at some low underlying base rate $1/\lambda_1$ but, after the term has occurred, then the probability of it occurring soon afterwards is increased to some higher rate $1/\lambda_2$. Specifically, the rate of re-occurrence is modeled by a mixture of two exponential distributions. Each of the exponential components is described as follows:

- The exponential component with larger mean (average), $1/\lambda_1$, determines the rate with which the particular term will occur if it has not occurred before or it has not occurred recently.

- The second component with smaller mean (average), $1/\lambda_2$, determines the rate of re-occurrence in a document or text chunk given that it has already occurred recently. This component captures the bursty nature of the term in the text (or document).

The mixture model for a gap x is described as follows:

$$\phi(x) \quad = \quad p\lambda_1 e^{-\lambda_1 x} + (1-p)\lambda_2 e^{-\lambda_2 x}$$

where p and $(1-p)$ denote, respectively, the probabilities of membership for the first and the second exponential distribution.

Now, if the "bag of words" homogeneity assumption is correct, then the above mixture model will be over-parameterized, as the gaps will be generated from a single exponential distribution. Then one of the following conditions must hold so as to dissolve one of the mixture components and end up with a single exponential distribution. These conditions are:

- $p = 0$ or $p = 1$
- $\lambda_1 = \lambda_2$

We first model the gaps based on a mixture of exponential distributions and then investigate the above claims with respect to the model.

3.3 Data

We choose the TIPSTER collection for our experiments because the dataset is of good quality, it is well understood, and it contains a range of different genres (Table 1).

Data Set	Contents of the documents
AP	Copyrighted AP Newswire stories from 1989.
DOE	Short abstracts from the Department of Energy.
FR	Issues of the Federal Register (1989), reporting source actions by government agencies.
PAT	U.S. Patent Documents for the years 1983-1991.
SJM	Copyrighted stories from the San Jose Mercury News (1991).
WSJ	Stories from Wall Street Journal 1987-89
ZF	Computer Select disks 89/90, Ziff-Davis Publishing Co.

Table 1: Description of contents of each of the datasets

We assembled some basic profiling data on these datasets. In Table 2, we list type to token ratios at 10 million words for each dataset. These ratios are calculated by dividing the number of words by the number of unique terms. They give a rough appreciation of the breadth of coverage, to the extent where breath of terminology can reflect this. The value is an indication of the average number of "old" words between occurrences of "new" words in running text.

Data Set	Average document length	Type to token ratio
AP	471.1	106.845
DOE	119.0	94.778
FR	1,370.7	144.866
PAT	4,790.9	134.017
SJM	438.1	102.149
WSJ	420.9	116.183
ZF	395.6	121.798

Table 2: Some basic statistics of each of the datasets

As part of the exercise, we inspected the 100 most frequent terms from each of the datasets by hand. Very frequent terms are not always function words, and lists were sensitive to the domain of some datasets[2] . The 10 most frequent terms (Table 3), nonetheless showed a high degree of overlap, and were clearly function words. The datasets were tokenized based on any space or punctuation, hence we had tokens like *s, o, m, p* occurring among the top 10 terms, but they have been removed from the table. Focusing on these ten terms in experiments across the different collections should yield information on the behaviour of a small collection of very frequent function words.

4 Experimental results

4.1 Homogeneity experiments

Experimental results for the χ^2 based homogeneity experiments are shown in Tables 4 and 5. The top value in each cell shows the CBDF and the bottom value the p-value. Both are averaged over iterations. Bold cells indicate cases where the homogeneity assumption has survived the test (p-value > 0.05). In Kilgarriff (97), inclusion of the most frequent terms means that the behaviour of function words will dominate the outcome of experiments, and that the CBDF measure examines mostly stylistic homogeneity. Here, to allow more detailed tracking of the distribution of very frequent terms, we calculated results at different

[2]Eg, *section* in position 19 in FR, *software* in position 21 in ZF, and *invention* in position 26 in PAT

Data Set	10 Most Frequent Terms
AP	the, of, to, a, in, and, said, for, that, on
DOE	the, of, and, in, a, to, is, for, with, are
FR	the, of, to, and, a, in, for, or, that, be
PAT	the, of, a, and, to, in, is, for, said, as
SJM	the, a, of, to, and, in, for, that, is, san
WSJ	the, of, to, a, in, and, that, for, is, said
ZF	the, and, to, of, a, in, is, for, that, with

Table 3: 10 most frequent terms for each of the TIPSTER datasets

values for N. To save space, we only show snapshots of our results, omitting intermediary results where p-values did not cross the 0.05 threshold.

4.1.1 docDiv experiment

The docDiv experiment maintains document boundaries and assigns whole documents randomly to either partition: it investigates homogeneity across documents in a collection. Table 4 shows that the homogeneity assumption is defeated (p-value < 0.05) quite readily. In the AP and DOE datasets the assumption cannot be defeated for the 10 and 20 most frequent terms, and in the WSJ and SJM datasets for the 10 most frequent terms. All the other datasets show heterogeneity with statistical significance, with p-values of 0 or close to it (very strong evidence against the homogeneity null-hypothesis).

CBDF values provide further insight. In most cases, these are very large, and associated with very low p-values, indicating high levels of non-homogeneity in the distribution of frequent words between documents. This is possibly an indicator of high stylistic variance in a collection.

4.1.2 halfdocDiv experiment

This experiment (Table 5) is sensitive to within-document homogeneity, assigning different halves of each document to each of the partitions. Again, the homogeneity assumption was defeated at some point for most datasets. The exception is the DOE collection where the null-hypothesis remains undefeated for the 20,000 most frequent

	N most frequent terms			
DataSet	**10**	**20**	**50**	**100**
AP	**2.107** **0.1216**	**1.576** **0.2139**	2.583 0.0003	2.290 0
DOE	**1.172** **0.463**	**1.450** **0.160**	1.755 0.0259	1.983 0
FR	54.524 0	41.715 0	72.093 0	66.787 0
PAT	21.074 0	29.315 0	62.494 0	55.353 0
SJM	**3.595** **0.1193**	2.768 0.0077	3.231 0	2.976 0
WSJ	**2.358** **0.178**	2.663 0.0019	2.364 0	2.335 0
ZF	11.947 0	8.133 0	6.907 0	6.576 0

Table 4: docDiv Results. Average CBDF and p-value per dataset using the N most frequent terms. Values in bold indicate cases where the homogeneity assumption has **not** been defeated (p-value > 0.05)

words. This dataset contains very short documents, each unlikely to deal with more than one topic. In stark contrast, the null hypothesis was resoundingly defeated even for the 10 most frequent terms for the FR and PAT datasets, with very low p-values, and comparatively high CBDF. Note that the FR and PAT datasets also have the longest average document lengths of the TIPSTER collection. In addition, these collections also appear the most diverse, with by far the highest type to token ratios (Table 2).

Generally speaking, the experiment finds statistically relevant heterogeneity much more often than the earlier docDiv experiment. Also, CBDF values are much lower here than in the corresponding docDiv table (with the exception of the PAT and FR collections) Comparing docDiv and halfdocDiv experiments suggests that very frequent terms distribute more homogeneously within documents than across document boundaries, but that document length may be a significant factor.

4.2 Modeling gaps

We model the gaps between successive occurrences of a particular term using a mixture of exponential distributions (Sarkar *et al.* 05). Modeling was based on a Bayesian framework which

	N most frequent terms			
DataSet	**10**	**20**	**50**	**500**
AP	**1.774**	**1.473**	**1.271**	1.171
	0.087	**0.117**	**0.066**	0.021
DOE	**0.728**	**0.931**	**1.043**	**1.061**
	0.655	**0.533**	**0.372**	**0.195**
FR	7.905	9.549	11.642	8.847
	0.001	0	0	0
PAT	20.360	15.568	11.886	7.694
	0	0	0	0
SJM	**1.323**	**1.569**	**1.469**	1.332
	0.386	**0.392**	**0.107**	0
WSJ	**1.563**	**1.618**	**1.298**	1.236
	0.279	**0.248**	**0.260**	0.017
ZF	**1.948**	**1.858**	1.609	1.559
	0.129	**0.116**	0.024	0

Table 5: halfdocDiv Results. Average CBDF and p-value per dataset using the N most frequent terms. Values in bold indicate cases where the homogeneity assumption has **not** been defeated (p-value > 0.05)

enables complex models to be fitted (Gelman *et al.* 95; Robert 96). The model provides estimates of the mean of each of the exponential distributions ($\widetilde{\lambda_1}$ and $\widetilde{\lambda_2}$) and estimates of the probability of a gap being generated from each of these distributions ($\widetilde{p}$ and $1-\widetilde{p}$).

To examine the homogeneity assumption, we have to investigate if any of the following claims are true:
$\widetilde{p}=0$ or $\widetilde{p}=1$ or $\widetilde{\lambda_1}=\widetilde{\lambda_2}$.
The validity of any of these claims would reduce the mixture model to a single component exponential distribution, which would be consistent with the assumption of homogeneity. We constructed the mixture models for the terms in Table 3, but due to lack of space we provide a full list of the parameter estimates for only three of those terms *the* (Table 6), *of* ((Table 7)) and *said* (Table 8).

For the term *the* (Table 6), $\widetilde{\lambda_1}$ and $\widetilde{\lambda_2}$ are very similar in the AP and WSJ datasets and $\widetilde{p}$ is close to 0 in the FR, PAT and SJM datasets, so in these datasets *the* may distribute homogeneously. In the DOE and ZF datasets, however, $\widetilde{p}$ is near neither 0 nor 1, and $\widetilde{\lambda_1}$ and $\widetilde{\lambda_2}$ differ markedly, so *the* does not appear to distribute homogeneously in these two datasets. Similarly, the term *of* (Table 7) has very similar values of $\widetilde{\lambda_1}$ and $\widetilde{\lambda_2}$ for

Data Set	$\widetilde{p}$	$1-\widetilde{p}$	$\widetilde{\lambda_1}$	$\widetilde{\lambda_2}$
AP	0.59	0.41	16.58	16.11
DOE	0.29	0.71	20.49	12.72
FR	0.01	0.99	194.89	13.47
PAT	0.03	0.97	58.96	10.61
SJM	0.02	0.98	168.52	17.80
WSJ	0.70	0.30	17.46	17.00
ZF	0.10	0.90	67.80	18.39

Table 6: Parameter estimates for the term *the* in each of the datasets

AP, DOE and WSJ datasets and $\widetilde{p}$ is close to 0 for the FR, PAT, SJM and ZF datasets. Hence for the term *of* each of the datasets provide little evidence against homogeneity either based on the values of $\widetilde{\lambda_1}$ and $\widetilde{\lambda_2}$ or $\widetilde{p}$. In contrast, the term *said* (Table 8) shows evidence of homogeneity only for the AP dataset, for which the value of $\widetilde{p}$ is close to 0.

Data Set	$\widetilde{p}$	$1-\widetilde{p}$	$\widetilde{\lambda_1}$	$\widetilde{\lambda_2}$
AP	0.65	0.35	38.37	36.44
DOE	0.62	0.38	21.10	19.72
FR	0.02	0.98	106.25	24.01
PAT	0.03	0.97	73.42	21.82
SJM	0.04	0.96	205.38	39.45
WSJ	0.42	0.58	36.91	35.39
ZF	0.01	0.99	262.47	46.51

Table 7: Parameter estimates for the term *of* in each of the datasets

Data Set	$\widetilde{p}$	$1-\widetilde{p}$	$\widetilde{\lambda_1}$	$\widetilde{\lambda_2}$
AP	0.04	0.96	696.38	69.01
DOE	0.67	0.33	61349.69	12224.94
FR	0.84	0.16	26385.22	392.62
PAT	0.06	0.94	2167.32	13.10
SJM	0.16	0.84	2499.38	92.42
WSJ	0.12	0.88	1608.49	72.62
ZF	0.42	0.58	8810.57	177.21

Table 8: Parameter estimates for the term *said* in each of the datasets

To investigate the homogeneity assumption for other common terms, we calculate the ratio between the two $\widetilde{\lambda}$s, $\widetilde{\lambda_1}/\widetilde{\lambda_2}$ and study how close this ratio is to 1. A $\widetilde{\lambda_1}/\widetilde{\lambda_2}$ ratio of 1 indicates that the two exponential distributions have equal means, and hence reduce to a single exponential distrib-

ution. A large deviation of the $\widetilde{\lambda_1}/\widetilde{\lambda_2}$ ratio from 1 reveals the presence of two very distinct exponential distributions and provides evidence against the homogeneity assumption of the term's distribution in the corpus provided the value of $\widetilde{p}$ is close to neither 0 or 1. If the value is very close to 0 or 1 (a difference of less than 0.05) we argue that one of the exponential distributions have negligible effect and there is little evidence against the term being homogeneously distributed. Table 9 provides the $\widetilde{\lambda_1}/\widetilde{\lambda_2}$ ratio and the values of $\widetilde{p}$ for the most frequent terms of each of the datasets.

In the table ratios of $\widetilde{\lambda_1}/\widetilde{\lambda_2}$ that are less than 1.2 are given in bold-face type, as are values of $\widetilde{p}$ that are below 0.05 or above 0.95. Combinations are underlined when one (or more) of the terms are in bold. For these terms the model does not suggest the assumption of homogeneity is violated, but the assumption seems poor for the terms that are not underlined in the table. (Formal statistical tests based on this model are currently being developed.) It can be observed from Table 9 that only the term *of* show signs of being homogeneously distributed across all the datasets either based on the $\widetilde{\lambda_1}/\widetilde{\lambda_2}$ ratio or values of $\widetilde{p}$ being close to 0. The terms *and*, *are*, *the* and *to* also seem to be homogeneously distributed across many of the datasets. The other 12 terms in Table 9 only appear to be homogeneously distributed in at most 2 of the 7 datasets.

Said is an interesting term in the table. It has very high values of the $\widetilde{\lambda_1}/\widetilde{\lambda_2}$ ratio and the values vary over a huge range. Also, the value of $\widetilde{p}$ for *said* is close to 0 for the AP dataset. This is because the term *said* has a huge dependence on the document's content and style, and these characteristics can be explored and studied by modeling the gaps. These findings allow the model to be used in characterizing genre and stylistic features.

The term *san* is an outlier in the list, as it is not a function word. But it featured in the list of top 10 terms in the SJM (stories from San Jose Mercury news) collection, being a very widely used term in that collection. As expected, based on the model, it is a very rare term, as indicated by a large rate of occurrence $\widetilde{\lambda_1}$, and it's bursty nature is indicated by small values of $\widetilde{\lambda_2}$, leading to large values of the $\widetilde{\lambda_1}/\widetilde{\lambda_2}$ ratio. Also, in contrast, the $\widetilde{\lambda_1}/\widetilde{\lambda_2}$ ratio for the SJM collection, is relatively small when compared to the values of the other collections, demonstrating the fact that the term *san* is a non-informative term in SJM, relative to the other collections.

The term *as* is also quite interesting, as it exhibits large values of $\widetilde{\lambda_1}/\widetilde{\lambda_2}$ ratio in all the collections other than FR and PAT. PAT has comparatively large values of $\widetilde{\lambda_1}/\widetilde{\lambda_2}$ ratio for most of the other terms, indicating the fact that the term has dependence on the content, style and structure of the document and collection. Under such circumstances will it be appropriate to apply a generic "stop-word" list for a collection of any documents? Based on the above experiments, we beleive that the answer is "no".

5 Conclusion

Our homogeneity experiments indicate that very frequent function words do not distribute homogeneously in general, across different documents, even when those documents are of the same, or a related genre (docDiv). They also show that such words do distribute more homogeneously within document boundaries, but that this behaviour is highly sensitive to document type, and may well depend on factors related to document length, and breadth of domain coverage per document. They demonstrate that the same very frequent function words take on very different distribution patterns in different collections, even where such collections belong to related genres. (halfdocDiv).

We further investigated the 10 most frequent terms of each of the collections by modeling the gaps between successive occurrences of a particular term based on a mixture of two exponential distributions. One of these distributions measure the inherent rate of occurrence of the term in the corpus, and the other measures the rate of re-occurrence after the term has occurred recently. The experiments demonstrate that terms distribute in this pattern, as compared to a "bag of words" homogeneity model where a single exponential distribution would be sufficient. Our experiments reveal that terms do occur in bursts, including most of the very frequent ones.

References

(Cavagli 02) Gabriela Cavagli. Measuring corpus homogeneity using a range of measures for inter-document distance. In *Third International Conference on Language Resources and Evaluation.*, 2002.

(Church 00) K. Church. Empirical estimates of adaptation: The chance of two noriega's is closer to p/2 than p^2. In *COLING*, pages 173–179, 2000.

Term	AP	DOE	FR	PAT	SJM	WSJ	ZF
a	1.98 (0.17)	3.62 (0.46)	3.88 (0.23)	3.34 (0.15)	2.33 (0.10)	2.00 (0.14)	2.59 (0.10)
and	**1.05** (0.30)	**1.05** (0.53)	2.74 (0.11)	3.62 (**0.02**)	3.62 (0.07)	**1.06** (0.14)	**1.05** (0.10)
are	**1.06** (0.69)	**1.05** (0.32)	3.01 (0.07)	5.09 (0.33)	10.43 (**0.01**)	**1.05** (0.69)	**1.16** (0.47)
as	31.64 (0.93)	31.47 (0.93)	3.97 (0.45)	4.46 (0.24)	40.73 (0.90)	65.09 (0.90)	56.38 (0.91)
be	3.03 (0.73)	1.30 (0.49)	6.06 (0.13)	5.71 (0.27)	3.48 (0.64)	2.13 (0.33)	2.23 (0.27)
for	2.04 (0.29)	3.19 (0.54)	4.40 (**0.05**)	4.09 (0.26)	2.61 (0.55)	1.89 (0.31)	15.94 (**0.01**)
in	2.23 (0.13)	2.49 (0.17)	7.08 (**0.02**)	4.01 (**0.05**)	2.94 (0.10)	1.93 (0.22)	2.83 (0.08)
is	2.93 (0.58)	4.67 (0.35)	3.87 (0.19)	5.34 (0.07)	4.04 (0.34)	2.43 (0.34)	5.76 (**0.02**)
of	**1.05** (0.65)	**1.07** (0.62)	4.43 (**0.02**)	3.37 (**0.03**)	5.21 (**0.04**)	**1.04** (0.42)	5.64 (**0.01**)
on	1.99 (0.31)	5.73 (0.72)	4.72 (0.21)	5.95 (0.25)	2.59 (0.46)	1.95 (0.55)	2.58 (0.22)
or	41.50 (**0.95**)	3.69 (0.48)	6.87 (0.36)	9.98 (0.28)	8.63 (0.81)	4.58 (0.71)	3.66 (0.78)
said	10.09 (**0.04**)	5.02 (0.67)	67.20 (0.84)	165.39 (0.06)	27.04 (0.16)	22.15 (0.12)	49.72 (0.42)
san	112.37 (0.92)	21.19 (0.74)	579.36 (0.93)	855.81 (0.93)	14.43 (0.46)	149.67 (0.96)	90.67 (0.80)
that	2.78 (0.16)	1.23 (0.59)	4.89 (0.15)	4.69 (0.21)	3.42 (0.20)	2.47 (0.11)	4.34 (**0.04**)
the	**1.03** (0.59)	1.61 (0.29)	14.47 (**0.01**)	5.56 (**0.03**)	9.47 (**0.02**)	**1.03** (0.70)	3.69 (0.10)
to	3.18 (0.10)	**1.15** (0.56)	12.45 (**0.01**)	3.81 (**0.05**)	6.78 (**0.02**)	**1.13** (0.41)	3.24 (**0.04**)
with	2.62 (0.40)	2.40 (0.28)	3.54 (0.23)	3.55 (0.24)	2.70 (0.64)	1.29 (0.45)	2.53 (0.12)

Table 9: Table showing values of the $\widetilde{\lambda_1}/\widetilde{\lambda_2}$ ratio and values of p for the frequent terms for all the datasets. $\widetilde{\lambda_1}/\widetilde{\lambda_2}$ ratios close to 1 are marked in bold (and underlined) and values of $\widetilde{p}$ close to 0 or 1 are also marked in bold (and underlined), providing evidence of the term being uniformly distributed in that dataset.

(De Roeck *et al.* 04) Anne De Roeck, Avik Sarkar, and Paul H Garthwaite. Defeating the homogeneity assumption. In Gerald Purnelle, Cedrick Fairo, and Anne Dister, editors, *Proceedings of 7th International Conference on the Statistical Analysis of Textual Data (JADT)*, pages 282–294, De Louvain, Belgium, 2004. UCL Presses Universitaries.

(Dunning 93) Ted E. Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74, 1993.

(Franz 97) Alexander Franz. Independence assumptions considered harmful. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pages 182–189, 1997.

(Gelman *et al.* 95) A. Gelman, J. Carlin, H.S. Stern, and D.B. Rubin. *Bayesian Data Analysis*. Chapman and Hall, London, UK, 1995.

(Katz 96) Slava M. Katz. Distribution of content words and phrases in text and language modelling. *Natural Language Engineering*, 2(1):15–60, 1996.

(Kilgarriff 97) A Kilgarriff. Using word frequency lists to measure corpus homogeneity and similarity between corpora. In *Proceedings of ACL-SIGDAT Workshop on very large corpora*, Hong Kong, 1997.

(Rayson & Garside 00) P. Rayson and R. Garside. Comparing corpora using frequency profiling. In *In proceedings of the Workshop on Comparing Corpora*, pages 1–6, 2000.

(Robert 96) Christian. P. Robert. Mixtures of distributions: inference and estimation. In W.R. Gilks, S. Richardson, and D.J. Spiegelhalter, editors, *Markov Chain Monte Carlo in Practice*, pages 441–464, 1996.

(Rose & Haddock 97) Tony Rose and Nick Haddock. The effects of corpus size and homogeneity on language model quality. In *In proceedings of the ACL-SIGDAT Workshop on Very Large Corpora*, pages 178–191, 1997.

(Sarkar *et al.* 05) Avik Sarkar, Paul H. Garthwaite, and Anne De Roeck. A Bayesian mixture model for term re-occurrence and burstiness. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 48–55, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

(Wilbur & Sirotkin 92) John Wilbur and Karl Sirotkin. The automatic identification of stop words. *Journal of Information Science*, 18(1):45–55, 1992.

(Yang & Wilbur 96) Yiming Yang and John Wilbur. Using corpus statistics to remove redundant words in text categorization. *Journal of the American Society of Information Science*, 47(5):357–369, 1996.

Using context-window overlapping in synonym discovery and ontology extension

María Ruiz-Casado, Enrique Alfonseca and **Pablo Castells**

Department of Computer Science
Universidad Autónoma de Madrid
28049 Madrid
{Maria.Ruiz,Enrique.Alfonseca,Pablo.Castells}@uam.es

Abstract

This paper describes a new, unsupervised procedure called *Context-window overlapping* for calculating the semantic distance between two terms. It is based on the distributional semantics hypothesis, and, in particular, in the fact that synonym words should be interchangeable in every context, and hyponyms can be substituted by their hyperonyms in most contexts.

The procedure has been applied to synonym identification, and to ontology extension. In the first task, it has been evaluated with 80 synonym test questions from the TOEFL which already constitute a standard test set in this problem, and attains results similar to most other non-ensemble procedures. Interestingly, it clearly outperforms Latent Semantic Analysis, other procedure grounded on the Distributional Semantic hypothesis. Concerning ontology enrichment, the results obtained are promising, although they can still be much improved. Conclusions are drawn from this result, and we outline several possibilities for future work.

1 Introduction

There is much work concerning modelling semantic similarity between words. Some use statistical models, and other represent contexts using the vector space model, or make use of conceptual hierarchies (Banerjee & Pedersen 03; Budanitsky & Hirst 01; Resnik 99). Such metrics have very useful applications for both Information Retrieval and Automatic Annotation in the semantic web, as they have been used for disambiguating word senses inside documents (Agirre *et al.* 01), automatically extending conceptual ontologies (Alfonseca & Manandhar 02), and extending user queries with synonyms discovered automatically (Turney 01).

In this paper, we describe a new simple algorithm, also grounded on the Distributional Semantics hypothesis. The results obtained so far are very promising, when compared to most of the previous-mentioned procedures. The procedure has been evaluated in two different tasks: synonym identification, and automatic ontology enrichment, with encouraging results.

The paper is structured as follows: Section 2 describes the metric used to measure the similarity between terms; Sections 3 and 4 describe the two applications in which it has been evaluated. Finally, Section 5 draws some conclusions and describes open lines for future work.

2 Similarity metric with context-window overlapping

The Distributional Semantics (DS) hypothesis states that the meaning of a word w is highly correlated to the contexts where w appears (Rajman & Bonnet 92). From this assumption, it is possible to develop statistical computational tools for calculating similarities in word meanings, which have been applied to Information Retrieval (Rajman & Bonnet 92; Salton 89), Text Summarisation (Lin 97), word-sense disambiguation (Yarowsky 92; Agirre *et al.* 00), and word clustering (Lee 97; Faure & Nédellec 98).

This section starts with some commonly agreed definitions of two semantic relations that are very relevant for characterising word meaning: hyponymy and synonymy. Next, the new procedure proposed is described.

2.1 A definition of hyponymy and synonymy based on contexts

Hyponymy is a semantic relationship which relates a concept with more general concepts, such as *horse* with *animal*. It can be defined in the following way:

> **Definition 1a.** Hyponymy is a relation of meaning inclusion between linguistic expressions. A is a hyponym of B if B is true for any concept x whenever A is true for x.
>
> Hyperonymy is the inverse relation to hyponymy.

For example (Resnik 93), every single QUEEN is a WOMAN, and therefore QUEEN is a hyponym of WOMAN. This implies that any utterance about a queen x entails the same utterance where x is referred to as being a woman, e.g. (1a) entails (1b).

(1) a. The Prime Minister honoured the queen with his presence.

b. The Prime Minister honoured the woman with his presence.

The example leads us to the definition of hyponymy in terms of interchangeability of linguistic expressions:

> **Definition 1b.** A is a hyponym of B if and only if for every sentence S containing A, S entails the same sentence with A substituted for B, $S[A/B]$ (Lyons 61).

Word Meanings	Word forms horse	heroin	junk	debris
horse, Equus sp.	×			
horse, heroin (drug)	×	×	×	
junk (Chinese boat)			×	
debris, detritus			×	×
...				

Table 1: Example of lexical matrix, showing some words and the concepts they lexicalise.

The second lexical relationship described in this section is **synonymy**, which relates words that convey the same meaning. In (Miller 95), synonymy is characterised as a matrix that relates word meanings to word forms. Word forms are typically sequences of characters delimited by spaces. However, in some contexts special symbols may be considered words and, as (Resnik 93) points out, provision must be made as well for multi-word expressions. Word meanings refer to "the lexicalised concept that a form can be used to express" (Miller 95). A particular example with some concepts and word forms is shown in Table 1. Here, {horse, heroin, junk} is a set of synonyms (a *synset*) that represents the concept *heroin* as a drug.

Some semanticists argue that the denotational meaning of a word is fully realised in contexts. As Firth (57, pg. 7) says, "The complete meaning of a word is always contextual, and no study of meaning apart from a complete context can be taken seriously", a theory agreed also by Cruse (86, p. 270) when he says that "natural languages abhor absolute synonyms just as nature abhors a vacuum". Under this premise, it is rare that two words have exactly the same meaning and are exchangeable in every possible context. Edmonds & Hirst (02) argue that many words are not absolute synonyms, but near-synonyms (also called *plesionyms*).

Even so, for practical purposes, we often use the relationship of synonymy between words, for instance, when explaining the meaning of a word in a context by giving other words which can be used in the same place (Resnik 93). In this way, we could define synonym words as words that convey the same meaning. Therefore, we can write parallel definitions to (1a) and (1b), using the fact that synonym words must be interchangeable in every context.

> **Definition 2a. Synonymy** is a relation of meaning identity between linguistic expressions. A and B are synonyms if and only if B is true for a concept x whenever A is true for x **and vice versa**.
>
> **Definition 2b.** Two word forms w_1 and w_2 are **synonyms** if and only if for every sentence S containing A, then S entails $S[A/B]$, and for every sentence T containing B, then T entails $T[B/A]$.

Corollary 2c comes straightforwardly from definition 2b. If two word forms w_1 and w_2 are exchangeable in every sentence where any one of them appears, then they can be used in exactly the same contexts in language:

> **Corollary 2c.** If two word forms w_1 and w_2 are synonym, then they can appear in exactly the same contexts, preserving the truth value.

Finally, we can define synonymy in terms of hyperonymy as in the following definition. It can be seen that, if we use definition (2d), then (2a) and (2b) can be derived from (1a) and (1b).

> **Definition 2d.** Two word forms w_1 and w_2 are synonyms if and only if both w_1 is a hyperonym of w_2 and w_2 is a hyperonym of w_1.

This said, we should also bear in mind that, although the notion of synonymy may be useful for practical purposes, it is very rare to find two words that are completely interchangeable. The difference in meaning may just be stylistic, or due to dialectal variations, but even in these cases we can expect that the sets of contexts in which we shall be able to find the two words will not be absolutely coincident.

2.2 Measuring similarity between contexts

These definitions of hyperonymy and synonymy give us ground to define metrics for semantic similarity between word forms which are based on the similarity between contexts. A popular technique to encode contexts and measure their similarity is the Vector Space Model (VSM), given a word w which appears in a corpus, we first define a context length (e.g. the words in the same sentence, or the words in a window of width L), and next we collect all the words in the context of every occurrence of w inside a bag. That *bag of words* will represent the meaning of w, and several semantic similarity metrics can be defined between the bags corresponding to two words. VSM can be extended with Latent Semantic Analysis (LSA), a dimensionality reduction procedure (Landauer & Dumais 97). We should note that, in VSM, there is much information lost, as all the words are put together in the bag and the syntactic dependences between the contextual terms will not be stored in the model.

A different approach, Pointwise Mutual Information (PMI) (Turney 01) is grounded on the slightly different assumption that two words with similar meanings will tend to appear near each other:

$$PMI(w_1, w_2) = \frac{hits(w_1\ NEAR\ w_2)}{hits(w_1)hits(w_2)}$$

So, for instance, many documents about *cars* are expected to contain the synonym word *automobile* as well.

one_way_similarity(w_1, w_2)
1. return count(w_1, w_2)

two_ways_similarity(w_1, w_2)
1. return count(w_1, w_2) + count(w_2, w_1)

count(w_1, w_2)
1. Collect, in S_1, $N_{snippets}$ Google snippets where w_1 appears
2. Set $n = 0$
3. For each snippet s_i in S_1,
   3.1. ctx = window of width L around w_1 in s_i.
   3.2. Remove the words from ctx if there is a sentence ending between them and w_1.
   3.3. If number of open-class words in $ctx < \theta$, *continue*.
   3.4. If ctx has already been seen, *continue*.
   3.5. Substitute w_1 by w_2 in ctx.
   3.6. Search in Google for ctx.
   3.7. If found any result, increment n.
4. Return n.

Figure 1: Pseudocode of the *Context-window overlapping* algorithm. $N_{snippets}$ is the number of snippets obtained from Google; L is the context width; θ is the minimum number of open-class words to consider a context.

2.3 Context-window overlapping

A possible drawback of the VSM technique is that much information is lost when all the sentences are reduced to a bag-of-words representation. In this operation:

- We lose information of which terms appeared in which contexts, as all the contexts are merged in a single vector.
- We lose information of the word-order and the phrasal structures inside each context.

If we want to calculate the similarity between two words, w_1 and w_2, ideally, it should be better to keep, for each of them, the complete contexts in which they can appear, and to compare the two sets of contexts, without any other transformation. If we consider sentences as contexts, we could describe the ideal procedure for calculating the similarity between two words in the following way:

1. Collect in S_1 every possible sentence in which w_1 can appear.
2. Collect in S_2 every possible sentence in which w_2 can appear.
3. Calculate the percentage of sentences in S_1 in which we can substitute w_1 for w_2 to obtain a sentence from S_2, and vice versa.

This procedure has two problems which stem from the current limitations of the technology:

- The number of possible sentences in which any word can appear can be arbitrarily large. If the sentences are collected from a textual corpus, we will necessarily have a sparse data problem. To overcome or reduce this problem, rather than collecting full sentences, we restrict the length of the context to a narrow *window*.
- It is highly unlikely that any corpus, apart from the Internet, will be large enough to let us collect enough contexts for both words. Therefore, we shall be forced to use the Internet. In this case, if we collect the contexts using a search engine, the time needed to get all the contexts in which w_1 appears (which may be hundreds of millions) will be so high that the procedure will not be usable at all.

  This problem might be reduced if (a) we collect a limited number of contexts for the first word, w_1, and (b) we directly substitute w_1 for w_2 inside those same contexts, to estimate the size of the intersection of S_1 and S_2.

Figure 1 shows the pseudocode of the *Context-window overlapping* algorithm. In a few words, it collects a list of contexts where w_1 appears, and it counts in how many of them it is possible to substitute w_1 with w_2, using the Internet as the reference corpus. In the version called *two_way_similarity*, the same is repeated exchanging the roles of w_1 and w_2.

The following sections describe the application of this algorithm for two different tasks: identification of synonym words, and ontology extension.

3 Synonym discovery

A particular application of semantic similarity metrics is the automatic identification of synonym words. Reported approaches for to solve this problem include LSA (Landauer & Dumais 97), PMI (Turney 01), metrics of proximity in documents combined with patterns of incompatibility (Lin & Zhao 03), thesaurus-based methods (Jarmasz & Szpakowicz 03), corpus-based similarity metrics (Terra & Clarke 03), and a combination of various procedures (Turney *et al.* 03). Several of the previous methods are grounded on the DS hypothesis.

All the systems reported here have been tested on a TOEFL test. The data set consists of 80 words, and for each of these words there are four possible

synonym candidates. The purpose of the task is to decide which of those candidates is the actual synonym. For instance, the first term in the data set is *enormously*, with candidate synonyms *appropriately, uniquely, tremendously* and *decidedly*. The system has to decide that *tremendously* is the synonym of the word. Recently, (Freitag *et al.* 05) have proposed a procedure for automatically generating TOEFL questions from WordNet.

In this approach, the two-way context-window overlapping procedure is used. As stated in section 2.1, the general idea is that, if two words are synonyms, then they are exchangeable in every context. Following the procedure introduced before, we can collect some snippets for the first word, substitute it by each of the candidate synonyms, and look how many of the context windows, with the original word substituted by the candidate, are also indexed by Google. Next, the same process is repeated by substituting the candidate synonym by the original word. The candidate that maximises the number of context windows in which we can interchange the two words will be selected.

There are three parameters have been set empirically:

- The window width (L) that has been taken is 5 words (the word under study, and two at each side). If this size is incremented, then the program returns a score of 0 for most of the candidates, because the windows would be too large and the probability of finding the same context window with the candidate synonym is very small.
- The threshold to consider that a context is informative is $\theta = 2$ (see step 3.3 in Figure 1). In this way, if Google has returned a context that is too small, for instance, because the original word is starting and ending a sentence, or because the context mainly contains closed-class words, then it will be ignored. With this threshold, context windows such as *for the WORD of the* will not be considered, because all the words at the left and at the right sides are closed-class words.
- Concerning the number of snippets to download, $N_{snippets}$, we have tried with several values, and we discovered that some of the candidates are more frequent than others. Hence, with a fixed number of snippets, it may be the case that all the candidates receive a similarity of 0. Therefore, $N_{snippets}$ is chosen dynamically to ensure that, from the several candidates, at least one of them reaches a count greater than 30. If there is a draw, more snippets are collected until it is untied. Note that these restrictions may require the collection of more than 1000 snippets from Google in some cases.

For instance, in the example mentioned above, *tremendously* had a score of 31, *uniquely* had a score of 5, *appropriately* had a score of 2, and *decidedly* had a score of 0. Therefore the first one was chosen as the candidate synonym for *enormously*.

Procedure	Acc.	95% conf.
(Landauer & Dumais 97)	64.40%	52.90–74.80%
non-native speakers	64.50%	53.01–74.88%
(Turney 01)	73.75%	62.71–82.96%
(Jarmasz & Szpakowicz 03)	78.75%	68.17–87.11%
(Terra & Clarke 03)	81.25%	70.97–89.11%
(Lin & Zhao 03)	81.25%	70.97–89.11%
CW overlapping	**82.50%**	72.38–90.09%
(Turney *et al.* 03)	97.50%	91.26–99.70%

Table 2: Results obtained (accuracy), and other published results on the TOEFL synonym results, from Turney *et al.* (03).

Results Table 2 shows the results obtained, compared to other published results on the TOEFL data set[1]. As can be seen, it outperforms all the previous approaches (although there is a statistical tie with some of them) except (Turney *et al.* 03). However, compared to this, our approach has the advantage that it does not require training, as it is fully unsupervised, and it is much more simple to implement.

4 Ontology extension

Ontologies are often described as "explicit specifications of a conceptualisation" (Gruber 93). They have proved to be a useful tool for knowledge representation. In many cases, ontologies are structured as hierarchies of concepts, by means of the hyperonymy relationship. Given the large cost of building and maintaining ontologies, there is already much work on procedures for automatically structuring concepts in ontologies, and for extending existing ontologies with new terms, and for populating an ontology with instances of its concepts. These tasks are usually called *ontology building*, *ontology enrichment* and *ontology population*, respectively. We may classify current approaches for ontology enrichment from text in the following groups:

- Systems based on distributional properties of words: they use some kind of distance metric based on co-occurrence information. This metric can be applied for clustering (Lee 97; Faure & Nédellec 98), for Formal Concept Analysis (Cimiano & Staab 04) or for classifying words inside existing ontologies (Hastings 94; Hahn & Schnattinger 98; Pekar & Staab 03; Alfonseca & Manandhar 02) or supersense categories (Curran 05).
- Systems based on pattern extraction and matching: these rely on lexical or lexicosemantic patterns to discover ontological and non-taxonomic relationships between concepts in unrestricted text. They may be based on manually defined regular expressions of words, (Hearst 92; Hearst 98; Berland & Charniak 99) or may learn such

[1]Obtained from Landauer and Praful Chandra Mangalath.

findHyperonyms(Word w)
1. Initialise a list *Candidates* with the top node.
2. While the list *Candidates* has changed in the previous iteration:
    2.1. Extend the list *Candidates* with the hyponyms of all the nodes that are already inside it.
    2.2. For every node n in *Candidates* (which is a set of synonym words),
        2.2.1. Initialise *n.score* to 0.
        2.2.2. For every synonym word s in that node,
            2.2.2.1. *n.score+ = one-way-similarity(w,s).*
    2.3. *Candidates* $\leftarrow$ the N nodes with the best scores.
3. Return *Candidates*.

Figure 2: Pseudo-code of the program for finding candidate hyperonyms for a given word. N is the beam width of the search.

Step	Top-5 Candidates	Score
1	(a) unit, whole, whole thing	31
	(b) location	17
	(c) body of water, water	17
	(d) building block, unit	16
	(e) part, piece	13
2	(a) unit, whole, whole thing	31
	(b) point	30
	(c) part, region	29
	(d) region	19
	(e) line	19
3	(a) area, country	34
	(b) point	16
	(c) district, territory	15
	(d) place, spot, topographic point	13
	(e) unit, whole, whole thing	11
4	(a) center, centre, eye, heart, middle	33
	(b) area, country	20
	(c) district, territory	13
	(d) place, spot, topographic point	9
	(e) point	9
5	(a) center, centre, eye, heart, middle	33
	(b) area, country	20
	(c) district, territory	13
	(d) place, spot, topographic point	9
	(e) point	9

Figure 3: Example showing the classification of Colchester

patterns from text (Finkelstein-Landau & Morin 99; Ruiz-Casado *et al.* 05). Navigli & Velardi (04) incorporates terminology extraction and ontology construction.

- Systems based on dictionary definitions analysis (Wilks *et al.* 90; Rigau 98; Richardson *et al.* 98) take advantage of the particular structure of dictionaries in order to extract hyperonymy relationships with which to arrange the concepts in an ontology. Concept definitions and glosses have been found very useful, as they are usually concise descriptions of the concepts and include the most salient information about them (Harabagiu & Moldovan 98).

This section describes a procedure for automatically extending WordNet with new terms using the context-window overlapping algorithm. If we follow definition (1b), we can assume that a term, in a sentence, can, in principle, be substituted by any of its hyperonyms. On the other hand, the inverse does not necessarily hold. Therefore, in this case, the procedure used is the one-way overlapping.

The algorithm is a top-down beam search procedure, in which we start at the top node in the ontology, and we proceed downwards, considering that node and all its children as candidate hyperonyms. The process is described in Figure 2.

Evaluation and results For the moment, the algorithm has been tested using the taxonomy of *entities* from WordNet, and 23 terms from the Simple English Wikipedia which did not appear in WordNet. The choice of these resources was done because our final purpose is to apply these techniques in a project about automatic knowledge acquisition from the Wikipedia. To choose the terms for the experiment, we first identified all the terms in the Wikipedia which were not in WordNet (around 600). Next, we removed from the beginning of the list, manually, those which were not hyponyms of *entity*. The 23 terms were chosen in order to have a representation of several kinds of concepts: persons, animals, locations and objects. Furthermore, in order to speed up the process, WordNet has been pruned to the 483 synsets that have an Information Content less than 7 (Resnik 99).

As an example, Figure 3 shows the classification performed for the concept Colchester. In the first step, *Colchester* is compared to the WordNet synset *entity*, and all its hyponyms, and the five ones with the highest scores (1a-1d) are kept for the next iteration. In the second step, Colchester is compared with these five synsets and all their hyponyms. This procedure is repeated until the set of five hyponyms does not change in one interation. In the example, this happens in iteration 5, moment in which they are returned to the user as candidates. In this example, the proposed hyperonyms at the end are *centre*, *area*, *district*, *place* and *point*.

Table 3 shows the results obtained when classifying the 23 terms inside WordNet. In the table, the candidates which are correct appear in bold-font, and

Term	Candidate 1	Candidate 2	Candidate 3	Candidate 4	Candidate 5
A Brief History of Time	human	piece	**whole**	body of water	organism
Alanis Morisette	**human**	**adult**	animal	**being**	part
Alaskan Native	**human**	*female person*	whole	*male person*	*woman*
Alpha male	**male person**	**human**	*male child*	**adult male**	**chief**
Angelina Jolie	**human**	animal	**adult**	part	compeer
Audrey Hepburn	**human**	animal	part	unit	**adult**
Bangalore	human	part	flora	compeer	line
Basque Country	*centre*	human	**area, country**	**region**	animal
Brad Pitt	**human**	animal	flora	**adult**	*friend*
Breakfast sausage	human	**whole**	part	body of water	friend
Britney Spears	part	body of water	unit	whole	cell
Brixton	human	whole	**part, region**	*line*	body of water
Burnham-on-Sea	**place, stop**	**location**	**part, region**	*line*	whole
Buzz Aldrin	**human**	animal	flora	part	whole
Caenorhabditis elegans	human	whole	**being**	cell	flora
Carl Sagan	**human**	animal	unit	**compeer**	part
Chorizo	human	animal	**whole**	body of water	part
Christina Ricci	part, region	whole	body of water	part	line
Christmas cracker	human	part	region	compeer	body of water
Christopher Columbus	**human**	part	unit	compeer	body of water
Coca-Cola	body of water	unit	part	line	region
Colchester	*centre*	**area, country**	**district**	**place**	*point*
Crewkerne	**place**	**part, region**	whole	*point*	part

Table 3: Results obtained for each of the the 23 terms.

those which are near correct hyperonyms in the ontology appear in italics. There are two terms which appear underlined; they correspond to the cases in which it was possible to identify automatically that the classification had not been successful, because the five candidates were all very far apart from each other in WordNet. Note that none of the candidates is located far too deep in the hierarchy; that is due to the pruning performed to WordNet.

It is possible to draw interesting observations from the results obtained:

- In some cases, the name chosen, in the ontology, for the hyperonym, is not easily associated to the word that we want to classify. This is the case of most objects and artifacts, which should be classified as *whole,unit* in order to proceed with the classification. Things such as book titles can probably be substituted by the term *book*, but they are hardly exchangeable by *whole* or *unit*. In these cases, the algorithm usually remains in the upper parts of the ontology and does not reach the most specific candidate hyperonym, as has happened with most objects in our experiment.
  This problem might improve if we modify the classification algorithm to force it proceed down deeper in the ontology.
- In some of the nodes, there are some synonyms terms which are used most of the times with a different sense. For instance, there is a node in WordNet, which is a hyponym of *location*, with the synonym terms {*center, centre, middle, heart, eye*}. The score for this node is inflated because there are many pages in the Internet containing the words *heart* and *eye*, but used with a different sense (as body parts).
  A possible solution might be to start by pruning the words in all the synsets to remove the meanings of the words that are rarely used.
  A clear weakness of this algorithm is its inability to treat polisemous words. This has been seen in the example with *eye*, but it would also happen with examples such as *horse* in Table 1 (meaning both Equus and heroin).
- Some terms are more common in the Internet that others. For instance, the words in the synset {*person, individual, someone, somebody, mortal, human, soul*} appear, as indexed by Google, with a frequency that is one order of magnitude higher than the words in many of the other synsets. Therefore, it will be more probable to find context windows with these words, just because they are more common. In fact, *person* was one of the five hyperonym candidates in 17 out of the 23 cases.
  This may indicate that it should be useful to adjust the frequencies using a statistical test, such as the χ^2 or the log likelihood.

5 Conclusions and future work

In this paper, we describe a procedure for calculating a semantic similarity metric between terms, based on their interchangeability in textual contexts. The metric has been tested on two different tasks: synonym discovery, consisting on identifying amongst four candidates, which one is the synonym of a given word; and ontology enrichment with new terms. The results for synonym detection are very good, being either equal or higher than all the other unsupervised methods. In the case of ontology enrichment, the results seem promising for the moment, and we also describe several ways in which we believe that the algorithm can be improved.

Some lines open for future work include to study more in-depth how the performance changes if we vary the parameters L, θ and $N_{snippets}$; and to check whether this procedure also outperforms VSM or LSA

in other problems.

Concerning ontology enrichment, we believe that the system could be much improved if we apply the solutions proposed in Section 4: to modify the algorithm to search deeper in the ontology; to work with weights calculated with an statistical test, rather than working with frequencies, and to remove, from each synset, the words which are generally used with a different sense, for instance, using Semcor to calculate the frequency of each sense.

References

(Agirre *et al.* 00) E. Agirre, O. Ansa, E. Hovy, and D. Martinez. Enriching very large ontologies using the www. In *Ontology Learning Workshop, ECAI*, Berlin, Germany, 2000.

(Agirre *et al.* 01) E. Agirre, O. Ansa, D. Martínez, and E. Hovy. Enriching wordnet concepts with topic signatures. In *Proceedings of the NAACL workshop on WordNet and Other lexical Resources: Applications, Extensions and Customizations*, 2001.

(Alfonseca & Manandhar 02) E. Alfonseca and S. Manandhar. Extending a lexical ontology by a combination of distributional semantics signatures. In *Knowledge Engineering and Knowledge Management*, volume 2473 of *Lecture Notes in Artificial Intelligence*, pages 1–7. Springer Verlag, 2002.

(Banerjee & Pedersen 03) P. Banerjee and T. Pedersen. Using measures of semantic relatedness for word sense disambiguation. In *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*, Mexico City, February 2003.

(Berland & Charniak 99) M. Berland and E. Charniak. Finding parts in very large corpora. In *Proceedings of ACL-99*, 1999.

(Budanitsky & Hirst 01) A. Budanitsky and G. Hirst. Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures. In *Workshop on WordNet and Other Lexical Resources, Second meeting of NAACL*, Pittsburgh, 2001.

(Cimiano & Staab 04) P. Cimiano and S. Staab. Clustering concept hierarchies from text. In *Proceedings of LREC-2004*, 2004.

(Cruse 86) D. A. Cruse. *Lexical Semantics.* Cambridge University Press, 1986.

(Curran 05) J. Curran. Supersense tagging of unknown nouns using semantic similarity. In *Procs. of ACL'05*, pages 26–33, 2005.

(Edmonds & Hirst 02) P. Edmonds and G. Hirst. Near synonymy and lexical choice. *Computational Linguistics*, 2002.

(Faure & Nédellec 98) D. Faure and C. Nédellec. A corpus-based conceptual clustering method for verb frames and ontology acquisition. In *LREC workshop on Adapting lexical and corpus resources to sublanguages and applications*, Granada, Spain, 1998.

(Finkelstein-Landau & Morin 99) M. Finkelstein-Landau and E. Morin. Extracting semantic relationships between terms: supervised vs. unsupervised methods. In *Proceedings of the International Workshop on Ontologial Engineering on the Global Information Infrastructure*, 1999.

(Firth 57) J. Firth. *Papers in Linguistics 1934-1951.* Oxford University Press, London, 1957.

(Freitag *et al.* 05) D. Freitag, M. Blume, J. Byrnes, E. Chow, S. Kapadia, R. Rohwer, and Z. Wang. New experiments in distributional representations of synonymy. In *Proceedings of CoNLL-2005*, pages 25–32, 2005.

(Gruber 93) T. R. Gruber. A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2):199–220, 1993.

(Hahn & Schnattinger 98) U. Hahn and K. Schnattinger. Towards text knowledge engineering. In *AAAI/IAAI*, pages 524–531, 1998.

(Harabagiu & Moldovan 98) A. M. Harabagiu and D. I. Moldovan. *Knowledge Processing. In (C. Fellbaum (Ed.) WordNet: An Electronic Lexical Database*, pages 379–405. MIT Press, 1998.

(Hastings 94) P. M. Hastings. *Automatic acquisition of word meaning from context.* University of Michigan, Ph. D. Thesis, 1994.

(Hearst 92) M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of COLING-92*, Nantes, France, 1992.

(Hearst 98) M. A. Hearst. *Automated Discovery of WordNet Relations. In Christiane Fellbaum (Ed.) WordNet: An Electronic Lexical Database*, pages 132–152. MIT Press, 1998.

(Jarmasz & Szpakowicz 03) M. Jarmasz and S. Szpakowicz. Roget's thesaurus and semantic similarity. In *Proceedings of RANLP-03*, 2003.

(Landauer & Dumais 97) T. K. Landauer and S. T. Dumais. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review*, 104(2):211–240, 1997.

(Lee 97) L. Lee. *Similarity-Based Approaches to Natural Language Processing.* Ph.D. thesis. Harvard University Technical Report TR-11-97, 1997.

(Lin & Zhao 03) D. Lin and S. Zhao. Identifying synonyms among distributinally similar words. In *Proceedings of the IJCAI-2003 Conference*, pages 1492–1493, 2003.

(Lin 97) C.-Y. Lin. *Robust Automated Topic Identification.* Ph.D. Thesis. University of Southern California, 1997.

(Lyons 61) J. Lyons. *A structural theory of semantics and its applications to lexical sub-systems in the vocabulary of Plato.* Ph. D. thesis, University of Cambridge, England. Published as *Structural Semantics*, No. 20 of the Publications of the Philological Society, Oxford, 1963, 1961.

(Miller 95) G. A. Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

(Navigli & Velardi 04) R. Navigli and P. Velardi. Learning domain ontologies from document warehouses and dedicated websites. *Computational Linguistics*, 30(2), 2004.

(Pekar & Staab 03) V. Pekar and S. Staab. Word classification based on combined measures of distributional and semantic similarity. In *Proceedings of Research Notes of the 10th Conference of the European Chapter of the Association for Computational Linguistics*, Budapest, Hungary, 2003.

(Rajman & Bonnet 92) M. Rajman and A. Bonnet. Corpora-based linguistics: new tools for natural language processing. In *1st Annual Conference of the Association for Global Strategic Information*, Germany, 1992. Bad Kreuznach.

(Resnik 93) P. Resnik. *Selection and Information: A Class-Based Approach to Lexical Relationships.* Ph.D. thesis. Dept. of Computer and Information Science, University of Pennsylvania, 1993.

(Resnik 99) P. S. Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11:95–130, 1999.

(Richardson *et al.* 98) S. D. Richardson, W. B. Dolan, and L. Vanderwende. MindNet: acquiring and structuring semantic information from text. In *Proceedings of COLING-ACL'98*, volume 2, pages 1098–1102, Montreal, Canada, 1998.

(Rigau 98) G. Rigau. *Automatic Acquisition of Lexical Knowledge from MRDs.* PhD Thesis, Departament de Llenguatges i Sistemes Informàtics.– Universitat Politècnica de Catalunya. – Barcelona, 1998.

(Ruiz-Casado *et al.* 05) M. Ruiz-Casado, E. Alfonseca, and P. Castells. Automatic extraction of semantic relationships for wordnet by means of pattern learning from wikipedia. In *Proceedings of NLDB-05*, 2005.

(Salton 89) G. Salton. *Automatic text processing.* Addison-Wesley, 1989.

(Terra & Clarke 03) E. Terra and C. L. A. Clarke. Frequency estimates for statistical word similarity measures. In *Proceedings of HLT/NAACL-2003*, pages 244–251, 2003.

(Turney 01) P. D. Turney. Mining the web for synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings of the 12th European Conference on Machine Learning (ECML-2001)*, pages 491–502, 2001.

(Turney *et al.* 03) P. D. Turney, M. L. Littman, J. Bigham, and V. Schnayder. Combining independent modules to solve multi-choice synonym and analogy problems. In *Proceedings of RANLP-03*, 2003.

(Wilks *et al.* 90) Y. Wilks, D. Fass, C. Ming Guo, J. McDonald, T. Plate, and B. Slator. Providing machine tractable dictionary tools. *Journal of Computers and Translation*, 2, 1990.

(Yarowsky 92) D. Yarowsky. Word-Sense Disambiguation using statistical models of Roget's categories trained on large corpora. In *Proceedings of COLING-92*, pages 454–460, France, 1992.

Lexico-Syntactic Subsumption for Textual Entailment

Vasile Rus and **Art Graesser** and **Kirtan Desai**
Department of Computer Science
Department of Psychology
Institute of Intelligent Systems
The University of Memphis
Memphis, TN 38152, USA
{vrus, a-graesser}@memphis.edu

Abstract

In this paper a graph-based approach to the task of textual entailment between a Text and Hypothesis is presented. The approach takes into account the full lexico-syntactic context of both the Text and Hypothesis and relies heavily on the concept of subsumption. It starts with mapping the Text and Hypothesis into graph-structures where nodes represent concepts and edges represent lexico-syntactic relations among concepts. Based on a subsumption score between the Text-graph and Hypothesis-graph an entailment decision is made. A novel feature of our approach is the handling of negation. The results obtained from a standard entailment test data set are better than results reported by systems that use similar knowledge resources. We also found that a tf-idf approach performs close to chance for sentence-like Texts and Hypotheses.

1 Introduction

Recognizing textual entailment (RTE) (Dagan *et al.* 05) is the task of deciding, given two text fragments, whether the meaning of one text is entailed (can be strongly inferred) from another text (Dagan *et al.* 05). We say that T (the entailing text) entails H (the entailed hypothesis). The task is relevant to a large number of applications, including machine translation, question answering, and information retrieval.

In this paper we present a novel approach to RTE that uses minimal knowledge resources. The approach relies on lexico-syntactic information and synonymy specified in a thesaurus. The results obtained are better than state-of-the-art solutions that use same array of resources.

In our approach each T-H pair is first mapped into two graphs, one for H and one for T, with nodes representing main concepts and edges indicating dependencies among concepts as encoded in H and T, respectively. An entailment score, $entail(T, H)$, is then computed that quantifies the degree to which the T-graph subsumes the H-graph. The score is so defined to be non-reflexive, i.e. $entail(T, H) \neq entail(H, T)$. We evaluate the approach on the standard RTE challenge data (Dagan *et al.* 05).

The application of this work to Intelligent Tutoring Systems (ITS), namely AutoTutor (Graesser *et al.* 04)(Graesser *et al.* 01), is under investigation. AutoTutor is an ITS that works by having a conversation with the learner. AutoTutor appears as an animated agent that acts as a dialog partner with the learner. The animated agent delivers AutoTutor's dialog moves with synthesized speech, intonation, facial expressions, and gestures. Students are encouraged to articulate lengthy answers that exhibit deep reasoning, rather than to recite small bits of shallow knowledge. It is in the deep knowledge and reasoning part where we explore the potential help of entailment.

The rest of the paper is structured as follows. Section 2 outlines related work. Section 3 describes our approach in detail. Section 4 presents the experiments we performed, results and a comparative view of different approaches. Discussion and Conclusions wrap up the paper.

2 Related Work

The task of textual entailment has been recently treated, in one form or another, by research groups ranging from informational retrieval to language processing leading to a large spectrum of approaches from tf-idf to knowledge-heavy.

In one of the earliest explicit treatments of entailment (Monz & deRijke 01) proposed a weighted bag of words approach. They argued that traditional inference systems based on first order logic are limited to yes/no judgements when it comes to entailment tasks whereas their approach delivered "graded outcomes". They established entailment relations among larger pieces of text (4 sentences on average) than the proposed RTE setup where the text size is a sentence (seldom two) or part of a sentence (phrase). We replicated their approach, for comparison purposes,

Pair ID	Type	Content	Solution
2132	Text	Ralph Fiennes, who has played memorable villains in such films as 'Red Dragon' and 'Schindler's List,' is to portray Voldemort, the wicked warlock, in the next Harry Potter movie.	remote dependencies, lexical relations, paraphrasing
	Hypo	*Ralph Fiennes will play Harry Potter in the next movie.*	
1981	Text	The bombers had **not** managed to enter the embassy compounds.	remote dependencies, negation
	Hypo	*The bombers entered the embassy compounds.*	
878	Text	A British oil executive was one of 16 people killed in the Saudi Arabian terror attack.	remote dependencies
	Hypo	*A British oil company executive was killed in the terrorist attack in Saudi Arabia*	

Table 1: Examples of text-hypothesis pairs from Recognizing Textual Entailment (RTE) Challenge.

and report its performance on the RTE test data.

Pazienza and colleagues (Pazienza *et al.* 05) use a syntactic graph distance approach for the task of textual entailment. Among the drawbacks of their work, as compared to ours, is lack of negation handling. Additionally, they ignore the importance of the threshold t that represents the cut-off point above which entailment holds. We show that this threshold can play an important role in the overall score.

Recently, Dagan and Glickman (Dagan & Glickman 04) presented a probabilistic approach to textual entailment based on lexico-syntactic structures. They use a knowledge base with entailment patterns and a set of inference rules. The patterns are composed of a pattern structure (entailing template $\rightarrow$ entailed template) and a quantity that tells the probability that a text which entails the entailing template also entails the entailed template. This is a good example of a knowledge intensive approach.

A closely related effort, although not labeled as entailment (probably because it was ahead of the RTE era), is presented in (Moldovan & Rus 01). They show how to use unification and matching to address the answer correctness problem. Answer correctness can be viewed as entailment: Is a candidate answer entailing the ideal answer to the question? Initially, the question is paired with an answer from a list of candidate answers. The resulting pair is mapped into a first-order logic representation and a unification process between the question and the answer follows. As a back-off step, for the case when no full unification is possible, the answer with highest unification score is top ranked. The task they describe is different than the RTE task because a list of candidate answers to rank are available. The granularity of candidate answers and questions is similar to the RTE data.

3 Approach

The task of entailment requires an arsenal of language processing components. Table 1 shows some examples from the RTE data sets and what type of knowledge would be necessary to solve them. The first column indicates the pair id as assigned by RTE Challenge organizers. The second column contains the type of the text fragment which is pasted in the third column. The last column provides clues on what information we need (beyond bag-of-words) in order to recognize the entailment for the pair.

Our solution for recognizing textual entailment is based on the idea of subsumption. In general, an object X subsumes an object Y if X is more general than or identical to Y, or alternatively we say Y is more specific than X. The same idea applies to more complex objects, such as structures of interrelated objects. Applied to textual entailment, subsumption translates into the following: hypothesis H is entailed from text T if and only if text T subsumes H.

The two text fragments involved in a textual entailment decision are initially mapped into a graph representation that has its roots

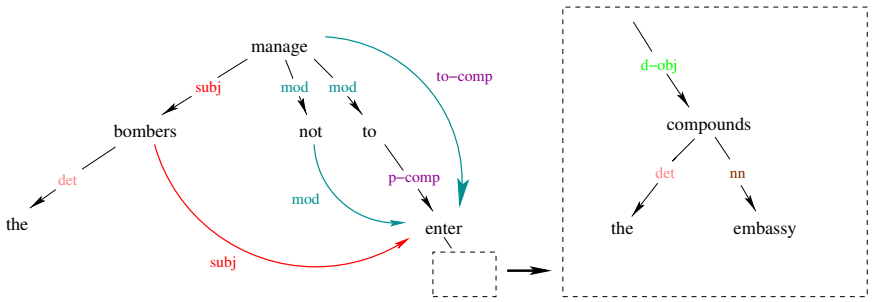


Figure 1: Example of dependency graph for test pair #1981 from rte. (Edges are in grayscale to better visualize the correspondence.)

in the dependency-graph formalisms of (Hays 64)(Mel'cuk 98). The mapping process has three phases: preprocessing, dependency graph generation and final graph generation.

In the preprocessing phase we do tokenization, lemmatization, part-of-speech tagging and parsing. The preprocessing continues with a step in which parse trees are transformed in a way that helps the graph generation process in the next phase. For example, auxiliaries and passive voice are eliminated but their important information is kept: voices are marked as additional labels to the verb tags, while aspect information (derived from modals such as *may*) is recorded as an extra marker of the node generated for a particular verb. An important step, part of the preprocessing phase, indentifies major concepts in the input: named entities, collocations, postmodifiers, existentials, etc. If we represented named entities composed of multiple words (e.g. *Overture_Services_Inc*) as a single concept, we would be in trouble when only a subset of the words is used in the other text fragment (*Overture*). To avoid this kind of problem, collocations in input are represented as a single concept by replacing the consecutive words forming a collocation with a new concept composed of the individual words glued with an underscore. A dictionary of collocations (compiled from WordNet) and a simple algorithm help us detect collocations in the input.

The mapping from text to the graph-representation is based on information from parse trees. We use Charniak's (Charniak 00) parser to obtain parse trees and head-detection rules (Magerman 94) to obtain the head of each phrase. A dependency tree is generated by linking the head of each phrase to its modifiers in a straightforward mapping step. The problem with the dependency tree is that it only encodes local dependencies (head-modifiers). Remote dependencies, as the subject relation between *bombers* and *entered* in Figure 1, are not marked in such dependency trees. An extra step transforms the previous dependency tree into a dependency graph (top part of Figure 2) in which remote dependencies are explicitly marked. Furthermore, the dependency graph is transformed into a final graph in which direct relations among content words are coded (bottom part of Figure 2). For instance, a *mod* dependency between a noun and its attached preposition is replaced by a direct dependency between the prepositional head and prepositional object.

The remote dependencies are obtained using a naive-bayes functional tagger (Rus & Desai 05). The naive bayesian model relies on more than a dozen linguistic features automatically extracted from parse trees (phrase label, head, part of speech, parent's head, parent's label, etc.). The model was trained on annotated data from Wall Street Journal section of Penn Treebank (Marcus *et al.* 93).

As soon as the graph representation is obtained a graph matching operation is initialized. A node in the Hypothesis is paired with a node in Text. If no direct matching (called *identity match*) is possible we use synonyms from WordNet. Nodes in the Text graph that have corresponding entries in WordNet (i.e. nodes for content words) are mapped to all possible senses in the database and the matching continues with comparing all the words in a given synset to the to-be-matched Hypothesis node. Possible multiples matchings are solved using criteria elaborated later (see Section 3.2).

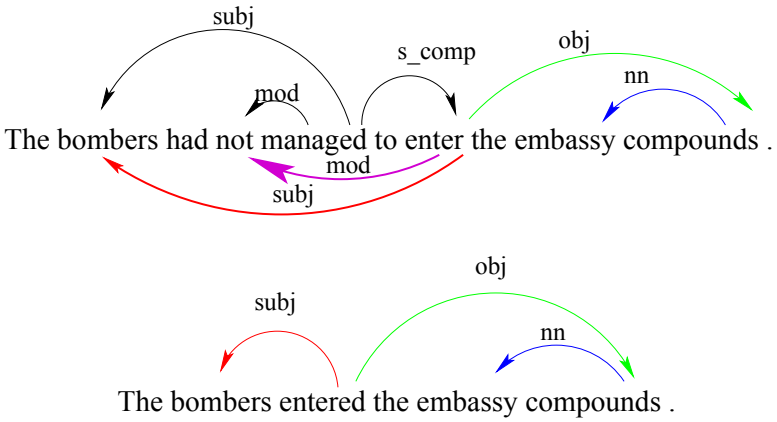


Figure 2: Example of graph representation for test pair #1981 from RTE.(Edges are in grayscale to better visualize the correspondence.)

3.1 Graph Matching and Subsumption

A graph $G = (V, E)$ consists of a set of nodes or vertices V and a set of edges E. Graphs are important since they can represent any relationship. Graphs can model the country's highway system with cities as nodes and routes between cities as edges, or electrical and electronic circuits with wire-crossings as vertices and components as edges (Skiena 98). We use graphs to model the linguistic information embedded in a sentence: nodes represent concepts and edges represent syntactic relations among concepts. Furthermore, we map the textual entailment problem into a graph isomorphism problem: the Text entails the Hypothesis if the hypothesis graph is contained or subsumed by the text graph.

Isomorphism in graph theory is the problem of testing whether two graphs are really the same (Skiena 98). Several variations of graph isomorphism exist in practice of which the subsumption or containment problem best fits our task. Is graph H contained in (not identical) to graph T? Graph subsumption consists of finding a mapping from vertices in H to T such as edges among nodes in H hold among mapped edges in T. In our case the problem can be further relaxed: attempt a subsumption and if not possible back-off to a partial subsumption. The important aspect is to quantify the degree of subsumption of H by T.

The subsumption algorithm for textual entailment has three major steps: (1) find an isomorphism between H_v (set of vertices of the Hypothesis graph) and T_v, (2) check whether the labeled edges in H, E_H, have correspondents in E_T, and (3) compute score. Step 1 is more than a simple word-matching method since if a node in H doesn't have a direct correspondent in T a thesaurus is used to find all possible synonyms for nodes in T. Nodes in H have different priorities: head words are most important followed by modifiers. Modifiers that indicate negation are handled separately from the bare lexico-syntactic subsumption since if H is subsumed at large by T, and T is not negated but H is, or vice versa (see example in Figure 2), the overall score should be dropped with high confidence to indicate no entailment. Step 2 takes each relation in H and checks its presence in T. It is augmented with relation equivalences among appositions, possessives and linking verbs (*be, have*). Lastly, a normalized score for node and edge mapping is computed. The score for the entire entailment is the sum of each individual node and relation matching score. The node match consists of lexical matching and aspect matching (for verbs). The aspect matching is not yet integrated. The overall score is adjusted after considering negation relations. More details on scoring are found in Section 3.5. The formula to compute the score is given by Equation 1.

3.2 Lexical Matching

A component of the overall score (see Equation 1) counts for lexical matching. Each concept in H is ideally mapped to a unique node in T. The ideal situation seldom occurs and one needs to find alternative ways to determine best matchings. Table 2 lists categories of lexical matchings that need to be accounted for. Perfect lexical matching occurs when one node V_h in H_v, has an identity match V_t in T_v and all dependencies of

Pair ID	Concept in H	Concept in T
864	eight	8
864	Qazvin province	province of Qazvin
116	teens	teen-age girls and young women
n/a	Overture services inc	Overture

Table 2: Examples of lexical matchings.

V_h are among the dependencies of V_t. An identity or direct match corresponds to a word with same base form and same part of speech. Penalties are paid for different deviations from the ideal case, for instance for different parts of speech.

Let us take a closer look at the lexical matching step. Although we already specified our choice for some of the following issues, we provide now evidence that justifies the choices. An important issue regards what concepts in the input sentences are to be mapped as nodes into graphs and thus considered for node matching. For instance, should auxiliaries be ignored? This is different from the multi-word concept issue discussed earlier. In all our experiments we lemmatize input sentences, ignore auxiliaries and discard modals but keep their impact, as an extra tag, on the main verb they modify, for later processing. These decisions were based on analyses of the development data. For instance, we compared the degree of lexical matching of T-H pairs with and without auxiliaries and modals, respectively. The number of fully matched pairs (all nodes in H-graph can be mapped in T-graph) increased by less than 1% (0.97% to be precise) if auxiliaries and modals are dropped. The number of *only-one* lexical-mismatches (all words but one are matched) almost does not change (0.963% difference). The impact of lemmatization is significant: full matching increases by more than 100% while only-one lexical mismatch increases by 61%.

Once we have the nodes to be matched all we need to do is the matching. This process starts with nouns and verbs and continues with their modifiers. An identical match is initially attempted with two possible problems: multiple direct matches or no direct match. The former case is handled by using syntactic information: if a node has multiple matchings, then choose the match that has similar syntactic relations. This is important since almost half of the T-H pairs exhibit this type of problem. For the latter problem, synonymy relations in WordNet are used. A hypothesis concept is matched against all synonyms of a text concept as indicated by its senses in WordNet. If one of its synonyms matches the hypothesis node, we end up with a successful match at the expense of a small penalty on the lexical score to indicate no direct match was found and that the match could be through a wrong sense in WordNet - explained next. There might be the case that more than one synonym (from different senses of same word) match the hypothesis node. The solution is to choose the synonym belonging to the most frequent sense because this choice leads to the best (most probable) interpretation of the input.

Verbs have some peculiarities that need to be addressed when we apply the previous node-matching methodology. In particular, one needs to pay attention to the impact of auxiliaries and modals (*will* is an auxiliary for us, although some linguistic theories classify it as a modal). We plan to use in the future an entailment table for verbs. In the table there are two rows and two columns for each modal - one indicating verbs accompanied by modals and another indicating unaccompanied verbs. Each cell indicates if the row entails the column. For instance, the cell corresponding to row *can verb* (indicating verb phrases where the modal *can* is followed by a *verb*) and column *verb* contains **does not entail**, while vice versa is true entailment and thus the mirrored cell will read **does entail**.

3.3 Dependency Matching

Syntactic information is extremely important for textual entailment as shown by the example below, where all concepts in H have a match in T but one syntactic relation is missing: the preposition head - preposition object relation between *capital* and *France* in H cannot be found in T.
T: *Besancon is the capital of France's watch and clock-making industry and of high precision engineering .*
H: *Besancon is the capital of France .*

$$entscore(T,H) = (\alpha \times \frac{\sum_{V_h \in H_v} max_{V_t \in T_v} match(V_h, V_t)}{|V_h|} + \beta \times \frac{\sum_{E_h \in H_e} max_{E_t \in T_e} synt\_match(E_h, E_t)}{|E_h|} + \gamma) \times \frac{(1 + (-1)^{\#neg\_rel})}{2} \quad (1)$$

A perfect syntactic score is assigned to a H-T pair in which all dependencies in H, denoted as the set of edges E_h of H-graph, are found in E_t. The label of the edge and its nodes need to be matched for a perfect score. The syntactic score is the sum of individual scores for each dependency divided by the number of dependencies in E_h.

3.4 Negation

We look at two broad types of negation: explicit and implicit. In this work, we treated only explicit negation. Explicit negation is indicated by particles such as: *no, not, neither ... nor* and their shortened forms *'nt*. Implicit negation is present in text via deeper lexico-semantic relations among different linguistic expressions. The most obvious example is the *antonymy* relation among lemmas which can be retrieved from WordNet. Negation is regarded as a feature of both Text and Hypothesis and it is accounted for in the score after the entailment decision for the Text-Hypothesis pair without negation is made. If one of the text fragments is negated, the decision is reversed; if both are negated the decision is retained (double-negation). In Equation 1 the term *#neg_rel* represents the number of negation relations in T and H.

3.5 The Scoring

The formula to compute the overall score is provided in Equation 1. The weights of lexical and syntactic matching are given by parameters α and β, respectively. The last term of the equation represents the effect of negation on the entailment decision. An odd number of negation relations between T and H, denoted *#neg_rel*, would lead to an entailment score of 0 while an even number will not change the bare lexico-semantic score. The choice of α, β and γ can have a great impact on the overall score. The Experiments and Results section talks about how to estimate those parameters. From the way the score is defined, it is obvious that $entscore(H,T) \neq entscore(T,H)$.

4 Experiments and Results

Before we present results on several baselines and of our approach let us look at the experimental setup as defined by RTE.

4.1 Experimental Setup

The dataset of text-hypothesis pairs was collected by human annotators and it is described in (Dagan *et al.* 05). It is reportedly a mix of seven subsets, which correspond to success and failure examples in different applications: Question Answering, Information Retrieval, Comparable Documents, Reading Comprehension, Paraphrase Acquisition, Information Extraction, and Machine Translation. Within each application setting, the annotators selected both positive entailment examples (judged as TRUE), where T does entail H, as well as negative examples (FALSE), where entailment does not hold (50%-50% split).

The evaluation is automatic. The judgements returned by the system are compared to those manually assigned by the human annotators (the gold standard). Two measures were proposed: accuracy and Confidence-Weighted Score. The percentage of matching judgements provides the accuracy of the run, i.e. the fraction of correct responses. The Confidence-Weighted Score (CWS, also known as average precision) is computed by sorting the judgements of the test examples by their confidence (in decreasing order from the most certain to the least certain) and calculating the following:

$$\frac{1}{n} * \sum_{i=1}^{n} \frac{\#-correct-up-to-pair-i}{i} \quad (2)$$

n is the number of the pairs in the test set, and i ranges over the pairs. The Confidence-Weighted Score varies from 0 (no correct judgements at all)

to 1 (perfect score), and rewards the systems' ability to assign a higher confidence score to the correct judgements than to the wrong ones.

4.2 Results

We started our evaluation by considering two baselines: a lexical overlap method and the approach presented in (Monz & deRijke 01).

It is arguable what the baseline for entailment is. In (Dagan *et al.* 05), the first suggested baseline is the method of blindly and consistently guessing TRUE or FALSE for all test pairs. Since the test data was balanced between FALSE and TRUE outcomes, this blind baseline would provide an average accuracy of 0.50. Randomly predicting TRUE or FALSE is another blind method that leads to a run being better than chance for (cws>0.540)/(accuracy>0.535) at the 0.05 level or for a run with (cws>0.558)/(accuracy>0.546) at the 0.01 level.

We experimented here with more informed baselines. The first baseline we used is the lexical overlap: tokenize, lemmatize (using *wnstemm* in wn library), ignore punctuation and compute the degree of lexical overlap between H and T. We normalized the result by dividing the lexical overlap by the total number of words in H. Then if the normalized score is greater than 0.5, we assign a TRUE value meaning T entails H, otherwise we assign FALSE. The normalized score also plays the role of confidence score necessary to compute the CWS metric. The results (first row in Table 3) for CWS and accuracy are close to chance, a possible suggestion that the test corpus is balanced in terms of lexical overlap. The precision (only accounting for positive entailment cases) of 0.6111 on this lexical baseline method may indicate that higher lexical matching may be a good indicator of positive entailment. The second informed baseline is the approach presented in (Monz & deRijke 01), mentioned earlier. We decided to apply it to the RTE data to compare a pure word-level statistical method to our method and also to see to what extent tf-idf fits RTE-like data. RTE uses sentence-like Hs and Ts as opposed to paragraphs in (Monz & deRijke 01). A larger context, with more words in both H and T, can favor a word-level statistical method. Briefly, tf-idf uses *idf* (inverted document frequency) as a measure of word importance, or *weight*, in a document. The idf weights are derived from the development data and then an entailment score is computed according to the equation below.

$$entscore(t,h) = \frac{\sum_{t_k \in (t \cap h)} idf_k}{\sum_{t_k \in h} idf_k} \qquad (3)$$

Every score below a certain threshold leads to a false entailment and everything above leads to true entailment. We obtained the optimal threshold from different runs with different thresholds (0.1, 0.2, ..., 0.9) on the development data. The results for the test data presented in the second row in Table 3 are from the run with the optimal threshold.

The third row in the table shows the results on test data obtained with the proposed graph-based method. Initially we used linear regression to estimate the values of the parameters but then switched to a balanced weighting ($\alpha = \beta = 0.5$, $\gamma = 0$) which provided better results on development data. Depending on the value of the overall score three levels of confidence are assigned: 1, 0.75, 0.5. For instance, an overall score of 0 leads to FALSE entailment with maximum confidence of 1. The results reported on test data are significant at 0.01 level.

The bottom rows in Table 3 replicate, for comparison purposes, the results of systems that participated in the RTE Challenge (Dagan *et al.* 05). We picked the best results (some systems report results for more than one run) for runs that use similar resources to us: word overlap, WordNet and syntactic matching.

5 Discussion

There are two major aspects of textual entailment of two sentences that make it harder than other tasks. First, it is a fine-precision task that demands an absolute answer as opposed to, for example, the answer correctness task in QA where the best answer among a set of candidates is picked. Second, Textual Entailment is highly dependent on domain specific knowledge which is vaguely present in language resources of the kind we used in this work.

A quick error analysis suggests treating dependencies individually, weighting certain dependencies more than others. For instance, modifier-modifee relations should have a smaller impact on the overall outcome than a subject relation.

system	cws	accuracy
baseline	0.543	0.538
idf-baseline	0.497	0.505
graph-based	0.604	0.554
Zanzotto (Rome-Milan)	0.557	0.524
Punyakanok	0.569	0.561
Andreevskaia	0.519	0.515
Jijkoun	0.553	0.536

Table 3: Performance and comparison of different approaches on RTE test data.

6 Conclusions

We presented in this paper a lexico-syntactic approach to textual entailment. As compared to a tf-idf approach it performs significantly better and also shows better results than systems that use the same array of resources. A tf-idf scheme is not particularly suitable for the RTE-like entailment task due to data sparseness and the need to perform deeper language processing to capture finer nuances of language.

ACKNOWLEDGEMENTS

This research was partially funded by The University of Memphis and AutoTutor project. The research on AutoTutor was supported by the National Science Foundation (REC 106965, ITR 0325428) and the DoD Multidisciplinary University Research Initiative (MURI) administered by ONR under grant N00014-00-1-0600. Any opinions, findings, and conclusions or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of The University of Memphis, DoD, ONR, or NSF. We are also grateful to three anonymous reviewers for their valuable comments.

References

(Charniak 00) E. Charniak. A maximum-entropy-inspired parser. In *Proceedings of North American Chapter of Association for Computational Linguistics (NAACL-2000)*, Seattle, WA, April 29 - May 3 2000.

(Dagan & Glickman 04) I. Dagan and O. Glickman. Probabilistic textual entailment: Generic applied modeling of language variability. In *Proceedings of Learning Methods for Text Understanding and Mining*, Grenoble, France, January 26 - 29 2004.

(Dagan *et al.* 05) I. Dagan, O. Glickman, and B. Magnini. The PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the Recognizing Textual Entaiment Challenge Workshop*, Southampton, U.K., April 11 - 13 2005.

(Graesser *et al.* 01) A.C. Graesser, K. VanLehn, C. Rose, P. Jordan, and D. Harter. Intelligent tutoring systems with conversational dialogue. *AI Magazine*, 22:39–51, 2001.

(Graesser *et al.* 04) A.C. Graesser, S. Lu, G.T. Jackson, Mitchell, H., M. Ventura, A. Olney, and M.M. Louwerse. Autotutor: A tutor with dialogue in natural language. *Behavioral Research Methods, Instruments, and Computers*, pages 180–193, 2004.

(Hays 64) D. Hays. Dependency theory: a formalism and some observations. *Language*, 40:511–525, 1964.

(Magerman 94) D.M. Magerman. *Natural Language Parsing as Statistical Pattern Recognition*. Unpublished PhD thesis, Stanford University, February 1994.

(Marcus *et al.* 93) M. Marcus, B. Santorini, and Marcinkiewicz. Building a large annotated coprus of english: the penn treebank. *Computational Linguistic*, 19(2):313–330, 1993.

(Mel'cuk 98) I.A. Mel'cuk. *Dependency Syntax: theory and practice*. State University of New York Press, Albany, NY, 1998.

(Moldovan & Rus 01) D.I. Moldovan and V. Rus. Logic form transformation of wordnet and its applicability to question answering. In *Proceedings of the ACL Conference (ACL-2001)*, Toulouse, France, July 2001.

(Monz & deRijke 01) C. Monz and M. de Rijke. *Light-Weight Entailment Checking for Computational Semantics*, pages 59–72. 2001.

(Pazienza *et al.* 05) M.T. Pazienza, M. Pennacchiotti, and F.M. Zanzotto. Textual entailment as syntactic graph distance: A rule based and svm based approach. In *Proceedings of the RTE Challenge Workshop*, Southampton, U.K., April 11 - 13 2005.

(Rus & Desai 05) V. Rus and K. Desai. Assigning function tags with a simple model. In *Proceedings of Conference on Intelligent Text Processing and Computational Linguistics (CICLing) 2005*, Mexico City, Mexico 2005.

(Skiena 98) S.S. Skiena. *The Algorithm Design Manual*. Springer-Verlag, 1998.

Integrating NLP Tools to Support Information Access to News Archives

Horacio Saggion*, **Emma Barker***, **Robert Gaizauskas***

Jonathan Foster**

*Department of Computer Science - University of Sheffield
211 Portobello Street - Sheffield - S1 4DP - UK
{saggion,ejbarker,robertg}@dcs.shef.ac.uk

**Department of Journalism Studies - University of Sheffield
18-22 Regent Street - Sheffield - S1 3NJ - UK
j.foster@shef.ac.uk

Abstract

We describe Cubreporter, a project which investigates the use of advanced natural language processing techniques to enhance access to a news archive for the specific purpose of background writing. We describe the problem of background writing for a breaking news story and the requirement for advanced NLP tools. We focus on the description of the overall functionalities of our prototype and give an account of our methodology for evaluation.

1 Introduction

Cubreporter is a research project which investigates how language technologies might help journalists to access information in a news archive in the context of a background writing task. The function of background material is to support and contextualise a breaking news story. The specific characteristics of the background-writing scenario make recent advances in areas of natural language processing such as question answering and text summarization relevant to this task.

The main research questions we address in this project are: (i) what are the essential components of a background story and how does background information relate to the "foreground" breaking news story? (ii) how can background information for a breaking news story be accurately found in the archive given the initial breaking news story? (iii) how can human language technology assist a journalist to access the vast amount of information in a news archive? in particular can recent advances in NLP technologies, in areas such as question answering, summarisation, and information extraction, offer advantages in gathering background that standard information retrieval cannot? (iv) how is background writing quality affected by the use of human language technology?

To address these questions we have designed and implemented a prototype that incorporates a standard information retrieval engine as a baseline, as well as a question answering system and document summarization technology. Information extraction technology is also used to extract structured representations of events which are in turn used to populate a database to support similar event search. These information access technologies are embedded in a browser-based graphical user interface which allows users to combine them flexibly in an iterative information seeking process.

Here we give an overview of the project and describe our work on the background gathering task and the tools used to support it. The main contributions of the work are: (i) a descriptive theory characterising the nature of background in the news and its relation to the foreground news story; (ii) a design for an information access platform that integrates information retrieval, summarisation, question answering and information extraction capabilities within a single system operating over a text archive of significant size; (iii) a methodology for comparative evaluation of different combinations of language technologies for the task of background writing, allowing an assessment of the relative utility of more sophisticated natural language processing tools versus traditional information retrieval tools for the task of background writing.

The rest of the paper is organised as follows. In the following section we describe the task of writing background news. In Section 3, we describe the structure of the news archive. Section 4 gives an overview of the different NLP processes involved in the project. In Section 5, we describe our methodology for extrinsic evaluation. Section 6 closes with an account of work in progress and future developments. It should be noted that while we focus on the specific task of journalistic background writing, investigative intelligence gathering in response to a new event is by no means exclusive to the news-producing community and work described here is also relevant to information seeking professionals working in commercial, policing, military and scientific domains.

2 Writing Backgrounds

Our work to date has involved the study of journalists who either work for or with materials produced by the Press Association, the major UK domestic newswire

service which provides copy to all major national daily newspapers. While background figures in a number of ways, including simple descriptive phrases interjected into the current story (e.g. *former Chancellor of the Exchequer*) and fact sheets listing similar or relevant occurrences (e.g. a listing of previous train crashes), we shall focus on the most significant form of background material only, the so-called "backgrounder". Backgrounders are coherent documents, typically written when a news editor deems a particular story worthy of dedicated background material, but which can be read on their own, out of their production context. They are usually not released till sometime after a news story has broken as time is needed both to determine whether a story merits a backgrounder, but also for the research to be carried out to assemble the material. Their function is not to continue to report details of new events, but rather to provide text that supports and contextualises these events.

There has been no prior work, so far as we are aware, on gathering information for background writing. Attfield & Dowell (03) propose a general model of journalistic information gathering. However, the backgrounder task is different from other types of news writting and deserves special attention.

Interviews with journalists, observation during a controlled task and text analysis of a sizeable set of archived background stories show that backgrounds are composed of four types of material: (1) accounts of similar events in the past (e.g. other train crashes, scandals of similar nature, etc.); (2) accounts of events which have led up to the current event (e.g. a chronology of company takeovers, store openings, price cuts and profit warnings in the months leading up to a supermarket's announcement of low annual profits); (3) profiles of persons or organisations or locations (usually role players in the new event) comprising some highly structured factual information about the role player, for example date and place of birth, career appointments, spouse etc; accounts of the role player in events leading up to the event and accounts of the role player in similar events to the current event; and (4) comment (quotes) on any of the preceding by notable individuals.

Interestingly, these information gathering requirements are similar to those addressed in recent NLP challenge tasks. For example, finding profiles of people or organisations is a task dealt with in recent TREC Question Answering evaluations (Voorhees 04) and Document Understanding Conferences (Over & Yen 04) and can be supported by solutions proposed in these contexts. Finding events similar to one reported in breaking news can be implemented with information extraction technology: text in the archive could be mapped off-line into structured representations which could be stored in a database for on-line searching (Milward & Thomas 00). Question answering technology can be used to support fact gathering as well as fact checking in a background writing context. Consider as an illustration the news about the "kidnapping of UK-born Margaret Hassan". Of considerable importance for the UK public are answers to the following (among other) questions: *How many British citizens are living in Iraq?* and *Where was Margaret Hassan born?*. Techniques used on factoid question answering are relevant here.

3 The News Archive

Through our collaboration with the PA we have obtained access to 11 years of newswire copy from 1994 to 2004. The archive contains more than 8.5 million stories totalling 20GB of data. The raw corpus has been processed and encoded in XML following a strict Document Type Definition (DTD) specification which captures all meta-data delivered by the Press Association and which includes elements such as story date, category, topic, and structural information such as headlines, bylines, and paragraphs. One example story is shown in Figure 1. The archive is organised per dates following the logical organisation of the PA wire where years are composed of months, months are composed of days, and there are a number of stories per day. Stories in the PA archive are classified into a number of topics or news categories from a controlled vocabulary representing the subject matter of the story (e.g., *Courts*, *Politics*). Within the same topic, stories are further identified by a number of free-text keywords that the journalists would assign which are called *catch-lines*.

When a "news event" occurs, a reporter writes a *snap*, a line of text summarising the news and "moves" it to the wire. From that point on, stories follow an installment pattern where each installment carries an updated account of the story. Installments have names such as *snapfull*, one or two paragraph long text expanding the snap, *lead*, copy that summarises the major aspects of the story, and so on. These installment types reflect their position and significance in the publishing cycle of major newspapers.

Subscribers to the PA have access to the archive through the PA Digital Text Library and Mediapoint systems which have a number of functionalities for information access including: text, keyword, and topic search. Output to a specific query is presented as a ranked list of documents and associated 'lead' paragraphs. Access to the full document is done by following a link.

4 Advanced NLP Technology

One of the objectives of the project is to carry out experimentation in order to investigate the research questions identified in the introduction. In the future it might be possible to deploy, at least partially, the technology produced in this project in a real application to give journalists cutting-edge natural language

```
<?xml version="1.0"  encoding="ISO-8859-1"?>
<!DOCTYPE HSA SYSTEM "../../../../dtds/HSA.DTD">
<HSA  DATE="20042004"  DAY="20"  YEAR="2004"  MONTH="04"
ID="HSA7041"  PRIORITY="4" CATEGORY="NRG" COUNT="76"
MSGINFO="PA"  TOPIC="1 ROYAL Cockle Morecambe"
TIMEDATE="201407 APR 04">
<HEADLINE>COCKLE PICKERS RESCUED FROM NOTORIOUS SANDS
</HEADLINE>
<BODY>
<PARAGRAPH NRO="1"> Four cockle-pickers have been rescued
by lifeboatmen  after getting trapped on
the sands at Morecambe Bay.</PARAGRAPH>
<PARAGRAPH NRO="2"> A group of ten cocklers,
who were not Chinese,
were returning to Hest Bank on a tractor
which got stuck as the tide swept in.</PARAGRAPH>
<PARAGRAPH NRO="3"> Some of the group were washed off the
tractor  but managed to get to a rocky
outcrop called Priest Skier. The rest were rescued by
the RNLI Morecambe hovercraft.</PARAGRAPH>
</BODY>
</HSA>
```

Figure 1: Corpus Encoding

processing capabilities for information access. Cubreporter comprises an off-line *corpus processing* subsystem and an on-line *information access* subsystem (see Figure 2).

The off-line subsystem produces a *text index* for document retrieval, *generic summaries* at fixed length for each story, *generic multi-document summaries* for sets of known related stories, and *logical forms* for database population. The database is an entity-event-relation relational repository which stores the information resulting from a process of semantic interpretation of each story. The database contains tables to record references to entities (such as people and organisations), events, locations, and temporal information. Relations are a set of fixed logic relations including logical subject and object, apposition, qualification, etc. A table of attributes stores the different values that qualify entities and events such as adjectives, adverbials, and quantifiers.

The on-line system provides question answering, keyword search, similar-, and further ad hoc summarization capabilities.

4.1 Off-line processing

The whole archive is processed with tools adapted from the GATE Java library (Cunningham *et al.* 02). We perform tokenisation, sentence boundary identification, part-of-speech tagging, morphological analysis, and named entity recognition, keeping the results of the analysis for use by various language processing components. A text index is produced for the processed documents using Lucene[1], a Java-based open source tool for indexing and searching. The text of each story at textual and paragraph level as well as each metadata field are indexed. Search can be performed in any of the fields alone or in combination with boolean operators. Further linguistic processing of the archive is carried out with SUPPLE (Gaizauskas *et al.* 05), a freely-available parser, integrated in GATE, and with an in-house discourse interpreter. SUPPLE uses a feature-based context-free grammar in order to produce syntactic representations and logical forms. The grammar in use consists of a sequence of subgrammars for: noun phrases (NP), verb phrases (VP), prepositional phrases (PP), relative clauses (R) and sentences (S). The semantic rules produce unary predicates for entities and events and binary predicates for attributes and relations. Predicate names are: (i) the citation forms obtained during lemmatisation; (ii) forms used to code syntactic information (e.g. *lsubj* for the logical subject of a given verb); (iii) specific predicates are used to encode, for example, named entity information (e.g. *name* for the name of a person). The document semantics is further analysed by a discourse interpreter which maps entities into a discourse model and performs coreference resolution based on an ontology we are adapting for the purpose of this project. The results of this semantic discourse analysis is transformed into records that are used to populate the database. For example for the headline presented in Figure 1 an event of type `rescue` and two entities `cockle pickers` and `sand` would be created. A `patient` relation would be created between the event and the entity `cockle pickers` and a `from` relation would be created between the event and entity `sand`. We are currently looking at standard classification systems such as the Subject Code three level system for describing content produced by the International Press Telecommunications Council (`http://www.iptc.org`). This system is used to describe news content and seems appropriate for the creation of a Cubreporter ontology for news events and actors.

Summaries at fixed compression rate and ranked sentences (for on-line summary access) are computed for each story in the archive using an in-house sin-

[1] http://jakarta.apache.org/lucene

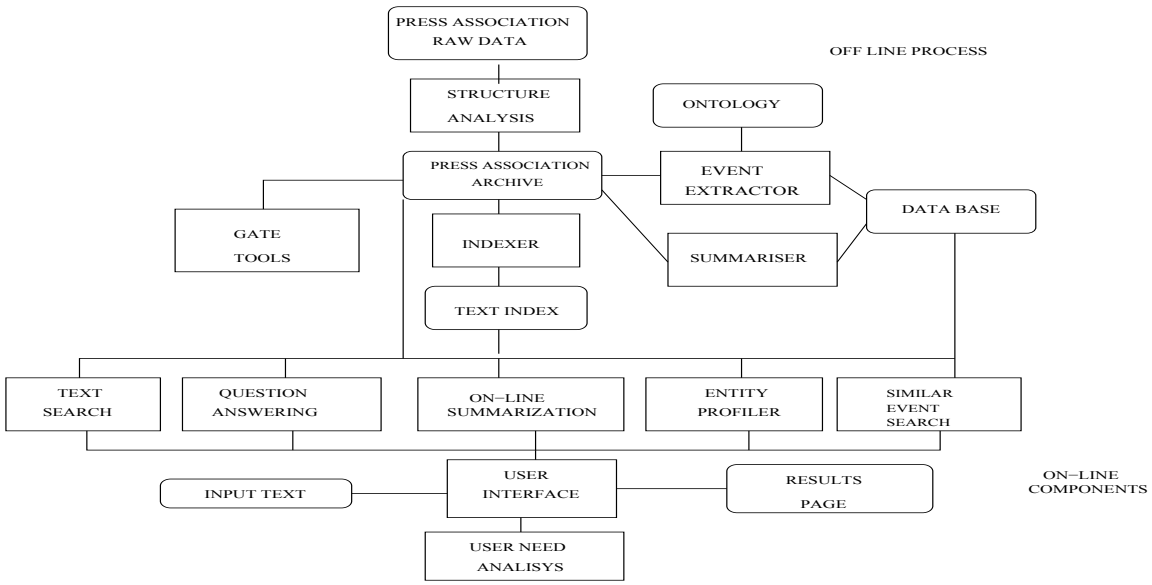


Figure 2: System Components

gle document summariser (Saggion 02). Sentences are ranked based on sentence-summary worthiness score obtained by combining scores for various features including sentence position, similarity of the sentence to the document headline, term distribution, named entity distribution, etc. Individual scores are combined using weights experimentally obtained from training corpus.

Off-line multi-document summarisation is carried out on a set of story-related documents. The tool extends the single document summariser by implementing a centroid-based summarisation system (Saggion & Gaizauskas 04a) which computes the similarity of each sentence to a cluster centroid and combines this value with single document summarization features. An n-gram similarity metric has been implemented to filter out redundant information, using a similarity threshold adjusted over training data. The weights used to combine the different features are trained over corpora.

4.2 On-line Processing

Access to the archive is through a user interface which is designed with the input text as its focus. The user enters a text which can be a sequence of keywords, a well formed natural language question, or a short snap-like text such as the initial report of a breaking news story. The system first carries out full text analysis of the fragment, and depending on the result of the analysis, additional options are made available including:

- access to full documents and summaries;
- answers and contexts to specific questions;
- profiles of persons, organisations, and locations;
- events similar to those described in the input.

Access to full documents and summaries In a pure document search situation – when the input text is a list of keywords – the journalist is presented with a results page containing access to full documents and to the previously computed story summaries, and installment multi-document summaries. The documents are ranked either by date or relevance – for the latter the standard `tf * idf` Lucene's default scoring mechanism is used. In addition query-focused summaries, tailored to the user's input text, are computed dynamically, in such a way that extracted sentences will be related to the user's assumed information need. Such summaries can be very effective when trying to identify the relevance of a document with respect to a query (Tombros *et al.* 98); generic sentence-summary worthiness features are combined with a query-based feature in a scoring function to obtain such summaries. A set of user-selected documents can be multi-document summarised on-line.

Question Answering Question Answering (QA) functionalities are used to provide the journalist with short, text units that answer their specific, well-formed natural language questions. We make use of a logic-based question answering system which given the logical form produced by the text analysis module, scores answer candidates in the database based on syntactic and semantic criteria (Gaizauskas *et al.* 03). Briefly, the scoring mechanism operates as follows. When the text analysis module finds a question, it produces an analysis which includes the expected answer type (EAT) and depending on the question, a special attribute created to refer to the attribute-value to be extracted from the answer entity. Each candidate answer gets a preliminary score according to (1) its semantic proximity to the EAT using WordNet and (2) the number of relations the candidate answer has with

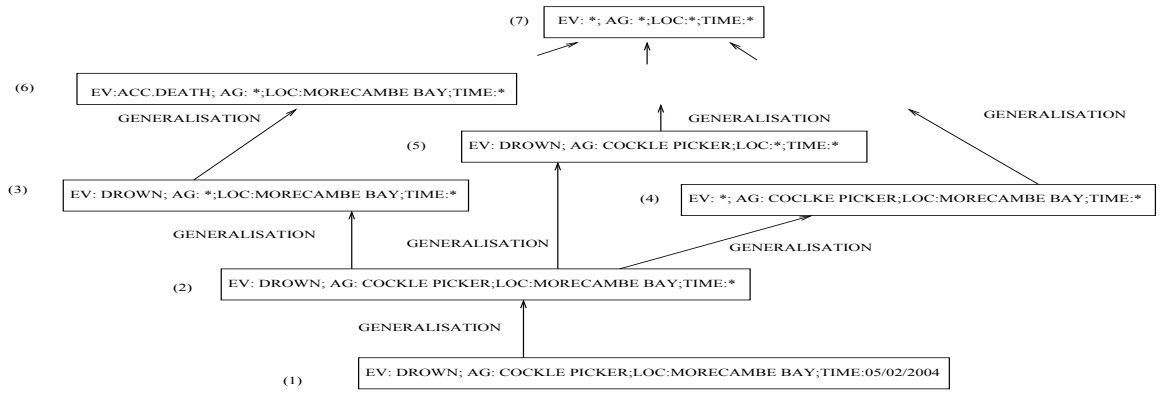


Figure 3: Similar : Generalisation

elements of the question. An overall score is computed for each entity as a function of its preliminary score plus a similarity value between the question and the sentence where the entity comes from (e.g., similar to word overlap).

Entity Profiles If the analysis of the input text recognises entities such as persons or organizations the system will return a profile. Tools for creating entity profiles are adapted from our definitional question answering system, developed for the TREC QA track, which uses pattern matching techniques against definition patterns to identify text fragments conveying profile information (Saggion & Gaizauskas 04b). Passages extracted from the collection are then filtered with the assistance of a similarity metric to avoid repetition of information.

Similar Event Search Our research into background news writing has shown that users are likely to be interested in past events similar to the new event that is the focus of the breaking news story. One strategy for extracting similar events is to use the IR component with the snap as a query in the hope that stories describing similar events will be returned at high ranks. However, this may be problematic as by definition the breaking story, to be "news", must be new and hence different in significant respects from previous events. Here, we propose a novel approach based on searching the database of extracted semantic representations of texts. Given a snap-like input text, a structured representation of the input is produced which includes a list of event-like representations which is then used to query the database.

Consider for example the following snap:

> *Eighteen Chinese Cockle pickers drowned in Morecambe Bay last night.*

The analysis of this text fragment produces the following template-like representation:

> Event: drown
> Agent: cockle picker
> Location: Morecambe Bay
> Time: 05/02/2004

In order to obtain similar events, this initial representation is transformed into a series of successively more general queries (see Figure 3). One possibility consists in replacing the time of the event by a wildcard: this will result in retrieving from the database "previous drownings in Morecambe Bay involving cockle pickers" (representation (2) in Figure 3). A further refinement would replace the agent of the event by a wildcard, resulting in a statement like "previous drownings in Morecambe Bay" (representation (3)). Yet, another possibility would be to replace the location of the event by a wildcard producing a statement like "previous drownings involving cockle pickers" (representation (5)). Yet another possibility would be to replace the actual arguments by generalisations: for example the type of event (drowning) could be replaced by other accidental deaths (using the information provided by the ontology as in (6) in Figure 3). Generalisations can be applied until all arguments have been replaced so as to effectively obtain "all events in the database."

The output of this process is a list of sentences from which each matching event was derived. For example, representation (3) in Figure 3 might return the following sentence:

> *Today a man drowned while he was walking his dog in Morecambe Bay ...*

We are currently investigating methods for presenting the results to the user, e.g., ranking and clustering.

4.3 Prototype Implementation

The off-line processing results in a Lucene inverted index, summaries and structured semantic representations of each story in the archive. The summaries and semantic representations are held in relational tables in a mySQL database. The user interacts with the system through a web client which communicates with a

web server (Tomcat). Both the text index and the relational database are accessed by the server as needed during on-line processing and web content is dynamically created for return to the client. A user database records details of users for security purposes and to allow search histories to recorded and revisited in subsequent sessions. Session management in the server allows multiple concurrent access to the archive.

5 Evaluation

A key criterion in evaluating our project is that of quality of the background stories which can be created by using the prototype. In order to measure in a scientific experiment whether new information technologies offer better access to information than conventional text search engines for the purpose of background information gathering, one has to articulate a theory of what constitutes a good background story. In order to address this issue we are following two complementary directions. First, we are investigating whether independent assessors can consistently rank and categorise backgrounds according to their quality. Secondly, we are working on a descriptive theory of background based on the semantic relation of content units in the background in relation to the breaking news event. This theory will help to predict the quality of a background as a function of its content and form. A pilot study has been conducted to investigate the first issue. The data for this study consisted of a collection of student assignments that were evaluated in terms of their respective quality by three independent journalist evaluators. Preliminary results reported elsewere (Barker & Gaizauskas 05) indicate reasonably high agreement among evaluators. Given the positive results of this experience, our plan is to construct a broader and more controlled corpus which will include different types of background written by professional journalists. In order to develop a theory of background, a set of relations is needed which indicate not only the relation between background and breaking news event, but also the relations between the different content units of the background.

We propose to adopt a framework such as that of Wolf and Gibson (Wolf & Gibson 04) or Marcu (Marcu 00) who have shown that it is possible to specify a set of discourse relations for text segments that are easy to code. Given such a descriptive framework a corpus of backgrounds will be annotated and experiments will be carried out to test the quality of background with respect to the descriptive theory.

While no extrinsic evaluation of the overall Cubreporter system has yet been carried out, some of the components have been evaluated. For example:

- the generic multi-document text summariser had a very good performance in DUC 2004, it was the second best system in task 2;
- the profile-based multi-document text summariser performed reasonably well in task 5 of DUC 2004 coming among the top nine participants. We have recently implemented a new method for extracting biographical information from text and obtained improved performance (Saggion & Gaizauskas 05);
- the QA system has participated in TREC/QA and in particular the definitional component placed fourth in 2004;
- in spite of the fact that our parser has never been formally evaluated, it has contributed to many successful information extraction projects in the past. We are currently assessing two approaches to evaluation: one is the evaluation of the logical forms produced by the parser using a resource such as Suzanne (Sampson 95), the other is to develop test suites for testing a range of grammatical phenomena and to support regression testing during grammar development.

While advanced NLP tools are far from perfect, they have the potential of offering improved access to news archives as compared with existing information access technologies, an hypothesis we are trying to validate.

6 Conclusion and Future Work

From a theoretical point of view, this work contributes with an in-depth examination of the background gathering task and with a methodological framework for extrinsic evaluation of information access systems.

From a technical point of view our work to date contributes to the creation, adaptation, and integration of NLP technology to support the task of background gathering. Much work has been done on specification and design of a web-based user interface and on in-house intrinsic evaluation of the different NLP components.

Current work involves the full integration of the NLP modules to carry out evaluation and testing of our research hypotheses.

Acknowledgements

We would like to thank four anonymous reviewers for their comments and suggestions. We acknowledge the support of the UK Engineering and Physical Sciences Research Council, research grant: R91465.

References

(Attfield & Dowell 03) S. Attfield and J. Dowell. Information seeking and use by newspaper journalists. *Journal of Documentation*, 59(2):187–204, 2003.

(Barker & Gaizauskas 05) E. J. Barker and R. Gaizauskas. Evaluating Cub Reporter: proposals for extrinsic evaluation of journalists using language technologies to access a news archive in background

research. In *Proceedings of the COLIS 2005 Workshop on Evaluating User Studies in Information Access*, 2005. To appear.

(Cunningham *et al.* 02) H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, 2002.

(Gaizauskas *et al.* 03) Robert Gaizauskas, Mark A. Greenwood, Mark Hepple, Ian Roberts, Horacio Saggion, and Matthew Sargaison. The University of Sheffield's TREC 2003 Q&A Experiments. In *Proceedings of the 12th Text REtrieval Conference*, 2003.

(Gaizauskas *et al.* 05) R. Gaizauskas, M. Hepple, H. Saggion, and M. Greenwood. SUPPLE: A Practical Parser for Natural Language Engineering Applications. In *International Workshop on Parsing Technologies*, 2005. Accepted.

(Marcu 00) D. Marcu. *The Theory and Practice of Discourse Parsing and Summarization.* MIT Press, Cambridge, Mass, 2000.

(Milward & Thomas 00) D. Milward and J. Thomas. From information retrieval to information extraction. In *Proceedings of the ACL Workshop on Recent Advances in Natural Language Processing and Information Retrieval*, 2000. Available at: http://www.cam.sri.com/html/highlight.html.

(Over & Yen 04) P. Over and J. Yen. Introduction to DUC-2004: An intrinsic evaluation of generic news text summarization systems. In *Proceedings of the HLT/NAACL 2004 Document Understanding Workshop (DUC-2004)*, 2004. Available at: http://www-nlpir.nist.gov/ projects/duc/pubs/ 2004slides/duc2004.intro.pdf.

(Saggion & Gaizauskas 04a) H. Saggion and R. Gaizauskas. Multi-document summarization by cluster/profile relevance and redundancy removal. In *Proceedings of Document Understanding Conference*, Boston, MA, May 6-7 2004. NIST.

(Saggion & Gaizauskas 04b) Horacio Saggion and Robert Gaizauskas. Mining on-line sources for definition knowledge. In *Proceedings of FLAIRS 2004*, Florida, USA, 2004. AAAI.

(Saggion & Gaizauskas 05) Horacio Saggion and Robert Gaizauskas. Experiments on Statistical and Pattern-based Biographical Summarization. In *Proceedings of the 12th Portuguese Conference on Artificial Intelligence - TeMA Workshop*, 2005. Accepted.

(Saggion 02) Horacio Saggion. Shallow-based Robust Summarization. In *Automatic Summarization: Solutions and Perspectives*, ATALA, December, 14 2002.

(Sampson 95) G. Sampson. *English for the Computer: The SUSANNE Corpus and Analytic Scheme.* Clarendon Press, Oxford, 1995.

(Tombros *et al.* 98) A. Tombros, M. Sanderson, and P. Gray. Advantages of Query Biased Summaries in Information retrieval. In *Intelligent Text Summarization. Papers from the 1998 AAAI Spring Symposium. Technical Report SS-98-06*, pages 34–43, Standford (CA), USA, March 23-25 1998. The AAAI Press.

(Voorhees 04) E. Voorhees. Overview of TREC 2003. In *Proceedings of the Twelfth Text Retrieval Conference (TREC 2003)*, NIST Special Publication 500-255, 2004. Available at: http://trec.nist.gov/ pubs/trec12/papers/ OVERVIEW.12.pdf.

(Wolf & Gibson 04) F. Wolf and E. Gibson. A response to Marcu (2003). Discourse structure: trees or graphs?, 2004. Available at: http://web.mit.edu/fwolf/www/discourse-annotation/Wolf_Gibson-coherence-representation.pdf.

A Proposal For An Arabic Named Entity Tagger Leveraging a Parallel Corpus*

Doaa Samy
Laboratorio de Lingüística Informática
Universidad Autónoma de Madrid
doaa@maria.lllf.uam.es

Antonio Moreno
Laboratorio de Lingüística Informática
Universidad Autónoma de Madrid
sandoval@maria.lllf.uam.es

José Mª Guirao
Dpto. de Lenguajes y Sistemas
Universidad de Granada
jmguirao@ugr.es

Abstract

The term Named Entity (NE), first introduced in 1995 by the Message Understanding Conference (MUC-6), is widely used in the field of Natural Language Processing and Information Retrieval. Since 1995, a lot of studies have addressed NE recognition, tagging and classification. These studies reflected its efficient role in IE systems (Sekine, 2004; Grishman and Sundheim, 1996; Hasegawa et al., 2004) as well as its effectiveness when used as anchor points in alignment techniques (Melamed, 2001; Samy et al., 2004). In this paper, we cover three main aspects concerning Arabic NE recognition and tagging. First, we present an overview of the linguistic nature and the studies concerning NE in Arabic texts. Second, we highlight the methodology of developing tools leveraging parallel corpora and previously developed tools for other languages. Third, we present our proposal for an Arabic NE tagger; its different modules, its coverage scope and the methodology used for its implementation. However, it could also be considered a method for aligning NE in parallel corpora. Finally, we evaluate the results against a gold standard. At the end, we discuss the final conclusions and future work.

1 Introduction

In this section, we will introduce an overview of the research held in the field of NE in general and a historical review of studies addressing the transliteration of Arabic Names.

1.1 Named Entities

NE recognition has proved to be an outstanding factor in the improvement of IR, CLIR and QA systems. In this paper, we try to highlight its importance in parallel text processing and alignment of parallel corpora.

The early NE classifications considered two main classes: names and numeric expressions. Both classes covered a range of 7 to 10 categories. Names might include categories such as: person names, organizations, location names, while numeric expressions cover the scope of: time, date, money and percent expressions (Sekine, 2004). These categories have been extended aiming at a wider coverage. An example of such expansion is the "200 category extended named entity hierarchy" proposed by Sekine (2004).

Although the idea of such an extensive categorization seems so appealing, it is quite beyond the

* This research has been supported by the grant TIN2004-07588-C03-02 (Spanish Ministry of Education and Science).

scope of our Arabic NE tagger for the time being, as it is a very laborious task in terms of time and annotation effort. Besides, we believe that in cases where languages lack resources for NE, which is the case in Arabic, it is more effective to start with basic categories. Once these resources are available, research should proceed on with its respective expansion.

1.2 Named Entities and Arabic Transliteration

Proper names constitute an important building block in the basic NE classifications. However, Semitic languages, in general, and Arabic scripted languages, in particular, present a challenge to the automated approaches for Proper Names and/or NE recognition. This fact could be explained if we take into consideration that a wide range of automated detection of Names (in Roman scripted languages) is based on formal orthographic criteria. These systems make use of the initial capitalisation of names of persons, locations, job titles and organizations. Also, upper case letters are used to indicate acronyms. Arabic scripted languages, on the other hand, do not provide such orthographic distinction, as they do not distinguish between upper case and lower case. That is why systems dealing with Semitic or Arabic Proper Names have to adopt different techniques to overcome such challenges.

To our knowledge, early studies tackling this issue in a computational context date to the early nineties (Roochnik, 1993; Arbabi et al., 1994). Such studies focused mainly on developing techniques and algorithms for transliteration. In this aspect, we consider it interesting to point out the following observations.

Reviewing the previous literature helped us establish the following key stages in the development of research concerning Arabic names:

Early beginnings (1993-1995): Interest in NE and Arabic Name transliteration almost coincided chronologically, although transliteration was prior to the concept of NE (first introduced in 1995).

The nineties: Despite the strong connections between both research fields, these fields remained unrelated, and each followed its own course independently. This situation prevailed because the target of transliteration focused mainly on machine translation systems (Stalls and Knight, 1998) or security issues, for example, border controls or passport checking as mentioned by Arbabi (1994); hence Information Retrieval as an important application field was not targeted at that time.

2000 to present: research in both fields (NE and Arabic Name Transliteration) began to converge in some way, although they have been limited to Arabic names transliteration and they did not include other categories of Arabic Named Entities. Besides, these studies had as a main target: IR and CLIR systems (AbdulJaleel et al., 2003; Darweesh et al., 2001; Al-Onaizan and Knight, 2002; Larkey at al., 2003; Gey and Oard, 2001, Cowie and Abdelali). The only occasion, where transliteration was mentioned within the general framework of NE, was in the study of Al-Onaizan (2002) on "*Translating Named Entities using monolingual and bilingual resources*", also designed and implemented from the perspective of IR/CLIR applications

After this review of previous work, it is clear that all approaches consider transliteration of Proper Names an indispensable step towards Arabic NE recognition. However, we would like to insist on the fact that transliteration covers only a subset of NE and that there is still a need for a comprehensive study that covers the rest of NE categories in Arabic scripted languages, in particular, without limiting the approaches to transliteration.

In this paper, we are trying to fill this gap by introducing a proposal for an Arabic NE recognition leveraging a Parallel Corpus (Spanish-Arabic) covering a wider scope of categories such as organization names, job titles and acronyms. Our approach is different in its resources and its main target application. Our main resource is an aligned parallel corpus and our final target is to identify the Arabic NE. In this way, the tagged NE would serve as anchor point for the alignment process.

2 Methodology

Developing a tagger is a task requiring the availability of either monolingual or bilingual resources. Almost all previous work in the field developed its techniques using data from bilingual dictionaries, lexicons or just simple lists of Proper and location names. The recent experiments, which try to adopt a totally statistical approach, depend mainly on lists of Proper Names and their corre-

sponding transliterations (Abduljaleel, 2003). Even the hybrid approaches combining linguistic and statistical methods validate their transliterations candidates against lists of proper names or against web counts (Al-Onaizan, 2002).

Our methodology, on the other hand, relies on two main types of resources; parallel corpora and previously developed tools for other languages.

2.1 Parallel Corpora

New approaches to develop NLP tools focus on the feasibility of using parallel corpora as resources. Such approach proved to be effective in terms of time and effort. Besides it provides the advantage of dealing with the different linguistic phenomena *in situ*, i.e. it offers an empirical data set for developing and testing the tools. Recent research on Word Sense Disambiguation makes use of parallel corpora (Diab and Resnik, 2002). Building Wordnets is another field which made use of parallel corpora (Diab, 2004).

For our tagger, we used an Arabic-Spanish parallel corpus aligned on the sentence level and tagged on the level of POS. The size of the subcorpus used for the experiment is not large (1200 sentence pairs), but due to its nature and its source, it contains a considerable number of NE. The corpus consists of UN documents published on the web. Since it was quite difficult to obtain parallel and reliable texts in this language pair (Spanish-Arabic), we opted for the UN documents as both Spanish and Arabic languages are official UN languages. The advantages of using this corpus can be summarized in the following points:

- *Reliability:* Considering the source, we could guarantee a *translation and transliteration* quality for the Named Entities.
- *Representativeness*: The corpus is a representation of Modern Standard Arabic on one hand, and of Standard Spanish on the other.

2.2 Previously developed tools for other languages

The second resource consists of previously developed tools for other languages. This resource used together with parallel corpora proved to give good results in many NLP applications.

Since we are using a Spanish-Arabic parallel corpus, the tools, which were mainly developed for processing the Spanish corpus, were used as a starting point for developing our Arabic tools. We, basically, relied on the output of the Spanish NE tagger. It is a rule-based tagger enriched with a monolingual Spanish lexicon. This tagger searches for patterns of Spanish NE and the patterns matched are tagged in xml with the tag:

```
<ne type = "" id = "">….</ne>
```

The Spanish NE tagger covered only two main NE categories: "np" (Nombre Propio */Proper Noun*) and "date". However, for the purpose of our experiment, the first type was extended to include:

- Person names
- Location names (Geographical locations and toponyms)
- Organizations (Political of Administrative Entities)
- Position (job titles)
- Acronyms

Following the new classification criteria, we had to modify the values of the *type* attribute in the Spanish Corpus.

3 Implementation

3.1 Scope and Structure

The above categorization is a semantic categorization. However, the implementation modules do not correspond strictly to this semantic classification. Instead, the implementation was based on pattern matching, lexical, orthographic and phonetic criteria. There are three basic modules:

- A module for date expressions
- A module for names based on simple transliteration. This covers the categories of person names, location names and some acronyms when phonetically transliterated.
- A module based on a bilingual lexicon. This module covers the categories of organizations and positions (job titles)

The "date" Module: Arabic date tagging depends mainly on regular patterns and a small lookup lexicon of months and days. The bilingual

lexicon of months includes months in Spanish and their equivalent in Arabic according to the Gregorian calendar (January, February, …etc) and the Lebanese calendar, since both are of common use in Arabic UN documents.

Transliteration Module: By transliteration, we mean the process of formulating a representation of words in one language using the alphabet of another language (Arbabi, 1994). In other words, it consists of the representation of a word in the closest corresponding letters or characters of a different alphabet or language, so that the pronunciation is as close as possible to the original word (Abdul-Jaleel, 2003).

Our implementation is a simple, straightforward one, but it proved to be efficient as it succeeded in meeting our main goal of detecting the Arabic names in the corpus. The main advantage over other more sophisticated approaches is that the parallel corpus plays a double role as a resource and a target at the same time. In addition to this, the fact that the parallel corpus is aligned reduces significantly the context and scope of search for valid transliterations.

To avoid encoding schemes problems or unrecognized characters, we decided to implement the transliteration module by means of numerical codification using the *Unicode* value for each Arabic character. Another solution was to use the Buckwalter's transliteration scheme considered almost a classic standard in Arabic NLP. However we decided to use Unicode as it supposes more portability to other languages if different phonetic/orthographic criteria are applicable.

On the other hand, and in the transliteration mappings from Roman characters, each character was given all its corresponding possibilities in the Arabic alphabet and consequently it is given the numeric *Unicode* value referring to each of these characters.

Arabic Character	Roman Character	Code
ب	[Pp]|[Bb]	0628
ر	[Rr]|[Rr]r	0631
ج	[Gg]|	062C
غ	[Gg]|	063A

Table 1. Example of Arabic characters and their codes

Expansion and Omission: In the transliteration module, we tried to deal with two phenomena: expansion and omission. Expansion consists in the possibility that one Roman character might be transliterated into two or more Arabic characters. For example, the "*t*" might have two possible transliterations in Arabic, either "ت" (062A) or "ط" (0637). The mapping, in this case, would be as follows: when a letter *t* is found, it could be transliterated either by character code 062A or 0637.

Omissions are common in short vowels' transliterations. Arabic scripted languages do not transliterate the short vowels. Instead, it uses the diacritics. But, in Modern Standard Arabic texts, words rarely appear with diacritics. This creates ambiguity for computational systems on all levels, starting from the tokenization till the semantic levels. In this aspect, transliteration is not an exception. However, the most practical way to deal with such phenomena is to handle the omissions. To do that, we used the regular expression operator "?" to indicate that the preceding character code might occur zero or one time(s).

Tokenization: To our knowledge, this feature has never been addressed in previous literature concerning the transliteration because almost all approaches were aiming at finding the best transliterations for a given name independently of its context. That is why tackling the tokenization problem was not considered. In our case, since we deal with a corpus, NE appear in their real context and one important issue, in this respect, is that NE as other nouns in Arabic may appear preceded by clitics. These clitics might be a conjunction "و", a preposition "ب ,"ل"or both "وب", "ول". To handle such feature, we had to expand the possibilities of matching by indicating that the string might be preceded by one or more pre-clitics.

Look-up module: In case of organizations and job titles, the Named Entity is either a one-word NE, such as `Embajador` *(Ambassador)*, `Presidente` *(President)*, or a compound NE; two or more tokens, such as `Naciones Unidas` (*United Nations*). Both types are looked up in the general lexicon used for POS tagging, since these words are originally common words, but they have passed from common words to NE through a semantic process to refer to a certain entity. This semantic phenomenon is reflected orthographically in the use of upper case. The look-up is easy and feasible, as it does not need especial effort for creating lists of NE referring to organizations or job titles.

3.2 Algorithm

This section explains how the tagging process takes place given that the Spanish NE have been previously annotated according to the above-mentioned classification. Our implementation relies on this basic assumption: "Given a pair of sentences where each is the translation of the other; and given that in one sentence one or more NE were detected, then the corresponding aligned sentence should contain the same NE either translated or transliterated".

This assumption is a simplistic one, as it doesn't take into consideration common phenomena in translation such as omission or addition. Despite this fact, NEs usually tend to be conserved in translations as they represent significant pieces of information. Such a semantic weight is reflected in the way translators deal with them. While a translator might have more flexibility in translating common nouns or expressions, when dealing with NE, the translator rather tries to keep the translation as close as possible to the source. Starting from this assumption, we follow this algorithm.

Input: The input consists of the file containing the aligned parallel corpus with Spanish NE tagged. The corpus is processed so that each pair of aligned sentences (*x, y*) is handled one at a time. We begin by processing the Spanish sentence in the following way:

- Previously tagged Spanish NEs are extracted from the Spanish sentence.
- Extracted NEs are classified in sub lists depending on their type.
- First, NEs of type date are passed to the date module.
- Given the list of tagged dates in Spanish in a sentence *x*. The system looks up the bilingual lexicon of months and numbers to find their equivalent in Arabic. Once found, the system searches the corresponding aligned Arabic sentence *y* for the pattern generated. If the generated pattern is found, it is tagged by the same tag as its Spanish equivalent and it is given the same ID number. If not, it exists this module.
- Second, NEs of type Person names, location names, toponyms and some acronyms[1] are passed to the transliteration module.
- For each Spanish NE and according to the mapping scheme, the system provides a combination of all possibilities of transliteration. The output consists of the Spanish NE together with a string with all transliteration possibilities. Different possible transliterations for each character are separated by "|". In case of vowels the specific numeric code is followed by "?" indicating that zero instances or one of the preceding character could occur. For example, given the proper name `Carl`, the transliteration module generates the following string

```
(0643|0633|062B|0642|062A0634)
(0629|0623|0639|0627|0647|0622
|0649|0621)? 0631 0644
```

- A list of all the Arabic words in the corresponding Arabic sentence is extracted. Each word is converted to a string of numeric codes, according to the codification scheme. In the example mentioned above, the Arabic word "كـارل" receives the following codification:

```
0643 0627 0631 0644
```

Comparing the Arabic string "0643 0627 0631 0644" against the above transliteration returns true. Thus, "كـارل" is the corresponding NE equivalent to "`Carl`".

- Finally, the valid candidate is automatically tagged by the same tag and is given the same ID number of its Spanish equivalent.
- Spanish Nes of type organization or job title are passed to the lookup module. The output of this stage is the looked-up Spanish NE, together with its Arabic translation obtained from the bilingual lexicon.
- Arabic translations are searched in the corresponding aligned sentence. If found, the

1 Acronyms are dealt with in the Arabic text by different ways. One possibility is to be transliterated phonetically. Another possibility is to use the name in its full form.

Arabic NE is tagged with the same tag and the same ID number of its corresponding Spanish NE.

Tagging Acronyms: Acronyms are handled in one of two ways. An acronym first is passed to the transliteration module. If found, then the Arabic translator has opted for a transliteration of the Acronym. Otherwise, the Acronym is returned to its full form, since usually the first occurrence of an acronym in a text is accompanied by its name in full form. We keep track of this name and if the transliteration module fails to find a candidate, it passes to the look up module where it searches for the equivalent translation. When found, it is tagged with the same tag and given the same ID number as its corresponding Spanish NE.

Unknown Named Entities: NE, which failed to be recognized through the previous stages, are names whose Arabic equivalents are totally different such as "Grecia" (*Greece*) "اليونان" or "Egipto" (Egypt) "مصر". This is explained in terms of the History of Language, which is far beyond our scope. The only way to tag such unknown words is either by human intervention, or by consulting a bilingual list of names if available.

Final Output: The final output consists of the same aligned corpus with the Arabic NE tagged indicating their type and given the same ID numbers of their corresponding Spanish ID.

4 Evaluation

The results of the NE tagger were evaluated against a gold standard set. From the 1200 pairs of sentences, 300 sentences from the Spanish corpus were selected randomly with their equivalent Arabic sentences. For each pair, the output of the NE tagger was compared to the manually annotated gold standard set.

The evaluation took place on the different tagging levels testing in that way the different tagging modules. The best results were achieved in the "date" module and the "look-up" module.

In the acronyms, sometimes due to the inconsistency in translating the acronyms to the Arabic, beside the extended length of the name, the tagger was not able to correctly identify all the Arabic corresponding NE. The acronyms were correctly identified only in 76% of the cases.

The transliteration module showed high coverage and accuracy in recognizing the transliterated NE. It correctly identified and tagged almost all transliterated NE (Recall 97.5%), even when the NE in Spanish and Arabic was not a precise transliteration; such as "Somalia" and its Arabic equivalent "الصومال". This is due to expanding the possibilities on one hand, and handling the vowels' omission and the tokenization, on the other hand. The only drawback of expansion is that the system in some cases wrongly identified words as NE (Precision 84%). To improve the precision, we applied a filter to the Arabic words, which omitted the Stop Words from the possible transliterated candidates. This increased the precision result significantly reaching (90%). Table 2 shows NE distribution in the evaluation and Table 3 shows the evaluation results.

	Arabic	**Spanish**
N. of sentences	307	300
Total N. of NE	721	743
Average NE/sent	2.41	2.54
Proper Names	39	40
Toponyms	164	167
Acronyms	11	27
Jobs	123	128
Organizations	275	277
Dates	109	104

Table 2. NE Distribution in the evaluation corpus

Recall	**Precision**	**Improved Precision**
97.5%	84%	90%

Table 3. Evaluation results

5 Conclusion and Future Work

NE recognition leveraging a parallel corpus and reusing previously developed tools for other languages proved to be an efficient methodology, as it supposes a feasible and cost effective solution to develop resources specially for languages with scarce resources.

Results obtained show that our basic assumption was practical and applicable. Although the transliteration module could be considered a shallow one, as it does not apply sophisticated statisti-

cal methods, but it was efficient for the task and it managed to meet the suggested goals.

Although the transliteration was implemented considering the Spanish-Arabic, we tried in the majority of cases to follow more general criteria, applicable on English-Arabic transliteration or French-Arabic transliteration. This is because the NEs tagged in the Spanish Corpus are not exclusively Spanish names. They are names proceeding from different languages; English, French, German, …etc.

For future work, we would consider applying statistical models for transliteration. Also a character bigram would be of great significance.

On the other hand, a phonological transcription tool for Spanish might be applied to the Spanish NE. The information concerning the syllables and their divisions might help us in improving the transliteration module.

Finally, the more trained the tagger, the more NE it would recognize, since in each training pass, the lexicon is enriched with the new NE. Such a resource would be very useful in working not only with parallel, but also with comparable corpora. Besides, such a list of NE extracted from real text would be a valuable resource for IR and/or CLIR applications.

Other applications might include Example Based Machine Translation, Translation Memories or Computer Assisted Language Learning since a parallel aligned corpus with both POS and NEs tagged, is considered a valuable resource especially for uncommon language pairs as Spanish and Arabic.

References

AbdulJaleel, N. and Larkey, L. 2003. English to Arabic Transliteration for Information Retrieval: A Statistical Approach, *CIIR Technical Report IR- 261*.

Al- Onaizan, Y. and Knight, K. 2002. Machine translation of names in Arabic text. *Proceedings of the ACL conference workshop on computational approaches to Semitic Languages*.

Al- Onaizan, Y. and Knight, K. 2002. Translating Named Entities Using Monolingual and Bilingual Resources. . *Proceedings of 40th ACL Conference,* Philadelphia, pp. 400-408.

Arbabi, M. Fischthal, S. M. Cheng, V. C. and Bart, E. 1994. Algorithms for Arabic name transliteration. *IBM Journal of Research and Developme*nt, 38(2): 183– 193.

Cowie, J and Abdelali, A. Interactive Cross Language Information Retrieval Using Transliterated Names Resolution. *Memoranda in Computer and Cognitive ScienceMCCS-04-331*.

Darwish, K., Doermann, D. Jones, R., Oard, D. and Rautiainen, M. 2001. TREC-10 experiments at Maryland: CLIR and video. In *TREC 2001*. Gaithersburg: NIST.

Diab, M. 2004. The feasibility of bootstrapping an Arabic Wordnet leveraging parallel corpora and English WordNet. *Proceedings of the International Conference on Arabic Language Resources and Tools (NEMLAR 2004),* Cairo, Egypt, pp.71-77.

Diab, M. and Resnik, P. 2002. Word sense tagging using parallel corpora. *Proceedings of 40th ACL Conference*, Pennsylvania, USA.

Gey, F. C. and Oard, D. W. 2001. The TREC-2001 cross-language information retrieval track: Searching Arabic using English, French, or Arabic queries. In *TREC 2001*. Gaithersburg: NIST.

Grishman, R. and Sundheim, B. "Message Understanding Conference - 6: A Brief History", *COLING-96*.

Hasegawa, T. Sekine, S. Grishman, R. 2004. Discovering Relations among Named Entities from Large Corpora. *Proceedings of ACL 04*; Barcelona, Spain.

Larkey, L., AbdulJaleel, N. and Connell, M. 2003. What's in a Name?: Proper Names in Arabic. Cross Language Information Retrieval. *CIIR Technical Report, IR- 278*.

Melamed, I. D. 2001. *Empirical Methods for Exploiting Parallel Text*, Cambridge/London: MIT Press.

Roochnik, P. 1993. *Computer-Based Solutions to Certain Linguistic Problems Arising from the Romanization of Arabic Names,* Ph.D. Dissertation, Georgetown University, Washington, DC.

Samy, D., Moreno Sandoval, A. and Guirao, J.M.2004. An Alignment Experiment of a Spanish-Arabic Parallel Corpus. *Proceedings of the International Conference on Arabic Language Resources and Tools (NEMLAR 2004),* Cairo, Egypt, pp.85-89.

Sekine, S. 2004. Named Entity: History and Future.

http://cs.nyu.edu/~sekine/papers/NEsurvey200402.pdf

Stalls, B. and Knight, K. 1998. Translating Names and Technical Terms in Arabic Text. *COLING/ACL Workshop on Computational Approaches to Semitic Language*s. Montreal, Quebéc.

Parameter reduction in unsupervisedly trained sliding-window part-of-speech taggers

Enrique Sánchez-Villamil and **Mikel L. Forcada** and **Rafael C. Carrasco**
{esvillamil,mlf,carrasco}@dlsi.ua.es
Transducens, Departament de Llenguatges i Sistemes Informàtics
Universitat d'Alacant, E-03071 Alacant, Spain

Abstract

A new, robust sliding-window part-of-speech tagger is presented, which itself is an approximation of an existing model, and a method is described to estimate its parameters from an untagged corpus. The approximation reduces the memory requirements without a significant loss in accuracy. Its performance is compared to that of the original sliding-window tagger as well as to that of a standard Baum-Welch-trained hidden-Markov-model part-of-speech tagger and a random tagger.

1 Introduction

A large fraction (typically 30%, but varying from one language to another) of the words in natural language texts are words that, in isolation, may be assigned more than one morphological analysis and, in particular, more than one part of speech (PoS). The correct resolution of PoS ambiguity for each occurrence of the word in the text is crucial in many natural language processing applications; for example, in machine translation, the correct equivalent of a word may be very different depending on its PoS.

This paper presents a new version of a sliding-window (SW) PoS tagger, that is, a system which assigns the PoS of a word based on the information provided by a fixed window of words around it. The SW tagger idea is not new (Sánchez-Villamil *et al.* 04), but the number of parameters required to achieve acceptable results is high compared to that of more usual approaches such as hidden Markov Models (HMM). The new light sliding-window (LSW) PoS tagger proposed here reduces greatly the number of parameters with a negligible loss of performance.

The paper is organized as follows: section 2 gives some definitions and describes the notation that will be used throughout the paper; section 3 describes the approximations that allow a SW tagger to be trained in an unsupervised manner and the training process itself; section 4 describes the LSW tagger in parallel to the SW tagger training algorithm; section 5 describes a series of experiments performed to compare the performance of a LSW tagger to that of a HMM tagger and to that of the SW tagger; and, finally, concluding remarks are given in section 6.

2 Preliminaries

Let $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_{|\Gamma|}\}$ be the *tagset* for the task, that is, the set of PoS tags a word may receive in a specific language, and $W = \{w_1, w_2, \ldots, w_{|W|}\}$ be the *vocabulary* of the task. A partition of W is established so that $w_i \equiv w_j$ (that is, both words belong to the same equivalence class) if and only if both are assigned the same subset of tags by the lexical categorizer.[1]

It is usual (Cutting *et al.* 92) to refine this partition so that, for high-frequency words, each word class contains just one word whereas, for lower-frequency words, word classes are made to correspond exactly to *ambiguity classes* containing all words receiving the same subset of PoS tags (although it would also be possible to use one-word classes for all words or to use only ambiguity classes). This refinement allows for improved performance on very frequent ambiguous words while keeping the number of parameters of the tagger under control.

Any such refinement will be denoted as $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_{|\Sigma|}\}$ where σ_i are word classes. In this paper, word classes will simply be ambiguity classes, without any refinement. We will call $T : \Sigma \to 2^\Gamma$ the function returning the set $T(\sigma)$ of PoS tags for each word class σ.

The PoS tagging problem may be formulated as follows: given a text $w[1]w[2]\ldots w[L] \in W^+$, each word $w[t]$ is assigned a word class $\sigma[t] \in \Sigma$ to obtain

[1] The lexical categorizer function may be implemented by a dictionary, a morphological analyser, a guesser, or any combination thereof.

an *ambiguously tagged* text $\sigma[1]\sigma[2]\ldots\sigma[L] \in \Sigma^+$; the task of the PoS tagger is to obtain a *tagged* text $\gamma[1]\gamma[2]\ldots\gamma[L] \in \Gamma^+$ (with all $\gamma[t] \in T(\sigma[t])$) as correct as possible.

Statistical PoS tagging looks for the *most likely* tagging $\gamma^*[1], \gamma^*[2], ..., \gamma^*[L]$ given an ambiguously tagged text $\sigma[1]\sigma[2]\ldots\sigma[L]$:

$$\gamma^*[1]\ldots\gamma^*[L] = \underset{\gamma[t]\in T(\sigma[t])}{\operatorname{argmax}} P(\gamma[1]\ldots\gamma[L] \mid \sigma[1]\ldots\sigma[L]). \quad (1)$$

ambiguously tagged sequence $\sigma[1]\ldots\sigma[L]$. In hidden Markov models (Rabiner 89), use of the Bayes' formula, modelling of tag sequences as first-order Markov processes, and additional approximations lead to

$$\gamma^*[1]\ldots\gamma^*[L] = \underset{\gamma[t]\in T(\sigma[t])}{\operatorname{argmax}} \prod_{t=0}^{t=L} p_S(\gamma[t+1] \mid \gamma[t]) \times \prod_{t=1}^{t=L} p_L(\sigma[t] \mid \gamma[t]), \quad (2)$$

where P_S is the *syntactical* probability modelling tag sequences and P_L is the *lexical* probability modelling the relations between tags and word classes, with $\gamma[0] = \gamma[L+1] = \gamma_\#$, a special delimiting tag analogous to a sentence boundary. The number of trainable parameters is $(|\Gamma| + |\Sigma|)|\Gamma|$. Tagging (searching for the optimal $\gamma^*[1]\gamma^*[2]\ldots\gamma^*[L]$) is implemented using an efficient, left-to-right algorithm usually known as Viterbi's algorithm (Cutting *et al.* 92; Rabiner 89), which, if conveniently implemented, can output a partial tagging each time a nonambiguous word is seen, but has to maintain multiple hypotheses when reading ambiguous words. HMM taggers may be trained either from tagged text (simply by counting and taking probabilities to be equal to frequencies) or from untagged text, using the well-known expectation-maximization backward-forward Baum-Welch algorithm (Rabiner 89; Cutting *et al.* 92).

3 The Sliding-Window PoS Tagger model

The sliding-window PoS tagger (Sánchez-Villamil *et al.* 04) approximates the probability in eq. (1) *directly* as follows:

$$P(\gamma[1]\gamma[2]\ldots\gamma[L] \mid \sigma[1]\sigma[2]\ldots\sigma[L]) \simeq \prod_{t=1}^{t=L} p(\gamma[t] \mid C_{(-)}[t]\sigma[t]C_{(+)}[t]) \quad (3)$$

where $C_{(-)}[t] = \sigma[t-N_{(-)}]\sigma[t-N_{(-)}+1]\cdots\sigma[t-1]$ is a *left context* of word classes of length $N_{(-)}$ and $C_{(+)}[t] = \sigma[t+1]\sigma[t+2]\cdots\sigma[t+N_{(+)}]$ is a *right context* of word classes of length $N_{(+)}$, so that for $t < 1$ and $t > L$, $\sigma[t] = \sigma_\#$, a special delimiting word class such that $T(\sigma_\#) = \{\gamma_\#\}$.

This *sliding window* method is local in nature; it does not consider any context beyond the window of $N_{(-)}+N_{(+)}+1$ words; its implementation is straightforward, even more than that of Viterbi's algorithm. The main problem is the estimation of the probabilities $p(\gamma[t] \mid C_{(-)}[t]\sigma[t]C_{(+)}[t])$. If a tagged corpus is available, these probabilities may be easily obtained by counting; however, the SW tagger has a specific way of estimating them from an untagged corpus, as we will see below. Another problem is the large number of parameters of the model ($|\Sigma|^{N_{(+)}+N_{(-)}}|\Gamma|$).

The main approximation in the model consists in assuming that the best tag $\gamma^*[t]$ contained in the window depends on the preceding context $C_{(-)}[t]$ and the succeeding context $C_{(+)}[t]$, and only *selectionally* on the word (one could say that it is the context which determines the probabilities of each tag, whereas the word just *selects* tags among those in $T(\sigma[t])$).

The most probable tag $\gamma^*[t]$ is

$$\gamma^*[t] = \underset{\gamma\in T(\sigma[t])}{\operatorname{argmax}} p(\gamma[t] = \gamma | C_{(-)}[t]\sigma[t]C_{(+)}[t]). \quad (4)$$

We will drop the position index $[t]$ because of time invariance; and write $p(\gamma|C_{(-)}\sigma\, C_{(+)})$. These probabilities are easily estimated from a tagged corpus (e.g., by counting) but estimating them from an untagged corpus involves an iterative process, which proceeds by estimating counts $\tilde{n}_{C_{(-)}\gamma\, C_{(+)}}$ which express the *effective* number of times that tag γ would appear in the text between contexts $C_{(-)}$ and $C_{(+)}$. Therefore,

$$p(\gamma|C_{(-)}\sigma C_{(+)}) = k_{C_{(-)}\sigma C_{(+)}}\tilde{n}_{C_{(-)}\gamma\, C_{(+)}} \quad (5)$$

if $\gamma \in T(\sigma)$ and zero otherwise, where $k_{C_{(-)}\sigma C_{(+)}} = (\sum_{\gamma'\in T(\sigma)} \tilde{n}_{C_{(-)}\gamma'\, C_{(+)}})^{-1}$ is a normalization factor. Accordingly, equation (4) could be written as:

$$\gamma^*[t] = \underset{\gamma\in T(\sigma[t])}{\operatorname{argmax}} \tilde{n}_{C_{(-)}[t]\gamma\, C_{(+)}[t]}, \quad (6)$$

where the dependence with respect to $\sigma[t]$ can be clearly seen to be only selectional.

But, how can the counts $\tilde{n}_{C_{(-)}\gamma\, C_{(+)}}$ be estimated? If the window probabilities $p(\gamma \mid C_{(-)}\sigma C_{(+)})$ were known, the effective counts could be easily obtained

from the text itself as follows:

$$\tilde{n}_{C_{(-)}\gamma C_{(+)}} = \sum_{\sigma:\gamma\in T(\sigma)} n_{C_{(-)}\sigma C_{(+)}} p(\gamma \mid C_{(-)}\sigma C_{(+)}), \tag{7}$$

where $n_{C_{(-)}\sigma C_{(+)}}$ is the number of times that ambiguity class σ appears between contexts $C_{(-)}$ and $C_{(+)}$; that is, one would add $p(\gamma \mid C_{(-)}\sigma C_{(+)})$ each time a word class σ containing tag γ appears between $C_{(-)}$ and $C_{(+)}$. Equations (5) and (7) may be iteratively solved until the $\tilde{n}_{C_{(-)}\gamma C_{(+)}}$ converge. For the computation to be more efficient, one can avoid storing the probabilities $p(\gamma \mid C_{(-)}\sigma C_{(+)})$ by organizing the iterations around the $\tilde{n}_{C_{(-)}\gamma C_{(+)}}$ as follows, by combining eqs. (5) and (7) and using an iteration index denoted with a superscript $[m]$,

$$\tilde{n}^{[m]}_{C_{(-)}\gamma C_{(+)}} = \tilde{n}^{[m-1]}_{C_{(-)}\gamma C_{(+)}} \times \sum_{\sigma:\gamma\in T(\sigma)} n_{C_{(-)}\sigma C_{(+)}} \left(\sum_{\gamma'\in T(\sigma)} \tilde{n}^{[m-1]}_{C_{(-)}\gamma' C_{(+)}}\right)^{-1}, \tag{8}$$

where the iteration may be easily seen as a process of successive multiplicative corrections to the effective counts $\tilde{n}_{C_{(-)}\gamma C_{(+)}}$. A convenient starting point is given by $p(\gamma \mid C_{(-)}\sigma C_{(+)}) = |T(\sigma)|^{-1}$ which is equivalent to assuming that initially all possible tags are equally probable for each word class.

Equation (8) contains the counts $n_{C_{(-)}\sigma C_{(+)}}$ which depend on $N_{(+)} + N_{(-)} + 1$ word classes; if memory is at a premium, instead of reading the text once to count these and then iterating, the text may be read in each iteration to avoid storing the $n_{C_{(-)}\sigma C_{(+)}}$, and the $\tilde{n}^{[k]}_{C_{(-)}\gamma C_{(+)}}$ may be computed *on the fly*. Iterations proceed until a selected convergence condition has been met (e.g. a comparison of the $\tilde{n}^{[k]}_{C_{(-)}\gamma C_{(+)}}$ with respect to the $\tilde{n}^{[k-1]}_{C_{(-)}\gamma C_{(+)}}$, or the completion of a predetermined number of iterations).

4 Light Sliding-Window PoS Tagger model

The model proposed in this paper may be considered as an approximation to the SW tagger just described, with the objective of reducing the number of parameters to estimate without a significant loss in tagging accuracy. The number of parameters of the LSW tagger in the worst case is $|\Gamma|^{N_{(-)}+N_{(+)}+1}$, compared to the $|\Sigma|^{N_{(-)}+N_{(+)}}|\Gamma|$ of the SW tagger. The number of parameters of the LSW tagger depends only on the size of the set of tags, which is much smaller than the number of word classes. However, as expected, the reduction of parameters makes the tagger slower, as the training and tagging equations are more complicated to compute.

The best tag γ^* is obtained by considering for each possible $\gamma[t]$ *all possible disambiguations* $E_{(-)}[t]\gamma[t]E_{(+)}[t]$ of the current window $C_{(-)}[t]\sigma C_{(+)}[t]$ and adding their probabilities $p(E_{(-)}[t]\gamma[t]E_{(+)}[t] \mid C_{(-)}[t]\sigma[t]C_{(+)}[t])$ as if they were independent.

The LSW tagger approximates eq. (4) as follows:

$$\gamma^*[t] = \operatorname*{argmax}_{\gamma\in T(\sigma[t])} \sum_{\substack{E_{(-)}\in T'(C_{(-)}[t])\\ E_{(+)}\in T'(C_{(+)}[t])}} p(E_{(-)}\gamma E_{(+)} \mid C_{(-)}[t]\sigma[t]C_{(+)}[t]) \tag{9}$$

where $E_{(-)}[t] = \gamma[t-N_{(-)}]\gamma[t-N_{(-)}+1]\ldots\gamma[t-1]$ is a *left context of tags* of size $N_{(-)}$, $[E_{(+)}[t] = \gamma[t+1]\gamma[t+2]\ldots\gamma[t+N_{(+)}]$ is a *right context of tags* of size $N_{(+)}$, and $\gamma[t]\ \forall t < 1, \forall t > L$ are all set to the special delimiting tag $\gamma_{\#}$.

Let $T' : \Sigma^* \rightarrow 2^{\Gamma^*}$ now be the function that returns all the tag sequences that can be assigned to a given sequence of ambiguity classes. The probabilities $p_{E_{(-)}\gamma E_{(+)}}$ may be easily estimated in an analogous way to equation (5), dropping time indices for invariance:

$$p(E_{(-)}\gamma E_{(+)} \mid C_{(-)}\sigma C_{(+)}) = k_{C_{(-)}\sigma C_{(+)}} \tilde{n}_{E_{(-)}\gamma E_{(+)}} \tag{10}$$

if $E_{(-)}\gamma E_{(+)} \in T'(C_{(-)}\sigma C_{(+)})$, and zero otherwise, where $k_{C_{(-)}\sigma C_{(+)}} = \left(\sum_{\gamma\in T(\sigma), E_{(-)}\in T'(C_{(-)}[t]), E_{(+)}\in T'(C_{(+)}[t])} \tilde{n}_{E_{(-)}\gamma E_{(+)}}\right)^{-1}$ is a normalization factor. Thus, equation (9) could be written similarly to eq. (6) as:

$$\gamma^*[t] = \operatorname*{argmax}_{\gamma\in T(\sigma[t])} \sum_{\substack{\sigma:\gamma\in T(\sigma)\\ C_{(-)}:E_{(-)}\in T'(C_{(-)}[t])\\ C_{(+)}:E_{(+)}\in T'(C_{(+)}[t])}} \tilde{n}_{E_{(-)}\gamma E_{(+)}}; \tag{11}$$

as in (6), $\gamma^*[t]$ depends only selectionally on $\sigma[t]$.

The counts $\tilde{n}_{E_{(-)}\gamma E_{(+)}}$ could be easily estimated if the probabilities $p(E_{(-)}\gamma E_{(+)} \mid C_{(-)}\sigma C_{(+)})$ were known, using an equation parallel to (7):

$$\tilde{n}_{E_{(-)}\gamma E_{(+)}} = \sum_{\substack{\sigma:\gamma\in T(\sigma)\\ C_{(-)}:E_{(-)}\in T'(C_{(-)})\\ C_{(+)}:E_{(+)}\in T'(C_{(+)})}} \Big(n_{C_{(-)}\sigma C_{(+)}} \times p(E_{(-)}\gamma E_{(+)} \mid C_{(-)}\gamma C_{(+)})\Big) \tag{12}$$

But, since they are unknown, the counts are estimated through an adapted version of the iterative

equation (8) of the SW tagger, applied until it converges:

$$\tilde{n}^{[m]}_{E_{(-)}\gamma E_{(+)}} = \tilde{n}^{[m-1]}_{E_{(-)}\gamma E_{(+)}} \times \sum_{\substack{\sigma:\gamma\in T(\sigma) \\ C_{(-)}:E_{(-)}\in T'(C_{(-)}) \\ C_{(+)}:E_{(+)}\in T'(C_{(+)})}} n_{C_{(-)}\sigma C_{(+)}} k^{[m-1]}_{C_{(-)}\sigma C_{(+)}} \quad (13)$$

A convenient equiprobable initialization takes $p(E_{(-)}\gamma E_{(+)} \mid C_{(-)}\gamma C_{(+)}) = (|T'(C_{(-)}\sigma C_{(+)})|)^{-1}$.

As has been advanced, the main difference between the SW and LSW models is the number of parameters needed; while the SW tagger keeps all $\tilde{n}_{C_{(-)}\gamma C_{(+)}}$, the LSW tagger keeps only the $\tilde{n}_{E_{(-)}\gamma E_{(+)}}$; this results in a lower complexity in the worst case at the expense of an increase in tagging time, given that the computation of $\gamma^*[t]$ needs to consider $|\Gamma|^{N_{(-)}+N_{(+)}}$ effective counts instead of only one, as in the original. Moreover, it is clear that a reduction in the number of parameters means a loss of information, so the tagging accuracy is expected to be worse.

5 Experiments

This section reports experiments to assess the performance of the sliding-window PoS taggers using different amounts of context, and compares them with that of customary Baum-Welch-trained HMM taggers (Cutting *et al.* 92).

For training and testing we have used the Penn Treebank, version 3 (Marcus *et al.* 93; Marcus *et al.* 94), which has 1,014,377 PoS-tagged words of English text taken from *The Wall Street Journal*. The word classes Σ of the Treebank will be taken simply to be ambiguity classes. The Treebank uses 45 different PoS tags; 24.08% of the words are ambiguous.

The experiments use a lexicon extracted from the Penn Treebank, that is, a list of words with all the possible parts of speech observed.[2] Of course, the exact tag given in the Treebank for each occurrence of each word is taken into account only for testing but not for training. To simulate the effect of using a real, limited lexical categorizer, we have filtered the resulting lexicon to keep only the 14,276 most frequent words (95% text coverage), and to remove, for each word, any PoS tag occuring less than 5% of the time. Using this simplified, but realistic, lexicon, texts in the Penn Treebank show 218 ambiguity classes (the word classes for these experiments). Words not included in the lexicon are assigned to a special ambiguity class (the *open* class) containing all tags representing parts of speech that can grow (i.e. a new word can be a noun or a verb but hardly ever a preposition).[3]

In order to train the taggers we have applied the following strategy, so that we can use as much text as possible for training: the Treebank is divided into 20 similarly-sized sections; a leaving-one-out procedure is applied, using 19 sections for training and the remaining one for testing, so that our results are the average of all 20 different train–test configurations. In our experiments, the SW model was a 15% faster in training time than LSW, but in our implementation *there was no significant difference in tagging time* between the models. Both models tag around 70,000 words per second in a Pentium IV 2.8GHz.

5.1 Effect of the amount of context

First of all, we show the results of the sliding-window taggers using no context ($N_{(-)} = N_{(+)} = 0$) as a baseline, and compare them to those of a Baum-Welch-trained HMM tagger and to random tagging. As expected, the performance of the taggers without context is not much better than random tagging (see table 1). This happens because without context the SW tagger and the LSW tagger, whose behaviours are completely equivalent in this case, simply deliver an estimate of the most likely tag in each class. The HMM tagger accuracy (90.7%) is also given for comparison. In this and the rest of experiments reported here, standard deviations are in the range 0.25% – 0.30%, but they will not be shown for clarity. All results correspond to the 15th iteration of the SW, LSW an HMM models, although the SW and LSW taggers usually converge in 3 or 4 iterations.

In order to improve the results, one obviously needs to increase the context (i.e., widen the sliding window). The results of using a reduced context of only one word before the current word ($N_{(-)} = 1, N_{(+)} = 0$) (the results obtained using a context of one word *after* the current word are worse) are also shown in figure 1. It is worth noting that even using such a limited context the performance of the LSW tagger almost reaches that of the HMM tagger, and is comparable —within the standard deviation— to that of the SW tagger, which has five times more parameters.

If we increase the size of the context to two context words, we have three different possibilities: using the two immediately preceding words, using one preceding and one succeeding word , and using two succeed-

[2] Even if the Treebank were ambiguously tagged (i.e, with ambiguity classes), a lexicon could still be extracted.

[3] Our open class contains the Penn Treebank tags `CD`, `JJ`, `JJR`, `JJS`, `NN`, `NNP`, `NNPS`, `RB`, `RBR`, `RBS`, `UH`, `VB`, `VBD`, `VBG`, `VBN`, `VBP`, and `VBZ`.

Tagger	$N_{(-)}$	$N_{(+)}$	Number of parameters	Accuracy
RANDOM	-	-	0	85.0%
HMM	-	-	11,835	90.7%
LSW AND SW	0	0	45	86.4%
SW	1	0	9,810	90.4%
LSW	1	0	2,025	90.2%
SW	1	1	2,138,580	92.1%
LSW	1	1	91,125	91.8%

Table 1: Comparison of the accuracy and the number of parameters of sliding-window taggers to other tagging strategies, as a function of the size of the left and right contexts.

ing words; the best results are achieved when using one preceding and one succeeding word ($N_{(-)} = 1$ and $N_{(+)} = 1$), and are shown in table 1. The performance of the sliding-window taggers is now clearly better than that of the HMM tagger, in exchange for a large increase in the number of parameters (still moderate in the case of the LSW tagger). Increasing the context a bit more, until using three context words in all possible geometries does not improve results (the corpus is not large enough to allow the estimation of so many parameters).

5.2 Effect of corpus size

To assess the effect of corpus size, we trained the taggers with corpora built using an increasing number of sections of the Treebank. The results show that the LSW tagger reaches its peak performance with smaller corpora than the SW tagger, which was expected in view of the difference in the number of parameters.

6 Concluding remarks

As commonly-used HMM taggers, simple and intuitive sliding-window PoS taggers (SW taggers, (Sánchez-Villamil *et al.* 04)) may be iteratively trained in an unsupervised manner using reasonable approximations to reduce the number of trainable parameters (LSW taggers, proposed here). Experimental results show that the performance of the sliding-window taggers and HMM taggers having a similar number of trainable parameters is comparable; the best results are obtained with a context of one preceding and one succeeding word. LSW tagger results are almost indistinguishable from SW tagger results. Besides, the reduction of parameters allows the LSW tagger to be trained with a smaller training set.

We are currently studying ways to improve the training algorithm, so that incremental training (i.e. automatically adding new text to the training set) can be done. We also plan to test the models with different corpora, using the morphological analysers and finer tagsets in the Spanish–Catalan translator `interNOSTRUM.com` (Canals-Marote *et al.* 01). In addition, we are studying the introduction of constraints (Laporte & Monceaux 00) and lexicalization (using word classes finer than ambiguity classes).

Acknowledgements: Work funded by the Spanish Government through grant TIC2003-08681-C02-01.

References

(Canals-Marote *et al.* 01) R. Canals-Marote, A. Esteve-Guillen, A. Garrido-Alenda, M. Guardiola-Savall, A. Iturraspe-Bellver, S. Montserrat-Buendia, S. Ortiz-Rojas, H. Pastor-Pina, P.M. Pérez-Antón, and M.L. Forcada. The Spanish-Catalan machine translation system interNOSTRUM. In B. Maegaard, editor, *Proceedings of MT Summit VIII: Machine Translation in the Information Age*, pages 73–76, 2001. Santiago de Compostela, Spain, 18–22 July 2001.

(Cutting *et al.* 92) D. Cutting, J. Kupiec, J. Pedersen, and P. Sibun. A practical part-of-speech tagger. In *Third Conference on Applied Natural Language Processing. Association for Computational Linguistics. Proceedings of the Conference*, pages 133–140, Trento, Italia, 31 marzo–3 abril 1992.

(Laporte & Monceaux 00) É. Laporte and A. Monceaux. Elimination of lexical ambiguities by grammars: The ELAG system. In John Benjamins Publishing Company, editor, *Liguisticae Investigationes*, volume 22, pages 341–367(27), 2000.

(Marcus *et al.* 93) Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: the Penn Treebank. *Computational linguistics*, 19:313–330, 1993. Reprinted in Susan Armstrong, ed. 1994, *Using large corpora*, Cambridge, MA: MIT Press, 273–290.

(Marcus *et al.* 94) Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. The Penn Treebank: Annotating predicate argument structure. In *Proc. ARPA Human Language Technology Workshop*, pages 110–115, 1994.

(Rabiner 89) Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

(Sánchez-Villamil *et al.* 04) E. Sánchez-Villamil, Mikel L. Forcada, and Rafael C. Carrasco. Unsupervised training of a finite-state sliding-window part-of-speech tagger. *Lecture Notes in Computer Science - Lecture Notes in Artificial Intelligence*, 3230(12):454–463, 2004.

Target-Language-Driven Agglomerative Part-of-Speech Tag Clustering for Machine Translation*

Felipe Sánchez-Martínez and **Juan Antonio Pérez-Ortiz** and **Mikel L. Forcada**
Transducens Group, Departament de Llenguatges i Sistemes Informàtics
Universitat d'Alacant
E-03071 Alacant. Spain
{fsanchez,japerez,mlf}@dlsi.ua.es

Abstract

This paper presents a method for reducing the set of different tags to be considered by a part-of-speech tagger. The method is based on a clustering algorithm performed over the states of a hidden Markov model, which is initially trained by considering information not only from the source language, but also from the target language, using a new unsupervised technique which has been recently proposed to obtain taggers involved in machine translation systems. Then, a bottom-up agglomerative clustering algorithm groups the states of the hidden Markov model according to a similarity measure based on their transition probabilities; this reduces the complexity by grouping the initial finer tags into coarser ones. The experiments show that part-of-speech taggers using the coarser tags have smaller error rates than those using the initial finest tags; moreover, considering unsupervised information from the target language results in better clusters compared to those unsupervisedly built from source language information only.

1 Introduction

This paper explores the automatic induction of hidden Markov model (HMM) topologies used for part-of-speech tagging in a machine translation (MT) system. Hidden Markov models (Rabiner 89) have been widely used for part-of-speech (PoS) tagging (Cutting *et al.* 92). In this case, the HMM topology is usually fixed (that is, manually defined following linguistics guidelines) and the training phase is restricted to the estimation of probabilities.

There have been some attempts to define the HMM topology automatically. (Stolcke & Omohundro 94) describe a technique for inducing the HMM structure from data, which is based in the general *model merging* strategy (Omohundro 92), but their work focuses on HMMs for speech recognition, not on HMMs used for PoS tagging where some additional restrictions have to be taken into account. On the other hand, the model merging method starts with a maximum likelihood HMM that directly encodes the training data, that is, where there is exactly one path for each element in the training corpus, and each path is used by one element only. This approximation is not a feasible approach when the resulting HMM will be used in a real environment such as a MT system, in which previously unseen events might occur.

A later work (Brants 95) focuses on the problem of finding the structure of a HMM used for PoS tagging. In that work the author also follows the model merging technique to find the tagset (set of PoS tags) to be used, but this time taking into account some restrictions in order to preserve the information provided by the fine states the initial HMM has. Furthermore, in this work the initial model has one state per part-of-speech, not per word occurrence, but it is trained following a supervised method.

In this paper we explore the use of a bottom-up agglomerative clustering algorithm to obtain the tagset to be used in a HMM-based PoS tagger within a MT system. The initial model is the one obtained using the fine tags delivered by the morphological analyzer of the MT system, trained following an unsupervised method that takes into account information from the target language (TL) (Sánchez-Martínez *et al.* 04a; Sánchez-Martínez *et al.* 04b) to estimate the HMM parameters. We apply the agglomerative clustering procedure both to taggers trained using the TL-driven procedure above and to taggers unsupervisedly trained using the Baum-Welch (Baum 72) algorithm.

The paper is organized as follows: Section 2 overviews the use of HMM for part-of-speech (PoS) tagging. In section 3 the principles of the TL-driven HMM training method are explained; then, in section 4 the clustering strategy is described, and section 5 explains the shallow-transfer MT system used for the TL-driven train-

* Work funded by the Spanish Ministry of Science and Technology through project TIC2003-08681-C02-01, and by the Spanish Ministry of Education and Science and the European Social Found through grant BES-2004-4711.

ing method and the experiments conducted. Finally, in sections 6 and 7 the results are discussed and future work is outlined.

2 Hidden Markov models for part-of-speech tagging

In this section we overview the application of HMMs in the natural language processing field as PoS taggers.

A HMM (Rabiner 89) is defined as $\lambda = (\Gamma, \Sigma, A, B, \pi)$, where Γ is the set of states, Σ is the set of observable outputs, A is the $|\Gamma|\times|\Gamma|$ matrix of state to state transition probabilities, B is the $|\Gamma|\times|\Sigma|$ matrix with the probability of each observable output σ being emitted from each state γ, and the vector π, with dimensionality $|\Gamma|$, defines the initial probability of each state. The system produces an output each time a state is reached after a transition.

When a HMM is used to perform PoS tagging, each HMM state γ is made to correspond to a different PoS tag,[1] and the set of observable outputs Σ are made to correspond to *word classes*. Typically a word class is an *ambiguity class* (Cutting *et al.* 92), that is, the set of all possible PoS tags that a word could receive. Moreover, when a HMM is used to perform PoS tagging, the estimation of the initial probability of each state can be avoided by assuming that each sentence begins with the end-of-sentence mark. In this case, $\pi(\gamma)$ is 1 when γ is the end-of-sentence mark, and 0 otherwise. A deeper description of the use of this kind of statistical models for PoS tagging may be found in (Cutting *et al.* 92) and (Manning & Schütze 99, ch. 9).

3 Target-language training overview

Typically the training of HMM-based PoS taggers is done using the *maximum-likelihood estimate* (MLE) (Gale & Church 90) method when tagged corpora[2] are available (supervised method) or using the Baum-Welch algorithm with untagged corpora[3] (unsupervised method). But, when the resulting PoS tagger is to be embedded as a module of a working MT system, the HMM training can be done in an unsupervised way using information not only from the source-language (SL), but also from the TL. This new training method has been previously described in (Sánchez-Martínez *et al.* 04a; Sánchez-Martínez *et al.* 04b), and is the method used to obtain the initial model that uses the largest possible tagset (that is, the one using the finest possible tags).

The main idea behind the use of TL information is that the correct disambiguation (tag assignment) of a given SL segment will produce a more likely TL translation than any of the remaining wrong disambiguations. In order to apply this method these steps are followed: first the SL text is segmented; then, the set of all possible disambiguations for each text segment are generated and translated into the TL; next, a TL statistical model is used to compute the likelihood of the translation of each disambiguation; and, finally, these likelihoods are used to adjust the parameters of the SL HMM: the higher the likelihood, the higher the probability of the original SL tag sequence in the model being trained.

Let us illustrate how this training method works with the following example. Consider the following segment in English, $s =$ *"He books the room"*, and that an indirect MT system translating between English and Spanish is available. The first step is to use a morphological analyzer to obtain the set of all possible PoS tags for each word. Suppose that the morphological analysis of the previous segment according to the lexicon is: *He* (pronoun), *books* (verb or noun), *the* (article) and *room* (verb or noun). As there are two ambiguous words (*books* and *room*) we have, for the given segment, four disambiguation choices or PoS combinations, that is to say:

- $\boldsymbol{g_1}$ = (pronoun, verb, article, noun),
- $\boldsymbol{g_2}$ = (pronoun, verb, article, verb),
- $\boldsymbol{g_3}$ = (pronoun, noun, article, noun), and
- $\boldsymbol{g_4}$ = (pronoun, noun, article, verb).

The next step is to translate the SL segment into the TL according to each disambiguation $\boldsymbol{g_i}$:

- $\tau(\boldsymbol{g_1}, s) =$ "*Él reserva la habitación*",
- $\tau(\boldsymbol{g_2}, s) =$"*Él reserva la aloja*",
- $\tau(\boldsymbol{g_3}, s) =$"*Él libros la habitación*", and

[1] This is only true when a first-order HMM is considered. In an n-th order HMM each state corresponds to a sequence of n PoS tags.

[2] In a tagged corpus each occurrence of each word (ambiguous or not) has been assigned the correct PoS tag.

[3] In an untagged corpus all words are assigned (using a morphological analyzer) the set of all possible PoS tags independently of context.

- $\tau(\boldsymbol{g_4}, s)$ ="*Él libros la aloja*".

It is expected that a Spanish language model will assign a higher likelihood to translation $\tau(\boldsymbol{g_1}, s)$ than to the other ones, which make little sense in Spanish. So the tag sequence $\boldsymbol{g_1}$ will have a higher probability than the other ones. Finally, the calculated probabilities for each disambiguation $\boldsymbol{g_i}$ are used to estimate the HMM parameters through the MLE method as if they were fractional counts.

4 Tagset clustering strategy

The reason for reducing the number of tags used by PoS taggers is due to the fact that the less tags the tagset has the better the HMM parameters are estimated, through the reduction of the data sparseness problem. Furthermore, as the number of transition probabilities to estimate is, for a first order HMM, quadratic with the number of tags, the number of parameters to store may be drastically reduced.

In order to obtain a coarser tagset we have not followed the model merging strategy already used by Brants (Brants 95) because it is a very time consuming method. Instead, we perform a bottom-up agglomerative clustering on an initial HMM that has as many states as different fine PoS tags the morphological analyzer delivers (see section 5 for details about the different PoS tags delivered by the morphological analyzer).

Bottom-up agglomerative clustering has been used for HMM state clustering (Rivlin *et al.* 97) in speech recognition tasks. One advantage of this clustering algorithm is that the number of clusters (coarse tags) to discover is automatically determined by providing the algorithm with a distance threshold. The algorithm begins with as many clusters as fine tags there are, and in each step those clusters that are closer are merged into a single one only if an additional constraint (see below) is met. The clustering stops when there are no clusters to be merged because their distance is larger than the specified threshold, or the constraint does not hold.

4.1 Constraint on the clustering

A very important property of the resulting tagset is that it must be possible to restore the original information (all grammatical features) represented by the fine tag from the coarser one; note that this is the information we are interested in, as it is used by the subsequent MT modules to carry out the translation. To ensure this property a constraint must hold; this constraint, already used in (Brants 95), establishes that two tags (states) cannot be merged in the same cluster if they share the emission of one or more word class (observables) outputs. This is because in this case, the PoS tagger would not be able to decide on a PoS tag for the observable output.

The previous constraint can be formally described as follows. Let f be a fine tag, c a coarse tag (cluster), σ an observable output, and F, C and Σ the fine tagset, the coarse one and the set of observable outputs, respectively. The original information of the fine tag f can be retrieved from the coarse one c by means of the injective function h defined as:

$$h : \Sigma \times C \rightarrow F \tag{1}$$

To ensure that this function is injective, that is, that for a given observable σ and a given coarse tag c there is only one fine tag f, the next constraint must be met:

$$\forall c \in C, \sigma \in \Sigma, f_1, f_2 \in c, f_1 \neq f_2 : f_1 \in \sigma \Rightarrow f_2 \notin \sigma, \tag{2}$$

where with $f \in c$ we mean that the fine tag f is in the cluster denoted by c, and with $f \in \sigma$ we mean that the observable output σ can be emitted from the fine tag f.

If the constraint expressed in (2) holds, function h is injective, and no information is lost when grouping fine tags into coarser ones.

4.2 Distance between clusters

As an agglomerative clustering will be applied, a distance measure between two clusters is needed in order to measure how similar they are.

Before defining how the distance between two clusters is calculated, let us define how the distance between two fine tags is calculated. The distance between two fine tags is based on the Kullback-Leibler *directed logarithmic divergence* (Kullback & Leibler 51) applied to the probabilistic distributions defined by the transition probabilities A between each fine tag and the rest. The directed logarithmic divergence measures the relative entropy between two probabilistic distributions $p(x)$ and $q(x)$:

$$d(p, q) = \sum_x p(x) \log_2 \frac{p(x)}{q(x)} \tag{3}$$

Since $d(p,q) \neq d(q,p)$, the relative entropy is not a true metric, but it satisfies some important mathematical properties: it is always nonnegative and equals zero only if $\forall x\, p(x) = q(x)$.

As for the clustering algorithm a symmetric distance measure is needed, we use the *intrinsic discrepancy* (Bernardo & Rueda 02) defined as:

$$\delta(p,q) = \min(d(p,q), d(q,p)) \tag{4}$$

Another possibility to make the distance measure symmetric would be to use the *divergence* (Brants 96) defined as

$$\mathrm{Div}(p,q) = d(p,q) + d(q,p) \tag{5}$$

but the intrinsic discrepancy is preferred, among other reasons, because if one probabilistic distribution has null values for some range of X and the other has not, the intrinsic discrepancy is still finite while the divergence is not.

Now that we know how to calculate the distance between two fine tags, we define the way in which the distance between two clusters is calculated. As the intrinsic discrepancy used does not hold the *triangle inequality*, the search space is not a metric one, and calculating a representative for each cluster is not a trivial task. Because of this, the distance between two clusters will be the *unweighted pair-group average*:

$$\delta(c_1,c_2) = \frac{\sum_{t_1 \in c_i} \sum_{t_2 \in c_2} \delta(t_1,t_2)}{\mathrm{card}(c_1)\mathrm{card}(c_2)}, \tag{6}$$

although other distances such as the *weighted pair-group average* or the *minimum/maximum pair-group distance* could also be suitable.

5 Experiments

As has been already mentioned, before applying the clustering algorithm a HMM-based PoS tagger for Spanish is trained using the fine tags delivered by the morphological analyzer. These fine tags have all the morphological information used by the rest of the modules of the MT system. For example, the Spanish word *señal* has the next morphological analysis (fine tag): "noun, feminine, singular", which is different from the fine tag "noun, feminine, plural" given for the word *señales*.

As the previous example illustrates, fine tags discriminate gender, number or, in a verb case, the person who performs the action, among other grammatical features. This causes the number of fine tags to be very large: 1 328 fine tags grouped into 1 594 ambiguity classes in our Spanish lexicon. Notice that the number of HMM transition probabilities to be estimated is quadratic with the number of tags, and the larger the tagset the worse the data sparseness problem.

We have conducted two different experiments for Spanish, one with the initial model trained using information from the TL,[4] as already explained above, and another one in which the initial model is trained using the classical Baum-Welch algorithm; in both cases the training is fully unsupervised.

As has been mentioned, in order to train a HMM-based PoS tagger using information from the TL a working MT system is required. In the next section we overview the MT system used in our experiments. Then we report the results achieved by the TL-driven training method and the Baum-Welch algorithm with the fine tagset, and the results achieved with the tagsets automatically obtained through the bottom-up agglomerative clustering already discussed.

5.1 Machine translation engine

Now we briefly introduce the MT system used in the experiments, although almost any other MT architecture (using a HMM-based PoS tagger) may also be suitable for the TL-driven training algorithm.

We used the Spanish–Catalan (two related languages) MT system interNOSTRUM[5] (Canals *et al.* 00) which basically follows a shallow transfer architecture consisting of the following sequence of stages:[6]

- A *morphological analyzer* tokenizes the text in surface forms (SF) and delivers, for each SF, one or more lexical forms (LF) consisting of *lemma*, *lexical category* and morphological inflection information. The lexical category and the morphological inflection information constitute the fine tag for each LF.
- A *PoS tagger* chooses, using a hidden Markov

[4] For the experiments we use as a TL model a classical trigram language model like the one used in (Sánchez-Martínez *et al.* 04b)

[5] The MT system and the morphological analyzer may be accessed at `http://www.internostrum.com`.

[6] A complete rewriting of this MT engine (Corbí-Bellot *et al.* 05) has been recently released under an open source license (`http://apertium.sourceforge.net`).

Training method	Avg. PoS error
Baum-Welch	$28.7 \pm 2.0\%$
TL based	$25.5 \pm 0.3\%$

Table 1: Average PoS tagging error rate (over ambiguous words only, and without considering unknown words) for the initial HMM that uses the large fine tagset. The error rate reported when the Baum-Welch training algorithm is used is the result of the best of 100 iterations. As can be seen, the standard deviation for the Baum-Welch algorithm is much larger than for the TL-driven algorithm, this is because the Baum-Welch algorithm can fall in a local maxima for some corpora.

model (HMM), one of the LFs corresponding to an ambiguous SF. This is the module whose training is considered in this paper.

- A *lexical transfer* module reads each SL LF and delivers the corresponding TL LF.
- A *structural transfer* module (parallel to the lexical transfer) uses a finite-state chunker to detect patterns of LFs which need to be processed for word reorderings, agreement, etc. and performs these operations.
- A *morphological generator* delivers a TL SF for each TL LF, by suitably inflecting it, and performs other orthographical transformations such as contractions.

5.2 Results

We have applied the presented bottom-up agglomerative clustering on a HMM previously trained using the large (indeed largest possible) initial tagset. Once the initial HMM has been trained the transition probabilities A are used to obtain the coarser tagset. Note that the final number of coarse tags is indirectly determined because the clustering algorithm is provided with a distance threshold.

The experiments have been done with three different corpora in order to know how the clustering algorithm behaves. When using the Baum-Welch algorithm to train the initial model we use three disjoint corpora with around 1 000 000 words each. For the TL-driven training method the corpora used were smaller, around 300 000 words each, because the training algorithm takes much more time, and convergence was reached before processing the whole 300 000 words.

Table 1 shows the average PoS tagging error rate for the two training methods used to obtain the initial HMM used to perform the bottom-up agglomerative clustering. As may be seen, the results achieved by the TL-driven training method are (expectedly) better as was already reported in previous works (Sánchez-Martínez *et al.* 04b). The error rates reported in Table 1 are over ambiguous words only, not over all words, and do not take into account unknown words. The PoS tagging error rate is evaluated using an independent 8 031-word hand-tagged Spanish corpus. The percentage of ambiguous words in that corpus is 26.7% and the percentage of unknown words is 2.0%.

In order to find the threshold that produces the best tagset we have performed the bottom-up agglomerative clustering for thresholds varying from 0 to 2.5 in increments of 0.05. Figure 1 shows the evolution of the PoS tagging error rate with the threshold for one of the corpora used (the remaining two corpora behave in a similar way, the error rate improvement being slightly lower) when using the TL-driven training method to obtain the initial HMM. The PoS tagging error corresponding to the negative threshold is the error rate of the initial HMM using the largest tagset. In that figure the number of coarse tags obtained automatically with each threshold is also shown. It has to be noted that after applying the clustering algorithm the HMM parameters are recalculated using the fractional counts collected during the TL-driven training (this would be equivalent to retraining with the new tagset). Thus, there is no need to retrain the model for each tagset; one simply recalculates the transition and emission probabilities.

As can be seen in Figure 1, with a null threshold value the number of clusters is 327, that is, there are around 1 000 fine tags that have exactly the same transition probabilities. This is because these fine tags are mostly for verbs receiving one (*dame* = "give+me") or two (*dámelo* = "give+me+it") enclitic pronouns, which rarely appear in the training corpus; therefore, the clustering algorithm puts all these fine tags in the same cluster. Furthermore, it can be seen that the best PoS tagger is obtained with a threshold of 1.25, which produces a tagset with only 241 coarse tags. The 241-tag tagset groups in the same cluster, for example, the third person singular tonic pronouns (*consigo* = "with himself/herself/itself", *usted* = "you"), the third person masculine plural tonic pronoun (*ellos* =

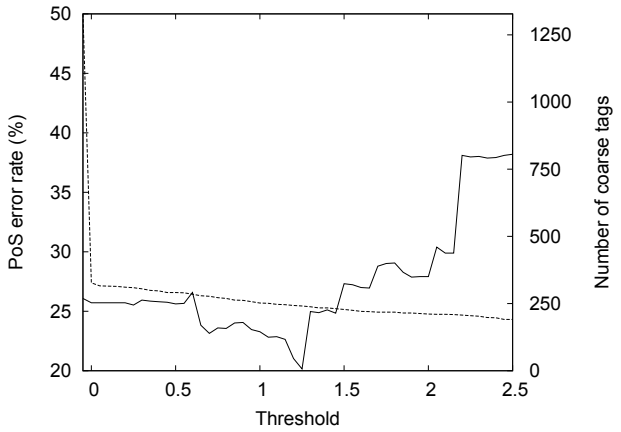


Figure 1: Evolution of the PoS tagging error (solid line with values on the left vertical axe) according to the different threshold values for $d(c_1, c_2)$ used in the experiments, when using as an initial model the one obtained with the TL-driven training method. The number of tags of the obtained tagset for each threshold is also given (dotted line with values on the right vertical axe).

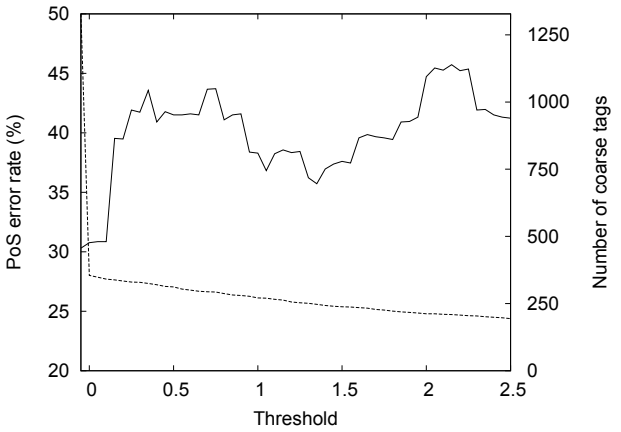


Figure 2: Evolution of the PoS tagging error (solid line with values on the left vertical axe) according to the different threshold values for $d(c_1, c_2)$ used in the experiments, when using as an initial model the one obtain with the Baum-Welch algorithm. The number of tags of the inferred tagset for each threshold is also given (dotted line with values on the right vertical axe).

"they"), the third person neutral tonic pronoun (*ello* = "it"), the third singular tonic pronouns (*nadie* = "no one", *alguien* = "someone", etc.), the third person reflexive tonic pronoun (*sí* = "himself/herself/itself"), and the relative *quien* (= "who/whom"). Furthermore, contrary to what it may be expected some specializations of the same category (for example, feminine adjective and masculine adjective) are assigned to different coarse tags (clusters).

Figure 2 shows the evolution of the PoS tagging error rate and the number of tags for each of the inferred tagset when the initial model is the one obtained using the Baum-Welch algorithm on one of the corpora used (the other two corpora behave in the same way). In this case, after running the clustering algorithm, the HMM was retrained with the new tagset for 100 Baum-Welch iterations.[7] The PoS tagging error rate given in that figure for each threshold is the one provided by the best Baum-Welch iteration. Notice that because of the presence of local maxima in which the Baum-Welch algorithm can fall, the PoS tagging error rate may behave erratically.

As can be seen in Figure 2 clustering does not improve the PoS tagging error rate, and the number of tags of the obtained tagsets for the same threshold values is similar to the number of tags obtained from the TL-trained initial model.

[7] In principle, one could also recalculate the probabilities from the forward-backward auxiliary variables, but we found it easier to simply retrain.

6 Discussion

We have explored the automatic tagset reduction, starting from a large fine tagset, by means of a bottom-up agglomerative clustering algorithm. We have conducted two different experiments: one that uses the Baum-Welch algorithm to obtain the initial HMM with all the fine tags, and another one that uses information from the TL to obtain that initial model.

The results reported show that using the TL-driven training method slightly improves the tagging accuracy, proving that the TL-driven training method is a good unsupervised approach that gives better results than the classical Baum-Welch algorithm.

In the experiments reported in this paper we have not used any smoothing technique to avoid null transition and emission probabilities for those unseen events in the training corpus.

Preliminary experiments using the *expected-likelihood estimate* (ELE) method (Gale & Church 90), which use a very rudimentary smoothing technique, show that the resulting coarse tagset is smaller for equal threshold values. We plan to test whether this still happens when applying a smoothing technique in the maximization step of the Baum-Welch algorithm.

7 Future work

The bottom-up agglomerative clustering uses a distance between clusters. In this paper we have

used the unweighted pair-group average of the intrinsic discrepancy, but other distance measures could also be suitable. We plan to test the minimum pair-group distance which is reported to produce clusters with more disperse elements and the maximum pair-group distance which usually gives more compacted clusters.

In this paper the *intrinsic discrepancy* was used to measure the distance between two fine tags. This measure is finite if one distribution has null values in some range of X and the other not. But, when one probabilistic distribution has null values where the other does this measure becomes infinity. In order to avoid this problem we plan to use the Jensen-Shannon divergence (Grosse *et al.* 02) which is finite for all pairs of distributions.

In one of the papers presenting the TL-driven training method (Sánchez-Martínez *et al.* 04b) the coarse tagset used was manually defined following linguistic guidelines and the method behaved unstably because of the *free-ride phenomenon* (different disambiguations leading to the same translation). We plan to test whether this problem persists with the best automatically inferred tagset.

References

(Baum 72) L. E. Baum. An inequality and associated maximization technique in statistical estimation of probabilistic functions of a Markov process. *Inequalities*, 3:1–8, 1972.

(Bernardo & Rueda 02) J. M. Bernardo and R. Rueda. Bayesian hypothesis testing: A reference approach. *International Statistical Review*, 70:351–372, 2002.

(Brants 95) Thorsten Brants. Tagset reduction without information loss. In *33rd Annual Meeting of the Association for Computational Linguistics (ACLANNUAL'95)*, Cambridge, Massachussetts, USA, 1995.

(Brants 96) Thorsten Brants. Estimating Markov model structures. In H. T. Bunnell and W. Idsardi, editors, *4th International Conference on Spoken Language Processing (ICSLP'96), October 3-6*, volume 2, pages 893–896, Philadelphia, USA, 1996.

(Canals *et al.* 00) Raül Canals, Anna Esteve, Alicia Garrido, M. Isabel Guardiola, Amaia Iturraspe-Bellver, Sandra Montserrat, Pedro Pérez-Antón, Sergio Ortiz, Herminia Pastor, and Mikel L. Forcada. InterNOSTRUM: a Spanish-Catalan machine translation system. *Machine Translation Review*, 11:21–25, 2000.

(Corbí-Bellot *et al.* 05) Antonio M. Corbí-Bellot, Mikel L. Forcada, Sergio Ortiz-Rojas, Juan Antonio Pérez-Ortiz, Gema Ramírez-Sánchez, Felipe Sánchez-Martínez, Iñaki Alegria, Aingeru Mayor, and Kepa Sarasola. An open-source shallow-transfer machine translation engine for the Romance languages of Spain. In *Proceedings of the 10th European Associtation for Machine Translation Conference*, pages 79–86, Budapest, Hungary, May 2005.

(Cutting *et al.* 92) D. Cutting, J. Kupiec, J. Pedersen, and P. Sibun. A practical part-of-speech tagger. In *Third Conference on Applied Natural Language Processing. Association for Computational Linguistics. Proceedings of the Conference.*, pages 133–140, Trento, Italia, 1992.

(Gale & Church 90) William A. Gale and Kenneth W. Church. Poor estimates of context are worse than none. In *Proceedings of a workshop on Speech and natural language*, pages 283–287. Morgan Kaufmann Publishers Inc., 1990.

(Grosse *et al.* 02) Ivo Grosse, Pedro Bernaola-Galván, Pedro Carpena, Ramón Román-Roldán, Jose Oliver, and H. Eugene Standley. Analisys of symbolic sequences using the jensen-shannon divergence. *Physical Review E*, 65(4), 2002.

(Kullback & Leibler 51) S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Math. Stats.*, 22:79–86, 1951.

(Manning & Schütze 99) Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT press, 1999.

(Omohundro 92) Stephen M. Omohundro. Best-first model merging for dynamic learning and recognition. In John E. Moody, Steve J. Hanson, and Richard P. Lippmann, editors, *Advances in Neural Information Processing Systems*, volume 4, pages 958–965. Morgan Kaufmann Publishers, Inc., 1992.

(Rabiner 89) L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

(Rivlin *et al.* 97) Ze'ev Rivlin, Ananth Sankar, and Harry Bratt. HMM state clustering across allophone class boundaries. In *Proc. Eurospeech '97*, pages 127–130, Rhodes, Greece, 1997.

(Sánchez-Martínez *et al.* 04a) F. Sánchez-Martínez, J. A. Pérez-Ortiz, and M. L. Forcada. Cooperative unsupervised training of the part-of-speech taggers in a bidirectional machine translation system. In *Proceedings of TMI, The Tenth Conference on Theoretical and Methodological Issues in Machine Translation*, pages 135–144, October 2004.

(Sánchez-Martínez *et al.* 04b) F. Sánchez-Martínez, J. A. Pérez-Ortiz, and M. L. Forcada. Exploring the use of target-language information to train the part-of-speech tagger of machine translation systems. In *Advances in Natural Language Processing, Proceedings of 4th International Conference EsTAL*, volume 3230 of *Lecture Notes in Computer Science*, pages 137–148. Springer-Verlag, October 2004.

(Stolcke & Omohundro 94) A. Stolcke and S. M. Omohundro. Best-first model merging for hidden Markov model induction. Technical Report TR-94-003, 1947 Center Street, Berkeley, CA, 1994.

Estimating Suboptimal Grammaticality from a Small Latin Corpus

Asad B. Sayeed and **Stan Szpakowicz**
School of Information Technology and Engineering
University of Ottawa
800 King Edward Avenue
Ottawa, Ontario, Canada
K1G 3J2
asayeed@mbl.ca,szpak@site.uottawa.ca

Abstract

In (Sayeed & Szpakowicz 04), we proposed a syntactic representation formalism in a minimalist framework, and a parsing algorithm presented on example derivations. The algorithm parses sentences with discontinuous noun phrases, a phenomenon that occurs in languages such as Latin, using an incremental process informed by a greedy heuristic. While the process handles most types of noun phrase discontinuity, the examples shown exclude a limited number of particularly complicated discontinuities with noun phrases that have at least a noun head and an agreeing adjunct. Our minimalist MERGE operator only merges adjacent words. If the discontinuity is caused by a compatible item, such as a verb, the parser still can build a single tree. It works even if the intervening item is incompatible, but an item compatible both with the head noun and the intervening item is also itself intervening. This accounts for the overwhelming majority of cases, but a few are still problematic. Since they involve very unlikely sequences of case alternations, we posit that these are extremely marked (or ungrammatical) sentences in Latin. In this paper, we describe how to examine a corpus based on the speeches of Cicero to show that such sentences are unlikely to appear in Latin texts. We use a morphological analyzer for Latin that does not disambiguate inflections. A final stage employs human observation on a significantly reduced set of candidate sentences. We find that the types of sentences in question do not exist in Cicero's speeches, so they probably were highly marked.

1 Motivation

Linguistic research is increasingly turning to the use of corpus-based techniques for determining the prevalence of use for various linguistic structures in actual texts. One of the challenges involved is developing tools and techniques to find evidence for the presence of certain structures or to support the scarcity or absence of these forms. Some of these tools can be already found for English; work such as (Resnik & Elkiss 04) takes advantage of the relatively rigid word order of English in order to search databases of sentences by their parse trees. It is harder to use parse trees as search keys in free word order languages such as Latin—we must rely on morphological information, which is often ambiguous. We must also expect a greater number of false positives, given the number of permutations of the structure that we are looking for. That is why we must widen our scope. We present in this work a technique for reducing the number of candidate sentences in a corpus to a manageable size in order to determine the absence of certain linguistic structures, based on predictions from a parser that we discussed in prior work.

In (Sayeed & Szpakowicz 04), we discussed a grammatical formalism and an algorithm for minimalist parsing inspired by (Stabler 97) and (Stabler 01). This minimalist parsing algorithm was designed to be more flexible than Stabler's in order to handle sentences from free word order languages more efficiently. Stabler's formalism employs feature cancellation and strict feature orderings; free word order languages would require a proliferation of permutations of features in the lexicon in order for Stabler's algorithm to parse a significant number of permutations of words in a given sentence.

We illustrated our formalism and algorithm in (Sayeed & Szpakowicz 04) using Latin[1] sentences similar to the following:

[1]There are a few reasons why we work primarily with Latin. We have a wider interest in the grammar of ancient languages. We are familiar with Latin, and it exhibits the grammatical characteristics in which we are most interested, such as noun phrase discontinuous constituency (Sayeed & Szpakowicz 04) and subject extraction from tensed embedded clauses (Sayeed 05b). It also avoids certain complications in other languages that share these characteristics (Sayeed 05a). Finally, it gives a very good rationale for pursuing work (such as this paper) on negative evidence for grammaticality.

pater laetus amat filium laetum
$\text{father}_{\text{nom}}$ $\text{happy}_{\text{nom}}$ $\text{loves}_{\text{3sg}}$ son_{acc} $\text{happy}_{\text{acc}}$
'The happy father loves the happy son.'

Our parser handles most word permutations in sentences such as this. Still, some permutations with certain discontinuous noun phrases cannot be parsed without relaxing many of the major restrictions on the parser that decide on its incremental nature. (Such restrictions have a reasonable psycholinguistic justification.) nature. The parser depends on a definition of the minimalist operator MERGE[2] that requires its operands to be adjacent in the sentence or on the list of intermediate trees built from the sentence. In the spirit of minimalism, we consider such relaxations of the restrictions on the formalism—in order to allow a small number of word orders to be parsed by our algorithm—as a last resort[3].

Thus we must first determine whether these sentences (which we describe shortly) need to be parsed in the first place—in other words, whether they are actually valid in Latin. It is usually assumed that the complex morphology and attendant agreement requirements of languages like Latin ensure that any permutation is permitted. Even native speakers of languages such as Russian may overintellectualize[4] the word order liberty they have, convincing themselves that otherwise awkward-sounding word orders that never appear may actually be valid. In addition, the lack of native speakers for a language such as Latin precludes even determining whether a word order sounds awkward.

Consequently, we decided to embark on the development of tools that aid in the analysis of Latin texts to determine what word orders we can legitimately consider unparsable in order to minimize the changes we have to make to the parser. Our hypothesis was that the problematic word orders never occur; we designed the experiment to look for sentences that challenge this hypothesis and demonstrate that they are absent from the texts we examined. This paper describes the data we used, the software we developed to analyze the data, and the results of our exploration. Observe that a thorough corpus examination is a much larger project than what we have undertaken so far, requiring a great deal of manual examination of texts; we are describing a pilot study that has given us noteworthy preliminary results.

[2] MERGE combines trees at their roots, given that the root of one tree is a constituent of the other.

[3] For a more detailed discussion of the algorithm and formalism itself, see (Sayeed & Szpakowicz 04).

[4] In other words, they may assume facts about Russian grammar from such sources as their formal education. Such a possibility has been recognized as far back as (Chomsky 77), who makes a similar point in another context about this issue.

2 The Data

The data we used were provided by the Perseus Project[5] from compilations of speeches by M. Tullius Cicero. We chose Cicero because he was a prose writer whose use of language was considered the most skilled among Roman writers and speakers for generations; his language was varied and complex, and he is more likely to provide the full range of plausible Latin sentences than most other writers. In Appendix A we list the speeches we used.

The corpus is segmented into sentences each with its own reference code; there is no division into phrases, meaning that a sentence can have multiple complete clauses. The Perseus Project also provided a lexicon in XML containing morphological analyses of every word in this Cicero corpus. For most words, there were several analyses—massive morphological ambiguity is the rule rather than the exception in Latin. We converted this lexicon into a Prolog database for the use described in the later sections.

3 Methodology and Implementation

We are interested in sentences containing the following items:

1. A noun in the nominative ($\mathbf{N}_{nom}$).
2. An adjective in the nominative that agrees in gender and number with the noun in the nominative ($\mathbf{A}_{nom}$).

[5] `http://www.perseus.tufts.edu/`

3. A noun in the accusative ($\mathbf{N}_{acc}$).

4. An adjective in the accusative that agrees in gender and number with the noun in the accusative ($\mathbf{A}_{acc}$).

5. A verb ($\mathbf{V}$) that agrees in person and number with $\mathbf{N}_{nom}$.

We encoded these definitions and relationships as Prolog predicates in a way that could detect the presence of these items in a list of words.

The orders that we want to exclude are:

- $\mathbf{V}\ \mathbf{A}_{nom}\ \mathbf{A}_{acc}\ \mathbf{N}_{nom}\ \mathbf{N}_{acc}$
- $\mathbf{V}\ \mathbf{A}_{acc}\ \mathbf{A}_{nom}\ \mathbf{N}_{acc}\ \mathbf{N}_{nom}$
- $\mathbf{V}\ \mathbf{A}_{acc}\ \mathbf{A}_{nom}\ \mathbf{N}_{nom}\ \mathbf{N}_{acc}$
- $\mathbf{V}\ \mathbf{A}_{acc}\ \mathbf{N}_{nom}\ \mathbf{N}_{acc}\ \mathbf{A}_{nom}$
- $\mathbf{V}\ \mathbf{A}_{acc}\ \mathbf{N}_{nom}\ \mathbf{A}_{nom}\ \mathbf{N}_{acc}$
- $\mathbf{V}\ \mathbf{N}_{nom}\ \mathbf{A}_{acc}\ \mathbf{A}_{nom}\ \mathbf{N}_{acc}$
- $\mathbf{A}_{nom}\ \mathbf{V}\ \mathbf{A}_{acc}\ \mathbf{N}_{nom}\ \mathbf{N}_{acc}$
- $\mathbf{N}_{nom}\ \mathbf{V}\ \mathbf{A}_{acc}\ \mathbf{A}_{nom}\ \mathbf{N}_{acc}$
- $\mathbf{N}_{acc}\ \mathbf{A}_{nom}\ \mathbf{A}_{acc}\ \mathbf{V}\ \mathbf{N}_{nom}$
- $\mathbf{N}_{acc}\ \mathbf{N}_{nom}\ \mathbf{A}_{acc}\ \mathbf{V}\ \mathbf{A}_{nom}$
- $\mathbf{A}_{nom}\ \mathbf{N}_{acc}\ \mathbf{N}_{nom}\ \mathbf{A}_{acc}\ \mathbf{V}$
- $\mathbf{N}_{acc}\ \mathbf{A}_{nom}\ \mathbf{A}_{acc}\ \mathbf{N}_{nom}\ \mathbf{V}$
- $\mathbf{N}_{acc}\ \mathbf{A}_{nom}\ \mathbf{N}_{nom}\ \mathbf{A}_{acc}\ \mathbf{V}$
- $\mathbf{N}_{acc}\ \mathbf{N}_{nom}\ \mathbf{A}_{acc}\ \mathbf{A}_{nom}\ \mathbf{V}$
- $\mathbf{N}_{nom}\ \mathbf{N}_{acc}\ \mathbf{A}_{nom}\ \mathbf{A}_{acc}\ \mathbf{V}$
- $\mathbf{N}_{acc}\ \mathbf{N}_{nom}\ \mathbf{A}_{nom}\ \mathbf{A}_{acc}\ \mathbf{V}$

We refer to permutations of our five types as *sentence classes*. We call these 16 permutations *problematic sentence classes* in order to emphasize their undesirability. What do they have in common? In all of them, there is an $\mathbf{A}_{acc}$ somewhere between a nominative form and a verb without also being next to a $\mathbf{N}_{acc}$. Since MERGE in (Sayeed & Szpakowicz 04) imposes agreement requirements, $\mathbf{A}_{acc}$ forms cannot MERGE with nominative forms (being accusative), and they also cannot MERGE with $\mathbf{V}$ forms, unlike $\mathbf{A}_{nom}$ forms—we assume that verbs take optional subjects, but not optional objects[6]. This $\mathbf{A}_{acc}$ form obstructs the nominative form from becoming adjacent to the verb in order to permit MERGE; however, were it next to its corresponding $\mathbf{N}_{acc}$, it would MERGE with the $\mathbf{N}_{acc}$, which can in turn MERGE with $\mathbf{V}$, no longer obstructing the nominative forms from also merging with the verb[7].

Our overall process was the following. For each sentence in the corpus:

1. Strip the punctuation.

2. Break the string into words.

3. Find the first five words that match the five types above.

4. Find all the type-assignments for these words. (There may be many due to morphological ambiguity.)

5. Determine to which sentence class each type-assignment belongs. For each type-assignment:

   (a) Determine if it belongs to a problematic sentence class.

   (b) If so, add the type-assignment (consisting of the words used associated with one of the above types) and the reference code of the sentence to which the type-assignment belongs to the list of sentences in that class (which is written out to a file).

We could have chosen every set of five consecutive relevant words in the sentence, but

[6] We obtain an elegant generalization from this which we discuss in forthcoming work. In short, allowing optional subjects implies that adjuncts to the subject may appear in the sentence without the explicit subject itself; this means that we must allow them to MERGE with the verb.

[7] A complete explanation why this is so would be neither simple nor straightforward. Readers interested in looking at the underlying issues further are invited to see (Sayeed & Szpakowicz 04) and (Sayeed 05a), the latter in particular, for a thorough background on the problem. We do not go into further detail here, as it is well beyond the specific scope of the work we are presenting.

this would have vastly increased the number of situations in which a five-word set would cross a clause boundary. Already, with only the first five relevant words being considered, there are many instances that straddle the clause boundary. Having effectively a five-word "window" traversing the sentences would increase this many times, drowning out any valid examples of these undesirable sentence classes, *if* they exist. The only way to go meaningfully beyond the restriction is to develop robust Latin clause chunking. The restriction to the first five relevant words at least provides an increased likelihood that all five will belong to the same clause.

Since we cannot completely avoid five-word sets that do cross clause boundaries, there may be many spurious identifications of sentences in problematic classes; morphological ambiguity also contributes to this. We hypothesize, in fact, that *all* of them are spurious. In effect, we needed to examine any that appear after the algorithm is run on the texts and identify them as incorrectly classifed; or we needed to do this to enough of them that we would be relatively confident that we would not encounter a genuine example of a sentence in a problematic class.

Given that the Cicero corpus is quite large for a task of this nature (20,082 sentences), how would we compute the number of type-assignments we need to check to get an acceptable confidence interval and level? To do this, we use the formula for sample size determination:

$$n = \left(\frac{z}{\delta}\right)^2 \hat{\Pi}(1 - \hat{\Pi}) \qquad (1)$$

n is the sample size, z – a value related to the confidence level required (usually 95%, giving $z = 1.96$), δ is the confidence interval, and $\hat{\Pi}$ is the proportion of the sample expected to give a certain value (Mansfield 91) (in our case, whether the type-assignment was spurious). We can also turn it around and solve for the confidence interval:

$$\delta = z\sqrt{\frac{\hat{\Pi}(1 - \hat{\Pi})}{n}} \qquad (2)$$

We then randomly select n type-assignments from the population and examine whether they are valid assignments given the structure and context of the sentences to which they are assigned. This step is manual and very time-consuming; it requires a careful examination not only of the sentence in question, but of potentially many sentences around it, in order to determine whether a word in a sentence has been assigned the correct type given the massive type-ambiguity that can exist in a Latin sentence.

If we ever encounter a single clear example of an undesirable sentence, then this investigation can, in theory, stop; and we can say that sentences in the undesirable classes exist in Latin, and in future work, we would have to find a way to include them. But is this really so? Perhaps, if it turns out that there are extremely few sentences in undesirable classes compared to other classes, we could still attribute this to other factors, such as stylistic factors or even a simple mistake.

If we do not encounter any examples of an undesirable sentence in our sample, then obviously we cannot *directly* assume that there are none in the corpus, or that such sentences are completely ungrammatical. Using the formulae above, however, we would be able to compute the maximum likelihood (δ) that we might encounter one in the future, given that all of the previous did not belong to the undesirable classes. If that likelihood is low, we would be able to build a case that a parser reflective of human linguistic competence need not be able to parse these sentences.

4 Results and Analysis

Though the process of collecting the results was fairly complicated, the results themselves are quite simple:

- In the corpus, there were 20,082 sentences in total, sentences often composed of many clauses.
- Of these, 18,414 had more than five words.

- Of these, 16,719 did not contain the requisite five types. This means that 1,695 sentences contained the five types.
- These 1,695 sentences produced 5190 possible type assignments in total.
- Of these, only 356 were *potentially* of the undesirable sentence classes.

In other words, only a small percentage (9.2%) of the sentences longer than five words were actually relevant to this study in containing the five word types. Of those, only 21% of the sentences containing those word types were potential members of undesirable sentences classes (compared with the undesirable sentence classes that make up $16/120 = 13.3\%$ of the total number of sentence classes).

We examined 176 of the analyses. All of them were spurious. Here is an example of a sentence for which a spurious analysis was found:

> *ac si* **quis est talis qualis** *esse* **omnis** *oportebat, qui in hoc ipso in quo exsultat et triumphat oratio mea me vehementer accuset, quod tam capitalem hostem non comprehenderim potius quam emiserim, non est ista mea culpa, Quirites, sed temporum.* (Cic. Catil. 2.3)

The words in bold are those that were identified as the first words in the sentence that could potentially fit an undesirable analysis.

- *quis* is $\mathbf{N}_{nom}$.
- *est* is $\mathbf{V}$.
- *talis* is $\mathbf{A}_{acc}$.
- *qualis* is $\mathbf{A}_{nom}$.
- *omnis* is $\mathbf{N}_{acc}$.

Given the parsing algorithm described in this work, we can see that neither adjective can reach its corresponding noun; they entirely block each other. But this is only true assuming that this type-assignment is correct. In reality, *talis* and *qualis* (together, "of such a nature") are intended to be of the same case, not different cases; it so happens that for both of them their nominative and accusative cases are identical. So the example is spurious and can be crossed off the list.

Since we have examined a significant number of such examples selected at random, we can now apply the formula for δ to find the confidence interval given a confidence level of 95%. Since all of them were spurious, we can make $\hat{\Pi} = 1$ (100%). This makes $1 - \hat{\Pi} = 0$. So $\delta = 0$; this result, though extremely desirable, is however probably not a useful representation of the situation. It predicts a perfect relationship between the sample data and the actual population partly by assuming that the lone human sentence-classifier made no mistakes in the rejection of all the undesirable analyses. To resolve this, we must assume a small amount of error on the part of the person doing the classification; in which case, if we assume that 1% of the spurious analyses might actually be real (even though the classifier did not report this), then we actually get a confidence interval of 1.47%.

The actual size of the population is 356, so we sampled and classified about half of them. In that case, the actual interval given a 95% confidence level—and assuming that (although the human classifier found 100% of the sample to be spurious) we will use 99% as the number that we are certain must be spurious—would probably be less than 1.47%.

5 Conclusions and Future Work

We have outlined a method of substituting partially automated studies of corpora for grammaticality judgements. It can apply when grammaticality judgements are impossible to obtain, as in the case of an ancient language like Latin. We applied the method to some of the works of Cicero to determine whether limitations on the kinds of noun phrase discontinuities accepted by our minimalist parser would be detrimental to its coverage of Latin sentences. We found that, at least in the Cicerine corpus we used, it was

unlikely that the sentences we were excluding from the parser (on psycholinguistic grounds) would actually appear. We can apply these techniques in the future to evaluating the validity of other predictions made by our approach to minimalist parsing, such as (Sayeed 05b).

This method substitutes the judgement of a non-native speaker on whether some sentences fit into some sentence class for the judgement of a native speaker who could give, in theory, a clear and direct response on the grammaticality of a sentence. One vital element, however, is missing: an assessment of the accuracy of the non-native-speaker evaluator. Due to resource constraints, we could only employ one evaluator (it is difficult to find people proficient in Latin willing to perform time-consuming evaluation tasks).

Specific to this experiment, we used only some of the material from a single Latin author. Selecting a single orator such as Cicero with interestingly varied prose allows us to use this technique to simulate the competence of a Latin speaker without having to worry about language variation over dialect and time, which may have been the result of using multiple authors. It is nevertheless necessary to expand the corpus so that we can find more than 356 eligible sentences to sample and test for membership in the problematic classes. This we may be able to do by including some of Cicero's contemporaries, such as Julius Caesar. To ensure maximum correspondence in their internal knowledge of Latin, we would have to examine the biographies of such authors so as to ensure that they would have absorbed Latin in roughly the same linguistic environments.

It remains a somewhat philosophical problem whether the results obtained from such investigations actually reflect linguistic competence in such a way that a parser based on them can be held to be also a reflection of a native human parser. After all, it could be that some of the sentences that do not appear in corpora are simply unrealized potential of the linguistic competence of the writer. Nevertheless, for a "dead" language, it is necessary to accept that we may never know the answer to this; we are, however, confident that in the specific parts of Latin syntax on which we have decided to focus, our corpus investigation has brought us to the closest approximation of that knowledge that we are likely ever to get. There is also other recent work for languages such as German that makes use of negative evidence from corpora to estimate graded grammaticality judgements (Kepser *et al.* 04).

Acknowledgements

Partial support for this work comes from the Natural Sciences and Engineering Research Council of Canada. The Perseus Project at Tufts University generously provided us with some of their materials in a machine-readable format, namely the texts of Cicero and the output of their morphological analyser. Michael Duda, an undergraduate student at the University of Ottawa, wrote parts of the software in the course of his honours project.

References

(Chomsky 77) Noam Chomsky. On wh-movement. In Peter Culicover, Thomas Wasow, and Adrian Akmajian, editors, *Formal Syntax*, pages 71–132. Academic Press, New York, 1977.

(Kepser *et al.* 04) Stephan Kepser, Ilona Steiner, and Wolfgang Sternefeld. Annotating and querying a treebank of suboptimal structures. In Sandra Kübler and Joakim Nivre, editors, *Treebanks and Linguistic Theories 2004*, pages 63–74, Tübingen, Germany, 2004.

(Mansfield 91) Edwin Mansfield. *Statistics for Business and Economics*. W. W. Norton, New York, 4th edition, 1991.

(Resnik & Elkiss 04) Philip Resnik and Aaron Elkiss. The linguist's search engine: Getting started guide. Technical Report LAMP-TR-108/CS-TR-4541/UMIACS-TR-2003-109, University of Maryland, College Park, 2004. Update of 20 January 2004 (`http://lse.umiacs.umd.edu/lse_guide.html`).

(Sayeed & Szpakowicz 04) Asad B. Sayeed and Stan Szpakowicz. Developing a minimalist parser for free word order languages with discontinuous constituency. In José Luis Vicedo González, Patricio Martínez-Barco, Rafael Muñoz, and Maximiliano Saiz-Noeda, editors, *EsTAL*, volume 3230 of *Lecture Notes in Computer Science*, pages 115–126. Springer, 2004.

(Sayeed 05a) Asad Sayeed. Developing a minimalist parser for free word order languages. Unpublished M.Sc. thesis, University of Ottawa, 2005. `http://www.umiacs.umd.edu/~asayeed/uottawa/thesis/thesis.pdf`.

(Sayeed 05b) Asad Sayeed. Minimalist parsing of subjects displaced from embedded clauses in free word order languages. In *Proceedings of the Association for Computational Linguistics 43rd annual meeting, student session*, Ann Arbor, Michigan, 2005.

(Stabler 97) Edward P. Stabler. Derivational minimalism. In *LACL '96: Selected papers from the First International Conference on Logical Aspects of Computational Linguistics*, pages 68--95, London, UK, 1997. Springer-Verlag.

(Stabler 01) Edward P. Stabler. Minimalist grammars and recognition. In Christian Rohrer, Antje Roßdeutscher, and Hans Kamp, editors, *Linguistic Form and its Computation*, pages 327--352. CSLI Publications, Stanford, 2001.

A The Corpus

In this paper, we wrote that we used compilations of some of Cicero's speeches for our corpus study. We list them in this appendix.

The speeches were all compiled and edited by Albert Clark.

- *Orationes: Cum Senatui gratias egit, Cum populo gratias egit, De domo sua, De haruspicum responso, Pro Sestio, In Vatinium, De provinciis consularibus, Pro Balbo*
- *Orationes: Divinatio in Q. Caecilium, In C. Verrem*
- *Orationes: Pro Milone, Pro Marcello, Pro Ligario, Pro rege Deiotaro, Philippicae I-XIV*
- *Orationes: Pro P. Quinctio, Pro Q. Roscio comoedo, Pro A. Caecina, De lege agraria contra Rullum, Pro C. Rabiro perduellionis reo, Pro L. Flacco, In L. Pisonem, Pro C. Rabiro Postumo*
- *Orationes: Pro Sex. Roscio, De imperio Cn. Pompei, Pro Cluentio, In Catilinam, Pro Murena, Pro Caelio*
- *Orationes: Pro Tullio, Pro Fonteio, Pro Sulla, Pro Archia, Pro Plancio, Pro Scauro*

Semantic Indexing using Minimum Redundancy Cut in Ontologies*

Florian Seydoux and **Jean-Cédric Chappelier**
School of Computer and Communication Sciences
École Polytechnique Fédérale de Lausanne (EPFL)
CH-1015 Lausanne, Switzerland
{florian.seydoux,jean-cedric.chappelier}@epfl.ch

Abstract

This paper presents a new method that improves semantic indexing while reducing the number of indexing terms. Indexing terms are determined using a minimum redundancy cut in a hierarchy of conceptual hypernyms provided by an ontology (e.g. *WordNet*, *EDR*). The results of some information retrieval experiments carried out on several standard document collections using the *WordNet* and *EDR* ontologies are presented, illustrating the benefit of the method.

1 Introduction

Three fields are mainly reported in the literature about the use of semantic knowledge for Information Retrieval: *query expansion* (Voorhees 94; Moldovan & Mihalcea 00), *Word Sense Disambiguation* (Ide & Véronis 98; Wilks & Stevenson 98; Besançon *et al.* 01) and *semantic indexing*. This contribution relates to the latest, the main idea of which is to use word senses rather than, or in addition to, the words[1] for indexing document, in order to improve both recall (by handling synonymy) and precision (by handling homonymy and polysemy). However, the experiments reported in the litterature lead to contradicting results: some claim that it degrades the performance (Salton 68; Harman 88; Voorhees 93; Voorhees 98); whereas for others the gain seems significant (Richardson & Smeaton 95; Smeaton & Quigley 96; Gonzalo *et al.* 98a; Gonzalo *et al.* 98b; Mihalcea & Moldovan 00).

Although it is definitely seems desirable for IR systems to take a maximum of semantic information into account, the resulting expansion of the data processed may not develop its full potential. Indeed, the growth of the number of index terms not only increases the processing time but could also reduce the precision as discriminating documents by using a very large number of index terms is a hard task.

This problem is not new, and various techniques aiming at reducing the size of the indexing set already exist: filtering by stoplist, part of speech tags, frequencies, or through statistical techniques as in LSI (Deerwester *et al.* 90) or PLSI (Hofmann 99). However, most of these techniques are not adapted to the case where an explicit semantic information is available, for example in the form of a thesaurus or an ontology (i.e. with some underlying formal – not statistical – structure).

The focus of the work presented here is to use external[2] structured semantic resources such as an ontology in order to limit the semantic indexing set. This work, which is a continuation of (Seydoux & Chappelier 05), relates, but from a different point of view, with experiments described in (Gonzalo *et al.* 98b), (Whaley 99) or (Mihalcea & Moldovan 00), which uses the synsets (or hypernyms synsets (Mihalcea & Moldovan 00)) of *WordNet* as indexing terms. We follow the *onto-matching* technique described in (Kiryakov & Simov 99), but here selecting the indexing set using an information theory based criterion, the *Minimum Redundancy Cut* (*MRC*, see figure 1), applied to the inclusive "is-a" relation (hypernyms) provided by the *WordNet* (Fellbaum 98; Miller 95) and *EDR* taxonomies (Miyoshi *et al.* 96).

2 Ontology-Cut Model

2.1 Goals

The choice of the appropriate hypernym (a "concept" in the ontology) to be used for representing a word is not easy: be it too general, the performance of the system will degrade (lack of precision); be it too specific, the indexing set will

* This work was partially supported by the Swiss National Fund for Scientific Research (SNFSR) under grant n°200020–103529.

[1] Usually lemmas or stems.

[2] By "external" we mean "not directly related to the document collection itself".

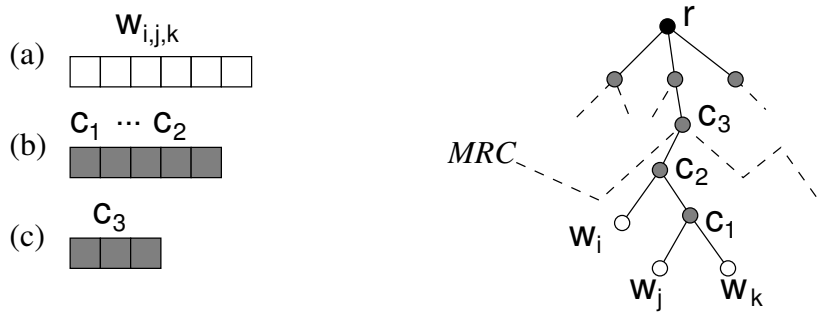


Figure 1: Several indexing scheme: (a) usual indexing with words, stems or lemmas; (b) synset (or hypernyms synsets) indexing: each indexing term is replaced by its (hypernyms) synset; this, in principle, reduces the size of the indexing set since all the indexing terms that are shared by the same hypernym are regrouped in one single indexing feature; (c) Minimum Redundancy Cut (*MRC*) indexing: each indexing term is replaced by its dominating concept chosen with *MRC*. This furthermore reduces the size of the indexing set since all the indexing terms that are subsumed by the same concept in the *MRC* are regrouped in one single indexing feature.

not reduce enough, preserving some distinction between words with close senses (lack of recall).

To select the appropriate level of conceptual indexing, we consider cuts in the ontology. A cut in the directed acyclic graph (DAG) representing the ontology is defined as a minimal subset[3] of nodes in the ontology defining a coverage of all the leaf nodes (i.e. words). Each node in the cut then represents every leaf node it dominates.

The problem is to find a computable strategy to select an optimal cut. We propose to use an information theory based criterion, that selects a cut for which the redundancy is minimal.

2.2 Minimum Redundancy Criterion

Let $\mathcal{N} = \{n_i\}$ represent the set of nodes and $\mathcal{W}$ the set of words in the ontology. A cut Γ is defined as a minimal subset[3] of $\mathcal{N}$ which covers $\mathcal{W}$. A probabilized cut $M = (\Gamma, P)$ is a couple consisting of a cut Γ and a probability distribution P on Γ. Finally, $|\Gamma|$ denotes the number of nodes in the cut Γ (and similarly $|M| = |\Gamma|$).

From now on, the probabilized cut $M = (\Gamma, P_f)$ is considered, where P_f is defined using the relative frequencies of the words in the collection:

$$P_f(n_i) = \frac{f(n_i)}{|D|},$$

$f(n_i)$ being the number of occurrences of the node n_i in a document collection D. To compute $f(n_i)$, we consider that an occurrence of n_i happens when any of the hyponym words of n_i occurs.

[3] "minimal subset" means that no node can be removed from the set without decreasing it's coverage.

The redundancy $R(M)$ of a probabilized cut $M = (\Gamma, P)$ is defined as (Shannon 48):

$$R(M) = 1 - \frac{H(M)}{\log |M|},$$
$$\text{where } H(M) = -\sum_{n \in \Gamma} P(n) \cdot \log P(n).$$

Minimizing the redundancy is thus equivalent to maximizing the ratio between the entropy $H(M)$ of the cut and its maximum possible value ($\log |M|$), i.e. balancing as much as possible the probabilities of the nodes in the cut.

Notice that R does not necessarily have a unique minimum, but the thesaurus may rather have several equally minimal cuts. In practice, this can easily be overcome, considering for instance any of the minimal cuts, or those having a minimal number of nodes, or the minimum average depth of the nodes, etc.

In order to identify global *MRC*, the whole set of possible cuts has to be considered. We thus decided to give up global optimality for the sake of tractability and focussed rather on more efficient heuristics.

The proposed algorithm consists, starting from the leaves, in iteratively modifying a given cut by systematically replacing a node by its parent or its children that minimizes the redundancy. For each node n_i in the current cut, we consider on one hand ${n_i}^{\downarrow}$ the (set of) children of n_i, and on the other hand ${n_i}^{\uparrow}$ the (set of) parents of n_i. Due to the DAG structure, this replacement can involve other nodes in the cut. In fact, when replacing

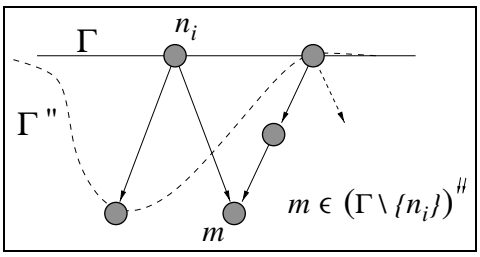


Figure 2: Lower search: node n_i is replaced by $n_i{}^{\downarrow}$ without the nodes already covered by other nodes in Γ (e.g. m), i.e. $(\Gamma \setminus \{n_i\})^{\Downarrow}$.

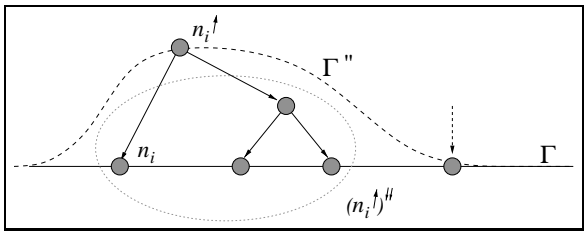


Figure 3: Upper search: node n_i is replaced by $n_i{}^{\uparrow}$, and all nodes covered by $n_i{}^{\uparrow}$ (i.e. $(n_i{}^{\uparrow})^{\Downarrow}$) are removed from Γ.

n_i by $n_i{}^{\downarrow}$, those nodes which are already covered by other nodes in the cut must be excluded, i.e. consider $n_i{}^{\downarrow} \setminus (\Gamma \setminus \{n_i\})^{\Downarrow}$ instead of $n_i{}^{\downarrow}$ (see fig. 2), where $n^{\Downarrow}$ stands for the transitive closure of $n^{\downarrow}$; and similarly for $n_i{}^{\uparrow}$ (see fig. 3).

Then, the cut with minimal redundancy among these new considered cuts and the current one is kept, and the search continue as long as better cuts are found. The full algorithm[4] is given hereafter (Algorithm 1).

This algorithm converges towards a local minimum redundancy cut close to the leaves. Note that this algorithm can be stopped at any time, if required, since it always works on a complete cut.

2.3 Example

Let us illustrate the *MRC* on the toy example of the ontology given in figure 4. With this data, the redundancy of the example cut Γ = [Animal, Plant, Transport] is given by:

n	Animal	Plant	Transport
$f(n)$	20	33	2
$P_f(n)$	0.3704	0.5926	0.0370
$-P_f(n)\log_2 P_f(n)$	0.5307	0.4473	0.1761
$R(\Gamma) = 1 - \frac{1.1541}{\log_2(3)} = 0.2718$			

[4] In practice, several optimizations can be made, which do not conceptually change the algorithm and are thus not presented here for the sake of clarity.

Algorithm 1 *MRC* local search algorithm

Requires: a hierarchy $\mathcal{N}$ (the leaves of which are $\mathcal{W}$)
Provides: a cut Γ with (local) minimal redundancy

$\Gamma \leftarrow \mathcal{W}$ *# current cut, start from the leaves*
repeat
  $\Gamma' \leftarrow \varnothing$ *# best new cut*
  $\Gamma'' \leftarrow \varnothing$ *# tested candidate*
  continue $\leftarrow$ false *# search-loop control flag*
  for all $n_i \in \Gamma$ **do**
    # Evaluate the children's cut:
    $\Gamma'' \leftarrow (\Gamma \setminus \{n_i\}) \cup \left(n_i{}^{\downarrow} \setminus (\Gamma \setminus \{n_i\})^{\Downarrow}\right)$
    $\Gamma' \leftarrow \text{Argmin}\left(R(\Gamma'), R(\Gamma'')\right)$
    # Evaluate each parent's cut:
    for all $n_j \in n_i{}^{\uparrow}$ **do**
      $\Gamma'' \leftarrow (\Gamma \cup \{n_j\}) \setminus n_j{}^{\Downarrow}$
      $\Gamma' \leftarrow \text{Argmin}\left(R(\Gamma'), R(\Gamma'')\right)$
  if $R(\Gamma') < R(\Gamma)$ **then**
    $\Gamma \leftarrow \Gamma'$ *# keep the best cut*
    continue $\leftarrow$ true *# the search goes on*
    # some watchdog or timer can be put here
until continue is false
return Γ
with $R(\varnothing) = R(\{c\}) = 1$, by convention.

In this case, by examining each of the 2036 possible cuts, one can check that the global *MRC* is also the local one found by the local search algorithm (with only 117 evaluation), and for which the redundancy is 0.07092 (see figure 4).

Regarding the considered indexing schemas (see fig. 1), consider the following three documents:

d_1	myosotis tree bicycle myosotis lion
d_2	lion cow carnivore
d_3	violet car fir carnivore

The baseline words indexing (scheme (a)) gives:

$\text{Id}_1^{(a)}$	bicycle:1 lion:1 myosotis:2 tree:1
$\text{Id}_2^{(a)}$	carnivore:1 cow:1 lion:1
$\text{Id}_3^{(a)}$	car:1 carnivore:1 fir:1 violet:1

Indexing by direct hypernyms (scheme (b)) gives:

$\text{Id}_1^{(c)}$	BlueFl:2 Carnivore:1 Ecological:1 Tree:1
$\text{Id}_2^{(c)}$	Carnivore:2 Herbivore:1
$\text{Id}_3^{(c)}$	BlueFl:1 Carnivore:1 Pollutant:1 Tree:1

Indexing by example cut Γ gives:

$\text{Id}_1^{(\text{ex. }\Gamma)}$	Animal:1 Plant:3 Transport:1
$\text{Id}_2^{(\text{ex. }\Gamma)}$	Animal:3
$\text{Id}_3^{(\text{ex. }\Gamma)}$	Animal:1 Plant:2 Transport:1

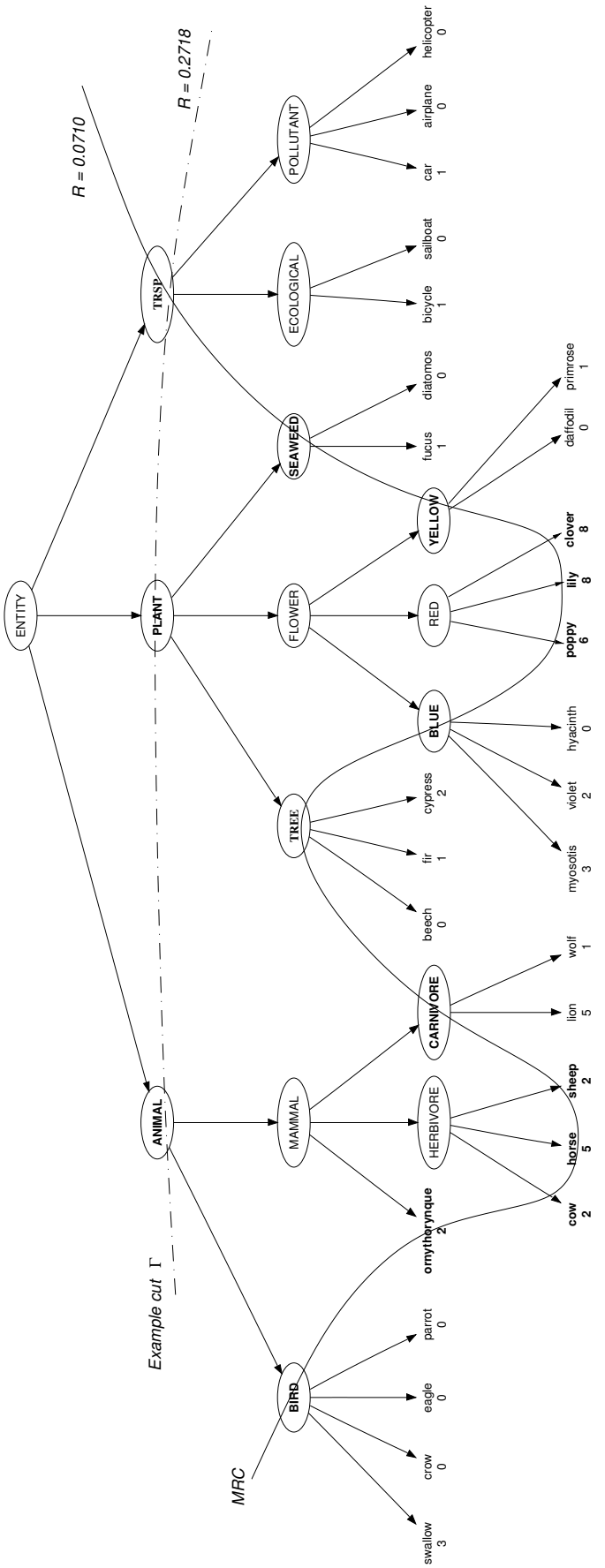


Figure 4: Examples of cuts and their redundancy value in a toy ontology. Frequencies $f(n)$ of words and concepts (from the dataset) are indicated. Items of the *MRC* cut are in bold; items of the example cut Γ = [Animal, Plant, Transport] are in italic. The redundancy is 0.071 for the *MRC*, 0.272 for Γ and 0.093 for the set of leaves.

Finally, indexing by *MRC* cut (scheme (c)) gives:

$Id_1^{(d)}$	BLUEFL:2 CARNIVORE:1 TRANSPORT:1 TREE:1
$Id_2^{(d)}$	cow:1 CARNIVORE:2
$Id_3^{(d)}$	BLUEFL:1 CARNIVORE:1 TRANSPORT:1 TREE:1

3 Experiments

We carried out several experiments with standard english document collections of the SMART system[5], and ontologies generated from the MYSQL port of *WordNet* [6] and the english part of *EDR Electronic Dictionary*.

WordNet gather information about approximatively 200,000 "words" ($\approx$ 150,000 different lexical strings including compounds) of type noun, verb, adjective and adverb; organized into $\approx$ 115,000 synsets, with $\approx$ 100,000 hypernyms relations between them.

EDR gather information about approximatively 420,000 "words" ($\approx$ 240,000 different lexical strings, including compounds and idiomatic expressions) of differents type (whithout restriction on POS); organized into $\approx$ 490,000 concepts, with $\approx$ 500,000 super/sub relations between them; we gather here the two differents ontologies provided by *EDR* (a very large scale general ontology and a smallest one specialized on information science).

3.1 Processing chain

All the experiments presented here were optained following the same processing chain:

1. First of all, textual information from documents and queries are tokenized and lemmatized by an external tool[7]; tokens are then filtered, according to their POS tag (only nouns, verbs, adverbs and djectives are kept).

2. Then we look for the correspondences between the tokens in a document and the entries (leaves) in the ontology, with the lexical string first and the lemmatized form then, if necessary. Tokens without correspondence in the considered ontology (either *WordNet* or *EDR*) are indexed in the standard way. The dynamic coverage rate of the collections by the ontology[8] was 90% in average, for both *WordNet* and *EDR*.

3. Then the hierarchy of concepts related to the tokens found in the ontology is expanded by selecting all possible senses for *WordNet* (relying on the mutual reinforcement induced by collocations to have a sort of disambiguation), and only the most frequent sense for *EDR* [9].

4. A *MRC* cut is then computed with the algorithm previously presented.

5. The index of documents and queries are then computed; each token is substituted by the concepts from the cut which subordinate it. As the cut was computed only with nodes covering words contained in the documents (not the queries), tokens covered by the ontology but not by concepts in the selected cut can occur in the queries. We thus evaluated the three following strategies (see also the fig 5):

   0up the first strategy simply consists in ignoring these tokens (in the fig 5, 'b' and 'c' are not indexed);

   1up in a more sophisticated strategy, we look if the related concepts (or synsets) subordinate a part of the cut; in this case, the term is indexed by the subordinate part of the cut, otherwise it is ignored (in the figure, 'b' is ignored but 'c' is indexed by 'C');

   2up the most sophisticated strategy evaluated here consists in also looking for the hypernyms of the related concepts, i.e. $(t^{\uparrow})^{\uparrow}$ (in the figure, 'c' is always indexed by 'C', but 'b' is indexed as well, by 'B' and 'C').

   Tokens not covered by the ontology are indexed in the standard way (in the figure, the token 'a' is always indexed by himself)

6. Finally, search and evaluation are performed, using the vector-space SMART information retrieval system (Salton 71).

[5] Available online at *ftp://ftp.cs.cornell.edu/pub/smart/*.

[6] *WordNet* v.2.0, by Android Technologies.

[7] Sylex 1.7, © 1993-98 DECAN INGENIA.

[8] The dynamic coverage rate is the number of occurrences of words that are in the ontology divided by the number of occurrences of words in the collection.

[9] This technique gives better results rather than keeping all possible senses in *EDR* (Seydoux & Chappelier 05); this kind of WSD was not possible for *WordNet*, because the required information was not present in the version used.

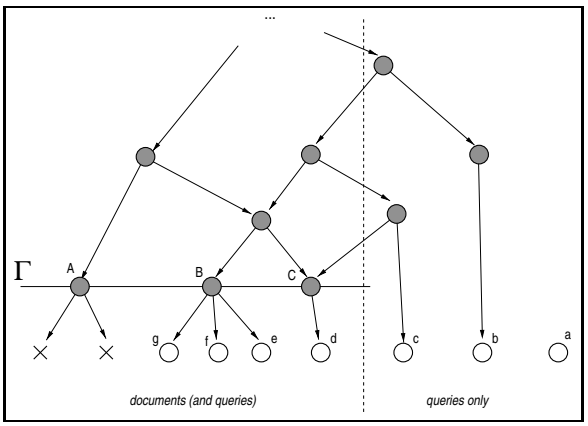


Figure 5: Possibles configuration when indexing the queries' terms. Word 'a' (not covered at all by the ontology) is always indexed by himself, while 'd' is always indexed by 'C' and 'e', 'f' and 'g' are always indexed by 'B'. In the 0up strategy, 'b' and 'c' are not indexed; in the 1up strategy, 'c' is indexed by 'C', and in the 2up strategy, 'b' is indexed as well by 'B' and 'C'.

Table 1 (hereafter) gather the 11-pt precision and the 30-doc recall[10] for the experiments carried out.

4 Discussion and Conclusion

Three main conclusions can be drawn out of these experiments:

1. Using adapted additional semantic information can enhance the indexing of documents, and thus the performance of a IR system. The results of semantic (ontology-based) indexing (columns (c)) are better than the baseline system for three of the collections, but clearly worse on the MED collection.

   This can be explained by the specificity of the vocabulary of these bases, and their adequacy with the semantic resource. ADI and CACM have an important technical vocabulary, but well covered by the *EDR* ontology[11]. CACM documents are extremely small, often restricted to a simple title of few words. Conversely, the vocabulary of TIME is very general and the length of each document is large. The CISI collection present documents of average size, but with a significant number of dates, proper names, etc., for which the POS filtering seems to have annoying consequences (the very low performance clearly indicate an initial loss of information). Finally, the MED collection has an extremely specific vocabulary, for which the used ontologies was not adapted.

2. Excepted for the TIME collection, the results on *EDR* seems globally better than those obtained on *WordNet*, but this is probably caused by the WSD technique used. Keeping all possible senses with *EDR* gives almost the same results as with *WordNet*, excepted for the TIME collection, for which the *WordNet* ontology seems really better. A semantic disambiguation, even rudimentary, appears to be necessary. The expected mutual reinforcement of collocations as a kind of "natural" disambiguation does actually not occur. The reason is probably that the hypernyms relations used do not constitute thematic links (for example, the thematically related terms "doctor", "drug", "hospital", "nurse", etc. are not linked together with the used ontologies).

   Anyway, this result enfavours the use of a proper WSD procedure for further improving the results.

3. The different strategies for dealing with tokens' terms not covered by the indexing set (the cut) lead to different outcomes, depending on the used ontology. As with *EDR*, the simple '0up' strategy gives better results, this strategy is clearly the worst with *WordNet*, while the '2up' seems to give the best results. This can perhaps be explained by the structure of the ontologies: all the synsets of *WordNet* directly cover several leaves, while for *EDR*, many of the concepts are just "pure concept", only related to others concepts.

As futur work, it is forseen to confront the results presented here with similar experiments using better WSD procedure (especially for *WordNet*), and also using different weighting scheme for the evaluation (as for example the Lnu-weighting, more

[10] These are (some of the) standard IR evaluation measure. 11-pt precision is the average precision on recall 0.0, 0.1, ..., 1.0 (where the precision at recall 0.0 is the maximum precision on the whole relevant documents retrieved by the system). 30-doc recall is the recall after the extraction of 30 documents.

[11] Even if in this case, the kind of WSD used with *EDR* is not adapted (especially for ADI; better results was obtained with *EDR* without WSD – see (Seydoux & Chappelier 05)).

reliable for handling noisy data and dealing with long documents).

It would be also interesting to generalize this technique in order to take in account all relationships between terms given by the ontology, in addition to the thesaurus of hypernyms.

Finally, we want to emphasize that the technique presented here does not limit to IR but could also be apply to other NLP domains, such as document clustering (Hotho *et al.* 03) and text summarization.

References

(Besançon *et al.* 01) Romaric Besançon, Jean-Cédric Chappelier, Martin Rajman, and Antoine Rozenknop. Improving text representations through probabilistic integration of synonymy relations. In *Proceedings of the Xth International Symposium on Applied Stochastic Models and Data Analysis (ASMDA'2001)*, volume 1, pages 200–205, 2001.

(Deerwester *et al.* 90) S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

(Fellbaum 98) Christiane Fellbaum, editor. *WordNet, An Electronic Lexical Database*. MIT Press, 1998.

(Gonzalo *et al.* 98a) J. Gonzalo, F. Verdejo, I. Chugur, and J. Cigarran. Indexing with WordNet synsets can improve text retrieval. In *Proc. of the COLING/ACL 1998 Workshop on Usage of WordNet for Natural Language Processing*, pages 38–44, 1998.

(Gonzalo *et al.* 98b) J. Gonzalo, F. Verdejo, C. Peters, and N. Calzolari. Applying EuroWordNet to multilingual text retrieval. *Journal of Computers and the Humanities*, 32(2-3):185–207, 1998.

(Harman 88) D. Harman. Towards interactive query expansion. In *Proc. of the 11th Annual Int. ACM-SIGIR Conference on Research and development in information retrieval*, pages 321–331, 1988.

(Hofmann 99) Thomas Hofmann. Probabilistic latent semantic indexing. In *proc. of the 22th International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 50–57, 1999.

(Hotho *et al.* 03) Andreas Hotho, Steffen Staab, and Gerd Stumme. Wordnet improves text document clustering. In *Proc. of the SIGIR 2003 Semantic Web Workshop*, 2003. None.

(Ide & Véronis 98) N. Ide and J. Véronis. Word sense disambiguation: The state of the art. *Computational Linguistics*, 24(1):1–40, 1998.

(Kiryakov & Simov 99) Atanas K. Kiryakov and Kiril Iv. Simov. Ontologically supported semantic matching. In *in Proceedings of NODALIDA'99: Nordic Conference on Computational Linguistics, Trondheim*, December 1999.

(Mihalcea & Moldovan 00) R. Mihalcea and D. Moldovan. Semantic indexing using WordNet senses. In *Proc. of ACL Workshop on IR & NLP*, 2000.

(Miller 95) George A. Miller. Wordnet: a lexical database for english. In *Communications of the ACM 38 (11)*, pages 39 – 41, november 1995. disponible sur http://www.cogsci.princeton.edu/ wn/index.shtml.

(Miyoshi *et al.* 96) H. Miyoshi, K. Sugiyama amd M. Kobayashi, and T. Ogino. An overview of the EDR electronic dictionary and the current status of its utilization. In *Proc. of COLING*, pages 1090–1093, 1996.

(Moldovan & Mihalcea 00) D. I. Moldovan and R. Mihalcea. Using wordnet and lexical operators to improve internet searches. *IEEE Internet Computing*, 4(1):34–43, 2000.

(Richardson & Smeaton 95) R. Richardson and A. F. Smeaton. Using WordNet in a knowledge-based approach to information retrieval. Technical Report CA-0395, Dublin City University, Glasnevin, Dublin 9, Ireland, 1995.

(Salton 68) G. Salton. *Automatic Information Organization and Retrieval*. McGraw-Hill, 1968.

(Salton 71) G. Salton. *The SMART Retrieval System – Experiments in Automatic Document Processing*. Prentice Hall, 1971.

(Seydoux & Chappelier 05) Florian Seydoux and Jean-Cédric Chappelier. Hypernyms ontologies for semantic indexing. In *Proc. of the SIGIR'05 Workshop on Methodologies and Evaluation of Lexical Cohesion Techniques in Real-world Applications (ELECTRA'2005), Brazil*, August 2005.

(Shannon 48) C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, July 1948.

(Smeaton & Quigley 96) A. F. Smeaton and I. Quigley. Experiments on using semantic distances between words in image caption retrieval. In *Proc. of 19th Int. Conf. on Research and Development in Information Retrieval*, pages 174–180, 1996.

(Voorhees 93) E. M. Voorhees. Using WordNet to disambiguate word senses for text retrieval. In *Proc. of 16th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 171–80, 1993.

(Voorhees 94) E. M. Voorhees. Query expansion using lexical-semantic relations. In *Proc. 17th Annual Int. ACM-SIGIR Conf. on Research and Development in Information Retrieval*, pages 61–69, 1994.

(Voorhees 98) E. M. Voorhees. Using WordNet for text retrieval. In C. Fellbaum, editor, *WordNet: An Electronic Lexical Database*, chapter 12, pages 285–303. MIT Press, 1998.

(Whaley 99) J. M. Whaley. An application of word sense disambiguation to information retrieval. Technical Report PCS-TR99-352, Dartmouth College, Computer Science, Hanover, NH, 1999.

(Wilks & Stevenson 98) Y. Wilks and M. Stevenson. Word sense disambiguation using optimised combinations of knowledge sources. In *Proc. of the 17th Int. Conf. on Computational Linguistics*, pages 1398–1402, 1998.

		(a)	(b)	(c)-0up	(c)-1up	(c)-2up
ADI collection (82 documents, 35 queries) [Documents from Information Science]						
EDR, most frequent concept, tf	*precision*	0.2497	**0.2939**	0.2924	**0.2933**	0.2844
	recall	0.5996	**0.7141**	0.6901	0.6996	0.6806
EDR, most frequent concept, tf.idf	*precision*	0.3578	**0.4274**	0.4266	0.4238	0.3700
	recall	0.6984	**0.7217**	0.7081	0.7176	0.7007
WordNet, all synsets, tf	*precision*	0.2497	**0.2671**	0.2593	0.2562	0.2567
	recall	0.5996	**0.6361**	0.6146	0.6064	0.6064
WordNet, all synsets, tf.idf	*precision*	**0.3578**	0.3564	0.3547	0.3450	0.3353
	recall	**0.6984**	0.6649	0.6767	0.6494	0.6422
TIME collection (423 documents, 83 queries) [General world news articles from the magazine Time (1963)]						
EDR, most frequent concept, tf	*precision*	0.3288	0.3692	**0.3964**	0.3908	0.3908
	recall	0.7755	0.7590	**0.8124**	0.8124	0.8124
EDR, most frequent concept, tf.idf	*precision*	0.5496	0.5565	**0.5602**	0.5543	0.5544
	recall	0.8901	**0.9053**	0.8968	0.8975	0.8975
WordNet, all synsets, tf	*precision*	0.3288	0.2579	0.2348	0.3449	**0.3451**
	recall	0.7755	0.6263	0.6122	0.7760	**0.7880**
WordNet, all synsets, tf.idf	*precision*	0.5496	0.5059	0.5565	0.5668	**0.5695**
	recall	0.8901	0.8421	0.9196	0.9161	**0.9281**
MED collection (1033 documents, 30 queries) [Collection of abstract from a medical journal]						
EDR, most frequent concept, tf	*precision*	**0.3623**	0.3229	0.3251	0.3253	0.3158
	recall	**0.4574**	0.4230	0.4214	0.4238	0.4143
EDR, most frequent concept, tf.idf	*precision*	**0.4607**	0.4518	0.4394	0.4380	0.3989
	recall	**0.5547**	0.5404	0.5344	0.5386	0.4862
WordNet, all synsets, tf	*precision*	**0.3623**	0.2750	0.1914	0.3174	0.3212
	recall	**0.4574**	0.3803	0.2894	0.4216	0.4241
WordNet, all synsets, tf.idf	*precision*	**0.4607**	0.4216	0.3329	0.4390	0.4369
	recall	**0.5547**	0.5120	0.4267	0.5263	0.5261
CISI collection (1460 documents, 112 queries) [Articles from Information science (Library science)]						
EDR, most frequent concept, tf	*precision*	0.0687	0.0805	**0.0817**	**0.0817**	**0.0814**
	recall	0.1239	**0.1300**	0.1243	0.1243	0.1257
EDR, most frequent concept, tf.idf	*precision*	0.1733	**0.1825**	0.1785	0.1672	0.1621
	recall	**0.2318**	**0.2313**	0.2243	0.2243	0.2211
WordNet, all synsets, tf	*precision*	0.0687	0.0588	0.0449	**0.0738**	**0.0739**
	recall	**0.1239**	0.0926	0.0745	0.1226	0.1224
WordNet, all synsets, tf.idf	*precision*	**0.1733**	0.1336	0.0875	0.1653	0.1644
	recall	**0.2318**	0.1979	0.1364	0.2204	0.2192
CACM collection (3204 documents, 64 queries) [Collection of titles and abstracts from a Computer science journal]						
EDR, most frequent concept, tf	*precision*	**0.1555**	0.1472	0.1525	0.1493	0.1482
	recall	**0.3082**	0.2926	0.2996	0.2982	0.2982
EDR, most frequent concept, tf.idf	*precision*	0.2865	0.2804	**0.2964**	0.2796	0.2724
	recall	0.4534	**0.4567**	0.4514	0.4489	0.4363
WordNet, all synsets, tf	*precision*	0.1555	0.1628	0.1101	**0.1637**	0.1625
	recall	**0.3082**	0.2736	0.2019	0.2993	0.2940
WordNet, all synsets, tf.idf	*precision*	**0.2865**	0.2390	0.2024	0.2621	0.2562
	recall	**0.4534**	0.3758	0.3366	0.4390	0.4354

Table 1: Evaluation of several indexing scheme (see figures 1 and 5) on several collections: (a): words only; (b): direct hypernyms; (c)-0up, (c)-1up and (c)-2up : hypernyms from the *MRC*.

A Simple WWW-based Method for Semantic Word Class Acquisition

Keiji Shinzato **Kentaro Torisawa**
School of Information Science
Japan Advanced Institute of Science and Technology (JAIST)
1-1 Asahidai, Nomi, Ishikawa, 923-1292 JAPAN
{skeiji, torisawa}@jaist.ac.jp

Abstract

This paper describes a simple method to obtain semantic word classes from HTML documents. Shinzato and Torisawa previously showed that itemizations in HTML documents can contain semantically coherent word classes. However, not all the itemizations are semantically coherent. Our goal is to provide a simple method to extract only semantically coherent itemizations from HTML documents. Our method can perform this task by obtaining hit counts from an existing search engine $2n$ times for an itemization consisting of n items.

1 Introduction

There are many natural language processing tasks in which semantically coherent word classes can play an important role, and many automatic methods for obtaining word classes (or semantic similarities) have been proposed (Church & Hanks 89; Hindle 90; Lin 98; Pantel & Lin 02). Most of these methods rely on particular types of word co-occurrence frequencies, such as noun-verb co-occurrences, obtained from parsed corpora.

On the other hand, Shinzato and Torisawa previously showed that itemizations in HTML documents on the World Wide Web (WWW), such as that in Figure 1, can contain *semantically coherent word classes* (Shinzato & Torisawa 04). We say a class of words is *coherent* if it contains only words that are semantically similar to each other and that have a common hypernym other than trivial hypernyms such as *things* or *objects*. The expressions in Figure 1 have a common non-trivial hypernym such as "Record shops" and constitute a semantically coherent word class. Since one can find a huge number of HTML itemizations throughout the WWW, we can expect a huge number of semantic classes to be available. However, itemizations do not always contain semantically coherent word classes. Many itemizations

```
<UL><LI>Favorite Record Shops</LI>
    <UL><LI>Tower Records</LI>
        <LI>HMV</LI>
        <LI>Virgin Megastores</LI>
        <LI>RECOfan</LI>
</UL></UL>
```

Figure 1: A sample code of an itemization layout

are used just for formatting HTML documents properly and contain semantically incoherent items. We therefore need an automatic method to filter out such itemizations.

In this paper, we present a simple filtering method to obtain only semantically coherent word classes from itemizations in HTML documents. The method calculates strength of association between items in a word class using hit counts obtained from a search engine, and tries to exclude semantically incoherent word classes according to this strength. Our method is simple in the sense that all it requires are hit counts from a search engine, mutual information regarding the hit counts, and an implementation of Support Vector Machines (Vapnik 95). Our method is also efficient and lightweight in the sense that it can perform this task just by obtaining hit counts at most $2n$ times for a word class consisting of n items. There is no need to download and analyze a large number of HTML documents.

We have tested the effectiveness of our method through experiments using HTML documents and human subjects. A problem is that it is difficult to set a rigorous evaluation criterion for semantic word classes. We try to solve this problem using a hierarchical structure in a manually tailored thesaurus.

In this paper, Section 2 reviews previous work and Section 3 describes our proposed method. Our exper-

imental results, obtained using Japanese HTML documents, are presented in Section 4.

2 Previous Work

An alternative to our filtering method for word classes is Shinzato's hyponymy relation acquisition method (Shinzato & Torisawa 04). It first extracts HTML itemizations and downloads documents including the expressions in the itemizations. The method then finds words that frequently appear in the downloaded documents and appear less frequently in general texts. Basically, it selects one among such words according to a score and other heuristics, and produces it as a common hypernym for the expressions in the itemization. In general, if the score value produced with a hypernym is large enough, the resulting hypernym is likely to be a proper hypernym and the itemization tends to include a semantically coherent word class. Because of this property, we can regard the whole procedure as an alternative to our method. More precisely, if we select the itemizations for which Shinzato's method produced a hypernym with a high score, then the selected itemizations tend to be semantically coherent word classes. The difference from our algorithm, though, is that Shinzato's method requires a large amount of time and is not appropriate as a filter to obtain a large number of word classes in a short time period. It needs to download a considerable number of texts, parse at least part of the texts, and count the occurrence frequencies of many words. Our aim is to skip such heavyweight processes and provide a lightweight filter.

As another type of method for obtaining semantically coherent word classes, there are a large number of methods to automatically generate semantically coherent word classes from normal texts. Most of these collect the contexts in which an expression appears and calculate similarities between the contexts for the expressions to constitute a word class. In Lin's work, for instance, the contexts for an expression are represented as a set of syntactic dependency relations in which the expression appears (Lin 98). Many others have taken similar approaches and here we cite just a few examples (Hindle 90; Pantel & Lin 02). The difference between these methods and our work is that they assume rather complex contexts such as dependency relations. We do not assume such complex contexts obtained by using parsers or parsed corpora. We only need to obtain the numbers of documents that include expressions in the same itemization using a search engine. This type of (co-occurrence) frequencies (or some statistical values computed from the frequencies) are known to be useful for detecting semantic *relatedness* between two expressions (Church & Hanks 89). The relatedness is not limited to semantic similarities, though, which we need to compute to obtain semantic word classes. For instance, Church found that while "doctor" and "nurse" have a large mutual information value, "doctor" and "bill" also have a large value. We do not think "doctor" and "bill" are *similar*, though they are somehow related. Our trick for excluding such word pairs from our semantic class is to use itemizations in HTML documents. We assume that expression pairs having semantic relatedness other than *strong* semantic similarities are unlikely to be included in the same itemization. For instance, "doctor" and "bill" will not appear in the same itemization.

3 Proposed Method

To be precise, our goal is to extract semantically coherent classes consisting of *single words* or *multi-word expressions* from HTML documents. In the following, we refer to single words and multi-word expressions simply as *expressions*.

Our procedure consists of two steps:

Step 1 Extract sets of expressions from itemizations in HTML documents. We call the obtained set an **Itemized Expression Set (IES)**.

Step 2 Select only semantically coherent classes from the IESs obtained in Step 1 using the document frequencies and mutual information.

The procedure was designed based on the following two assumptions. Step 1 corresponds to Assumption A, and Step 2 to Assumption B.

Assumption A At least, some IESs are semantically coherent.

Assumption B Expressions in a semantically coherent IES are likely to co-occur in the same document.

To judge the semantic coherence among the expressions in an IES, according to Assumption B, we estimate strength of co-occurrences between pairs of expressions. As this strength, we use simple document frequencies and pairwise mutual information (Church & Hanks 89). Our algorithm computes these values for pairs of expressions in the same IES. An important point is that our algorithm does not compute the

values for all the possible pairs in an IES, although a most straightforward implementation of Assumption B should be the algorithm that does so. Instead, our algorithm randomly generates only n pairs of expressions from an IES consisting of n expressions and then calculates document frequencies and mutual information for each pair. The parameters required to compute the values are obtained from an existing search engine. The number of queries to be given to the engine is just $2n$ for an IES consisting of n expressions. Note that if we compute the strength for all possible pairs, we need to throw $n(n-1)/2+n$ queries to the search engine. Our method is thus much more efficient than this exhaustive algorithm in terms of the number of queries.

Details of Steps 1 and 2 are described below.

3.1 Step 1: Extract IESs

The objective of Step 1 is to extract IESs from itemizations in HTML documents. We follow the approach described in (Shinzato & Torisawa 04). First, we associate each expression in an HTML document with a *path* which specifies both the HTML tags enclosing the expression and the order of the tags. Consider the HTML document in Figure 1. The expression "Favorite Record Shops" is enclosed by the tags `<LI>`,`</LI>` and `<UL>`,`</UL>`. If we sort these tags according to their nesting order, we obtain a path (UL, LI) and this path specifies the information regarding the place of the expression. We write ⟨(UL, LI), Favorite Record Shops⟩ if (UL, LI) is a path for the expression "Favorite Record Shops." We can then obtain the following paths for the expressions from the document.

⟨(UL, LI), Favorite Record Shops⟩,
⟨(UL, UL, LI), Tower Records⟩,
⟨(UL, UL, LI), HMV⟩,
⟨(UL, UL, LI), Virgin Megastores⟩,
⟨(UL, UL, LI), RECOfan⟩

Our method extracts the set of expressions associated with the same path as an IES. In the above example, we can obtain the IES {Tower Records, HMV, Virgin Megastores,RECOfan}.

3.2 Step 2: Select Semantically Coherent IESs

In Step 2, our procedure filters out semantically incoherent IESs from IESs obtained in Step 1. We use document frequencies and pairwise mutual information for this purpose. These values are given to Support Vector Machines (SVMs) (Vapnik 95) as features for selecting semantically coherent IESs.

To obtain the features given to the SVM, we first generate n pairs of two expressions in an IES consisting of n expressions. More precisely, for each expression in the IES, we randomly pick another expression from the set to generate the pairs. For the IES {Tower Records, HMV, Virgin Megastores, RECOfan}, we generate, for instance, the following set of expression pairs from the IES.

{⟨Tower Records, HMV⟩,
⟨HMV, Virgin Megastores⟩,
⟨Virgin Megastores, RECOfan⟩,
⟨RECOfan, Tower Records⟩}

Next, we estimate pairwise mutual information for each pair. We defined pairwise mutual information, $I(e_1, e_2)$, between expressions e_1 and e_2 as

$$I(e_1, e_2) = \log_2 \frac{\frac{docs(e_1, e_2)}{N}}{\frac{docs(e_1)}{N} \times \frac{docs(e_2)}{N}}$$

where $docs(e)$ is the number of documents including an expression e, $docs(e_i, e_j)$ is the number of documents including two expressions e_i and e_j. We estimate $docs(e)$ and $docs(e_i, e_j)$ using a search engine, which is "goo" (http://www.goo.ne.jp/) in our experiments. N is the total number of documents and we used 4.2×10^9 as N according to "goo." Note that we used -10^9 as a logarithm of 0 in calculating the mutual information values.

Consider the following IESs.

A {Tower Records, HMV, Virgin Megastores, RECOfan}
B {Gift Certificates, International, New Releases, Top Sellers, Today's Deals}

We think that Set A is a semantically coherent IES while Set B is semantically incoherent. The pairwise mutual information values computed for each IES are listed in Table 1. (We randomly generated pairs for each IES as described before, and computed the mutual information for the pairs.) The values for the pairs in Set A are all positive and larger than those for the pairs in Set B. This roughly means that every pair in Set A co-occurs much more frequently than expected only from the frequencies of each item in the pair with assuming independence of the occurrences of the items. In addition, the differences between the actual co-occurrence frequencies and the expected co-occurrence frequencies are larger in Set A than those in Set B. We expect to be able to select semantically coherent IESs by looking at such differences in mutual information values.

We used the features listed in Table 2. The major part of the features are mutual information and hit counts for expression pairs, but they also include hit

Table 1: Examples of pairwise mutual information.

set	e_i	$docs(e_i)$	e_j	$docs(e_j)$	$docs(e_i, e_j)$	$I(e_i, e_j)$
A	Tower Records	2.01×10^5	RECOfan	4.87×10^2	9.90×10^1	12.05
	Virgin Megastores	9.87×10^3	Tower Records	2.01×10^5	9.21×10^2	10.93
	RECOfan	4.87×10^2	HMV	9.71×10^5	2.52×10^2	10.13
	HMV	9.71×10^5	Virgin Megastores	9.87×10^3	1.29×10^3	9.14
B	International	7.88×10^7	Today's Deals	6.40×10^5	4.52×10^5	5.23
	Gift Certificates	2.64×10^6	New Releases	1.12×10^6	1.91×10^4	4.76
	Sell Your Stuff	1.35×10^5	Gift Certificates	2.64×10^6	4.21×10^2	2.31
	Top Sellers	3.09×10^6	Sell Your Stuff	1.35×10^5	2.81×10^2	1.50
	Today's Deals	6.40×10^5	Top Sellers	3.09×10^6	3.48×10^2	-0.44
	New Releases	1.12×10^6	Top Sellers	3.09×10^6	3.39×10^2	-1.42

Total number of documents $N = 4.2 \times 10^9$.

counts for single expressions and some other items which we expect to be useful in our task. Note that we used only the largest, the second largest, the smallest, and the second smallest hit counts, co-occurrence frequencies, and mutual information values as features (features with ID **3** to **6**, **9** to **12** and **18** to **21**.) Since we restricted IESs given to our method only to the ones that have more than three expressions, the feature values are always defined.

Let us examine the number of queries needed to obtain feature values for an IES consisting of n expressions. First, we need to obtain $docs(e)$ for every item e in the IES. This requires n queries. Then we obtain $docs(e_i, e_j)$ of the randomly generated n pairs, which also needs n queries. Note that we can obtain all the feature values in Table 2 only from the above hit counts. Then, the total number of required queries is $2n$. On the other hand, we can show that it is necessary to throw $n(n-1)/2 + n$ queries to a search engine if we compute the feature values for all the possible pairs in the IES. In a later section, we empirically show that our method, which requires much less queries than the exhaustive algorithm, achieves the performance comparable to the exhaustive algorithm.

Finally, our method ranks the IESs according to output values of the SVM (i.e., values of the decision function of the SVM) and produces only the top M IESs as final outputs. Note that SVMs are usually used as a binary classifier that classifies a sample into two categories according to the value of the decision function. But we use them in a slightly different way. We assume that a value of the decision function indicates the likeliness that a given IES is semantically coherent. More precisely, we made an assumption that the larger a value of the decision function for a given IES is, the more likely the IES is to be semantically coherent. Then, we expect that by producing only the IESs having large values of the decision function as final outputs, we can obtain the semantically coherent IESs with a relatively high precision.

4 Experiments

4.1 Setting

We downloaded 1.0×10^6 HTML documents (10.5 GB with HTML tags), and extracted $132,874$ IESs through the method described in Section 3.1. We randomly picked 800 sets from the extracted IESs as our test set. It contained $5,227$ expressions in total. As our training set for the SVMs, we randomly selected 400 sets, which included $2,541$ expressions. The training set was annotated with *Coherent/Incoherent* labels by the authors according to an evaluation scheme for IESs, as described in a later section. Note that the test set and the training set were chosen so that the two sets do not have any common expressions. We chose *TinySVM* [1] as an implementation of SVMs. As the kernel function, we used the ANOVA kernel of degree 2 provided in *TinySVM*. This choice was made according to the observations obtained in experiments using the training set. Other types of kernel provided in *TinySVM* did not converge during the training or did not indicate high performance on the training set.

4.2 Evaluation Scheme

In our experiments, we evaluated the IESs produced by our method according to the following criterion.

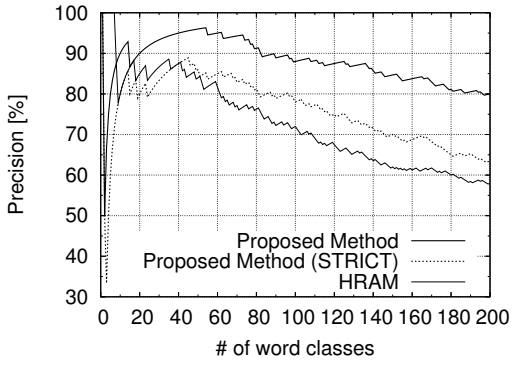
CRITERION If we can come up with a common hy-

[1] Available from http://chasen.org/~taku/software/TinySVM/

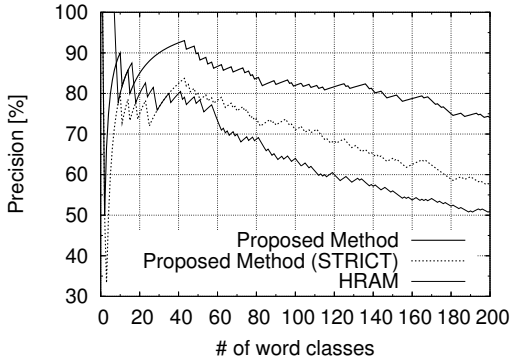
Table 2: Features used in our procedure.

ID	Descriptions
1	Sum of the $I(e_i, e_j)$ values of all pairs.
2	Average of the $I(e_i, e_j)$ values of all pairs.
3	Largest $I(e_i, e_j)$ of a pair in P.
4	2nd largest $I(e_i, e_j)$ of a pair in P.
5	Smallest $I(e_i, e_j)$ of a pair in P.
6	2nd smallest $I(e_i, e_j)$ of a pair in P.
7	Sum of the $docs(e_i, e_j)$ values of all pairs.
8	Average of the $docs(e_i, e_j)$ values of all pairs.
9	Largest $docs(e_i, e_j)$ of a pair in P.
10	2nd largest $docs(e_i, e_j)$ of a pair in P.
11	Smallest $docs(e_i, e_j)$ of a pair in P.
12	2nd smallest $docs(e_i, e_j)$ of a pair in P.
13	Number of pairs whose $docs(e_i, e_j)$ is 0.
14	Number of items in an IES.
15	Number of items whose $docs(e)$ is 0.
16	Sum of the hit count for all items in an IES.
17	Average hit count for all items in an IES.
18	Largest hit count for an item in an IES.
19	2nd largest hit count for an item in an IES.
20	Smallest hit count for an item in an IES.
21	2nd smallest hit count for an item in an IES.

P: A set of pairs of two randomly selected items in an IES.



(A) Three subjects



(B) Four subjects

Figure 2: Comparison with HRAM

pernym [2] for 70% of the expressions in a given IES, we regard the IES as a semantically coherent class. However, when we can think of only the words referring to an extremely wide range of objects, such as *things* and *objects*, as hypernyms, the class is not coherent.

In the following, we call the words that refer to an extremely wide range of objects, such as *things* and *objects*, *trivial hypernyms*. The trivial hypernyms are problematic since expressions that are not similar to each other may have a trivial hypernym as its common hypernym. For instance, consider a set of expressions $\{automobile, desk, human, idea\}$. It may be possible to regard "objects" as a common hypernym of the expressions, but it is difficult to regard the set as a *semantically coherent class*. This means that it is not sufficient to judge the semantic coherence of expressions according to only whether one can think of their common hypernym, and it is necessary to judge if we can come up with a *non-trivial* common hypernyms of the expressions.

Then, the problem is how we can make a list of non-trivial (possible) hypernyms. We used the Nihongo Goi Taikei thesaurus (Ikehara *et al.* 97) to solve this problem. The thesaurus contains 2,710 semantic classes, each of which are labeled by a Japanese expression naming the class, and the classes are organized into a hierarchical structure. We tried to make a list of trivial hypernyms from the thesaurus according to the following steps. First, we extracted 245 labels of all the classes that are located in the top five levels in the hierarchy, and then manually checked whether the extracted labels should be regarded as trivial hypernyms. As a result, we could obtain 164 trivial hypernyms such as "個体 (*individual*)" and "事象 (*phenomena*)." We then removed the trivial hypernyms from the set of general nouns in the thesaurus. As a result, we could obtain the list of 92,002 nouns, and assumed that it is a list of non-trivial (possible) hypernyms.

We evaluated acquired IESs using four human subjects. The subjects were asked if they can come up for each IES with a common hypernym in the list of the non-trivial hypernyms, which were generated by the

[2] In this study, class-instance relations are also regarded as hypernym-hyponym relations. Then, for instance, we can think of *common hypernyms* of proper nouns.

above procedure. More precisely, the subjects were asked to give a common hypernym of a given IES to our evaluation tool. The subjects could proceed to the evaluation of the next IES only when the tool finds the given hypernym in the non-trivial hypernym list, or when the subjects tell the tool that they could not find any non-trivial hypernyms. By this, we could prevent the subjects from choosing trivial hypernyms.

CRITERION is generous in the sense that a common hypernym have to cover only 70% of the expressions in a given IES. We also prepared a stricter version of the criterion, which asks the subjects to come up with a common hypernym covering *all* the expressions in an IES.

CRITERION (STRICT) If we can come up with a common hypernym for **100%** of the expressions in a given IES, we regard the IES as a semantically coherent class. However, when we can think of only trivial hypernyms as hypernyms, the class is not coherent.

Note that the precisions obtained according to **CRITERION (STRICT)** are always lower than those obtained according to **CRITERION**.

4.3 Experimental Results

We conducted three types of experiments. First, we compared the performance of our method with that of Shinzato's Hyponymy Relation Acquisition Method (HRAM) (Shinzato & Torisawa 04), which can be seen as an alternative to our method as pointed out in Section 2. Next, we checked the contributions of the features used in our method. Finally, we compared the precisions of our method with those of an exhaustive method which calculates mutual information and co-occurrence frequencies for all possible pairs of expressions in an IES. At the same time, we also checked the performance when we use a small homemade search engine. In the following, we present the results of above three experiments.

4.3.1 Comparison with HRAM

First, we compared the performances of our method with those of HRAM. As mentioned, the outputs of our method are sorted according to the decision function values in the SVM, while the outputs of HRAM are also sorted by its original score. In this series of experiments, we gave 800 IESs in our test set to our method and HRAM, and we then picked up top 200 IESs from the outputs of the both methods. The performances are shown in Figure 2. Graph (A) shows the precisions of the methods when we assume that semantically coherent IESs are only the IESs that are accepted by three or more human subjects in the four subjects, while graph (B) indicates the precisions of the methods when semantically coherent IESs are only the IESs that all the four subjects accept. In the both graphs, the y-axis indicates the precision of the acquired word classes, while the x-axis indicates the number of classes. "Proposed Method" refers to the precisions achieved by our method and "HRAM" indicates the precisions obtained by HRAM. The evaluation of both methods were done according to **CRITERION** defined before. "Proposed Method (STRICT)" is the results of our method when the used evaluation scheme was **CRITERION (STRICT)**.

From the both graphs, it is easy to see that our method outperforms HRAM[3].

If we look at the results evaluated according to **CRITERION**, the precision of our method in graph (A) reached 88% for the top 100 classes which was 12.5% of all the given IESs in the test set. For the top 200 classes (25% of all the IESs in the test set), the precision was about 80%. The kappa statistic for measuring the inter-rater agreement was 0.69 for our method. For HRAM, the statistic was 0.78. These values indicate that our subjects had *good* agreement, according to (Landis & Koch 77). Table 3 shows examples of the IESs produced by our method.

4.3.2 Contribution of features

Next, we removed some features from the training and test sets to check the contributions of these features. More precisely, we classified all the feature types into the following three categories. The numbers specify the feature IDs in Table 2.

MIs: $\{1, 2, 3, 4, 5, 6\}$,
Coocs: $\{7, 8, 9, 10, 11, 12, 13\}$,
Others: $\{14, 15, 16, 17, 18, 19, 20, 21\}$

The set *MIs* consisted of the features including the mutual information. The set *Coocs* was the set of the features regarding co-occurrence document frequencies. The features in *Others* were those that had nothing to do with the co-occurrence frequencies or the mutual information. We checked the performance of our method when we removed one category of fea-

[3]Note that we did not plot the performance of HRAM when we evaluated it according to **CRITERION(STRICT)**. But, since the precisions obtained according to **CRITERION (STRICT)** are always lower than those by **CRITERION**, and "Proposed Method (STRICT)" indicates better performance than "HRAM" obtained according to **CRITERION**, we can say our method outperformed HRAM in both criteria.

Table 3: Examples of acquired IESs

Semantically coherent IESs
Japanese actresses 春野寿美礼 (Sumire Haruno), 桜木絵美 (Emi Sakuragi), 天乃悠華 (Yuka Amano), 諸鳥あい (Ai Morotori), 綾奈舞 (Mai Ayana), 大伴れいか (Reika Oban), 南しずか (Shizuka Minami)
Football teams in Italy ピアチェンツァ(Piacenza), ユベントス (Juventus), トリノ (Torino), ブレシア (Brescia), キエーボ (Chievo), ローマ (AS Roma), モデナ (Modena), ＡＣミラン (AC Milan), ラツィオ (SS Lazio)
Semantically incoherent IESs
Events or items related to wedding ceremonies 挙式 (Celebration), ケーキ (Cakes), 音響照明 (Sound and lighting) 料理 (Cuisine), 飲物 (beverages), サービス料 (Service charges), 引出物 (Presents), 印刷物 (Printed matter)
Captions of private pictures listed in someone's webpage 国会議事堂前 (In front of a parliament building), パルテノン神殿 (The Parthenon), アテネ駅 (Sta. Athens), 大統領官邸を護る衛兵 (Guards for a presidential mansion)

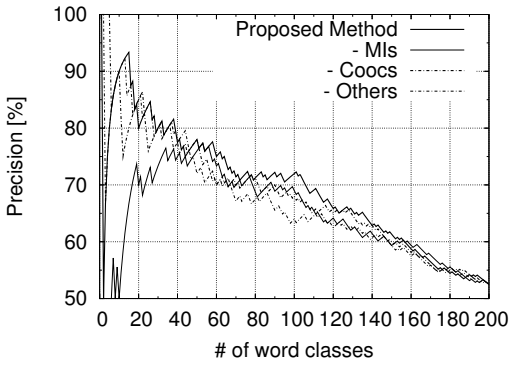


Figure 3: Contribution of each feature set

tures from training and test sets. As a test set, we used 200 IESs randomly selected from the *original* test set that consisted of the 800 IESs. This is because we wanted to see the performances over not only highly ranked IESs but also *all* the given IESs. The 800 IESs in the original test set are too many to be evaluated by human subjects. We regarded IESs such that three or more human subjects in the four subjects can come up with non-trivial hypernyms as semantically coherent classes. The judgments by subjects were done according to **CRITERION**. Figure 3 shows the obtained precisions. "-*X*" refers to the precisions obtained when the feature set *X* was eliminated. The graph indicates that each feature set contributed to the improvement of the precisions.

4.3.3 Comparison with Exhaustive method

Finally, we compared our method to a more exhaustive algorithm that computed the mutual information and the co-occurrence frequencies for all the possible pairs in an itemization. In this series of experiments, we used our home-made search engine that could search through 1.74×10^7 downloaded HTML documents (191 GB with tags) instead of the search engine "goo." We used it because (1) we wanted to check the performance obtained by non-commercial small search engines and (2) the number of required queries would become quite large and could cause a problem in using a commercial search engine. The other settings, including the test set and the criterion for the evaluation, were the same as the one in the previous experiments for checking the contribution of the feature sets.

The results are shown in Figure 4. "Random Pairs" refers to the precisions of our method, which calculates mutual information and hit counts for randomly selected n pairs of expressions, while "Exhaustive Pairs" indicates the precisions of an exhaustive algorithm which calculates mutual information and hit counts for all possible pairs in an itemization. "Proposed Method (Commercial)" refers to the precisions obtained by using the commercial search engine "goo." In other words, this is the same curve referring to "Proposed Method" in Figure 3.

Note that in obtaining the precisions of "Random Pairs", we conducted experiments 10 times, and then averaged the precisions obtained in all the trials. This is because we wanted to reduce the effect of random selections of pairs.

Interestingly, the exhaustive algorithm could not achieve significantly better performance than our method. Moreover, in some part, the precisions obtained by the exhaustive algorithm were lower than those of our method. In this series of experiments, the exhaustive algorithm required 5,714 queries, while our method used the search engine just 2,582 times. In short, our method can classify IESs in a shorter time than the exhaustive algorithm does, and the precisions obtained by our method are at least comparable to those of the exhaustive algorithm, at least, in our current settings.

Note that the precisions obtained by using our home-made search engine were lower than those achieved by using the commercial search engine "goo." The difference between these precisions is approximately 10% for the top 50 classes (25% of all the IESs in the test set.) We think that this deterioration in

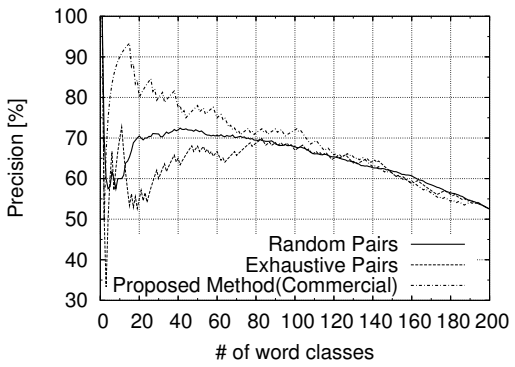


Figure 4: Comparison between our method and an exhaustive method.

the precisions is due to the difference of the amount of the documents accessible by the search engines. Actually, "goo" can search through more than 200 times as many documents as those that can be searched by the home-made search engine. Considering such a huge difference, we think that the precisions obtained by our engine are reasonably good.

5 Conclusion

We have proposed a method to extract semantic word classes from itemizations in HTML documents. Its major characteristics are that (1) it can be implemented easily using SVMs and an existing commercial search engine or a home-made search engine, and (2) it can perform its task using only hit counts obtained by a small number of queries given to a search engine. The method was evaluated by using four human subjects.

Our method ranks itemizations collected from the WWW according to the decision function value of SVMs and produces top itemizations in the ranking as final outputs. In our experiments, when the top 10% of collected itemizations were produced, at least, three of the four human subjects regarded about 80% of the itemizations as sets of semantically similar expressions for which the subjects could come up with non-trivial common hypernyms.

We used mutual information as a metric indicating association between the items in a semantic class. We would like to test other measures, such as log likelihood ratio, in our future work.

References

(Church & Hanks 89) Kenneth Ward Church and Patrick Hanks. Word association norms, mutual information, and lexicography. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*, pages 76–83, 1989.

(Hindle 90) Donald Hindle. Noun classification from predicate-argument structures. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, pages 268–275, 1990.

(Ikehara *et al.* 97) Satoru Ikehara, Miyazaki Masahiro, Shirai Satoshi, Yokoo Akio, Nakaiwa Hiromi, Ogura Kentaro, Ooyama Yoshihumi, and Hayashi Yoshihiko. *Nihongo Goi Taikei – A Japanese Lexicon*. Iwanami Syoten, 1997.

(Landis & Koch 77) J.R. Landis and G.G. Koch. The measurement of observer agreement for categorical data. In *Biometrics 33*, pages 159–174, 1977.

(Lin 98) Dekang Lin. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pages 768–774, 1998.

(Pantel & Lin 02) Patrick Pantel and Dekang Lin. Discovering word senses from text. In *SIGKDD-02*, pages 613–619, 2002.

(Shinzato & Torisawa 04) Keiji Shinzato and Kentaro Torisawa. Acquiring hyponymy relations from web documents. In *Human Language Technology conference/North American chapter of the Association for Computational Linguistics annual meeting*, pages 73–80, 2004.

(Vapnik 95) Vladimir Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

Using WordNet Similarity and Antonymy Relations to Aid Document Retrieval

Thomas de Simone and **Dimitar Kazakov**
Department of Computer Science
University of York
Heslington, York, YO10 5DD, UK
mrtom@tomandpete.co.uk, kazakov@cs.york.ac.uk

Abstract

This article proposes a document search technique that uses the language database WordNet to cluster search results into meaningful categories according to the words that modify the original search term in the text. The work focusses on the relevance of the semantic relations of similarity and antonymy as present in WordNet.

1 Introduction

The growing issue of information overload on the Internet has prompted the appearance of a number of clustered search strategies which all have the same goal of breaking down large sets of search results into smaller clusters, each cluster containing results which should be equally relevant to each other. The most popular current techniques found on the Internet follow an online clustering process whereby the user inputs a search term, and the returned documents are compared to each other to decide which ones belong together. It is this method of relating documents to each other which is under heavy investigation, with many different approaches being suggested. Clustering search engines[1] tend to use string matching, rather than true linguistic analysis, to identify key words and phrases which documents share in common, and generate clusters based on these phrases. In this work, the words are semantically analysed using WordNet (WN) (Miller *et al.*, 1993) to give a more informed analysis of how the documents might relate to each other. The WN relations studied are those of similarity and antonymy. If the search word is a noun or verb, the matching documents could be clustered according to the adjectives, resp. adverbs modifying the search term in the text.

In addition to WN, the work also makes use of PoS tagging, shallow parsing, Tgrep (Pito, 1992) and entropy (Shannon, 1948). The primary corpus used for the study is the Wall Street Journal (WSJ) treebank, but the Semcor corpus[2] is also used to investigate the potential benefits offered by Word Sense Disambiguation (WSD). Semcor is a text corpus which has been manually annotated with WN sense tags, giving the effect of being processed with a high quality WSD algorithm.

2 Related Work

Zamir and Etzioni's on-line clustering tool *Grouper* (Zamir and Etzioni, 1999) is representative of the current academic research using phrase matching. The tool uses a linear time agglomerative clustering algorithm known as Suffix Tree Clustering (STC). It begins by identifying the most frequent phrases, and then clustering together documents which share them, forming base clusters. Clusters are allowed to overlap, and can be merged if the overlap is considerable. Clusters are then named according to its most representative phrase. The latter is labelled with its relative frequency in the cluster.

Investigations in the use of WN in document clustering include work by Hotho *et al.* (2003), and Sedding and Kazakov (2004). Both works focus on off-line clustering, and preprocess documents using WN as follows (with some variations between the two). Each document is PoS tagged, stemmed, and converted into a "bag of words" representation, the result being that each document is represented as a list of words and their frequencies within the document. Words are then replaced with tokens representing their synonyms and hypernyms. This means that two words which are synonymous would now be represented by the same token. In the case of hypernyms, two words are represented by the same token when one is a more general concept subsuming the other. The frequencies of all features are then modified using the `tf idf` weighting scheme in order to highlight features with high

[1] See http://www.clusty.com, http://www.dogpile.com and Grouper (Zamir and Etzioni, 1999), among others.

[2] The Semcor files are freely available to download from http://www.cs.unt.edu/~rada/downloads.html

frequency in selected documents, but infrequent on the whole (Salton and Wong, 1975). A clustering algorithm (bisectional k-means) is then applied. The results indicate that while using the synonymy relation does indeed have a beneficial effect on the quality of clusters, the inclusion of hypernyms produces acceptable results only when carefully managed, as the `tf idf` weighting on its own is not sufficient to eliminate the negative influence of very general hypernyms.

3 Clustering with Synonymy and Antonymy

This work echoes some of the ideas presented in the previous section, but studies a different aspect of the possible contribution of WN to clustering, and makes no use of the hypernym relation. The main idea is as follows: a collection of documents containing a given noun or verb can be subdivided according to the modifiers (adjectives, resp. adverbs) of that search term in the set of documents. Documents containing the same adjective (resp. adverb) can be grouped in the same cluster. The resulting clusters can also be aggregated by bringing together modifiers linked through the WN similarity relation. In either case, the user could be automatically provided with meaningful alternatives (in the shape of pairs of opposite concepts) to further refine the search if one or more pairs of clusters formed in the above way corresponded to pairs of WN antonyms. Ideally, such pairs of clusters would cover a large proportion of documents, and be of equal (or similar) size.

If, in a given search, two such clusters can be found which are sufficiently large to cover the majority of the search results, then subsequent subclusters could be derived from them.

Given that there will invariably be documents which still do not belong to a cluster after this process, it is also investigated how the remaining results could be clustered. In this case, singleton clusters could also be produced based on modifiers which are linked together via the similarity relation. Again, it is hoped that the resulting clusters would be small in number, and between them would cover the majority of the set of search results.

4 Design

The tool is split into two separate functions: initial preprocessing and on-line searching. The first stage uses morpho-lexical analysis, PoS tagging and shallow parsing to replace word forms by their standard lexical entries, and reduce documents from a coherent collection of text to a "bag of features" representation, where each feature in a document contains a head and a corresponding modifier. Optionally, lexical entries can be replaced by their synsets for additional aggregation of features. In case of semantic ambiguity, only the most frequent sense/synset (as listed in WN) is used for the WSJ as a crude form of word sense disambiguation (WSD). In the case of Semcor, the correct synset listed in the corpus is used to emulate the performance of a perfect WSD tool.

The searchable database is represented as a hierarchical set of records, as depicted in Figure 1. The database is split into two "docspaces", which are collections of documents preprocessed for search with nouns, resp. verbs as keywords. For each docspace and document, there is a set of records representing each noun (verb) in that document, and the modifiers with which it is linked in the text. These records also store the frequencies with which the head word and each of its modifiers appear in the document.

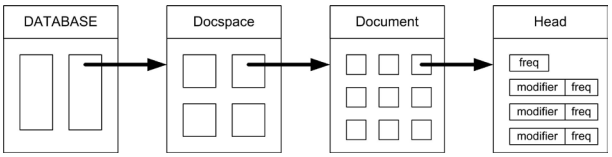


Figure 1: DB structure after preprocessing.

At search time, the keyword is preprocessed in the same way as the documents in the database (to ensure consistent treatment of words), and then all documents containing this keyword are retrieved. Each document is linked to the list of keyword (phrase head) modifiers it contains. An example of this is given in Figure 2. Note that at this point, it is no longer necessary to refer to the actual head, because all of these modifiers are describing the same head.

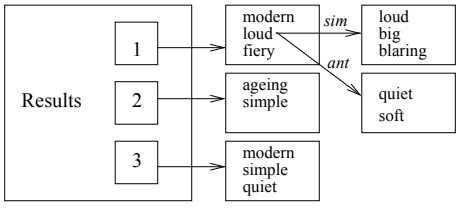


Figure 2: Representation of initial results.

Here the similarity relation can be used, in two separate ways. On its own, it allows to expand each modifier into a list (or adjective/adverb "sim" cluster), containing the original modifier and all the words (adjectives or adverbs) it is similar to. In combination with antonymy, similarity can produce the list of indirect antonyms to the modifier in question. Figure 2 shows how "loud" is expanded into two clusters, one representing adjectives with similar meaning, and another representing the opposite.

In all cases, the ultimate goal is to attempt splitting the documents into pairs of subsets, containing the search keyword with a modifier or its antonym. If neither similarity nor antonymy has been used in the preceding steps, the algorithm will explore only the direct antonymy relation to create these pairs of subsets. If similarity and antonymy are used in the previous step, it is also possible to generate pairs of clusters representing indirect antonyms.

The actual algorithm takes a document's list of modifiers and looks for occurrences of each modifier (and its synonyms and antonyms, if applicable) in the other documents. Given the example in Figure 2, the algorithm would start with the modifiers from document 1 and search for them within documents 2 and 3. It would then search for all the modifiers from document 2 in document 3 (but will not need to look back in document 1). In the illustrated example, documents 1 and 3 will seed a pair of antonym ("ant") clusters defined by "loud" and "quiet".

Entropy (Shannon, 1948) has been used to score the quality of a pair of ant clusters, or of a sim cluster, together with its complement. (In fact, a good case can also be made for the use of information gain (Quinlan, 1986)). In an interactive mode, this score helps list the best pairs of antonyms first, and could potentially be used when hierarchical clustering is considered. However, as the results on the coverage of antonym clusters in the next section suggest, hierarchical clustering is possible only in a small fraction of cases.

For two antonyms a and b, we can define the set of documents covered by them as A_1 and A_2 respectively. We can also define the set of all documents in the current search as the universe, U. The most useful pair of antonyms a and b will be that which comes closest to satisfying the following equations: $A_1 \cup A_2 = U$; $A_1 \cap A_2 = \emptyset$. We then define $m = |A_1 \setminus \{A_1 \cap A_2\}|$, and $n = |A_2 \setminus \{A_1 \cap A_2\}|$. Here m is the number of documents containing modifier a, but not modifier b (and vice versa for n). These can be used to express the entropy equation, as shown below.

$$Ent(S) = - \sum_{i \in \{m,\, n\}} \left(\frac{i}{m+n} \right) \log_2 \left(\frac{i}{m+n} \right)$$

The best pair of antonym clusters will be the one with maximum entropy.

5 Implementation

The main program is written in Perl and interacts with the WN database via the third party package `WordNet::QueryData`[3]. The preprocessing stages are largely taken care of by a subset of the WSJ treebank that has been fully parsed and PoS tagged, presented in a format searchable by Tgrep. This allows features to be extracted using Tgrep's powerful search language, which can search parse trees and use PoS tags and regular expressions to find features. For instance, all adjective-noun pairs in noun phrases ending in "`Adjs N`" can be extracted, e.g., "The short, wealthy, Irish director" → (`short director`), (`wealthy director`), etc. These features are written to a text file, which can then be parsed by the program and turned into a database structure, as illustrated in the previous section. When presented with PoS tags attached, these features can also be preprocessed by WN.

The search tool uses another third party package `Set::Scalar`[4] to convert the arrays of documents produced in a search into set representations, thus allowing the equations from the previous section to be used with ease in the program.

6 Results

Both WSJ and Semcor were converted into "bag of features" representations under a range of configurations. The noun-based docspace representing the WSJ treebank was used to produce the following four data sets:

wsj_jjnn Only stemming is performed on the corpus; no words are grouped into synsets.

[3] Available on `www.cpan.org`.

[4] Also available on `www.cpan.org`.

wsj_jjnn_psyn Both heads and modifiers are replaced with the synset ID representing the most common (primary) sense of the word.

wsj_jjnn_psh Only heads are replaced with primary synset IDs.

wsj_jjnn_psm Only modifiers are replaced with primary synset IDs.

Each of these configurations was batch tested using the complete list of possible search terms (nouns) present in the data. The clustering algorithm was run twice with the following options:

Sim, ant Similarity and indirect antonyms used for clustering.

No sim, no ant Neither of the above used.

In addition, a data set comprising nouns modified by other nouns, and a verb-based data set, were created and batch tested using the "No sim, no ant" strategy, since they were only needed to indicate rough figures for the potential inclusion of these two types of feature in the search tool. They were called **wsj_nnnn** and **wsj_vbrb** respectively.

The Semcor corpus was tested in two configurations for both nouns and verbs:

sem_1jjnn Noun-related data

sem_1jjnn_wsd Noun-related data with sense-tagged words

sem_vbrb Verb-related data

sem_vbrb_wsd Verb-related data with sense-tagged words.

Key results are presented below, with a full set of results available from the second author's Web publication list.[5]

The "Occurrence of Antonyms" test investigates how many results are found containing one or more antonym pairs over the set of search terms. The original hypothesis suggests that the majority of results will contain at least one antonym pair which can then be further broken down. Figure 3 shows the experimental results gathered for the key configurations. With the first three configurations, the total number of possible search terms is in the region of 2000, but when using sense tags with the Semcor corpus, this number rises to 2500.

[5] www.cs.york.ac.uk/~kazakov/my-publications.html

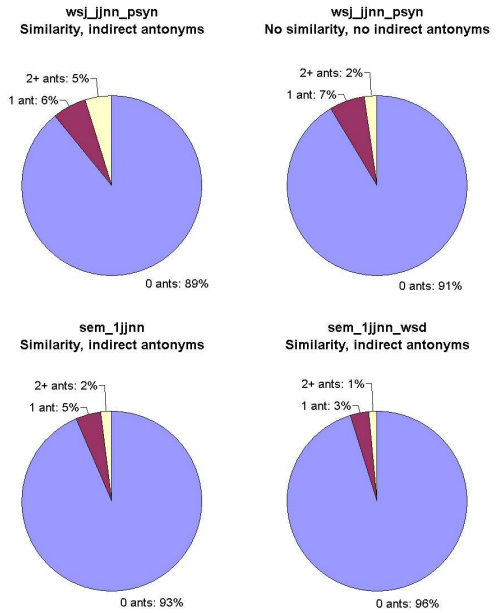


Figure 3: "Occurrence of Antonyms" results for WSJ.

These results show that over all possible searches, the vast majority (typically in the region of 90%) return results that contain no antonym pairs at all. Of the remaining 10%, roughly half contain only a single antonym pair. In some test cases, searches are found to produce as many as 12 pairs, but in most cases, there are rarely more than a handful of search terms in a set which produce more than four pairs. This means that there is simply not a great enough proportion of searches which return even one antonym pair to split further. This evidence is backed up by the results from Semcor, which show a very similar trend over a different corpus.

In terms of entropy and coverage, the average figures for ant pairs in **wsj_jjnn_psyn** with similarity enabled is 0.51, which is quite far from the ideal value of 1. This means that the separation seen in most ant pairs tends to be uneven. The average coverage of ant pairs (the proportion of documents containing the search term and a modifier from either antonym cluster) in the same configuration is just 15%, meaning that the pairs do not actually account for a very substantial portion of the search results. The figures are quoted for this configuration because it is the configuration which produced the best results.

It can also be seen from the first two pie charts in Figure 3 that the use of the similarity relation changes the proportions of results con-

Table 1: Impact of similarity on clusters.

	sim & ind ant	neither
Avg. no. sims	9.86	10.58
Avg. coverage	11%	6%

taining antonyms. When similarity and indirect antonyms are employed, the number of results containing no antonym pairs decreases, if only by a small amount. This means that the similarity relation causes more clusters to link via the antonymy relation, as would be expected.

The impact of the similarity relation is further highlighted in Table 1, where it can be seen that its use with **wsj_jjnn** causes the number of sim clusters to drop, and the average coverage of each cluster (its size) to increase. Again, this is in line with the expected result, since more clusters are being linked together, and joining to form larger clusters.

As for the use of verb-adverb features, the **wsj_vbrb** configuration proved to be of much more limited use. Semcor produced a grand total of 348 possible search terms, compared to the noun-adjective features, which produce in the region of 2000 search terms. For WSJ, the number of verb-adverb features was much lower, to the point of being useless.

The **wsj_nnnn** configuration (noun-noun features) produced 1257 possible search terms, which may be smaller than the number produced by the data sets for adjective-noun features, but is large enough to be significant. The search for singleton clusters (analogue to the sim clusters) based on an exact match of a noun modifier produced the results in Table 2.

7 Conclusions

The primary goal was to establish if the antonymy relation could be used on the modifiers found in documents to decompose a set of search results into a hierarchy of sub-clusters. The experiments performed suggest, with a convincing majority, that this is not possible. In all experiments, a small proportion of search terms produced results containing one antonym pair, and even then the values for entropy and coverage were not as high as was hoped for. This means that even for those search terms which do return antonym pairs, the quality of those pairs is far from ideal. The use of indirect antonyms did indeed increase the average number of antonym pairs found, as would be expected, but this was still not enough to use this clustering technique in isolation. This work has also shown that compared to nouns, verbs offer a limited amount of help for document clustering.

Table 2: Sim clusters of NN phrases.

	wsj_nnnn
Total search terms	1257
Average number of clusters	5.54
Average coverage	0.42

An analysis of the make-up of selected search results shows that the average search result can be expected to contain at most one or two large clusters, and the remainder of the results tend to be a large collection of very small clusters, typically only one or two documents in size. This is very much counter to the hypothesis expressed earlier in the report, and suggests that if modifiers are going to be used as the features which make up clusters, some sort of agglomerative approach needs to be implemented after the primary clusters are returned, in order to turn these insignificant clusters into a smaller collection of larger clusters. One possible idea would be to group clusters containing the same (or similar) modifier and heads linked through hypernymy (Hotho *et al.*, 2003; Kazakov and Sedding, 2004). In this way, the two approaches might offset each other's drawbacks somewhat.

Finally, one should seriously consider using the sim clusters of noun-noun `NP` phrases, the coverage of which appears nearly optimal.

References

A. Hotho, S. Staab, and G. Stumme. Wordnet Improves Text Document Clustering, 2003.

D. Kazakov and J. Sedding. Wordnet-Based Text Document Clustering. *Third Workshop on Robust Methods in Analysis of Natural Language Data (ROMAND)*, pages 104–113, 2004.

G. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. Five Papers on Wordnet. *Tech. report, Princeton University*, 1993.

R. Pito. Tgrep manual. Distributed with the Penn Treebank, 1992.

J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

G. Salton and A. Wong. A vector space model for automatic indexing. *Communications of the ACM*, 18:613–620, 1975.

C.E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27, 1948.

O. Zamir and O. Etzioni. Grouper: a dynamic clustering interface to Web search results. *Computer Networks*, 31(11–16):1361–1374, 1999.

Faking Errors to Avoid Making Errors: Very Weakly Supervised Learning for Error Detection in Writing

Jonas Sjöbergh
KTH KOD
SE-100 44 Stockholm, Sweden
jsh@nada.kth.se

Ola Knutsson
KTH KOD
SE-100 44 Stockholm, Sweden
knutsson@nada.kth.se

Abstract

This paper describes a method to create a grammar checker "for free". It requires no manual work, only unannotated text and a few basic NLP tools. The method used is to simply annotate a lot of errors in written text and train an off-the-shelf machine learning implementation to recognize such errors. To avoid manual annotation artificially created errors are used for training. Recall is comparable to other grammar checkers but precision is lower. Our method also complements traditional grammar checkers, i.e. they do not always find the same errors. The evaluation is performed on real errors.

1 Introduction

Automatic grammar checking is traditionally done using manually written rules, constructed by a computational linguist. We present a method that saves a lot of work by training a machine learning algorithm on artificially created errors.

Methods for detecting grammatical errors without using manually constructed rules have been presented before. (Atwell 87) uses the probabilities in a statistical part-of-speech tagger, detecting errors as low probability part-of-speech sequences. A similar method is presented in (Bigert & Knutsson 02), where new text is compared to known correct text and deviations from the "language norm" are flagged as suspected errors. (Chodorow & Leacock 00) present a method based on mutual information measurements to detect incorrect usage of difficult words.

(Mangu & Brill 97) use machine learning to detect when one word has been confused with another. (Golding 95) combines several methods to solve the same problem. (Hardt 01) treats comma placement and determiner-noun agreement in Danish as a confusion set problem in a similar way. He also uses artificial errors as negative examples. Another example of a confusion set problem is English article usage before noun phrases (Han *et al.* 04).

Unlike most of the methods mentioned our method is applicable to a wide range of error types. Our method is similar to the one presented by (Izumi *et al.* 03), who manually annotated errors in transcribed spoken language.

Our method is based on viewing grammar checking more or less as a tagging task. We simply train an available machine learning algorithm on annotated errors to create a grammar checker. The new idea in our approach is to use only artificial errors for training, and we show that while it might not be as good as training on real errors, it still produces a useful grammar checker. The strength of this approach is that it is very resource lean. No time consuming manual annotation of errors is needed, neither is access to large amounts of human produced (unintentional) errors. Almost no manual work at all is required, only unannotated text and a few basic NLP tools are used.

2 Method of Detecting Errors

The basic idea of our method is to treat grammar checking as a tagging task. Collect a lot of text, mark all errors with "ERROR" and all other words with "OK". Train an off-the-shelf tagger on this data and you have a grammar checker. To achieve better feedback it is possible to have different tags for different types of errors, i.e. "SPELLING", "VERB-TENSE", etc. Another way to achieve this is to train a new specialized classifier for each error type, which ignores other types of errors.

Finding these errors and annotating them requires a lot of work. Our method avoids this by using artificial errors. A lot of text without errors is used, and the text is then corrupted by adding errors. Since they are added automatically they can be annotated at the same time. When this is done we automatically annotate the resulting text with part-of-speech (PoS), using TnT (Brants 00). The words, PoS and error anno-

tation is then used as training data for the automatic grammar checker. Almost any machine learning implementation could be used for this. We use fnTBL (Ngai & Florian 01), a transformation based rule learner, which produces rules that are easily understood by humans.

Below is an example of an error generation program, for agreement errors. When implemented in a high level scripting language, the code is not much longer than this pseudo code. Since the main strength of our method is that it is resource lean, the simpler the error generation the better.

```
(1) Read lemma lexicon (or stems)
(2) Read PoS-tags with agreement constraints
(3) Run PoS-tagger
(4) For each tagged sentence:
(5)  Pick random word with agreement constraint
(6)  Get lemma (lexicon)
(7)  Get random word with this lemma (lexicon)
(8)  If not exact same word:
(9)   Change word, mark as error
```

If we run the error generation code on "I bought a car." we could get for instance "I/OK bought/OK a/OK cars/ERR ./OK".

The error generation programs sometimes change a sentence so that the result is still grammatical. One simple example would be a program that inserts word order errors by randomly changing the order of neighboring words. Not all changes will lead to errors, for example "I heard dogs barking" and "I heard barking dogs" are both correct, but "heard I dogs barking" is not. Such sentences will of course still be marked as erroneous. This is not a great problem, since if something is correct there are usually many examples of this which are not the result of changes, and thus marked as correct. This means that the learner will in general only learn rules for those artificial errors that result in text which is incorrect, since the other "errors" will be drowned out by all the correct examples.

Our method can be used on many error types. Some examples of errors that could be generated artificially include: word order errors (reorder randomly selected words), missing words (remove randomly selected words), "hard" spelling errors (replace words with another word with only a one letter difference), split compounds (replace all words that could be made from concatenating two other words in the corpus with these two words), agreement errors and verb tense errors (use a dictionary lookup to replace words with another inflectional form of the same word), prepositional use (change prepositions to other prepositions), etc.

The main strength is errors that are simple to generate, but where the resulting sentence structure is hard to predict. Word order errors and split compounds are examples of such errors. Errors such as repeated words for which it is straightforward to predict the result can also be handled by our method, but is probably better handled by traditional methods.

We have tested our method on two different error types: split compounds, an error type suited to our method, and agreement errors, suited to traditional grammar checking methods. Agreement errors were tested to see how our method holds up where the competition is the hardest. We compared our method to three other grammar checkers, evaluating them on Swedish texts of different genres.

2.1 Split Compounds

In compounding languages, such as Swedish and German, a common error is to split compound words, i.e. write "quick sand" when "quicksand" was intended. Two concrete examples from Swedish: (1) "en långhårig sjukgymnast" means "a physical therapist with long hair". Splitting the compounds to make "en lång hårig sjuk gymnast" is still grammatical but the meaning is changed to "a tall, hairy and sick gymnast". (2) If the compound "ett personnummer" ("social security number") is split to "ett person nummer" ("one person number") it would lead to an agreement error and be ungrammatical.

Training data for the erroneously split compounds experiments was a one million words corpus of written Swedish, the Stockholm-Umeå Corpus, SUC (Ejerhed *et al.* 92). A modified spelling checker, Stava (Domeij *et al.* 94; Kann *et al.* 01) was used to automatically split compounds.

While some manual work has been put into creating Stava (and thus in a sense made this type of error generation less independent of manual work), the part used here, i.e. the compound analysis component, was automatically constructed from a dictionary. If however there are tools available that someone already put a lot of manual

effort into creating, our method could use these. Our method would then be a method of creating a grammar checking component from other tools in an unsupervised way.

The training data consisted of the corpus texts, to show correct language use, and another copy of all the corpus texts. The second copy had all compounds recognized by the compound splitter split into their components, with the components marked "error".

The rule learner was given the word n-grams, PoS n-grams and error annotation n-grams. The n-grams were unigrams, bigrams and trigrams. Some combinations of these were also allowed, such as the current word and error annotation trigrams. The initial guess for the learner was that words more common in compounds than as a single word (in the training data) were probably errors and all other words correct. The best rules found by the learner used PoS bigrams and error annotation of one word and PoS of its neighbor.

To improve the precision of the learned rules one can use the fact that if a compound is split it will result in at least two components. Any single word marked error is thus probably a false positive (or one of its neighbors is a false negative), and can be removed. Since there was a spelling checker available we improved this a little by filtering the output through the spelling checker. If a suspicious word could not be combined into a correct compound by using a neighboring word also marked "error" it was considered a false alarm and the error was removed. This improved the precision but also removed many correctly detected split compounds, usually because they were misspelled as well as erroneously split (and would thus be found by the spelling checker instead).

Using the spelling checker gave only a small improvement over just removing errors with no neighboring error, while both methods improved the precision of the original rules significantly.

2.2 Agreement Errors

In Swedish, determiners, adjectives, possessives and nouns must agree in number, gender and definiteness. Agreement errors are quite common, especially when revising text using a computer. The agreement can span long reaches of text, which can make the errors hard to detect. Manually writing good rules for agreement errors is relatively straightforward, and it is one of the more popular error categories to detect among automatic grammar checkers.

To generate artificial errors the SUC corpus was used again. In each sentence a word from any word class with agreement restrictions was randomly selected. This word was then changed to another randomly selected form of the same word. This was done by a simple lexicon lookup were the lemma of the word was found and another word with the same lemma and a different surface form was selected. The selected word was marked as an error and all other words were marked as correct.

When an agreement error occurs, at least two words are involved. We only mark the changed word as an error, although it would be reasonable to mark all words with agreement restrictions related to the changed word. One reason for this is that it is easy to mark the changed word but hard to mark the other words. If we could find them, we would already have an agreement error detection method. Also, since we know which word was changed, we know which word should be corrected to retrieve the intended meaning, even though the agreement error itself could likely be corrected in several ways.

As features for the machine learner the gender, number and definiteness of the word were given (if applicable). All this information is included in the tagset we trained TnT on, and was automatically assigned. The PoS of the word and the error annotation were also included. Unigrams, bigrams, trigrams and combinations of these features were used. The best rules combined PoS and n-grams of the gender features.

The initial guess was that there were no errors in the text. A baseline was constructed by locating every occurrence of two consecutive words that had different gender, number or definiteness and marking the first of these as an error. This baseline could be used as initial guess for the learner, which gives higher precision than the original initial guess, since many rules are learned that remove alarms (mostly spurious alarms from the baseline), but lower recall.

3 Evaluation

We compared our method to three different grammar checkers for Swedish, one commercial grammar checker, one state of the art research product and one method not based on manually written rules.

	MS Word (manual)	ProbGr. (statistical)	Granska (manual)	SnålGr.	SnålGr. + Filter	Baseline	Baseline + Filter	Union	Inter-section
Detected errors	75	225	322	588	535	331	120	582	275
False negatives	-	-	490	224	277	481	692	230	537
False positives	-	-	6	49	24	162	6	29	1
Precision	-	-	98%	92%	96%	67%	95%	95%	100%
Recall	-	-	40%	72%	66%	41%	15%	72%	34%

Table 1: Detection of split compound components. The baseline is simply the most common tag for each word ("error" or "correct"), from the training data. "Union" is any word marked "error" by either the manual rules of the Granska grammar checker or the filtered automatic rules of SnålGranska (our method). "Intersection" is any word marked by both. MS Word and ProbGranska do not specifically address the problem of split compounds but find some anyway, but of course with a different diagnosis.

- The Swedish grammar checker in Microsoft Word 2000, which uses a grammar checker developed by Lingsoft (Arppe 00; Birn 00). It is based on manually constructed rules. The rules are tuned for high precision.

- Granska (Domeij *et al.* 00), a state of the art grammar checker, also based on manually constructed rules. Roughly 1 000 hours of manual work have been put into creating the rule set.

- ProbGranska (Bigert & Knutsson 02), a statistically based grammar checker. It detects errors by looking for things that are "different" from known correct text, based on PoS trigrams. ProbGranska is currently used as a complement to the manual rules of Granska in a grammar checking environment.

3.1 Evaluation on Collections of Errors

The first evaluation was performed on collections of examples of authentic split compounds and agreement errors. These were all taken from real texts, but since there is at least one error in each sentence it is a quite unrealistic data set, and it is easy for the grammar checkers to achieve high precision with so many errors available. The benefit of these collections is that all errors that occur have been manually annotated, so it is easy to check the precision and recall of the grammar checkers. Since these are real errors a grammar checker with a good result on these texts will likely work well on "real" texts too.

For split compounds examples were taken mostly from web pages and newspapers. There were 5 124 words, of which 812 were components from split compounds. Most compounds consisted of only two components. Sometimes two (but rarely more) adjacent compounds were both split. The results are shown in Table 1.

For split compounds the results are quite good. Compared to the other grammar checkers, the automatically learned rules have lower precision but the highest recall. Detecting split compounds is considered quite hard, and Granska is one of the few grammar checkers that actually tries to detect split compounds. It is likely the best grammar checker currently available for this.

The grammar checker in MS Word 2000 does not look for split compounds but these errors sometimes look like other types of errors that MS Word recognizes. On the test data MS Word classed 75% of the detected split compounds as spelling errors. One third of these were caused by the split compound also being miss-spelled, one third by the compound containing a word which was not recognized (e.g. "Rambo") and one third by the morphological change of the head of the compound. MS Word classed the remaining detected split compounds as agreement errors.

The ProbGranska extension to Granska often finds split compounds. In the test data most of the alarms generated by ProbGranska are caused by split compounds.

For agreement errors the data consisted of 4 556 words, also mostly from newspapers or the Internet. There were 221 agreement errors in the test data, the results are shown in Table 2.

For agreement errors the results are not as impressive, which is to be expected since agreement errors are one of the best covered error types of traditional grammar checkers. While the automatic rules are outperformed by the manually cre-

	MS Word (manual)	ProbGranska (statistical)	Granska (manual)	SnålGranska	Baseline	Union	Intersection
Detected errors	71	17	101	88	100	134	54
False negatives	155	-	125	138	126	92	172
False positives	1	-	5	15	143	19	1
Precision	99%	-	95%	85%	41%	88%	98%
Recall	31%	-	45%	39%	44%	60%	24%

Table 2: Detection of agreement errors. The baseline marks the first of any two consecutive words that have different gender, number or definiteness as an error. "Union" is any word marked "error" by either the manual rules of the Granska grammar checker or the automatic rules of SnålGranska (our method). "Intersection" is any word marked by both. ProbGranska does not specifically look for agreement errors.

ated rules, the results are still good enough to be useful.

The main reason for the lower recall of the automatic rules is that they only work in a small local window. Many of the errors detected by the manual rules span tens of words. Since the automatic rules find none of these errors and still manages to find almost as many errors, there are a lot of errors detected by the automatic rules not found by the manual rules. Combining the two methods thus gives better results than either method individually, as shown in Table 2. They also complement each other, though not as much, on split compounds, as shown in Table 1.

3.2 Evaluation on Real Texts

To evaluate the performance on real texts a few sample texts were collected. All grammar checkers were then run on the texts. All words suspected to contain errors by any of the grammar checkers were manually checked to see if it was a real error. The texts were not manually checked to find all errors, since that would require a lot of work and the time was not available. This gives the precision of the grammar checkers, but not the recall since there could be many errors not detected by any of the grammar checkers. It is possible to get an upper bound on the recall though, using the errors missed by one grammar checker and detected by another.

The first genre we evaluated the grammar checkers on was old newspaper articles. These were taken from the Swedish Parole corpus (Gellerstam *et al.* 00), which also contains other genres though only newspaper texts were used here. These texts are very hard for the grammar checkers, since they are well proofread and contain almost no errors. The results are shown in Table 3. The results are not impressive, the precision is very low for all grammar checkers. Since there are almost no errors to find, this is to be expected. The number of false positives (false alarms) gives an indication of whether the grammar checkers would be usable for writers who make few errors. 50 false alarms, as for our presented method, in 10 000 words is probably tolerable, considering that the commercial grammar checker produces about twice as many when including spelling error reports, though of course it also tries to capture more error types.

The second genre was essays written by people learning Swedish as a second language. These were taken from the SSM-corpus (Hammarberg 77). These texts contain a lot of errors, which is generally good for the grammar checkers (easier to get high precision). It also leads to problems though, since many errors overlap and there is often very little correct text to base any analysis on. Results are shown in Table 4. There are a lot of errors that no grammar checker detects, in a sample that was manually checked to find all errors less than half the errors were detected.

The grammar checkers using manually constructed rules show much higher precision (about 95%) than our presented method (about 86%). They also detect many more errors, mainly because they also look for spelling errors, which are common and much easier to detect. When it comes to grammatical errors the recall is comparable to the manual rules. On split compound errors, which our method is well suited for and which are hard to describe with rules, our method performs very well. On agreement errors, which are one of the best covered error types using man-

	MS Word	ProbGranska	Granska	SnålGranska	Total
All detected errors	10	1	8	3	13
All false positives	92	36	35	50	200
Detected spelling errors	8	0	6	1	9
False positives	89	-	20	-	101
Detected grammatical errors	2	1	2	2	4
False positives	3	36	15	50	99
Detected agreement errors	0	0	0	1	1
Detected split compounds	0	0	0	0	0

Table 3: Evaluation on proofread newspaper texts, 10 000 words. Since there are very few remaining errors to detect, performance is less than impressive.

	MS Word	ProbGranska	Granska	SnålGranska	Total
All detected errors	392	101	411	122	592
All false positives	21	19	13	19	67
Detected spelling errors	334	34	293	26	362
False positives	18	-	5	-	21
Detected grammatical errors	58	67	118	96	230
False positives	3	19	8	19	46
Detected agreement errors	32	9	49	43	74
Detected split compounds	5	8	20	27	35

Table 4: Evaluation on second language learner essays, 10 000 words. With many errors in the text high precision is to be expected. Less than half of all errors are detected, though.

ual rules, its performance is still quite good, with similar recall but lower precision compared to the manual rules.

It is also interesting to note that the grammar checkers do not overlap very much in which errors they detect. A total of 230 grammatical errors are detected but no individual grammar checker detects more than 118. Combining different methods, for instance by signaling an error whenever at least one grammar checker believes something is wrong, would thus give much higher recall.

The final genre was student essays written by native speakers, Table 5. Again, the results are not impressive for any of the grammar checkers. Many false alarms stem from quotations, law books and old texts such as the Bible are quoted. These contain text that is grammatical but differs a lot from "normal" language use. There are also false alarms when spoken language constructions that are rare in written texts are used. This is especially true for the two automatic methods, which both compare new texts to the "language norm" they were trained on (in this case written language).

4 Conclusions and Discussion

We have presented an error detection method that requires almost no manual work. It works quite well for detecting errors. It has lower precision than state of the art grammar checkers based on manually constructed rules, but the precision is high enough to be useful. For some error types the recall of the new method is much higher than the recall of other grammar checkers.

The greatest advantage of this method of creating a grammar checker is that it is very resource lean. A few minutes were spent on generating artificial errors. Some other resources are also needed but only commonly available resources: unannotated text, a part-of-speech tagger and a spelling checker was all that was used.

If several different modules are trained to detect different types of errors they can be combined into one framework that detects many error types. In this case false alarms become a problem, since even if each module only produces a few false alarms the sum of them might be too high. In our tests many false alarms were caused by some other type of error occurring. This kind of false

	MS Word	ProbGranska	Granska	SnålGranska	Total
All detected errors	38	23	48	28	90
All false positives	31	45	13	31	111
Detected spelling errors	24	3	17	1	25
False positives	28	-	0	-	28
Detected grammatical errors	14	20	31	27	65
False positives	3	45	13	31	83
Detected agreement errors	5	0	11	8	15
Detected split compounds	0	1	1	1	1

Table 5: Evaluation on essays written by native speakers, 10 000 words. Frequent use of spoken language style and quotations from for instance legal documents lead to a lot of false alarms in these essays.

alarm might not be a serious problem, since they are caused by real errors and just have the wrong classification. Possibly the module which should find this type of error will also find those errors and the correct classification will also be available. It is also possible to steer the machine learner towards high precision (few false alarms).

It is especially interesting that the method works so well for split compounds. This is a common problem for second language learners of Swedish and also quite common in informal texts by native speakers. It is also a hard problem to write rules for manually. Few grammar checkers address these errors.

Another interesting and useful result is that the automatically learned rules complement the manually constructed rules. This means that they do not find the same errors, so combining the two methods to achieve better results than each individual method is possible.

Acknowledgments

We thank Viggo Kann for contributing useful ideas and helpful suggestions.

This work has been funded by The Swedish Agency for Innovation Systems (VINNOVA).

References

(Arppe 00) Antti Arppe. Developing a grammar checker for Swedish. In T. Nordgård, editor, *Proceedings of Nodalida '99*, pages 13–27. Trondheim, Norway, 2000.

(Atwell 87) Eric Steven Atwell. How to detect grammatical errors in a text without parsing it. In *Proceedings of the 3rd EACL*, pages 38–45, Copenhagen, Denmark, 1987.

(Bigert & Knutsson 02) Johnny Bigert and Ola Knutsson. Robust error detection: A hybrid approach combining unsupervised error detection and linguistic knowledge. In *Proceedings of Romand 2002, Robust Methods in Analysis of Natural language Data*, pages 10–19, 2002.

(Birn 00) Juhani Birn. Detecting grammar errors with lingsoft's Swedish grammar checker. In T. Nordgård, editor, *Proceedings of Nodalida '99*, pages 28–40. Trondheim, Norway, 2000.

(Brants 00) Thorsten Brants. TnT – a statistical part-of-speech tagger. In *Proceedings of the 6th Applied NLP Conference, ANLP-2000*, pages 224–231, Seattle, USA, 2000.

(Chodorow & Leacock 00) Martin Chodorow and Claudia Leacock. An unsupervised method for detecting grammatical errors. In *Proceedings of NAACL'00*, pages 140–147, Seattle, USA, 2000.

(Domeij *et al.* 94) Rickard Domeij, Joachim Hollman, and Viggo Kann. Detection of spelling errors in Swedish not using a word list en clair. *Journal of Quantitative Linguistics*, 1:195–201, 1994.

(Domeij *et al.* 00) Richard Domeij, Ola Knutsson, Johan Carlberger, and Viggo Kann. Granska – an efficient hybrid system for Swedish grammar checking. In *Proceedings of Nodalida '99*, pages 49–56, Trondheim, Norway, 2000.

(Ejerhed *et al.* 92) Eva Ejerhed, Gunnel Källgren, Ola Wennstedt, and Magnus Åström. The linguistic annotation system of the Stockholm-Umeå Corpus project. Technical report, Department of General Linguistics, University of Umeå (DGL-UUM-R-33), Umeå, Sweden, 1992.

(Gellerstam *et al.* 00) Martin Gellerstam, Yvonne Cederholm, and Torgny Rasmark. The bank of Swedish. In *Proceedings of LREC 2000*, pages 329–333, Athens, Greece, 2000.

(Golding 95) Andrew Golding. A bayesian hybrid for context sensitive spelling correction. In *Proceedings of the 3rd Workshop on Very Large Corpora*, pages 39–53, Cambridge, USA, 1995.

(Hammarberg 77) Björn Hammarberg. Svenskan i ljuset av invandrares språkfel. *Nysvenska studier*, 57:60–73, 1977.

(Han *et al.* 04) Na-Rae Han, Martin Chodorow, and Claudia Leacock. Detecting errors in english article usage with a maximum entropy classifier trained on a large, diverse corpus. In *Proceedings of LREC-2004*, pages 1625–1628, Lisbon, Portugal, 2004.

(Hardt 01) Daniel Hardt. Transformation-based learning of Danish grammar correction. In *Proceedings of RANLP 2001*, Tzigov Chark, Bulgaria, 2001.

(Izumi *et al.* 03) Emi Izumi, Kiyotaka Uchimoto, Toyomi Saiga, Thepchai Supnithi, and Hitoshi Isahara. Automatic error detection in the Japanese learners' English spoken data. In *Companion Volume to the Proceedings of ACL '03*, pages 145–148, Sapporo, Japan, 2003.

(Kann *et al.* 01) Viggo Kann, Rickard Domeij, Joachim Hollman, and Mikael Tillenius. Implementation aspects and applications of a spelling correction algorithm. In L. Uhlirova, G. Wimmer, G. Altmann, and R. Koehler, editors, *Text as a Linguistic Paradigm: Levels, Constituents, Constructs. Festschrift in honour of Ludek Hrebicek*, volume 60 of *Quantitative Linguistics*, pages 108–123. WVT, Trier, Germany, 2001.

(Mangu & Brill 97) Lidia Mangu and Eric Brill. Automatic rule acquisition for spelling correction. In *Proceedings of the 14th International Conference on Machine Learning*, pages 187–194, 1997.

(Ngai & Florian 01) Grace Ngai and Radu Florian. Transformation-based learning in the fast lane. In *Proceedings of NAACL-2001*, pages 40–47, Carnegie Mellon University, Pittsburgh, USA, 2001.

Syntactic Identification of Attribution in the RST Treebank

Peter Rossen Skadhauge and **Daniel Hardt**
CMOL/Department of Computational Linguistics
Copenhagen Business School
DENMARK
{prs,dh}@id.cbs.dk

Abstract

We present a system that automatically identifies Attribution, an intra-sentential relation in the RST Treebank. The system uses uses syntactic information from Penn Treebank parse trees. It identifies Attributions as structures in which a verb takes an SBAR complement, and achieves a f-score of .92. This supports our claim that the Attribution relation should be eliminated from a discourse treebank, since it represents information that is already present in the Penn Treebank, in a different form. More generally, we suggest that intra-sentential relations in the RST Treebank might all be eliminable in this way.

1 Introduction

There has been a growing interest in recent years in Discourse Structure. A prominent example of this is the RST Treebank(Carlson *et al.* 02), which imposes hierarchical structures on multi-sentence discourses. Since the texts in the RST Treebank are taken from the syntactically annotated Penn Treebank(Marcus *et al.* 93), it is natural to ask what the relation is between the discourse structures in the RST Treebank and the syntactic structures of the Penn Treebank.

In our view, the most natural relationship would be that discourse structures always relate well-formed syntactic expressions, typically sentences. Discourse trees would then be seen as elaborations of syntactic trees, adding relations between sentential nodes that are not linked by syntactic relations. This would allow discourse structures and syntactic structures to coexist in a combined hierarchical structure.

Surprisingly, this is not what we have found in examining the syntax-discourse relation in the RST Treebank. A large proportion of relations apply to subsentential spans of text;[1] spans that may or may not correspond to nodes in the syntax tree. Is this complicated relation between syntax and discourse necessary? Our hypothesis is that the subsentential relations in the RST Treebank are in fact redundant; if this is true it should be possible to automatically infer these relations based solely on Penn Treebank syntactic information.

In this paper, we present the results of an initial study that strongly supports our hypothesis. We examine the Attribution relation, which is of particular interest for the following reasons:

- It appears quite frequently in the RST Treebank (15% of all relations, according to (Marcu *et al.* 99))
- It always appears within, rather than across, sentence boundaries
- It conflicts with Penn Treebank syntax, always relating text spans that do not correspond to nodes in the syntax tree

We describe a system that identifies Attributions by simple, clearly defined syntactic features. This system identifies RST Attributions within precision and recall over 90%. In our view, this strongly supports the view that Attribution is in fact a syntactic relation. The system performs dramatically better than the results reported in (Soricut & Marcu 03) for automatic identification of such relations, where the precision and recall were reported at below .76. Furthermore, human annotator agreement reported in the RST Treebank project is also well below our results, with reported f-scores no higher than .77.(Soricut & Marcu 03)

In what follows, we first describe Attributions as they are understood in the RST Treebank project. Next we present the Attribution identification procedure, followed by a presentation of results. We compare these results with related work, as well as with inter-coder agreement re-

[1] In the TRAINING portion of the RST Treebank, we found 17213 Elementary Discourse Units (EDU's). Of these only 6068 occurred at sentence boundaries.

ported in the RST Treebank project. Finally, we discuss plans for future work.

2 Attributions in the RST Treebank

The RST coding manual(Carlson & Marcu 01) gives the following definition of Attribution:

> Instances of reported speech, both direct and indirect, should be marked for the rhetorical relation of ATTRIBUTION. The satellite is the source of the attribution (a clause con- taining a reporting verb, or a phrase beginning with according to), and the nucleus is the content of the reported message (which must be in a separate clause). The ATTRIBUTION relation is also used with cognitive predicates, to include feelings, thoughts, hopes, etc.

The following is an example cited in the coding manual:

> [The legendary GM chairman declared] [that his company would make "a car for every purse and purpose."]wsj_1377

According to the RST Treebank, the attribution verb is grouped with the subject into a single text span. This constitutes the Attribution Satellite, while the Nucleus is the SBAR complement of the attribution verb, as shown below in Figure 1.

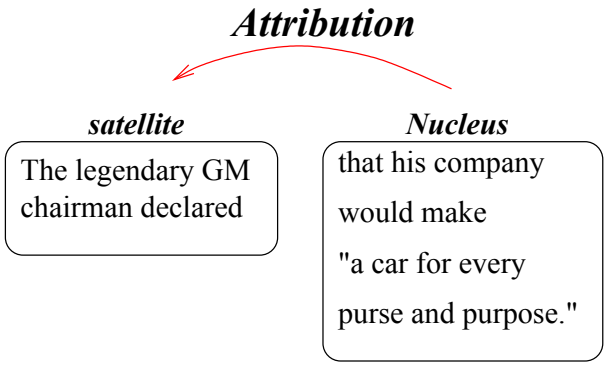


Figure 1: Attribution in the RST Treebank

This conflicts with the syntactic structure in the Penn Treebank. As shown in Figure 2, the attribution verb is grouped with its SBAR complement, forming a VP, which is related to the subject.

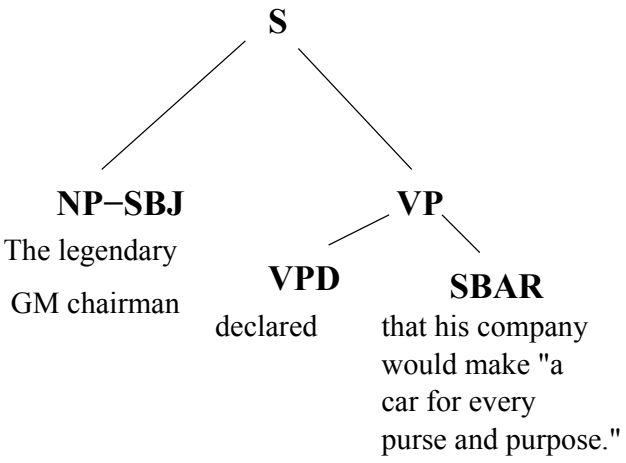


Figure 2: Attribution in the Penn Treebank

The main difference in the two structures regards the position of the verb; in the RST Treebank, the verb is grouped with the subject, while in the Penn Treebank, it is grouped with the SBAR complement. In the following section, we describe our method for identifying RST Attributions, based on the Penn Treebank syntactic structure.

3 Identifying Attributions

We define three forms of Attribution relations:

- **Basic:** A verb is followed by a sentential complement position
- **Backwards:** The sentential complement precedes the verb. In these cases, a trace appears as complement to the verb, and is coindexed with the sentential complement
- **According-To:** the phrase "according to" occurs

3.1 Basic Attributions

In this form, a sentential object immediately follows a verb.

Consider the example

(1) Now, the firm says it's at a turning point.

In PTB, the sentence is annotated as in :

(2)

```
( (S
    (ADVP-TMP (RB Now) )
    (, ,)
    (NP-SBJ (DT the) (NN firm) )
    (VP (VBZ says)
```

```
    (SBAR (-NONE- 0)
      (S
        (NP-SBJ (PRP it) )
        (VP (VBZ 's)
          (PP-LOC-PRD (IN at)
            (NP (DT a) (NN turning)
                       (NN point) ))))))
  (. .) ))
```

Sentential objects are annotated as SBAR regardless of the presence of complementizers. Thus, the subroutine searches the corpus for structures matching the template (3), which matches verb phrases in which a verb is followed by an SBAR.

(3) `(VP ... (V.. ...) (SBAR ...) ... )`

The SBAR must follow immediately after the verb, which may be the last verb in a verbal cluster. This represents a simplification, since adverbials may occur between the verb and its SBAR complement. Our implementation correctly identifies 1497 occurrences, and incorrectly identifies 215 occurrences of attributions, corresponding to a contribution to the total recall of 0.615 with a precision of 0.874.

3.2 Backwards Attributions

Where a sentential object does not immediately follow its corresponding verb, it is represented as a trace which is coindexed with the S. In the following example, the sentential complement precedes the sentence:

(4) "I believe that any good lawyer should be able to figure out and understand patent law",$_i$ Judge Mayer says t_i

The example is represented as follows in PTB:

(5)

```
((S-6 (`` ``)
  (NP-SBJ-2 (PRP I) )
  (VP (VBP believe)
   (SBAR (IN that)
    (S
     (NP-SBJ-4 (DT any) (JJ good)
                        (NN lawyer) )
      (VP (MD should)
       (VP (VB be)
        (ADJP-PRD (JJ able)
         (S
          (NP-SBJ (-NONE- *-4) )
          (VP (TO to)
           (VP
            (VP (VB figure)
             (PRT (RP out) )
             (NP (-NONE- *RNR*-5) ))
            (CC and)
            (VP (VB understand)
             (NP (-NONE- *RNR*-5) ))
            NP-5 (NN patent)
                 (NN law) )))))))))
  (PRN
   (, ,)
   (`` ``)
   (S
    (NP-SBJ (NNP Judge) (NNP Mayer) )
    (VP (VBZ says)
       (S (-NONE- *T*-6) )))
```

The sentential object of "says" is represented by the trace (`(S (-NONE- *T*-6) ))`), which is coindexed with the outer sentence (`(S-6)`).

The procedure searches for sentences of the types `S`, `S/SBAR`, and `VP/S-TPC` which are linked to a trace in the surrounding sentence. Thus, it covers cases of topicalization and sentence inversion which are the most frequent reasons for sentential objects not occurring immediately after the verb.

The subroutine covering sentential objects linked by traces make 700 correct and 4 incorrect predictions, corresponding to a recall contribution of 0.287 with a precision of 0.994.

3.3 According-To Attributions

Also categorized as attributions are "according to" expressions. These are identified with a separate subroutine which simply identifies occurrences of the two words "according" and "to" in sequence.

Example:

(6) Now, according to a Kidder World story about Mr. Megargel, all the firm has to do is "position ourselves more in the deal flow."

(7)

```
( (S
```

```
(ADVP-TMP (RB Now) )
(, ,)
(PP (VBG according)
  (PP (TO to)
    (NP
      (NP (DT a) (NNP Kidder)
          (NNP World) (NN story) )
      (PP (IN about)
        (NP (NNP Mr.)
            (NNP Megargel) )))))
(, ,)
(NP-SBJ
  (NP (DT all) )
  (SBAR
    (WHNP-1 (-NONE- 0) )
    (S
      (NP-SBJ-2 (DT the) (NN firm) )
      (VP (VBZ has)
        (S
          (NP-SBJ (-NONE- *-2) )
          (VP (TO to)
            (VP (VB do)
              (NP (-NONE- *T*-1)
                        ))))))))
(VP (VBZ is) (`` ``)
  (VP (VB position)
    (NP (PRP ourselves) )
    (ADVP-MNR (RBR more)
      (PP (IN in)
        (NP (DT the)
            (NN deal) (NN flow) )))))
(. .) ('' '') ))
```

The subroutine identifies 87 "according to" expressions correctly, and 1 incorrectly.

4 Discussion of Results

Our system for recognizing Attributions is a quite direct implementation of the description of Attribution given in the RST Tagging Manual, relying on simple structural characteristics. In developing the system, we examined data in the Training portion of the RST Treebank. To ensure that our implementation was not tuned to any idiosyncrasies of the data we examined, we performed two tests of our system, on the Test portion of the RST Treebank as well as the Training portion. We avoided any examination of data in the Test portion of the Treebank.

Given the general nature of the syntactic characteristics of our system, it is not surprising that the results on the Training and Test portions of the Treebank our quite similar. We present the overall results on both portions of the Treebank, followed by more detailed results, giving the contributions of the main subparts of the system.

4.1 Overall Results

The following figure summarizes the results of executing the procedure on the two portions of the Treebank.

Corpus	Precision	Recall	F-score
Training	0.912	0.938	0.925
Test	0.897	0.944	0.920

Figure 3: Overall results

4.2 Subparts of the System

Next, we present the contribution of each of the three subparts of the system.

	+	−	Prec	Rec
Basic	1497	215	0.874	
Backwards	700	4	0.994	
According-to	87	1	0.989	
Total	2284	220	0.912	0.938

Figure 4: Breakdown of system results (Training corpus)

	+	−	Prec	Rec
Basic	193	33	0.854	
Backwards	90	0	1.000	
According-to	4	0	1.000	
Total	286	33	0.897	0.994

Figure 5: Breakdown of system results (Test corpus)

5 Related Work

(Soricut & Marcu 03) describe a *Discourse Parser* – a system that uses Penn Treebank syntax to identify intra-sentential discourse relations in the RST Treebank. Since this applies to *all* intra-sentential relations in the RST Treebank, while our system is limited to Attribution, the systems are not directly comparable. Still, the results and discussion from (Soricut & Marcu 03) provide some useful perspective on our results.

(Soricut & Marcu 03) evaluate their Discourse Parser under a variety of scenarios; the most favorable has human-corrected syntax trees and discourse segmentation. In this scenario, the system achieves an f-score of .703 with the full set of 110 Relation Labels, and 75.5 with the relation label set collapsed to 18 labels. (Soricut & Marcu 03) note that human annotator agreement receives comparable f-scores, of .719 and .77 respectively. In the light of these numbers, our Attribution system f-score of .92 is quite impressive. This provides some measure of support for our hypothesis that the intra-sentential relations in the RST Treebank are in fact properly viewed as alternative notations for syntactic information that is already present in the Penn Treebank.

Of course, it may well be that some of the other intra-sentential relations present much greater difficulties than Attribution. But these results suggest that it is worth pursuing our project of attempting to automatically derive the intra-sentential RST Treebank relations from specific syntactic features.

6 Conclusion and Future Work

We have shown that Attribution relations can be identified successfully by using the syntactic structure of the Penn Treebank. In a sense, then, notating Attribution relations in syntactically parsed texts adds no information. Our hypothesis is that all intra-sentential relations in the RST Treebank are of this character.

This is important for several reasons. First, it is clear that the relations *across* sentences in the RST Treebank are not directly derivable from syntax, at least not in any obvious way. Our approach to identifying Attributions is a direct implementation of the description in the RST Treebank tagging manual. For inter-sentential relations such as CONTRAST or EXPLANATION-EVIDENCE, the situation is quite different. Syntactic criteria are relevant, but clearly not decisive, as can be observed in (Marcu & Echihabi 02). Finally, the elimination of intra-sentential relations like Attribution would appear to be more in line with the original vision behind RST; for example, according to (Mann & Thompson 88), the basic unit for RST relations is the clause.

References

(Carlson & Marcu 01) Lynn Carlson and Daniel Marcu. Discourse tagging manual. ISI Tech Report ISI-TR-545, 2001.

(Carlson *et al.* 02) Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In Jan van Kuppevelt and Ronnie Smith, editors, *Current Directions in Discourse and Dialogue*. Kluwer Academic Publishers, 2002.

(Mann & Thompson 88) William Mann and Sandra Thompson. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281, 1988.

(Marcu & Echihabi 02) Daniel Marcu and Abdessamad Echihabi. An unsupervised approach to recognizing discourse relations. In *Proceedings, 40th Annual Meeting of the Association for Computational Linguistics (ACL-02)*, Philadelphia, PA, 2002.

(Marcu *et al.* 99) Daniel Marcu, Magdalena Romera, and Estibaliz Amorrortu. Experiments in constructing a corpus of discourse trees: Problems, annotation choices, issues. In *Proceedings of the Workshop on Levels of Representation in Discourse*, pages 71–78, Edinburgh, Scotland, 1999.

(Marcus *et al.* 93) Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2), 1993.

(Soricut & Marcu 03) Radu Soricut and Daniel Marcu. Sentence level discourse parsing using syntactic and lexical information. In *Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference (HLT/NAACL)*, Edmonton, Canada, 2003.

Feature Selection for Electronic Negotiation Texts

Marina Sokolova, Vivi Nastase, Mohak Shah and **Stan Szpakowicz**
School of Information Technology and Engineering,
University of Ottawa,
Ottawa ON, K1N 6N5, Canada,
{sokolova,vnastase,mshah,szpak}@site.uottawa.ca

Abstract

Various feature selection and data representation methods have been proposed for text data collected from electronic negotiations. We compare two broad classes: process-based and corpus-based feature selection. In particular, we study the informativeness and representativeness of each method from these classes with respect to the classification of outcomes of electronic negotiations. Our empirical results are a quantitative basis for our analysis.

1 Introduction

Texts exchanged in electronic negotiations (e-negotiations) contain signals that may indicate the successful or unsuccessful outcome. In order to extract such signals, it is essential to find an effective feature selection method and a suitable data representation to enable learning in this environment. Various methods to address this issue, with various biases, have been proposed. In this paper, we introduce two broad classes of such methods, with important commonalities. We further analyze the methods in each class, looking for those that result in an optimum feature subset and data representation for texts coming from e-negotiations. We focus on identifying the learning settings that better assist the prediction of negotiation outcomes. The important components of such settings are features, their representation and the learning paradigm. The quality of the classification of the negotiation outcomes is one of the evaluation measures. Note that although we reduce the classification of negotiation outcomes to the classification of negotiation texts, our procedure differs from standard text classification. For an overview of machine learning methods and their application to text classification, including different types of features refer to (Sebastiani, 2002).

The first class that we consider contains the methods that exploit the knowledge of the negotiation process and the strategies employed when two parties negotiate. The former, based on the identification of negotiation-related words, was introduced in (Shah et al, 2004). The latter, using strategy-related features, was introduced in (Sokolova and Szpakowicz, 2005). The data representation based on negotiation-related features benefits from the knowledge of the negotiation. On the other hand, the strategy-related data representation relies on more general knowledge of the influence strategies that negotiators employ to reach a beneficial agreement. However, both these methods rely on the knowledge of the *process* of negotiation, though at different levels. Hence, we place them together under the umbrella of *process-based* data representation.

The second class that we discuss here contains methods that identify a representative subset of features by considering the statistical characteristics of the data under investigation. One such method, quite popular, represents data with the most frequent n-grams; often it is a unigram representation ($n = 1$). We also introduce an approach that relies on features whose frequency behaviour varies between data classes. Those are features that occur more frequently in one class than in the other (for example, in successful rather than unsuccessful negotiations). All these methods work with corpus statistics; we name them collectively *corpus-based* data representation.

Having defined the classes of data representation and feature selection methods for e-negotiations, we continue our analysis to address two issues:

- which set of features is better suited to represent e-negotiation texts so as to classify them on negotiation outcomes;

- which representation gives better insights into the negotiations themselves.

We employ various learning paradigms to examine the behaviour and usefulness of each representation.

In addition, we also examine whether the presence of selected features is important or the frequency of occurrence matters equally to each candidate feature selection method. Finally, we show that the *process-based* approach fares better in terms of the classification accuracy than the *corpus-based* approach. The correct identification of successful and unsuccessful negotiations increases when the feature sets result from process-based approaches. Pinpointing the most representative features should help predict the negotiation outcome better during the negotiation itself, and warn the negotiators when their language use may lead to a failure.

The insights gained will be useful in studying and extracting knowledge about specific negotiation problems such as strategies, tactics, negotiation moves and ways in which negotiation partners exert influence on each other and in identifying the appropriate feature sets for such tasks.

The remainder of the paper is organized as follows: Section 2 introduces the environment of e-negotiations and the specifics of the e-negotiation data. Section 3 describes the feature selection methods that we investigate; they all come from the two broad classes discussed earlier in the paper. Section 4 discusses in detail the experimental setting and reports the classification results. Section 5 presents an analysis of, and insights into, the behaviour and usefulness of various methods in the light of our experiments. Finally, Section 6 highlights the main findings and future directions.

2 E-negotiation Data

E-negotiations occur in various domains (for example, labour or business) and involve various users (for example, negotiators or facilitators). As in traditional negotiations, e-negotiation participants have established goals and exhibit strategic behaviour (Brett, 2001). The negotiation outcome (success or failure) results from these strategic choices. E-negotiations held by humans, however, share the uncertainty intrinsic to any human behaviour.

Text messages exchanged in e-negotiations reflect the negotiation traits and trends; Figure 1 shows an example from the beginning of a negotiation (Kersten et al, 2002). (Kersten and Zhang, 2003) used the

(Buyer) *Hi Joe, I'm Lisa and I represent Cypress Cycles in this negotiation. After extensive deliberation we have prepared an offer to purchase sprockets and gear assemblies. We think it is a fairly good offer and hope you find it acceptable.*
(Seller)*Hi Lisa, I am Joe, the representative of Itex Manufacturing and I am very delighted to get in touch with you. First of all, thank you very much for the possibility to negotiate with you and your company. Despite your really interesting offer, it is not possible for me and my company to accept it under all circumstances. For that reason I would like to make the following proposal to you. I am very interested in what you are thinking about, so I am looking forward to hearing from you. Bye, Joe.*

Figure 1: A sample of e-negotiation

history records of e-negotiations to study how the negotiation outcome depends on the intensity and distribution of offers exchanged during negotiation. However, such records and statistics might not be available in practice (esp. when, say for instance, the negotiation is not held via a negotiation support system). In such cases, the text used by the negotiators in their message-exchanges can prove to be useful. We examine this realm and hence work with the transcripts of the Inspire negotiations.

The *Inspire* text data (Kersten et al, 2002) is the largest text collection gathered through e-negotiations (held between people who learn to negotiate and may exchange short free-form messages). Negotiation between a buyer and a seller is successful if the virtual purchase has occurred within the designated time, and is unsuccessful otherwise. The system registers the outcome. We use the transcripts of 2557 negotiations, 1427 of them successful. We consider a transcript as a single example, with all messages concatenated chronologically, preserving the original punctuation and spelling. A successful negotiations is a positive example, an unsuccessful negotiation – a negative example. The *Inspire* data contain 27,055 word types

which constitute the initial feature set. That is, we apply feature selection to the data that contain 2557 examples and 27,055 features.

3 Feature-Selection Approaches

We want to compare two broad classes of feature selection methods and the feature subsets that these methods produce. As an evaluation criterion we use the results of the learning of classifiers on data represented using each of these feature subsets with respect to the outcome of negotiations.

We consider two *process-based* feature selection methods, negotiation-related and strategy-related, and two *corpus-based* methods, which represent the data with the most frequent unigrams and with *indicative* words. There is a major difference between the methods of the two classes. The former relies on expert knowledge about the domain from which the data originate. The latter requires feature scoring based purely on the statistical properties of the data. There is another difference: the extent of automation. *Process-based* approaches are inevitably semi-automatic, unlike the fully automatic *corpus-based* approaches that do not require integrating any expert knowledge.

3.1 Process-based Approaches

This type of feature selection is based on two different criteria. The *negotiation-related* feature selection identifies features specific to the process of negotiations. We can also build on the knowledge of influence-strategies that the negotiators employ. The features thus identified are called *strategy-related* feature selection.

Negotiation-related features (Shah et al, 2004) include words with specific negotiation-related meanings. Such words have been found to be unusually frequent compared to the typical word distribution in standard corpora. Selection of the negotiation-related features is based on the idea of identifying the elements of the communication model(Hargie and Dickson, 2002) of negotiations and works as follows:

- Consider the key elements of negotiations and identify these elements for the specific negotiations. Examples of such elements include: Environment (in the *Inspire* data – business), Goal (reaching an agreement), Topic (the purchase of good), Social roles within negotiations (buyers and sellers) and outside negotiations (students).
- Build the N-gram models from the data for N = 1, 2, 3.
- Identify semantic categories for the elements of negotiations; for example, the categories "hobbies" and "studies" can be identified for the social roles outside negotiations, the category "negotiation-specific" – for the goal, topic and environment.
- With respect to these categories, disambiguate each word – if necessary – using the most frequent bigrams and trigrams in which it appears.
- Build a semantic lexicon from the text data. Tag each word type[1] with one or more semantic category, using a lexical resource with semantic information (a machine-tractable form of (Summers, 2003)). In case of multiple candidate tags, select the one that corresponds to the most frequent sense of the word.
- Select the words tagged as negotiation-specific.

Strategy-related feature selection approach is based on the influence strategies most commonly used in negotiations. We present the general framework; see (Sokolova and Szpakowicz, 2005) for the details of the theoretical background and the implementation. To deliver the strategies, negotiators use appeal, logical necessity, and the indicators of intentions towards the subject of the negotiations and the negotiation process. In language, these strategic tools are exhibited in persuasion, substantiation, exchanges of offers, agreement and refusal (Brett, 2001); they reflect the reasoning, opinions and emotions of the participants. They are signalled by pronouns, negations, modal verbs, mental verbs, volition verbs and adjectives. Selection of the strategic features works as follows.

- Identify the influence strategies used in negotiations. *Direct* strategies are used when a negotiator directly influences the counterpart to

[1] A word type represents all occurrences of the same string in a text.

Negotiation-related features	
Word categories	Word types
nouns	offer, price, delivery...
action verbs	reduce, return, prepare...
volition verbs	agree, accept, refuse...
adjectives	recent, unacceptable...
mental verbs	think, know...

Table 1: Examples of negotiation-related features.

Strategy-related features	
Word categories	Word types
personal pronouns	I, we, you...
negations	no, none, nothing,...
modal verbs	can, will, should...
volition verbs	accept, promise, refuse...
adjectives	next, last, final, ...
mental verbs	think, understand, consider...

Table 2: Examples of strategy-related features.

make desirable concessions, *indirect* strategies – when attempts to influence the counterpart are not explicit.

- Represent influence strategies with the expression of persuasion, argumentation, substantiation, rejection and denial, and so on.

- Find a mapping between the word categories and the categories representing these strategies: negations are mapped to rejection and denial, modal verbs – to argumentation, mental verbs are associated with the intention towards the process of negotiations, and so on.

- Build the list of word categories including modals, volition verbs, negations, mental verbs, superlative adjectives. Finally, automatically extract from the data the words belonging to these categories.

Tables 1 and 2 give examples of negotiation-related and strategy-related features for the *Inspire* data[2].

3.2 Corpus-based Approaches

We evaluate the effectiveness of automatic corpus-based feature selection on two approaches. First, we use 200 most frequent unigrams counted in the e-negotiation corpora (one built from the data of successful negotiations, the other from the data of unsuccessful negotiations). These unigrams are chosen so that their frequencies are approximately the same in both successful and unsuccessful negotiations. With this set of features, we want to investigate if the features most frequently used in *both the negotiation classes* assist in binary classification. As opposed to most frequent words, *indicative* words are the unigrams whose frequency differs considerably in successful and unsuccessful negotiations. To identify these words we separate the data into two sets – successful and unsuccessful negotiations – and calculate the log-likelihood statistics LL for each word w (Rayson and Garside, 2000).

$$LL(w) = 2 * \left(\left(a * log\left(\frac{a*(a+b)}{c}\right)\right) + \left(b * log\left(\frac{b*(a+b)}{d}\right)\right)\right)$$

where a and c are the number of occurrences of w and the number of word tokens respectively, in the first corpus; b and d, in the second corpus. The higher the $LL(w)$, the larger the difference between frequencies of the word w in the two corpora.

3.3 The Datasets

For sets of features selected by each of the approaches described in subsections 3.1 and 3.2, we form bags of features from their unigrams. In each case, we build two datasets:

1. with the numerical attributes whose values are the numbers of occurrences of the word in negotiation; in this case we add one more attribute, whose value is the number of occurrences of other unigrams in the negotiation[3];

2. with the binary attributes showing whether the feature appears in the negotiation; there is no additional attribute.

4 Empirical Results

We have introduced several feature selection methods for e-negotiation. Now, we evaluate them using three learning paradigms. Paradigms with different

[2]The lists of negotiation-related features and strategic features intersect on seven features.

[3]To show that this attribute is relevant to the outcomes, we filter the attributes with Weka-based filters (Witten and Frank, 2000); this always selects the additional attribute as relevant.

learning biases give us an insight into the consistency of the results across them. We use C5.0, a version of C4.5 (Quinlan, 1993), a decision-tree learner that classifies entries by separating them into classes according to information gain of the attributes. Kernel methods, especially Support Vector Machines (SVM) (Cristianini and Shawe-Taylor, 2000), have been successfully used for text classification. They are also resistant to noise and work well on data with arbitrary distributions. We apply a linear kernel SVM. We also apply the probabilistic Naive Bayes classifier (NB) (Duda et al., 2000). NB was used with kernel density estimation and with the normal distribution estimation to model the numerical values (Witten and Frank, 2000). NB with kernel density estimation has shown better accuracy. We therefore report results only for NB with kernel density approximation.

We present tenfold cross-validation estimates of accuracy. To find out how the classifiers work on individual data classes, we use the standard text classification metrics: precision (P), recall (R) and equally-weighted F-measure. We have performed an exhaustive search on the adjustable parameters for every method. The classifiers were run on both sets of features: numerical, with the attribute values taking into account the frequency of occurrence of the selected set of features for each method; binary, with attribute values 0 for the absence and 1 for presence of the selected feature. Because of the identical performance of all classifiers on the sets of the most frequent and indicative features, we exclude the latter from the binary experiments.

Tables 3, 4 and 5 report the highest accuracy and corresponding P, R, F achieved by each classifier on every feature set and feature representation. For both numerical and binary representations, we *italicize* the highest accuracy for each classifier and put in ***bold*** the highest accuracy among them. The highest precision and recall are shown in **bold**. In our experiments, the baseline accuracy and precision are 55.8%, recall is 100%, and F-measure is 71.6% when we classify all negotiations as successful.

We do not present statistical significance because our results do not give enough material for a thorough *ANOVA* test for differences among groups; *ANOVA* would be the best method of exploring the difference in performance of combinations of the data features, their representation, and a classifier. *t-test*, used for a pair-wise comparison, is clearly not a suitable candidate. Additionally, Tables 4 and 5 show that the process-based features give the highest precision and recall for both numerical and binary representations. In the next section we explain how the process-based data representations affect the classification of positive and negative examples, that is, successful and unsuccessful negotiations.

5 The Informativeness of the Feature Sets

The features selected by the process-based approaches give higher classification accuracy than the features selected by the corpus-based approaches, but the two feature selection methods differ in what characteristics they extract from the data.

- The negotiation-related feature set is *specific* to negotiation; it captures the negotiators' main goal with respect to the negotiation issues, preferences and scope (width, depth, generality, specificity), and the numerical representation features reveal the intensity of the discussion of negotiation issues.

- The strategy-related feature set is *generic* in the sense that it does not relate specifically to negotiation issues; it rather captures the intentions to continue a negotiation, the influence on the partner, self-obligations and motivations, openness to feedback or the opposite, the boundaries within personal communication, and so on.

Negotiation-related and strategy-related features, although process-based, represent different aspects of the same process and therefore vary in their informative capacity. These differences allow learning of negotiation outcomes from various perspectives. Figures 2 and 3 report the true positive and true negative rates corresponding to the accuracies reported above. The results show that the negotiation-related features give higher accuracy in correct identification of positive examples and lead to the following explanation:
- the positive class either is homogenous or consists of a few well-represented subclasses;
- the negative class is divided into several small subclasses, and some of these subclasses are underrepresented.

Features	attr	NB	SVM	C5.0	attr	NB	SVM	C5.0
negotiation-related	num	*69.3*	71.7	***75.4***	bin	69.4	*74.0*	***74.8***
strategy-related	num	65.3	71.3	74.5	bin	*71.1*	72.7	73.7
most frequent	num	64.3	*73.4*	71.5	bin	64.2	71.5	73.3
indicative	num	64.2	72	74.4	bin	n/a	n/a	n/a

Table 3: Classification accuracy.

Features	# of attr	NB			SVM			C5.0		
		P	R	F	P	R	F	P	R	F
negotiation-related	124	72.3	72.5	72.5	72.5	75.8	74	73.3	**87.7**	79.9
strategy-related	100	74	58.3	56.7	**74.8**	73.2	74.0	72.5	87.6	79.3
most frequent	201	74.6	54.4	62.6	72.9	75.3	74.1	72.4	84.2	80.0
indicative	201	74.6	54.6	62.9	73.2	75.8	74.5	73.0	85.9	79

Table 4: Precision and recall; numerical representations.

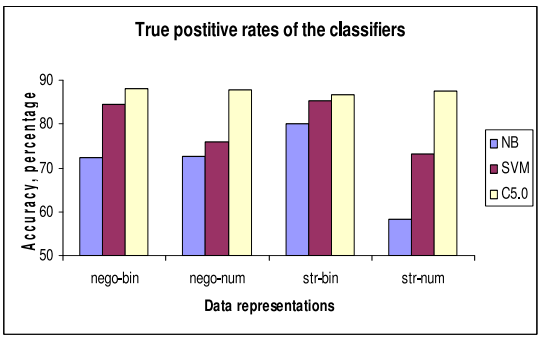


Figure 2: Classification of positive examples

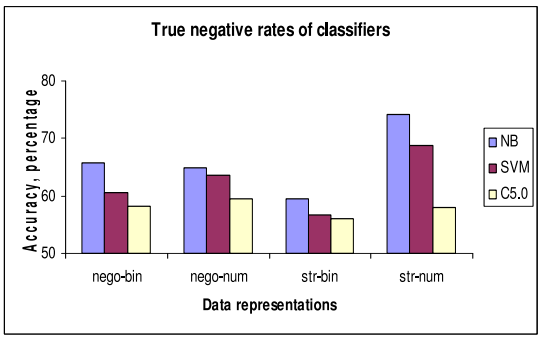


Figure 3: Classification of negative examples

This means that similarities among successful negotiations are easily revealed through the use of negotiation-related features and are strong enough to build a homogenous class, whereas for unsuccessful negotiations this assumption does not hold. The strategy-related features improve the classification accuracy by correctly identifying negative examples, especially when the binary representation is used. These features extract stronger similarities from the negative class than from the positive one. In the context of negotiations this suggests that discussing the topic of negotiation helps identify successful negotiations, while studying the implementation of influence-strategies helps identify unsuccessful negotiations.

We have shown that the two process-based approaches are complementary in the sense that they address different problems in learning from e-negotiation texts. It is natural to ask whether the benefits of both the sets of features can be exploited simultaneously. One possible direction of investigation would be to continue work with the features, either by constructing new ones, for example, building collocations of negotiation-related and strategy-related features, or suggesting an elaborate features selection method. Another opportunity to benefit from both sets of features comes from building an ensemble of classifiers, where the classifiers built by the same learner use different sets of features to classify the data and then combine their results. SVMs with the high accuracy and the most balanced performance on the data are the reasonable candidates.

6 Conclusion and future work

We have categorized, empirically compared and analyzed various feature selection and data represen-

Features	# of attr	NB			SVM			C5.0		
		P	R	F	P	R	F	P	R	F
negotiation-related	123	66.4	72.3	69.5	**73.1**	84.6	78.4	72.6	**88**	77.3
strategy-related	99	71.5	80.2	75.6	71.4	85.3	77.7	71.3	87.4	78.9
most frequent	200	74.6	54.6	63.1	72.9	75.3	74.2	72.3	84.2	77.8

Table 5: Precision and recall; binary representations.

tation methods for the text data collected during electronic negotiations. In particular, we compared two broad classes: the process-based and corpus-based feature selection methods. For each method from these two classes, we have studied their informativeness and representativeness with respect to the classification of the outcomes of e-negotiations. We have focused on the problem of identifying the learning settings that better assist the prediction of negotiation outcomes, where the settings include features, their representation and the learning paradigm. The classification of the negotiation outcomes was one of the evaluation measures.

We have shown empirically that the sets of features selected by the process-based approaches provide better classification of negotiation outcomes than the sets of features selected by the corpus-based approaches. We have confirmed this conclusion for NB, SVM and C5.0. Our analysis has shown that within the process-based feature selection approaches, the negotiation-related and strategy-related features complement each other on the classification of successful negotiations and unsuccessful negotiations. Thus, the features are good candidates for the future work on classification of the negotiation outcomes from texts.

The empirical results and their analysis should be helpful in work on knowledge-based electronic negotiation systems. We suggest the means of predicting the negotiation outcome and warning the negotiators when their language use may lead to the failure of negotiations.

Acknowledgments

This work has been partially supported by the Natural Sciences and Engineering Research Council of Canada and by the Social Sciences and Humanities Research Council of Canada.

References

Brett J. M. 2001. *Negotiating Globally*. Jossey-Bass.

Cristianini, N., J. Shawe-Taylor. 2000. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press.

Duda, R., P. Hart, D. Stork 2000. *Pattern Classification*. Wiley, 2nd ed..

Hargie, O., D. Dickson. 2004. *Skilled Interpersonal Communication: Research, Theory and Practice*. Routledge, 4th ed.

Kersten G. E. and others. 2002-2006 *Electronic negotiations, media and transactions for socio-economic interactions*. interneg.org/enegotiation/.

Kersten, G. E. and G. Zhang. 2003. Mining Inspire Data for the Determinants of Successful Internet Negotiations. *Central European Journal of Operational Research*. 11(3): 297–316.

Summers, D. (ed). 2003 *Longman Dictionary of Contemporary English*. Pearson Education: Longman. 4th ed.

Quinlan, J. R. *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, San Mateo, CA, 1993.

Rayson, P., R. Garside. 2000. Comparing Corpora using Frequency Profi ling. Proc Workshop on Comparing Corpora (ACL 2000), 1–6.

Sebastiani, F. 2002 "Machine Learning in Automate Text Categoriazation". *ACM Computing Surveys*. 1-47.

Shah, M., M. Sokolova, S. Szpakowicz. 2004 "The Role of Domain-Specifi c Knowledge in Classifying the Language of E-negotiations". *Natural Language Processing (Proc of ICON'2004)*. 99–108.

Sokolova, M., S. Szpakowicz. 2005 "Classifi cation and Strategy Analysis in Electronic Negotiations". *Advances in Artificial Intelligence (Proc of AI'2005)*. 145–157.

Witten, I., E. Frank. 2000. *Data Mining*. Morgan Kaufmann. www.cs.waikato.ac.nz/ml/weka/

Exploiting Parallel Texts to Produce a Multilingual Sense Tagged Corpus for Word Sense Disambiguation

Lucia Specia, Maria das Graças Volpe Nunes
ICMC – University of São Paulo
Av. do Trabalhador São-Carlense, 400
São Carlos, 13560-970, Brazil
{lspecia,gracan}@icmc.usp.br

Mark Stevenson
DCS – University of Sheffield
Regent Court, 211 Portobello Street
Sheffield, S1 4DP, UK
M.Stevenson@dcs.shef.ac.uk

Abstract

We describe an approach to the automatic creation of a sense tagged corpus intended to train a word sense disambiguation (WSD) system for English-Portuguese machine translation. The approach uses parallel corpora, translation dictionaries and a set of straightforward heuristics. In an evaluation with nine corpora containing 10 ambiguous verbs, the approach achieved an average precision of 94%, compared with 58% when a state of the art statistical alignment tool was used. The resulting corpus consists of 113,802 instances tagged with the senses (i.e., translations) of the 10 verbs. Besides the word-sense tags, this corpus provides other useful information, such as POS-tags, and can be readily used as input to supervised machine learning algorithms in order to build WSD models for machine translation.

1 Introduction

Word Sense Disambiguation (WSD) is concerned with the identification of the sense of an ambiguous word in a given context, that is, one among its possible meanings. For example, the noun *pen* has at least two unrelated meanings: *writing device* and *enclosure*. The verb *to run*, in turn, has at least two possible related meanings: *to move quickly* and *to go*.

Although WSD can be thought of as an independent task, its importance is more straightforwardly realized when it is used in an application, such as Information Retrieval or Machine Translation (MT) (Wilks & Stevenson, 1998). In MT, which is the focus of this paper, WSD can be used to identify the most appropriate translation for a source language word when the target language offers more than one option with different meanings, but the same part-of-speech. However, there is not always a direct relation between the number of possible senses and translations of a word; different senses of a word in the source language can be translated by the same target word, and a non-ambiguous source word can have two or more possible translations (Hutchins & Somers, 1992). In this context, thus, "sense" means, in fact, "translation". For example, assuming the translation from English to Portuguese, explored in this work, *bank* can be translated as *banco* (*financial institution* or *seat*) or *margem* (*land along the side of a river*). *Financial institution* and *land along the side of a river* are both senses of the English word *bank*, however, the *seat* sense is valid only in the translation.

Sense ambiguity has been recognized as one of the most important problems in MT (Bar-Hillel, 1960). Nowadays, despite the great advances in WSD, this problem is still considered a serious barrier to the progress in MT. The problem was recently investigated for English-Portuguese MT (Specia, 2005). The study showed that the current MT systems do not handle sense ambiguity appropriately and that this is one of the reasons for the unsatisfactory translations.

The various approaches to WSD are generally aimed at monolingual contexts. Recent approaches have focused on the use of corpus-based and machine learning techniques in order to avoid the massive effort required to codify linguistic knowledge. These approaches have shown good results, especially those using supervised learning (Edmonds & Cotton, 2001). However, supervised approaches are dependent on a sense tagged corpus. The lack or inadequacy of such corpora is one of the main drawbacks of those approaches.

For monolingual applications, there are some available sense tagged corpora, such as SemCor (Miller et al., 1994). However, for multilingual applications there are only few corpora for certain languages. For English-Portuguese, in particular, there are no available corpora. The creation of an expressive corpus would represent an important step towards achieving effective WSD between this pair of languages. Certainly, automating this process would avoid the effort required to carry out manual tagging.

Although a good strategy, the automatic creation of sense tagged corpora is still little explored. Some approaches aimed at the creation of English tagged sense corpora include the work of Agirre & Martínez (2004), who exploited Wordnet relations and monolingual corpora, and Diab & Resnik (2002), who made use of bilingual parallel corpora and word alignment methods. Dinh (2002) also explored bilingual parallel corpora and word alignment methods to create an English-Vietnamese sense tagged corpus.

Given the large amount of multilingual machine readable texts currently available, identifying the correspondent word pairs in the source and target languages of parallel corpora seems to be a very practical strategy to automatically create sense tagged data. Parallel corpora are also good knowledge sources to directly carry out the sense disambiguation, especially for MT purposes. In fact, parallel corpora have been explored in several ways for MT since (Brown et al., 1991). They have also been used for monolingual WSD (Dagan & Itai, 1994, Ide et al., 2002; Ng et al., 2003).

Most of these works rely on the existence of accurate word alignment methods. However, current word alignment methods do not present a satisfactory performance, when applied to English-Portuguese. Indeed, experiments with several alignment methods on English-Portuguese reported a precision of 57% and a recall of 61% for the best method (Caseli et al., 2004).

Considering these issues in the context of our ultimate goal of building a WSD system for English-Portuguese MT, we developed a hybrid approach, mixing linguistic and statistical knowledge, to automatically create a sense tagged corpus. The approach makes use of parallel corpora, bilingual dictionaries, and a set of simple heuristics. We experimented with nine parallel corpora containing 10 ambiguous verbs, and compared the results to those produced by the word alignment tool GIZA++ (Och & Ney, 2003).

In the remaining of this paper, we first present our approach, including its scope, the parallel corpora explored, and the sense tagging process (Section 2). We then present the evaluation of the approach, discussing its results (Sections 3 and 4), and conclude with some remarks and future work (Section 5).

2 The sense tagging approach

2.1 Scope

This work focuses on verbs; these represent difficult cases for WSD and, once disambiguated, can help to disambiguate other words in the sentence, especially their arguments. In this stage, we are dealing with seven frequent[1] and highly ambiguous verbs identified as very problematic to MT systems according to a previous study (Specia, 2005). We also consider other three frequent but not so ambiguous verbs. These three verbs were selected in order to analyze the effect of polysemy level on our method. The complete list of verbs, along with their number of possible translations[2], is given in Table 1.

Possible translations are single words, including synonyms, and phrasal verb usages. Phrasal verb senses are considered because the occurrence of a verb followed by a preposition / particle does not necessarily indicate a phrasal verb. Multiword translations are not considered for these experiments and will be tackled in future work. The average number of translations for the seven highly ambiguous verbs (*come*, *get*, *give*, *go look*, *make* and *take*) is 203. The average for the three other verbs (*ask*, *live* and *tell*) is 19.

Verb	# translations	Verb	# translations
come	226	make	239
get	242	take	331
give	128	ask	16
go	197	live	15
look	63	tell	28

Table 1: Verbs and its possible translations

2.2 Parallel corpora

The original untagged corpus, consisting of English sentences containing the 10 verbs along with their manually translated Portuguese sentences, was collected from nine sources, including a mixture of genres and domains, as shown in Table 2. *Europarl* (Koehn, 2002) comprises bilingual versions of the European Parliament texts. *Compara* (Frankenberg-Garcia & Santos, 2003) comprises fiction books. *Messages* contains input / output messages used by Linux software[3]. *Bible* contains versions of the Christian Bible. *Red Badge* is the novel The Red Badge of Courage, by Stephen Crane. *PHP* consists of the user manual to the PHP programming language[4]. *ALCA* comprises bilingual versions of documents from Free Trade Area of the America[5]. *NYT* comprises some on-line daily news of the New York Times newspaper[6]. Finally, *CPR* consists of 65 abstracts of Computer Science thesis from the University of São Paulo.

All these corpora were already sentence aligned. Sentences in a many-to-one or one-to-many relationship with sentences in the translation were grouped together to form a "unit". So, the number of units is the same for both languages. Using specific concordancers, we selected the sentences from these corpora containing one of the 10 verbs. The number of resulting units (in one language), and English (E) and Portuguese (P) words for each corpus are illustrated in Table 2.

Corpus	# units	# E words	# P words
Europarl	167,339	6,193,904	6,299,686
Compara	19,706	518,710	475,679
Messages	16,844	385,539	394,095
Bible	15,189	474,459	443,349
Red Badge	823	15,172	12,555
PHP	226	7,964	6,342
ALCA	191	7,478	7,386
NYT	47	1,585	1,575
CPR	41	1,339	1,381
Total	**220,406**	**7,606,150**	**7,642,048**

Table 2: Numbers of sentences and words

The proportion of units for each verb varies from corpus to corpus. The smallest corpora did not contain any occur-

[1] According to the frequency list of the British National Corpus (Burnard, 2000).

[2] According to the DIC Prático Michaelis® machine readable English-Portuguese dictionary, version 5.1.

[3] www.gnome.org

[4] www.php.net/download-docs.php

[5] www.ftaa-alca.org/alca_p.as

[6] www.nytimes.com

rences of some verbs.

2.3 Pre-processing

Some pre-processing steps were carried out to filter units and to transform the corpus into an adequate format:

1. English units were lemmatized using the Minipar parser (Lin, 1993).
2. Unit pairs containing English idioms involving one of the 10 verbs were eliminated.
3. POS tag the units in both languages, using the Mxpost tagger (Ratnaparkhi, 1996).
4. Portuguese verbs and verbal expressions were lemmatized (Feltrim, 2004).
5. Pairs of units for which the English verb under consideration has no valid verb tag in the English unit were eliminated; likewise, when the Portuguese unit has no word with a verb tag.

Units containing idioms were eliminated to avoid tagging errors, since idiom translations are usually non-literal. For that, we created a list of idioms containing the verbs based on the on-line version of the Cambridge Dictionary of Idioms[7].

The filter of the fifth step intended to isolate cases referring to tagger and concordancer problems, as well as to avoid errors due to modified translations, that is, when the verb in the English unit was paraphrased by words other than verbs.

The units from each of the corpora were handled separately, since we intend to analyze the genre / domain influence in our WSD model. The outputs of the pre-processing steps are English and Portuguese filtered units, being all words POS tagged, and English words and Portuguese verbs lemmatized. The total number of sentences was 206,913.

2.4 Sense identification

In order to identify the translation of each verb occurrence, the following assumptions were made:

- Given a sentence aligned parallel corpus, the translation of the verb in an English unit can be found in its corresponding Portuguese unit.
- Every English verb has a pre-defined set of possible translations, including those referring to phrasal verbs, and this set can be extracted from bilingual dictionaries.
- Phrasal verbs have specific translations; so, if a verb occurs in such constructions, the translations of the complete construction should be considered first. Some verb plus particle / preposition constructions may also be used as non-phrasal verbs. In this case, the translations of the verb itself should be also considered.
- Translations have different probabilities of being used in a given corpus, and these probabilities can be identified through a statistical co-occurrence analysis of the corpus.
- If there are two or more possible translations for an English verb, the more similar to the position of the English verb is the position of the translation in its respective unit, the more likely it is the correct one.

Based on these assumptions, a sense tagging process was created, relying in the following resources and heuristics.

2.4.1 Resources

To define the set of possible single-word translations for each verb, we used machine readable versions of bilingual dictionaries. We used the same dictionaries to identify a list of phrasal verbs and their translations. We consulted the on-line version of the Cambridge Dictionary of Phrasal Verbs[8] in order to create lists of separable and inseparable phrasal verbs, that is, phrasal verbs that can and can not have words between the verb and the particle. We consulted occurrences of each construction in the British National Corpus to elaborate a list of verbs plus particles / prepositions that can be used both as phrasal verb and as non-phrasal verb.

The NATools package (Simões & Almeida, 2003) was used to produce a list of translation probabilities. NATools uses statistical techniques to create bilingual dictionaries from sentence aligned parallel corpora. It generates bidirectional lists of at most 20 possible translations for all the words in the parallel corpus, along with their probabilities. Although the tool does not make use of any language-dependent resource, we pre-processed the parallel corpora in order to improve the produced dictionaries. Processing the units for all verbs in a given corpus together, we performed the following steps:

1. POS tag units in both languages.
2. Lemmatize English (Lin, 1993) and Portuguese verbs (Feltrim, 2004).
3. Eliminate the unit pairs containing idioms in the English version, using the list of idioms previously mentioned.
4. Remove stop words, punctuation, and other symbols from units in both languages.

In Table 3 we illustrate the list of translation probabilities produced by NATools for the verb *to give*, in the Compara corpus.

Translation	Prob.	Translation	Prob.
ceder_v	0.0117	**lançar_v**	0.0131
devolver_v	0.0053	pergunta	0.0063
\(null\)	0.1520	**entregar_v**	0.0252
renunciar_v	0.0055	provocar_v	0.0077
desistir_v	0.0225	fazer_v	0.0309
soltar_v	0.0060	**dar_v**	0.5783
deixar_v	0.0065	ser_v	0.0230
receber_v	0.0079		

Table 3: Translation probabilities for *to give*

In general, the lists produced contain mostly verbs appropriate as translations (bold face in Table 3), but also some

[7] http://dictionary.cambridge.org/default.asp?dict=I

[8] http://dictionary.cambridge.org/default.asp?dict=P

verbs that are not possible translation according to our dictionary (other words with a _*v* tag), words with other POS, and a null translation probability, that is, the probability of the verb not being translated. Since we assume that at least one possible translation of the verb is in the Portuguese unit, we normalized the resulting list to eliminate the null translation probability.

The lists produced do not include all the possible translation belonging to our dictionaries, because many of them may not occur in the corpus, or may occur with a very low frequency. For those translations, we assigned a zero probability.

Since the probabilities vary from corpus to corpus, the translation probabilities were generated individually for each corpus.

2.4.2 Heuristics

Given the assumptions and the resources created, we defined a set of heuristics to find, in the Portuguese unit (PU), the most adequate translation for each occurrence of the verb in an English unit (EU). The general procedure is shown in Figure 1. In detail, the heuristics comprises the following steps:

1. Identify inseparable phrasal verbs in the EU, annotating the unit when they occur. We compare the lemmas of the words tagged as verbs and the following 1-5 words to the list of inseparable phrasal verbs.
2. Identify, in the remaining EUs, separable phrasal verb, annotating the unit when they occur. Again, we compare the lemmas of the words tagged as verbs and the following 1-8 words to our list of separable phrasal verbs, allowing 2-3 words between the verb and the particle. We assume the remaining EUs do not contain any phrasal verb.
3. Identify the absolute positions of the verb / phrasal verb in the EU, ignoring punctuation signals and other symbols.
4. In the verb lemmas of the PU, search for all possible translations of the verb, consulting specific dictionaries for inseparable, separable, or non-phrasal verbs. Three possible situations arise:
    a. No translation is found – go to step 5.
    b. Only one translation is found – go to step 6.
    c. Two or more translations are found – go to step 7.
5. If the occurrence is a non-phrasal verb, finalize the process, considering that no adequate translation was found. Otherwise, first verify if the verb plus particle / preposition can be used as non-phrasal verb. If yes, go back to the step 4, now looking for possible translations of the verb in the non-phrasal dictionary. If it can not be used as a non-phrasal verb, finalize the process, considering that no adequate translation was found.
6. Select the only possible translation and use it to annotate the EU.
7. Identify the absolute positions of each translation in the PU and assign a position weight (PosW) to the translation, penalizing translations in distant positions from the position of the EU verb, according to the following:

$$PosW = 1 - \left(\frac{|EUposition - PUposition|}{10} \right)$$

8. Verify the translation probability for each of the possible translation, calculating the final translation weight (TraW) as follows:

$$TraW = PosW + probability$$

9. Choose the translation with the highest weight (TraW) to annotate the EU.

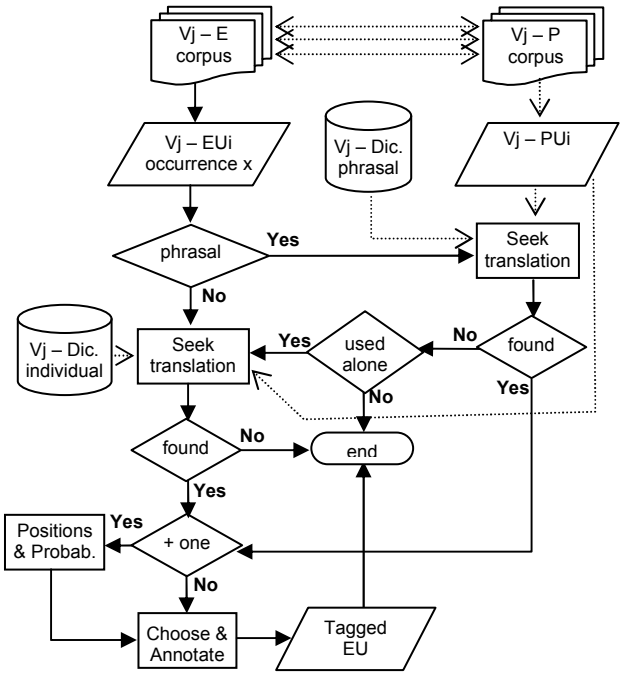


Figure 1: Sense identification process

The position plus probability weighting schema adopted in the case of more than one possible translation was empirically defined after experimenting with different schemas. As an example of its use, consider the pair of sentences shown in Figure 2, for *to come* (EU position = 7). The system correctly identifies the translation as *vir*, the lemma of *vindo* (PU position = 9, PosW = 0.8, probability = 0.432, TraW = 1.232), although there are two more possible translations in the sentence, according to our list of possible translations: *sair* (PU position = 2, PosW = 0.5, probability = 0.053, TraW = 0.553) and *ir* (lemma of *for*) (PU position = 6, PosW = 0.9, probability = 0.04, TraW = 0.94). If we had considered only the position of the words, without the weighting schema, the system would have chosen the wrong translation: *ir*.

"I'd rather leave without whatever I **came** for."
"Prefiro sair sem o que for que tenha **vindo** buscar."

Figure 2: Example of parallel sentences

It is worth noticing that the word position plays the most important role in this example. The probabilities generally

take effect when the possible translations are close to each other.

3 Evaluation and discussion

Our approach determined a translation for 55% of all verbs (113,802 units) in the nine corpora (Table 2). Similar identification percentages were observed among verbs and corpora. The lack of identification for the remaining occurrences was due to three main reasons: (a) we do not consider multi-word translations; (b) errors from the tools used in the pre-processing steps, especially POS tagging errors; and (c) modified translations, including cases of omission and addition of words.

Although the coverage of our approach in automatically tagging a corpus can be considered low, it is important to mention that we give preference to the precision of the sense tagging to the detriment of wide coverage. Our intention is to use this corpus to train a WSD model and we therefore require data to be as accurate as possible.

In order to estimate the precision of the sense tagging process, we randomly selected 30 tagged EU from each corpus, for each verb, including units without phrasal verbs and with both kinds of phrasal verbs. We grouped the five smallest corpora (Miscellaneous) for this evaluation. The total number of evaluated units was 1,500. The precision for each corpus and verb is shown in Table 4.

Verb	Europarl	Compara	Messages	Bible	Misc.
come	80%	84%	95%	90%	91%
get	93%	87%	100%	95%	82%
give	97%	95%	95%	97%	93%
go	90%	90%	95%	85%	95%
look	100%	98%	95%	90%	100%
make	87%	86%	100%	93%	97%
take	80%	88%	91%	90%	93%
ask	100%	98%	100%	100%	100%
live	100%	100%	100%	100%	100%
tell	100%	94%	100%	100%	96%
Ave.	**93%**	**92%**	**97%**	**94%**	**95%**

Table 4: Precision of the sense tagging process

On average, our approach was able to identify the correct senses of 94.2% of the analyzed units. It achieved a very high average precision (99.2%) for the less ambiguous verbs (the three last in Table 4). Of the seven highly ambiguous verbs, *to look* and *to give* have lower numbers of possible senses than the rest, and for them the system also achieved a very high average precision (96%). For the remaining five verbs, the system achieved an average precision of 90.3%. Therefore, although there is no direct relation between the number of senses and the precision, the precision was generally lower for the most ambiguous verbs.

The tagging errors were consequences of the problems mentioned above, regarding the coverage of the system, but were also due to limitations of our heuristics. The distribution of the errors sources for each corpus is shown in Table 5.

Corpus	Idiom / slang	Modified translation	Tagger error	Heuristics
Europarl	6%	66%	8%	20%
Compara	8%	71%	0%	21%
Messages	0%	100%	0%	0%
Bible	6%	74%	10%	10%
Mics.	10%	69%	16%	5%

Table 5: Tagging error sources

Most of the errors were due to modified translations, including omissions and paraphrases (such as active voice sentences being translated by different verbs in a passive voice). In fact, with exception of the technical corpora (*Messages* and *PHP*), the translations were far from literal. In those cases, as in the case of idioms or slang usages, the actual translation was not in the sentence, or was written using words that were not in the dictionary, but the system found other possible translation, corresponding to other verb. Tagger errors refer to the incorrect tagging of the verbs with any other POS. In this case, the system also pointed out other possible translations in the PU. Errors due to the choices made by our heuristics are also related to the other mentioned errors. For example, considering the position of the words as the main evidence can be an inappropriate strategy when translations are modified by the inclusion or omission of words.

It is important to remember that some units are very long (for example, 180 words), containing many possible translations. In fact, an EU can have many verbs and the words used to translate other verbs may also be translations of the verb under consideration. The sentence alignment certainly reduced the number of possible translations, however, even after that process, the average number of possible translations in a PU, in all corpora and for all verbs, was 1.5. If we consider only the seven most ambiguous verbs, the average was 2.4 (from 1 to 15 possible translations in a PU).

4 Comparison with an alternative approach

We compared the precision of the system to the precision of the GIZA++ word alignment package (Och & Ney, 2003). Every pre-processed corpus was individually submitted to GIZA++ (the five smallest corpora were grouped in order to provide enough data for the statistical processing). We then analyzed the alignment produced for the verbs using the same sentences used to evaluate our system. The average precision for each corpus is shown in Table 6.

We considered as correct alignments all those including the verb translation, even if they were not one-to-one, that is, if they included other words. As shown in Table 6, the precision of the alignment produced by GIZA++ is considerably lower than the precision of our system. Unsurprisingly, the difference between the performances of the two approaches is statistically significant ($p < 0.05$, Wilcoxon Signed Ranks Test). Since statistical evidence is the only information used by GIZA++, it was not successful in identifying non-frequent translations. More-

over, it rarely found the correct alignment in the case of modified translations.

Corpus	Precision
Europarl	51%
Compara	61%
Messages	70%
Bible	42%
Miscellaneous	66%

Table 6: Precision of the GIZA++ word alignment

It is important to note that in this analysis we considered only the cases for which our system had proposed a possible translation. As previously mentioned, filters were used to avoid tagging errors. In order to find out GIZA++ outputs for those cases that were not tagged by our system, we analyzed 10 cases, for every verb and corpus, amounting to 500 parallel units. In average (all verbs and corpora), only 1% of these non-tagged units corresponded to GIZA++ null alignments for the verb. In 29% of the cases GIZA++ produced a correct alignment; while in 70%, the alignment pointed was incorrect. Although we analyzed the pre-processed corpora, again, in most of the cases, the incorrect GIZA++ alignments were due to modified translations. In those cases, the actual translation was not in the sentence, but the alignment system indicated a non-null alignment, since it does not include any linguistic knowledge about possible translations.

This comparison shows that the precision of our approach is, indeed, superior to those of the most relevant current word-alignment methods. It also shows that the use of the dictionaries avoided many tagging errors. Moreover, though our approach uses statistical information as one of the clues during the tagging process, it will still work if that information is not available. As a consequence, the performance for very small corpora will not be severely affected. So, we believe that the precision achieved by our system is satisfactory and that the resulting instances are thus appropriate to be used as a training corpus to produce WSD models.

5 Conclusion

We presented an approach to create a sense tagged corpus aimed at MT, based on parallel corpora, linguistic knowledge and statistical evidence. The results of an evaluation using a subset of nine parallel corpora and 10 verbs showed that the approach is effective, achieving an average precision of 94%. Most of the tagging errors were related to characteristics of the corpora: non-literal translations and use of language constructions that are very difficult to process automatically (idioms, e.g.). Nevertheless, the use of filters and elaborated heuristics avoided many errors, reducing the coverage of the system, but increasing its precision.

The resultant corpus of 113,802 instances provides, in addition to the sense tags, other kinds of useful information: POS-tags, lemmas and the neighbour words. This corpus will be used to train a supervised machine learning algorithm in order to produce a WSD model.

Although applied to a small set of words, the approach can be extended to wider contexts. Besides the parallel corpora, the required resources can be extracted from machine readable sources. In addition, the evaluation reported here was carried out on difficult cases, and thus the results on other lexical items are likely to be as good, if not better, than those reported.

In future work, we will experiment with different weighting schemes, in order to explore more deeply the statistical analysis of the parallel corpora. We plan to consider as possible translations also those indicated by the statistical analysis, but which are not included in the bilingual dictionaries. With this, we hope to minimize the dependence on the knowledge resources and allow unusual, but valid, translations to be identified.

References

(Agirre & Martínez, 2004) E. Agirre, D. Martínez. Unsupervised WSD Based on Automatically Retrieved Examples: The Importance of Bias. In *Proceedings of the Conference on Empirical Methods in NLP*, pp. 25-32, 2004.

(Bar-Hillel, 1960) Y. Bar-Hillel. Automatic Translation of Languages. *Advances in Computers*. Academic Press, New York, 1960.

(Brown et al., 1991) P.F. Brown, S.A. Della Pietra, V.J Della Pietra, R.L. Mercer. Word Sense Disambiguation Using Statistical Methods. In *Proceedings of the 29th Annual Meeting of ALC*, pp. 264-270, 1991.

(Burnard, 2000) L. Burnard. Reference Guide for the British National Corpus. Oxford University Press, 2000.

(Caseli et al., 2004) H.M. Caseli, A.M.P. Silva, M.G.V. Nunes. Evaluation of Methods for Sentence and Lexical Alignment of Brazilian Portuguese and English Parallel Texts. In *Proceedings of the 7th* SBIA, Sao Luiz, pp. 184-193, 2004.

(Dagan & Itai, 1994) I. Dagan, A. Itai. Word Sense Disambiguation Using a Second Language Monolingual Corpus. *Computational Linguistics*, 20:563-596, 1994.

(Diab & Resnik, 2002) M. Diab, P. Resnik. An Unsupervised Method for Word Sense Tagging using Parallel Corpora. In *Proceedings of the 40th Anniversary Meeting of the ACL,* Philadelphia, 2002.

(Dinh, 2002) D. Dinh. Building a training corpus for word sense disambiguation in the English-to-Vietnamese Machine Translation. In *Proceedings of Workshop on Machine Translation in Asia*, pp. 26-32, 2002.

(Edmonds & Cotton, 2001) P. Edmonds, S. Cotton. SENSEVAL-2: Overview. In *Proceedings of the 2nd Workshop on Evaluating Word Sense Disambiguation Systems*, pp. 1-5, 2001.

(Feltrim, 2004) V.D. Feltrim. *Uma abordagem baseada em córpus e em sistemas de crítica para a construção de ambientes Web de auxílio à escrita acadêmica em português*. Tese de Doutorado, Universidade de São Paulo, São Carlos, 2004.

(Frankenberg-Garcia & Santos, 2003) A. Frankenberg-Garcia, D. Santos. Introducing COMPARA: the Portuguese-English Parallel Corpus. *Corpora in translator education*, pp. 71-87, 2003.

(Hutchins & Somers, 1992) W.J. Hutchins, H.L. Somers. *An Introduction to Machine Translation*. Academic Press, UK, 1992.

(Ide et al., 2002) N. Ide, T. Erjavec, D. Tufis. Sense Discrimination with Parallel Corpora. In *Proceedings of the SIGLEX Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, Philadelphia, pp. 56-60, 2002.

(Koehn, 2002) P. Koehn. *Europarl: A Multilingual Corpus for Evaluation of Machine Translation*, 2002.,

(www.isi.edu/~koehn/publications/europarl).
(Lin, 1993) D. Lin. Principle based parsing without overgeneration. In *Proceedings of the 31st Annual Meeting of the ACL,* Columbus, pp. 112-120, 1993.
(Miller et al., 1994) G.A. Miller, M. Chorodow, S. Landes, C. Leacock, R.G. Thomas. Using a Semantic Concordancer for Sense Identification. In *Proceedings of the ARPA Human Language Technology Workshop - ACL*, Washington, pp. 240-243, 1994.
(Ng et al., 2003) H.T. Ng, B. Wang, Y.S. Chan. Exploiting Parallel Texts for Word Sense Disambiguation: An Empirical Study. In *Proceedings of the ALC-2003,* Sapporo, pp. 455-462, 2003.
(Och & Ney, 2003) F.J. Och, H. Ney. A Systematic Comparison of Various Statistical Alignment Models, *Computational Linguistics*, 29(1):19-51, 2003.
(Ratnaparkhi, 1996) A. Ratnaparkhi. A Maximum Entropy Part-Of-Speech Tagger. In *Proceedings of the Conference on Empirical Methods in NLP,* Pennsylvania, 1996.
(Simões & Almeida, 2003) A.M. Simões, J.J. Almeida. NATools - A Statistical Word Aligner Workbench. In *Proceedings da Sociedade Española para el Procesamiento del Lenguaje Natural*, Madrid, 2003.
(Specia, 2005) L. Specia. A Hybrid Model for Word Sense Disambiguation in English-Portuguese Machine Translation. In *Proceedings of the 8th Research Colloquium of the UK Special-interest Group in Computational Linguistics*, Manchester, pp. 71-78, 2005.
(Wilks & Stevenson, 1998) Y. Wilks, M. Stevenson. The Grammar of Sense: Using part-of-speech tags as a first step in semantic disambiguation. *Journal of Natural Language Engineering*, 4(2):135-144, 1998.

Exploiting Linguistic Cues to Classify Rhetorical Relations

Caroline Sporleder and **Alex Lascarides**
School of Informatics
University of Edinburgh
2 Buccleuch Place
Edinburgh EH8 9LW
{csporled,alex}@inf.ed.ac.uk

Abstract

We propose a method for automatically identifying rhetorical relations. We use supervised machine learning but exploit cue phrases to automatically extract and label training data. Our models draw on a variety of linguistic cues to distinguish between the relations. We show that these feature-rich models outperform the previously suggested bigram models by more than 20%, at least for small training sets. Our approach is therefore better suited to deal with relations for which it is difficult to automatically label a lot of training data because they are rarely signalled by unambiguous cue phrases (e.g., CONTINUATION).

1 Introduction

Clauses in a text relate to each other via rhetorical relations such as CONTRAST, EXPLANATION or RESULT (see, e.g., (Mann & Thompson 87)). For example, (1b) relates to (1a) with RESULT:

(1) a. A train hit a car on a level crossing.
b. It derailed.

Many NLP applications would benefit from a method which automatically identifies such relations. Question-answering and information extraction systems, for instance, could use them to answer complex queries about the cause or result of an event. Rhetorical relations have also been shown to be useful for automatic text summarisation (Marcu 98).

While rhetorical relations are sometimes signalled by cue phrases (also known as *discourse connectives*) such as *but*, *since* or *consequently*, these are often ambiguous. For example, *since* can indicate either a temporal or an explanation relation (examples (2a) and (2b), respectively). Furthermore, cue phrases are often missing (as in (1) above). Hence, it is not possible to rely on cue phrases alone.

(2) a. She has worked in retail since she moved to Britain.
b. I don't believe he's here since his car isn't parked outside.

In this paper, we present a machine learning method which uses a variety of (relatively shallow) linguistic and textual features, such as word stems, part-of-speech tags or tense information, to determine the rhetorical relation between two adjacent text spans (sentences or clauses) *in the absence* of a cue phrase. We employ a supervised machine learning technique based on decision trees and boosting (Schapire & Singer 00). However, to avoid manual annotation of large amounts of training data, we train on automatically labelled examples, building on earlier work by (Marcu & Echihabi 02), who extracted examples from large text corpora and used cue phrases to label them with the correct rhetorical relation. The cue phrases were then removed before the classifiers were trained.

This approach works because there is often a certain amount of redundancy between the cue phrase and the general linguistic context. For example, the two clauses in example (3a) are in a CONTRAST relation signalled by *but*. However, this relation can also be inferred if no cue phrase is present (see (3b)).

(3) a. She doesn't make bookings but she fills notebooks with itinerary recommendations.
b. She doesn't make bookings; she fills notebooks with itinerary recommendations.

(Hobbs *et al.* 93) and (Asher & Lascarides 03) propose a *logical* approach to inferring relations, which in this case would rely on the linguistic cues of a negation in the first span, syntactic parallelism of the two spans, and the fact that they both have the same subject. We intend to explore whether such cues can also be exploited as features in a statistical model for recognising rhetorical relations.

Thus, the main difference between our research and the earlier work by (Marcu & Echihabi 02) is that their models rely on word co-occurrence

statistics alone while we use a variety of linguistic features, similar to those used by (Lapata & Lascarides 04) and inspired by symbolic approaches to the task (Hobbs *et al.* 93; Corston-Oliver 98). We also use a different set of relations.

2 Related Research

(Marcu & Echihabi 02) present a machine learning approach to automatically identify four rhetorical relations (CONTRAST, CAUSE-EXPLANATION-EVIDENCE, CONDITION and ELABORATION) from the inventory of relations described in (Mann & Thompson 87). Two types of non-relations (NO-RELATION-SAME-TEXT, NO-RELATION-DIFFERENT-TEXTS) are also included. The training data are extracted automatically from a large text corpus (around 40 million sentences) using manually constructed extraction patterns containing cue phrases which typically signal one of these relations. For example, if a sentence begins with the word *but*, it is extracted together with the immediately preceding sentence and labelled with the relation CONTRAST. Examples of non-relations are created artificially by selecting non-adjacent text spans (from the same or different texts). Because the text spans are non-adjacent and randomly selected, it is relatively unlikely that a relation holds between them. Using this method, the authors obtain between 900,000 and 4 million examples per relation.

The cue phrases were then removed from the extracted data and a set of Naive Bayes classifiers was trained to distinguish between relations on the basis of co-occurrences between pairs of lexical items. (Marcu & Echihabi 02) report a test set accuracy of 49.7% for the six-way classifier.

(Lapata & Lascarides 04) present a method for inferring temporal connectives. They, too, extract training data automatically, using connectives such as *while* or *since*. But their task differs from ours and Marcu and Echihabi's, in that they aim to predict the original temporal connective (which was removed from the test set) rather than the underlying rhetorical relation. They thus tackle connectives which are ambiguous with respect to the rhetorical relations they signal, such as *since*, and they do not address how to disambiguate them. To achieve their task, they train simple probabilistic models based on nine types of linguistically motivated features. They report accuracies of up to 70.7%.

There have also been a variety of non-statistical approaches to the problem. (Corston-Oliver 98), for instance, presents a system which takes fully syntactically analysed sentences as input and determines rhetorical relations by applying heuristics which take a variety of linguistic cues into account, such as clausal status, anaphora and deixis. (Le Thanh *et al.* 04) use heuristics based on syntactic properties and cue phrases to split sentences into discourse spans and to determine which intra-sentential spans should be related. In a second step, they then combine several cues, such as syntactic properties, cue words and semantic information (e.g. synonyms) to determine which relations hold between these spans. Finally, they derive a discourse structure for the complete text by incrementally combining sub-trees into larger textual units.

3 Our Approach

3.1 Relations and Cue Phrase Selection

We chose a subset of rhetorical relations from SDRT's inventory (Asher & Lascarides 03), namely: CONTRAST, RESULT, EXPLANATION, SUMMARY and CONTINUATION. These relations were selected on the basis that for each of them, there are *unambiguous* cue phrases but these relations also frequently occur *without* a cue phrase; so it is beneficial to be able to determine them automatically if no cue phrase is present. This is in marked contrast to relations such as CONDITION, which always require a cue phrase (e.g., *if... then* or *suppose that ...*).

SDRT relations are defined purely on the basis of truth conditional semantics and therefore tend to be less fine-grained than those used in Rhetorical Structure Theory (RST) (Mann & Thompson 87) (see below). Let $R(a, b)$ denote the fact that a relation R connects two spans a and b. For each of the five relations it holds that $R(a, b)$ is true only if the the contents of a and b are true too. In addition, **contrast(a,b)** entails that a and b have parallel syntactic structures that induce contrasting themes, **result(a,b)** entails that a causes b, **summary(a,b)** entails that a and b are semantically equivalent, **continuation(a,b)** means that a and b have a contingent, common topic and **explanation(a,b)** means that b is an answer to the question *why a?* (cf. (Bromberger 62)).

To identify mappings from cue phrases to the SDRT relations they signal, and in particular to

identify unambiguous cue phrases, we undertook an extensive corpus study, using 30 randomly selected examples for each cue phrase (i.e., around 2,000 examples in all), as well as linguistic introspection given SDRT's dynamic semantic interpretation. The differences between SDRT and RST mean that some cue phrases which are ambiguous in RST are unambiguous in SDRT. For example, *in other words* can signal either SUMMARY or RESTATEMENT in RST, but SDRT does not not distinguish these relations since the length of the related spans is irrelevant to SDRT's semantics. Similarly, SDRT does not distinguish EXPLANATION and EVIDENCE, and therefore, while *because* is ambiguous in RST, it is unambiguous in SDRT, signalling only EXPLANATION. SDRT also does not distinguish CONTRAST, ANTITHESIS and CONCESSION, making *but* unambiguous.

Sentences (4) to (8) below show one automatically extracted example for each relation (cue phrases which were used for the extraction and removed before training are underlined, and the two spans are indicated by square brackets).

(4) [We can't win] [but we must keep trying.]
(CONTRAST)

(5) [The ability to operate at these temperatures is advantageous,] [because the devices need less thermal insulation.]
(EXPLANATION)

(6) [By the early eighteenth century in Scotland, the bulk of crops were housed in ricks,] [the barns were consequently small.]
(RESULT)

(7) [The starfish is an ancient inhabitant of tropical oceans.] [In other words, the reef grew up in the presence of the starfish.]
(SUMMARY)

(8) [First, only a handful of people have spent more than a few weeks in space.] [Secondly, it has been impractical or impossible to gather data beyond some blood and tissue samples.]
(CONTINUATION)

3.2 Data

We used three corpora, mainly from the news domain, to extract our data set: the British National Corpus (BNC, 100 million words), the North American News Text Corpus (350 million words) and the English Gigaword Corpus (1.7 million words). We took care to remove duplicate texts. Since we were mainly interested in written texts, we also excluded all BNC files which are transcripts of speech.

Most of our corpora were not annotated with sentence boundaries, so we used a publicly available sentence splitter (Reynar & Ratnaparkhi 97), which was pre-trained on news texts, to automatically insert sentence boundaries.

The extraction happened in two steps. First, we processed the raw text corpora to extract *potential* training examples using manually written extraction patterns based on 55 (relatively unambiguous) cue phrases. All extracted examples were then parsed with the RASP parser (Carroll & Briscoe 02) and the parse trees were processed to (i) identify the two spans using simple heuristics (based on clause boundaries and the position of the cue phrases) and (ii) filter out any false positives that could not be filtered out using the raw texts alone.

An example of the latter is sentence (9), which was extracted as an example of a SUMMARY relation based on the apparent presence of the cue phrase *in short*. However, the parser correctly identified this string as part of the prepositional phrase *in short order* and the example was discarded. Examples which could not be parsed (or only partially parsed) were also discarded at this stage. For each of the extracted training examples, we also kept track of its position in the paragraph as we used this information in one of our features.

(9) In short order I was to fly with 'Deemy' on Friday morning.

Using this two step extraction method we were able to extract both intra- and inter-sentential relations (see (4) and (7) above, respectively). However, we limited the length of the extracted spans to one sentence as we specifically wanted to focus on relations between small units of text.

There are three potential sources of noise in the extraction process: (i) the two spans are not related, (ii) they are related but the wrong relation is hypothesised and (iii) the hypothesised span boundaries are wrong. The latter applies particularly to SUMMARY and RESULT, where either span can contain more than one sentence. In this case we would only extract the first (or last) sentence of the span. However, this will not cause any harm provided the partially extracted span already contains enough cues for our model to correctly learn the relation.

In our extraction method we went for high precision at the expense of recall. A small-scale evaluation using 100 randomly selected, hand-corrected examples (20 per relation) revealed 11 extraction errors overall. In no case was the wrong relation predicted. Three errors were due to hypothesising a relation where there was none. The remaining 8 errors were wrong boundary predictions (partly due to our "one sentence per span" limit, partly due to sentence-splitting errors). Hence we achieved an overall precision of 89% (97% if the less important boundary errors are excluded).

The number of training examples we could extract automatically differed for every relation: for CONTINUATION we obtained less than 2,000 examples whereas for the most frequently extracted relation, CONTRAST, we obtained around 50,000 examples. On the whole, our data set is much smaller than the one used by (Marcu & Echihabi 02), which contained around 10 million examples for six relations. Our task is thus more challenging in the sense that we are classifying rhetorical relations on the basis of a smaller training set.

3.3 Machine Learning

We used BoosTexter (Schapire & Singer 00) as our machine learning system. BoosTexter was originally developed for text categorisation. It combines a boosting algorithm with simple decision rules and allows a variety of feature types, such as nominal, numerical or text-based features. For the latter, BoosTexter applies n-gram models when forming classification hypotheses. We used BoosTexter's default settings in all experiments discussed below.

3.4 Features

We implemented a variety of linguistically motivated features (72 in total), roughly falling into 9 classes: positional features, length features, lexical features, part-of-speech features, temporal features, syntactic features and cohesion features.

Positional Features We defined three positional features. The first encodes whether the relation holds intra- or inter-sententially. The second and third encode whether the example occurs towards the beginning or end of a paragraph. The motivation for these features is that the likelihood of different relations varies with both their paragraph position and the position of sentence boundaries relative to span boundaries. For instance, CONTRAST is more likely to hold between two clauses within a sentence than CONTINUATION. And a SUMMARY relation is probably more frequent at the beginning or end of a paragraph than in the middle of it.

Length Features Information about the length of the spans might be equally useful. For example, it is possible that the average span length for CONTINUATION is longer than for CONTRAST.

Lexical Features Lexical information is also likely to provide useful cues for identifying the correct relation (cf. (Marcu & Echihabi 02)). For example, word overlap may be evidence for a SUMMARY relation. Furthermore, while we do not use cue phrases as our model features (as they provide the basis on which the data is labelled), there may be words not in our cue phrase inventory which hint at the presence of a particular relation. For instance, *still* often occurs in contrasts.

We incorporated a variety of lexical features. For each of the spans, we included the string of lemmas and stems of all words as a text-based feature. We also separately included the lemmas of all content words. Encoding lexical items as text-based features allows BoosTexter to automatically identify n-grams that may be good cues for a particular relation. Note that BoosTexter will only consider n-grams that form a continuous string. Hence bigrams in BoosTexter are different from the (non-adjacent) word-pairs used in (Marcu & Echihabi 02).

As a further feature, we calculated the overlap between the spans, i.e., what proportion of stems, lemmas, and content-word lemmas occurs in both, and added this as a numerical feature.

Part-of-Speech Features We encoded the string of part-of-speech tags for both spans as a text-based feature as it is possible that certain part-of-speech tags (e.g., certain pronouns) are more likely for some relations than for others. Following (Lapata & Lascarides 04), we also encoded specific information about the verbs, nouns and adjectives in the spans. In particular, we included the string of verb (noun, adjective) lemmas contained in each span as text-based features. For instance, the strings of verb lemmas in example (5), repeated as (10) below, are "*operate be*"

(left span) and "*need*" (right span).

(10) The ability to operate at these temperatures is advantageous because the devices need less thermal insulation.

We also mapped the lemmas to their most general WordNet (Fellbaum 98) class (e.g., verb-of-cognition or verb-of-change for verbs, event or substance for nouns etc.). Ambiguous lemmas which belong to more than one class, were mapped to the class of their most frequent sense. If a lemma was not in WordNet, the lemma itself was used. Finally, we also calculated the overlaps between lemmas and between WordNet classes for each part-of-speech class and included these as numerical features.

Temporal Features Tense and aspect provide clues about temporal relations among events and may also influence the probabilities of different rhetorical relations. We therefore included temporal features in the model. To do so, we first extracted all verbal complexes from the parse trees and then used simple heuristics to classify each of them in terms of finiteness, modality, aspect, voice and negation (Lapata & Lascarides 04). For example, *need* in example (10) maps to: present, 0, imperfective, active, positive. We also introduced an additional feature where we only encoded this information for the main verbal complex in each span.

Syntactic Features It is likely that some relations (e.g., SUMMARY) have syntactically less complex spans than others (e.g., CONTINUATION). To estimate syntactic complexity we determined the number of NPs, VPs, PPs, ADJPs, and ADVPs contained in each span. Information about the argument structure of a clause may serve as another measure of syntactic complexity. We therefore encoded several aspects of argument structure as well, e.g., whether a verb has a direct or indirect object or whether it is modified by an adverbial. This information can be easily extracted from the RASP parse trees. We also included information about the subjects, i.e., their part-of-speech tags, whether they have a negative aspect (e.g. *nobody*, *nowhere*) and the WordNet classes to which they map (see above).

Cohesion Features The degree of cohesion between two spans may be another informative feature. To estimate it we looked at the distribution of pronouns and at the presence or absence of ellipses (cf. (Hutchinson 04)). For the former, we kept track of the number of first, second and third person pronouns in each span. We also used simple heuristics to identify whether either span ends in a VP ellipsis and included this information as a feature.

4 Experiments

We conducted three main experiments. First we assessed how well humans can determine rhetorical relations in the absence of cue phrases. This gives a measure of the difficulty of the task. We then determined the performance of our machine learning models and compared it to two baselines. Finally, we looked at which features are particularly useful for predicting the correct relation.

4.1 Experiment 1: Human Agreement

As we mentioned earlier, automatically extracting and labelling training data for a supervised machine learning paradigm in the way suggested in this paper and in earlier work (Marcu & Echihabi 02) relies on the existence of a certain amount of redundancy between the cue phrase and other linguistic features in signalling which rhetorical relation holds. If cue phrases were only used in cases where a relation cannot be inferred from the linguistic context alone, any approach which aims to train a classifier on automatically extracted examples from which the cue phrases have been removed would fail.

The presence of redundancy in some cases is evident from examples like (3), where CONTRAST can be inferred even when the cue phrase is removed. However, there may be other cases where this is more difficult. To assess the difficulty of determining the rhetorical relation in examples from which the cue phrase has been removed, we conducted a small pilot study with human subjects.

We used our extraction patterns to automatically extract examples for the four rhetorical relations CONTRAST, EXPLANATION, RESULT and SUMMARY (CONTINUATION was added after the pilot study). We then manually checked the extracted examples to filter out false positives and randomly selected 10 examples per relation from which we then removed the cue phrases. We also semi-automatically selected 10 examples of adjacent sentences or clauses which were not related by any of the four relations. For each example,

we also included the two preceding and following sentences as context, keeping track of any paragraph markings. We then asked three subjects who were trained in discourse annotation to classify each of the 50 examples as one of the four relations or as NONE. All subjects were aware that cue phrases had been removed from the examples but did not know the location of the removed cue phrase. We evaluated the annotations against the gold standard and calculated the average accuracy. To estimate inter-annotator agreement, we also determined the Kappa coefficient (Siegel & Castellan 88). The results are shown in Table 1.

Avg. Accuracy	Kappa (pairwise, avg.)
71.25	.61

Table 1: Human performance

While the agreement is far from perfect, it is relatively high for a discourse annotation task. Hence it seems that the task of predicting the correct relation for sentences from which the cue phrase has been removed is feasible for humans. However, the accuracy was not equally high for all relations: RESULT (90%), CONTRAST (83%) and EXPLANATION (75%) seem to be relatively easy, while SUMMARY (57%) is more difficult, and the accuracy was lowest for the NONE class (50%).

Interestingly, our findings regarding the relative ease with which a given relation can be inferred if the original cue phrase is removed, deviate from those obtained by (Soria & Ferrari 98), who conducted a similar experiment for Italian. They found that "additive relations" (like SUMMARY) are easiest to infer, followed by "consequential relations" (e.g., RESULT and EXPLANATION) and "contrastive relations" (e.g., CONTRAST), which were found to be the most difficult by far. Without further research it is difficult to say where the difference between our and Soria & Ferrari's findings stem from. They could be language-specific (i.e., English vs. Italian), domain-specific (mainly news texts vs. mixed genres) or due to the different taxonomies of relations.

4.2 Experiment 2: Probabilistic Modelling

Our machine learning experiments involved five relations: CONTRAST, EXPLANATION, RESULT, SUMMARY and CONTINUATION. The automatic extraction method yielded very different amounts of training data for each of them (see section 3.2). However, machine learning from skewed data is highly problematic as it often leads to classifiers which always predict the majority class (Japkowicz 00). To avoid this problem, we decided to create uniform training (and test) sets which contained an equal number of examples for each relation. The number of examples for the least frequent relation (CONTINUATION) was 1,732 and we randomly selected the same number of examples for each of the other relations. We used 90% of this data set for training (7,795 examples) and 10% for testing (865 examples), making sure that the distribution of the relations was uniform in both data sets, and evaluated BoosTexter's performance using 10-fold cross-validation.

For comparison, we also used two baselines. For the first, a relation was predicted at random. As there are five relations and all are equally frequent in the test set, the average accuracy achieved by this strategy will be 20%. For the second baseline, we implemented a bigram model along the lines of (Marcu & Echihabi 02). Table 2 shows the average accuracies of the three classifiers for all relations and also for each individual relation.

It can be seen that our feature-rich BoosTexter model performs notably better than either of the other two classifiers. It outperforms the random baseline by nearly 40% and the bigram model by more than 20%. This difference is statistically significant ($\chi^2 = 208.12$, DoF = 1, p <= 0.01). Furthermore, the performance gain achieved by our model holds for every relation with the exception of EXPLANATION where the bigram model performs better.

	Avg. Accuracy		
Relation	random	bigrams	BT
contrast	20.00	33.11	43.64
explanation	20.00	75.39	64.45
result	20.00	16.21	47.86
summary	20.00	19.34	48.44
continuation	20.00	25.48	83.35
all	**20.00**	**33.96**	**57.55**

Table 2: Results for BoosTexter (BT) and two baselines (10-fold cross-validation)

The comparison with the bigram model is not entirely fair as this method is geared towards large training sets. For example, (Marcu & Echihabi 02) use it on a data set of nearly 10 million examples, and their 6-way classifier achieves

49.7% compared with the 5-way classifier reported here with 33.96% accuracy. However, while it is possible that the bigram model outperforms our feature-rich BoosTexter model on large training sets, obtaining large amounts of training data is not always feasible, even if these are extracted automatically. As we have mentioned, some relations occur relatively infrequently. Others may appear more often but usually without an unambiguous cue phrase signalling the relation. In these cases even very large text corpora may not be big enough to extract sufficient training data for a bigram model to perform well. In our experiments, this case arose with the CONTINUATION relation, for which less than 2,000 examples could be extracted from a text corpus of 450 million words. For such relations, our approach seems a better choice than the bigram model proposed by (Marcu & Echihabi 02).

It is interesting that our model and the bigram model differ with respect to which relations are identified most reliably. Our model achieves the highest accuracy for CONTINUATION and the lowest for CONTRAST, while the bigram model achieves the highest accuracy for EXPLANATION and the lowest for RESULT. This suggests that it might be possible to achieve even better results by combining both models, for example, by incorporating the bigram model as a feature in our BoosTexter model.

Since our model already achieves fairly good results for the relation for which we could extract the fewest training examples (CONTINUATION), but less good results for relations for which we could extract a larger set of training examples, such as CONTRAST, it may also be possible to further improve performance by including more training data for the latter.

4.3 Experiment 3: Feature Exploration

To determine which features are particularly useful for the task, we conducted a further experiment in which we trained an individual BoosTexter model for each of our features. We then tested these one-feature classifiers on an unseen test set (again using 10-fold cross-validation) and calculated the accuracies. Table 3 shows the 10 best performing features and their average accuracies.

This suggests that lexical features (stems, words, lemmas) are the most useful features. Table 4 shows some of the words chosen by BoosTexter as being particularly predictive of a given relation. Most of the choices seem fairly intuitive. For instance, an EXPLANATION relation is often signalled by tentatively qualifying adverbs such as *perhaps* or *probably*, while SUMMARY and CONTINUATION relations frequently contain pronouns and CONTRAST can be signalled by words such as *other*, *still* or *not* etc. Of course the predictive power of such words may be to some extent domain dependent. Our examples came largely from the news domain and the situation may be slightly different for other domains.

Feature	Avg. Accuracy
left stems	42.51
left words	41.79
intra/inter	39.18
left pos-tags	34.62
right words	32.82
right stems	32.58
right pos-tags	31.72
left content words	29.78
left noun lemmas	28.30
right span length	28.12

Table 3: Best features (10-fold cross-validation)

Table 3 also suggests that the lexical items in the left span are more important than those in the right span. For example, the feature *left stems* is 10% more accurate then the feature *right stems*. This makes sense from a processing perspective: if the relation is already signalled in the left span the sentence will be easier to process than if the signalling is delayed until the right span is read.

Relation	Predictive Words
contrast	other, still, not, ...
explanation	perhaps, probably, mainly, ...
result	undoubtedly, so, indeed, ...
summary	their, this, yet ...
continuation	you, it, there ...

Table 4: Words chosen as cues for a relation

Another feature which proves very useful is *intra/inter*, which encodes whether the relation is intra- or inter-sentential. BoosTexter predicts CONTINUATION if the relation is inter-sentential and EXPLANATION otherwise. This decision rule is probably responsible for the high accuracy achieved for CONTINUATION as most CONTINUATION relations are indeed inter-sentential (though there are exceptions).

5 Conclusion

We have presented a machine learning method for automatically classifying discourse relations in the absence of cue phrases. Our method uses feature-rich models which combine a wide variety of linguistic features. We employed supervised machine learning techniques to train these models but extracted and labelled our training data automatically using predefined extraction patterns. Consequently no annotation effort is required.

We tested our method on five rhetorical relations and compared the performance of our models to that achieved by a bigram model. We found that our feature-rich models significantly outperform the simpler bigram models, at least on relatively small training sets. This means that our method is particularly suitable for relations which are rarely signalled by (unambiguous) cue phrases (e.g., CONTINUATION). In such cases, it is difficult to obtain sufficiently large training sets that a bigram model will perform well, even if the training set is obtained automatically from very large text corpora (manually constructing sufficiently large training sets is, of course, equally problematic).

In future research, we plan to conduct classification experiments with the most frequent relations to investigate whether our models are indeed outperformed by bigram models on large training sets and if so at what point this happens.

So far we have only tested our method on examples from which the cue phrases had been removed and not on examples which occur naturally without a cue phrase. However, these are exactly the types of examples at which our method is aimed. So we also intend to create a small, manually labelled, test corpus containing naturally occurring examples without cue phrases and test our method on this to determine whether our results carry over to that data type; the RST Discourse Treebank (Carlson *et al.* 02) could be used as a starting point for this (cf. (Marcu & Echihabi 02)).

Acknowledgements

The research in this paper was supported by EPSRC grant number GR/R40036/01. We would like to thank Ben Hutchinson and Mirella Lapata for interesting discussions, participating in our labelling experiment and letting us use some of their Perl scripts. We are also grateful to the reviewers for their helpful comments.

References

(Asher & Lascarides 03) Nicholas Asher and Alex Lascarides. *Logics of Conversation*. Cambridge University Press, 2003.

(Bromberger 62) Sylvain Bromberger. An approach to explanation. In Ronald J. Butler, editor, *Analytical Philosophy*, pages 75–105. Oxford University Press, 1962.

(Carlson *et al.* 02) Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. RST Discourse Treebank. Linguistic Data Consortium, 2002.

(Carroll & Briscoe 02) John Carroll and Edward Briscoe. High precision extraction of grammatical relations. In *Proceedings of COLING-02*, pages 134–140, 2002.

(Corston-Oliver 98) Simon H. Corston-Oliver. Identifying the linguistic correlates of rhetorical relations. In *Proceedings of the ACL Workshop on Discourse Relations and Discourse Markers*, pages 8–14, 1998.

(Fellbaum 98) Christiane Fellbaum, editor. *WordNet: An Electronic Database*. MIT Press, Cambridge, MA, 1998.

(Hobbs *et al.* 93) Jerry R. Hobbs, Mark Stickel, Douglas Appelt, and Paul Martin. Interpretation as abduction. *Artificial Intelligence*, 63(1–2):69–142, 1993.

(Hutchinson 04) Ben Hutchinson. Acquiring the meaning of discourse markers. In *Proceedings of ACL-04*, pages 685–692, 2004.

(Japkowicz 00) Nathalie Japkowicz. The class imbalance problem: Significance and strategies. In *Proceedings of IJCAI-00*, pages 111–117, 2000.

(Lapata & Lascarides 04) Mirella Lapata and Alex Lascarides. Inferring sentence-internal temporal relations. In *Proceedings of NAACL-04*, pages 153–160, 2004.

(Le Thanh *et al.* 04) Huong Le Thanh, Geetha Abeysinghe, and Christian Huyck. Generating discourse structures for written text. In *Proceedings of COLING-04*, pages 329–335, 2004.

(Mann & Thompson 87) William C. Mann and Sandra A. Thompson. Rhetorical structure theory: A theory of text organization. Technical Report ISI/RS-87-190, ISI, Los Angeles, CA, 1987.

(Marcu & Echihabi 02) Daniel Marcu and Abdessamad Echihabi. An unsupervised approach to recognizing discourse relations. In *Proceedings of ACL-02*, pages 368–375, 2002.

(Marcu 98) Daniel Marcu. Improving summarization through rhetorical parsing tuning. In *The 6th Workshop on Very Large Corpora*, pages 206–215, 1998.

(Reynar & Ratnaparkhi 97) Jeffrey C. Reynar and Adwait Ratnaparkhi. A maximum entropy approach to identifying sentence boundaries. In *Proceedings of ANLP-97*, pages 16–19, 1997.

(Schapire & Singer 00) Robert E. Schapire and Yoram Singer. BoosTexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

(Siegel & Castellan 88) Sidney Siegel and N. John Castellan. *Nonparametric Statistics for the Behavioral Sciences*. McGraw-Hill, New York, 1988.

(Soria & Ferrari 98) Claudia Soria and Giacomo Ferrari. Lexical marking of discourse relations – some experimental findings. In *Proceedings of the ACL-98 Workshop on Discourse Relations and Discourse Markers*, 1998.

Optimizing information retrieval in question answering using syntactic annotation

Jörg Tiedemann
Alfa Informatica, University of Groningen
tiedeman@let.rug.nl

Abstract

One of the bottle-necks in open-domain question answering (QA) systems is the performance of the information retrieval (IR) component. In QA, IR is used to reduce the search space for answer extraction modules and therefore its performance is crucial for the success of the overall system. However, natural language questions are different to sets of keywords used in traditional IR. In this study we explore the possibilities of integrating linguistic information taken from machine annotated Dutch newspaper text into information retrieval. Various types of morphological and syntactic features are stored in a multi-layer index to improve IR queries derived from natural language input. The paper describes a genetic algorithm for optimizing queries send to such an enriched IR index. The experiments are based on the CLEF test sets for Dutch QA from the last two years. We could show an absolute improvement of about 8% in mean reciprocal rank scores compared to the base line using traditional IR with plain text keywords.

1 Introduction

One of the strategies in question answering (QA) systems is to identify possible answers in large document collections. The task of the information retrieval (IR) component in such a system is to reduce the search space for information extraction modules that look for possible answers in relevant text passages. Obviously, the system fails if IR does not provide appropriate segments to the subsequent modules. Hence, the performance of IR is crucial for the entire system.

The main problem for IR is to match a given query with relevant documents. This is usually done in a bag-of-words approach, i.e. sets of query keywords are matched with word type vectors describing documents in the collection. However, in QA we start up with a well-formed natural language question from which an appropriate query has to be formulated to send to the IR component. The base line approach is simply to use all content words in the question as keywords to run traditional IR. In many cases this is not satisfactory especially where questions are short with only a few informative content words. In some cases we want to restrict the query to narrow down possible matches (to improve precision). In other cases, where keywords from the question are to restrictive, we want to widen the query to increase recall.

Natural language questions are more than bags of words and contain additional information besides possible keywords. Syntactic constructions and dependencies between constituents in the question bear valuable information about the given request. The challenge for QA is to take advantage of any linguistic clue in the question that might be necessary to find an appropriate answer. Therefore, natural language processing (NLP) is used in many components of QA systems, for example, in question analysis, answer extraction and off-line information extraction (see e.g. (Moldovan *et al.* 02; Jijkoun *et al.* 04; Bouma *et al.* 05)). The use of NLP tools in information retrieval has been studied mainly to find better and/or additional index terms, e.g. complex noun phrases, named entities, disambiguated root forms (see e.g. (Zhai 97; Prager *et al.* 00; Neumann & Sacaleanu 04)). Several studies also investigate the use of other syntactically derived word pairs (Fagan 87; Strzalkowski *et al.* 96). (Katz & Lin 03) argue that syntactic relations can be very effective in information retrieval in question answering when selected carefully. Following up on these ideas, we would like to combine various features and relations that can be extracted from linguistically analyzed documents in our IR component to find better matches between natural language questions and relevant text passages.

Our investigations are focused on open-domain question answering for Dutch using dependency relations. We use the wide-coverage dependency parser Alpino (Bouma *et al.* 01) to produce linguistic analyses of both questions as well as sentences in documents in which we expect to find

the answers. An example of a syntactic dependency tree produced by Alpino can be seen in figure 1.

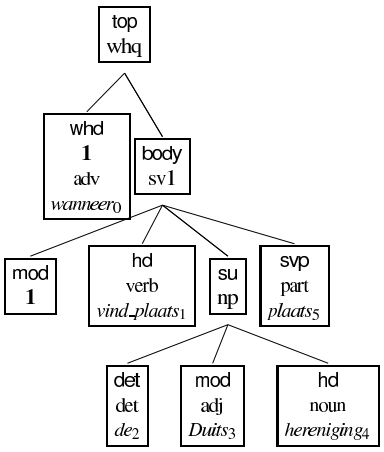


Figure 1: A dependency tree produced by Alpino for a Dutch CLEF question (When did the German re-unification take place?).

From the dependency trees produced by the parser we can extract features and relations that might be useful for IR, for example, part-of-speech information, named-entity labels, and, of course, syntactic dependency relations. The idea is to add this information to the index in some way to make it searchable via the IR component. Questions are analyzed in the same way and similar features and relations can be extracted from the annotation. Hence, we can match them with the enriched IR index to find relevant text passages. For this we assume that questions do not only share lexical items with relevant text passages but also other linguistic features such as syntactic relations. For example, if the question is about "winning the world cup" we might want to look for documents that include sentences where "world cup" is the direct object of any inflectional form of "to win". This would narrow down the query compared to a plain keyword search for "world", "cup" and "winning".

The nice thing about relevance ranking in IR is that we can also combine traditional keyword queries with more restrictive queries using, e.g., dependency relations. Documents that contain both types will be ranked higher than the ones where only one type is matched. In this way we influence the ranking but we do not reduce the number of selected documents.

Linguistic annotation can be used in many other ways. For example, part-of-speech information can be useful for disambiguation and weighting of keywords. Certain keyword types (e.g. nouns and names) can be marked as "required" or as "more important" than others. Named entity labels can be used to search for text passages that contain certain name types (for example, to match the expected answer type provided by question analysis). Morphological analyses can be used to split compositional compounds.

There is a large variety of possible features and feature combinations that can be included in a linguistically enriched IR index. There is also a wide range of possible queries to such an index using all the features extracted from analyzed questions. Finding appropriate features and query parameters is certainly not straightforward. In our experiments, we use data from the CLEF (Cross-Language Evaluation Forum) competition on Dutch QA to measure the success of linguistically extended queries. The following sections describe the IR component in our QA system and an iterative learning approach to feature selection and query formulation in the QA task.

2 The IR component

The IR component in our QA system (Joost) (Bouma *et al.* 05) is implemented as an interface to several off-the-shelf IR engines. The system may switch between seven engines that have been integrated in the system. One of the systems is based on the IR library Lucene from the Apache Jakarta project (Jakarta 04). Lucene is implemented in Java with a well-documented API. It implements a powerful query engine with relevance ranking and many additional interesting features. For example, Lucene indices may include several data fields connected to each document. This feature makes it very useful for our approach in which we want to store several layers of linguistic information for each document in the collection. Besides the data fields, Lucene also implements a powerful query language that makes it possible to adjust queries in various ways. For example, query terms can be weighted (using numeric "boost factors"), boolean operators are supported and proximity searches can be stated as well. It also allows for phrase searches and fuzzy matching. The support of data fields and the flexible query language are the main reasons for selecting Lucene as the IR engine in this study.

The IR interface can be used independently from Joost. In this way we can run batch calls on pre-defined queries without requiring other modules of the QA system.

2.1 The multi-layer index

The QA task in CLEF is corpus-based question answering. The corpus for the Dutch competition contains several years of newspaper texts, including about 190,000 documents with about 77 million words. Documents are marked with paragraph boundaries (which might be headers as well). We decided to use paragraphs for IR which gave the best balance between IR recall and precision. Paragraphs also seem to be a natural segmentation level for answer extraction even though the mark-up does not seem to be very homogeneous in the corpus. The entire corpus consists of about 1.1 million paragraphs that include altogether about 4 million sentences. The sentences have been parsed by Alpino and stored in XML tree structures.[1] From the parse trees, we extracted various kinds of features and feature combinations to be stored in different data fields in the Lucene index. Henceforth, we will call these data fields *index layers* and, thus, the index will be called a *multi-layer index*. We distinguish between *token layers*, *type layers* and *annotation layers*. Token layers include one item per token in the corpus. Table 1 lists token layers defined in our index.

Table 1: Token layers

text	stemmed plain text tokens
root	root forms
RootPos	root form + POS tag
RootHead	root form + head word
RootRel	root form + relation name
RootRelHead	root form + relation + head

As shown in the table above, certain features may appear in several layers combined with others. Features are simply concatenated (using special delimiting symbols between the various parts) to create individual items within the layer. Tokens in the *text* layer and in the *root* layer have also been split at hyphens and underscores to split compositional compounds (Alpino adds underscores between the compositional parts of words that have been identified to be compounds).

[1] About 0.35% of the sentences could not be analyzed because of parsing timeouts.

Type layers include only specific types of tokens in the corpus, e.g. named entities or compounds (see table 2).

Table 2: Type layers

compound	compounds (non-split root forms)
ne	named entities (non-split roots)
neLOC	location names
nePER	person names
neORG	organization names

Annotation layers include only the labels of (certain) token types. So far, we defined only one annotation layer for named entity labels. This layer may contain the items 'ORG', 'PER' or 'LOC' if such a named entity occurs in the paragraph.

2.2 Multi-layer IR queries

Features are extracted from analyzed questions in the same way as it was done for the entire corpus when creating the IR index (see section 2.1). Now, complex queries can be sent to the multi-layer index described above. Each individual layer can be queried using keywords of the same type. Furthermore, we can restrict keywords to exclude or include certain types using the linguistic labels of the analyzed question. For example, we can restrict RootPos keywords to nouns only. We can also add another restriction about the relation of these nouns within the dependency of the tree. We can, for example, use only the nouns that are in a object relation to their head in the tree. Now, we can also change weights of certain types (using Lucene's boost factors) and we can run proximity searches using pre-defined token window sizes. Here is a summary of query items that we may use in IR queries:

basic: a keyword in one of the index layers

restricted: *token-layer* keywords can be restricted to a certain word class ('noun', 'name', 'adj', 'verb') or/and a certain relation type ('obj1' (direct object), 'mod' (modifier), 'app' (apposition), 'su' (subject))

weighted: keywords can be weighted using a *boost factor*

proximity: a window can be defined for each set of *(restricted) token-layer* keywords

The restriction features (second keyword type) are limited to the ones listed above. We could

easily extend the list with additional POS labels or relation types. However we want to keep the number of possible keyword types at a reasonable level. Altogether there would be 304 different keyword types using all combinations of restrictions and basic index layers, although, some of them are pointless because they cannot be instantiated. For example, a verb is not to be found in an object relation to its head and therefore, such a combination of restriction is useless. For simplification, we consider only a small pre-defined set of combined POS/relation-type restrictions: *noun-obj1*, *name-obj1* (nouns or names as object); *noun-mod*, *name-mod* (nouns or names as modifiers); *noun-app*, *name-app* (nouns or names as appositions); and *noun-su*, *name-su* (nouns or names as subjects). In this way we get a total set of 208 keyword types.

Figure 2 shows a rather simple example query using different keyword types, weights and one proximity query.

Wanneer vond de Duitse hereniging plaats ?
(When did the German re-unification take place?)

```
RootRelHead:(Duits/mod/hereniging
             hereniging/su/vind_plaats)
root:((vind plaats)^0.2 Duits^0.2 hereniging^3)
text:("vond Duitse hereniging"~50)
```

Figure 2: An example query using linguistic features derived from a dependency tree using root-relation-head triples, roots with boost factor 0.2, noun roots with boost factor 3 and text tokens in a window of 50 words (stop words have been removed)

Note that all parts in the query are composed in a disjunctive way (which is the default operator in Lucene). In this way, each "sub-query" may influence the relevance of matching documents but does not restrict the query to documents for which each sub-query can be matched. In other words, no sub-query is required but all of them may influence the ranking according to their weights. An extension would be to allow even conjunctive parts in the query for items that are required in relevant documents. However, this is not part of the present study.

3 The CLEF test set

We used the CLEF test sets from the Dutch QA tracks in the years 2003 and 2004 as training and evaluation data. Both collections contain Dutch questions from the CLEF competitions that have been answered by the participating system. The test sets include the answer string(s) and document ID(s) of possible answers in the CLEF corpus. We excluded the questions for which no answer has been found. Most of the questions are factoid questions such as 'Hoeveel inwoners heeft Zweden?' (How many inhabitants does Sweden have?). Altogether there are 570 questions with 821 answers.[2]

For evaluation we used the mean reciprocal rank (MRR) of relevant paragraphs retrieved by IR:

$$MRR = \frac{1}{x} \sum_{x} \frac{1}{rank(first\_answer)}$$

We used the provided answer string rather than the document ID to judge if a retrieved paragraph was relevant or not. In this way, the IR engine may provide passages with correct answers from other documents than the ones marked in the test set. We do simple string matching between answer strings and words in the retrieved paragraphs. Obviously, this introduces errors where the matching string does not correspond to a valid answer in the context. However, we believe that this does not influence the global evaluation figure significantly and therefore we use this approach as a reasonable compromise when doing automatic evaluation.

4 Automatic query optimization

As described above, we have a large variety of possible keyword types that can be combined to query the multi-layer index. It would be possible to use intuition to set keyword restrictions, weights and window sizes. However, we like to carry out a more systematic search for optimizing queries using possible types and parameters. For this we use a simplified genetic algorithm in form of an iterative "trial-and-error beam search". The optimization loop works as follows (using a subset of the CLEF questions):

1. Run initial queries (one keyword type per IR run) with default weights and default window settings.

[2] Each question may have multiple possible answers. We also added some obvious answers which were not in the original test set when encountering them in the corpus. For example, names and numbers can be spelled differently (*Kim Jong Il* vs. *Kim Jong-Il*, *Saoedi-Arabië* vs. *Saudi-Arabië*, *bijna vijftig jaar* vs. *bijna 50 jaar*)

2. Combine the parameters of two of the N best IR runs (= *crossover*). For simplicity, we require each setting to be unique (i.e. we don't have to run a setting twice; the good ones survive anyway). Apply mutation operations (see next step) if crossover does not produce a unique setting. Do crossovers until we have a maximum number of new settings.

3. Change some settings at random (*mutation*).

4. Run the queries using the new settings and evaluate (determine *fitness*).

5. Continue with 2 until "bored".

This setting is very simple and straightforward. However, some additional parameters of the algorithm have to be set initially. First of all, we have to decide how many IR runs ("individuals") we like to keep in our "population". We decided to keep only a very small set of 25 individuals. "Fitness" is measured using the MRR scores for finding the answer strings in retrieved documents. Selecting "parents" for the combination of settings is simply done by randomly selecting two of the 25 "living individuals". We compute the arithmetic mean of weights (or window sizes) if we encounter identical keyword types in both parents. We also have to set the number of new settings ("children") that should be produced at a time. We set this value to a maximum of 50. Selection according to the fitness scores is done immediately when a new IR run is finished.

Finally, we have to define *mutation operations* and their probability. Settings may be mutated by adding a keyword type (with a probability of 0.2), removing a keyword type (with a probability of 0.1), or by increasing/decreasing weights or window sizes (with a probability of 0.2). Window sizes are changed by a random value between 1 and 10 (shrinking or expanding) and weights are changed by a random real value between 0 and 5 (decreasing or increasing). The initial weight is 1 (which is also the default for Lucene) and the initial window size is 20.

The optimization parameters are chosen intuitively. Probabilities for mutations are set at rather high values to enforce quicker changes within the process. *Natural selection* is simplified to a top-N selection without giving individuals with lower fitness values a chance to survive. Experimentally, we found out that this improves the convergence of the optimization process compared to a probabilistic selection method. Note that there is no obvious condition for termination. A simple approach would be to stop if we cannot improve the fitness scores anymore. However, this condition is too strict and would cause the process to stop too early. We simply stop it after a certain number of runs especially when we encounter that the optimization levels out.

5 Experiments

For our experiments, we put together the CLEF questions from the last two years of the competition. From this we randomly selected a training set of 420 questions (and their answers) and an evaluation set of 150 questions with answers (heldout data). The main reason for merging both sets and not using one year's data for training and another year's data for evaluation is simply to avoid unwanted training/evaluation-set mismatches. Each year, the CLEF tasks are slightly different from previous years to avoid over-training on certain question types. By merging both sets and selecting at random we hope to create a more general training set with similar properties as the evaluation set.

For optimization, we used the algorithm as described in the previous section together with the multi-layer index and the full set of keyword types as listed earlier. IR was run in parallel (3-7 Linux workstations on a local network) and a top 15 list was printed after each 10 runs. For each setting we also compute the "fitness" of the test data to compare training scores to scores on heldout data. Table 3 summarizes the optimization process by means of MRR scores and compares it to the base line of using traditional IR with plain text keywords. The algorithm was stopped after 1000 different settings. Figure 3 plots the training curve for 1000 settings on a logarithmic scale (left plot). In addition, the curve of the corresponding evaluation scores is plotted in the right part of the figure. The thin lines in figure 3 refer to the scores of individual settings tested in the optimization process. The solid bold lines refer to the top scores using the optimized queries.

Both plots illustrate the nature of the iterative optimization process. Random modifications cause the oscillation of the fitness scores (see the thin black lines). However, the algorithm picks up the advantageous features and promotes them in

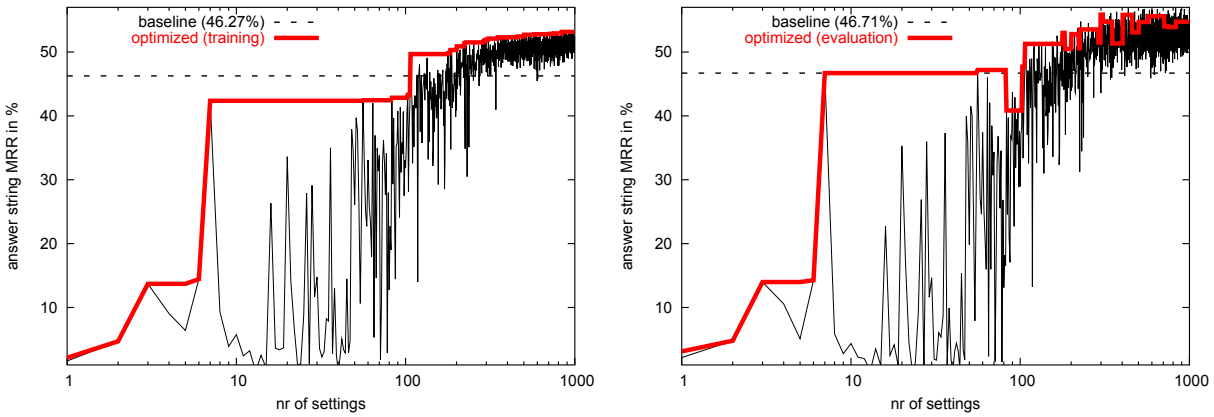


Figure 3: Parameter optimization. Training (left) and evaluation (right).

nr of settings	training	evaluation
baseline	46.27	46.71
10	42.36	46.70
150	49.69	51.28
250	51.51	53.55
450	52.32	55.82
600	52.79	55.62
1000	53.16	54.76

Table 3: Optimization of query parameters (MRR scores of answer strings). Baseline: IR with plain text tokens (+ stop word removal & Dutch stemming). All scores are in %

the competitive selection process. The scores after the optimization are well above the base line of using plain text tokens only (about 8% measured in MRR). Most of the improvements can be observed in the beginning of the optimization process[3] which is very common in machine learning approaches. The training curve levels out already after about 300 settings.

The two plots also illustrate the relation between scores in training and evaluation. There seems to be a strong correlation between training and evaluation data. The general tendency of evaluation scores is similar to the training curve with step-wise improvements throughout the optimization process even though the development of the evaluation score is not monotonic. Besides the drops in evaluation scores we can also observe a slight tendency for values to decrease after about 500 settings which is probably due to over-fitting. Note also that the evaluation scores for the optimized queries do not always reach the top scores among all tested individuals. However, the optimized queries are close to the best possible query according to the fitness scores measured on evaluation data.

Now, we are interested in the features that have been selected in the optimization process. Table 4 shows the settings of the best query after trying 1000 different settings. It also lists the total numbers of keywords that have been produced for the questions in the training set for each keyword type using these settings.

Eight token layers are used in the final query. It is somehow surprising that the named-entity layers are not used at all except the meta-layer that contains the named-entity labels (neTypes). However, features captured in these layers are also used in some of the query parts where the POS restriction is set to 'name'. This overlap probably makes it unnecessary to add named-entity keywords to the query.

Most keyword restrictions are applied to the root layer. The largest weight, however, is set to the plain text token layer. This seems to be reasonable when looking at the performance of the single layers (the text layer performs best, followed by the root-layer). The most popular constraints are 'nouns' and 'names' (among POS labels) and subject (su), direct object (obj1) among relation types. This also seems to be natural as noun phrases usually include the most informative part of a sentence.

Looking at the the proximity queries we can observe that the optimized query is quite strict with rather small windows. Many proximity queries use a window size below the initial setting of 20

[3]Note that the X scale in figure 3 is logarithmic for both, training and evaluation.

Table 4: Optimized query parameters after 1000 settings including the number of keywords produced for the training set using these parameters.

layer	restrictions POS	relation	nr of keywords	weight/window
text			1478	weight = 13.57
text			1478	window = 13
	verb		237	window = 9
	noun	mod	3	weight = 2.75
root			1570	weight = 1.66
root			1570	window = 12
	noun		520	weight = 1
		su	237	window = 20
	adj		125	window = 21
	name		88	window = 26
	noun	su	72	weight = 1
	name	obj1	54	window = 18
	name	su	20	weight = 4.21
RootPos			1209	weight = 1
RootPos			1209	window = 29
	noun		472	weight = 3.65
	noun	obj1	216	window = 36
	name		44	weight = 1
RootRel			1209	weight = 3.12
		obj1	413	weight = 1.76
		obj1	413	window = 18
RootHead			1209	weight = 1
		obj1	413	weight = 1
RootRelHead			1209	weight = 1
		obj1	413	weight = 2.55
	noun	su	60	weight = 5.19
	name	obj1	23	weight = 1
	name	obj1	23	window = 10
compound			208	weight = 2.52
neTypes			306	weight = 3.25

tokens. However, it is hard to judge the influence of the proximity queries and their parameters on the entire query where all parts are combined in a disjunctive way.

Altogether, the system makes extensive use of enriched index layers and also gives them significant weights (see for example the RootPos layer for nouns and the RootRelHead layer for noun subjects). They seem to contribute to the IR performance in a positive way.

6 Conclusions and future work

In this paper we describe the information retrieval component of our open-domain question answering system. We integrated linguistic features produced by a wide-coverage parser for Dutch, Alpino, in the IR index to improve the retrieval of relevant paragraphs. These features are stored in several index layers that can be queried by the QA system in various ways. We also use word class labels and syntactic relations to restrict keywords in queries. Furthermore, we use keyword weights and proximity queries in the retrieval system. In the paper, we demonstrate an iterative algorithm for optimizing query parameters to take advantage of the enriched IR index. Queries are optimized according to a training set of questions annotated with answers taken from the CLEF competitions on Dutch question answering. We could show that the performance of the IR component could be improved by about 8% on unseen evaluation data using the mean reciprocal rank of retrieved relevant paragraphs. We believe that this improvement also helps to boost the performance of the entire QA system. It will be part of future work to test the QA system with the adjusted IR component and the improved ranking of relevant passages. We also like to explore further techniques in integrating linguistic information in IR to optimize retrieval recall and precision even further.

References

(Bouma *et al.* 01) Gosse Bouma, Gertjan van Noord, and Robert Malouf. Alpino: Wide coverage computational analysis of Dutch. In *Computational Linguistics in the Netherlands CLIN, 2000.* Rodopi, 2001.

(Bouma *et al.* 05) Gosse Bouma, Jori Mur, and Gertjan van Noord. Reasoning over dependency relations for QA. In *Knowledge and Reasoning for Answering Questions (KRAQ'05)*, IJCAI Workshop, Edinburgh, Scotland, 2005.

(Fagan 87) Joel L. Fagan. Automatic phrase indexing for document retrieval. In *SIGIR '87: Proceedings of the 10th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 91–101, New York, NY, USA, 1987. ACM Press.

(Jakarta 04) Apache Jakarta. Apache Lucene - a high-performance, full-featured text search engine library. http://lucene.apache.org/java/docs/index.html, 2004.

(Jijkoun *et al.* 04) Valentin Jijkoun, Jori Mur, and Maarten de Rijke. Information extraction for question answering: Improving recall through syntactic patterns. In *Proceedings of COLING-2004*, 2004.

(Katz & Lin 03) Boris Katz and Jimmy Lin. Selectively using relations to improve precision in question answering. In *Proceedings of the EACL-2003 Workshop on Natural Language Processing for Question Answering*, 2003.

(Moldovan *et al.* 02) Dan Moldovan, Sanda Harabagiu, Roxana Girju, Paul Morarescu, Finley Lacatusu, Adrian Novischi, Adriana Badulescu, and Orest Bolohan. LCC tools for question answering. In *Proceedings of TREC-11*, 2002.

(Neumann & Sacaleanu 04) Günter Neumann and Bogdan Sacaleanu. Experiments on robust NL question interpretation and multi-layered document annotation for a cross-language question/answering system. In *Proceedings of the CLEF 2004 working notes of the QA@CLEF*, Bath, 2004.

(Prager *et al.* 00) John Prager, Eric Brown, Anni Cohen, Dragomir Radev, and Valerie Samn. Question-answering by predictive annotation. In *In Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Athens, Greece, July 2000.

(Strzalkowski *et al.* 96) Tomek Strzalkowski, Louise Guthrie, Jussi Karlgren, Jim Leistensnider, Fang Lin, Jos Prez-Carballo, Troy Straszheim, Jin Wang, and Jon Wilding. Natural language information retrieval: TREC-5 report, 1996.

(Zhai 97) Chengxiang Zhai. Fast statistical parsing of noun phrases for document indexing. In *Proceedings of the fifth conference on Applied natural language processing*, pages 312–319, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

Weak Named Entities Recognition using morphology and syntax *

Antonio Toral Rafael Muñoz and **Andrés Montoyo**
Departamento de Lenguajes y Sistemas Informáticos
University of Alicante
Carretera San Vicente S/N
Alicante 03690, Spain
{atoral,rafael,montoyo}@dlsi.ua.es

Abstract

A lot of research has been done regarding strong named entities recognition, both following the rule-based and the learning approaches. However, weak named entities have been not treated in detail yet. We propose a system for the detection and classification of these kind of entities that uses morphology (PoS) and syntax (shallow parsing) features. Our starting point is the text correctly tagged with strong entities. The results are encouraging as we obtain a F-score better than 60% without learning (corpus) nor knowledge (gazetteers or grammars) resources.

1 Introduction

Named Entity Recognition (NER) was defined at the MUC conferences (Chinchor 98) as the task consisting of detecting and classifying strings of text that refer to people, locations, organizations, dates, time, quantities, etc. NER can be understood then as a classification task in which each token of a text is to be classified into a set of categories: the different types of entities considered plus not an entity.

Although NER was initially considered just as a subtask of Information Extraction - one of the major tasks of Natural Language Processing (NLP) -, nowadays there is a general consensus about that the application of NER to other NLP tasks may improve their results. For example, NER has been succesfully applied to tasks such as Question Answering or Machine Translation to mention just a few.

Named entities could be divided in two groups according to their complexity: strong and weak entities. Strong entities are the simpler ones and basically consist of proper nouns. On the other hand, weak entities are more complex and in the simpler case are made up of a trigger word and a proper noun. An example of strong entity is "Portugal" while an example of weak entity could be "the president of Portugal" which has a trigger word (president) and a proper noun (Portugal).

Until the present moment, NER research has focused in strong entities, both using knowledge and learning approaches. The results for this kind of entities have reached more than acceptable values (last systems are around 80% for classification and around 90% for identification (Carreras *et al.* 02)). Therefore, we think it is the time to focus on weak entities.

In this work we propose a system to deal with the identification and classification of weak entities. Our approach is based on the morphologic and syntactic features of weak entities. We made a prior study about the structure of this kind of entities and found out some characteristics that we suggest may help for their identification and classification.

The paper is organized as follows. In Section two the background on the elements used in our system is briefly pointed out. In the third section our system is outlined. Fourth section presents the evaluation of our system. Finally, conclusions and future work proposals are treated in section five.

2 Background

The state of the art about the different elements that are used in our system is described in this section. These are Part-of-speech (PoS) tagging, shallow parsing and NER.

2.1 PoS

PoS tagging may be defined as the task that consists of classifying the words of a text according to their PoS and morphosyntactic features such as gender, number and so on. Therefore, these taggers are applied at a low level of language (morphologic) and provide important information for subsequent NLP processing (i.e. syntactic and semantic levels).

Being a basic tool for NLP, there has been a

* This research has been partially funded by the Spanish Government under project CICyT number TIC2003-07158-C04-01 and by the Valencia Government under project numbers GV04B-276 and GV04B-268.

lot of research in this area during the last years. Because of this, nowadays we have mature PoS-taggers like the one included in Freeling (Carreras *et al.* 04) for the most used natural languages.

2.2 Shallow parsing

Shallow parsing is an alternative to full-sentence parsers. Instead of producing a complete analysis of sentences, it performs only partial analysis of the syntactic structures in a text. The motivation for doing this is that the results obtained are much better for shallow parsing than for complete parsing. Besides, the information provided by a shallow parser, although may seem to be very simple, has proved to be valuable for NLP tasks.

2.3 NER

Research in NER started in the ninetees during the MUC conferences. At that time most research was done in the knowledge based approach. NER systems using basically rules and gazetteers were built and integrated into larger Information Extraction systems. After the last MUC conference, research started to focus in the learning paradigm. Hence, today there are several systems which obtain satisfactory results belonging to both approaches.

Regarding weak entities, not that much research has been carried out already. However, a research carried out at the University of Barcelona should be emphasized. They have developed a Module for Named Entities Recognition and Classification called MICE (Arévalo Rodríguez *et al.* 04). This system recognizes both strong and weak entities. The first are solved using Machine Learning while the later are treated with hand-made syntactic patterns.

The important matter about this work is that it introduces several aspects about weak entities that provide valuable information for the research in this field. This way, the authors provide two classifications of weak entities, both according to syntax and to semantics. The authors differentiate three types of weak entities according to their semantics:

- Core NEs, which have a trigger word belonging to the gazetteers.
- Related NEs, which have a hyponim, hypernym or synonym or a trigger word from the gazetteers.
- General NEs, any noun phrase.

With reference to syntax, they distinguish two kinds of weak NEs:

- Syntactically simple weak NEs. Made up of a single noun phrase (i.e. "the president of Portugal") or very simple cases of coordination (i.e. "the representants of Portugal and France").
- Syntactically complex weak NEs. Formed by complex noun phrases including plurals, anaphora, etc.

MICE recognizes and classifies core and related weak NEs which are syntactically simple using a context free grammar enriched with the semantic information of the trigger words. Finally, they do not provide a evaluation based on effectivenes but a qualitative one.

3 Method

One central aim of this research is to treat weak entities without applying a hand-made grammar. The motivation for applying this approach comes from the study of the structure of syntactic syntagmas. After doing that we concluded that their structure most of the times concides with the structure of weak entities. Because of this fact we have focused on the treatment of core and syntactically simple weak NEs.

Our system recognizes and classifies weak named entities using PoS, syntactic features and strong NEs information. These characteristics, but the strong NEs, are provided by a PoS tagger and a shallow parser respectively, both included in the Freeling software package. With this information, the core NER system (a modified version of DRAMNERI (Toral 05) enriched with morphosyntactic capabilities) performs this task.

The input to our NER module is text where each word is tagged with its PoS, the syntactic syntagma that the word belongs to and its strong named entity category. An example of input fragment would be:

```
el el DA0MS0 sn 4 O
alcalde alcalde NCMS000 sn 4 O
de de SPS00 sn 4 O
Arevalo Arevalo NP00000 sn 4 B-LOC
, , Fc !! 5 O
Francisco Francisco NP00000 sn 6 B-PER
Leon Leon NP00000 sn 6 I-PER
```

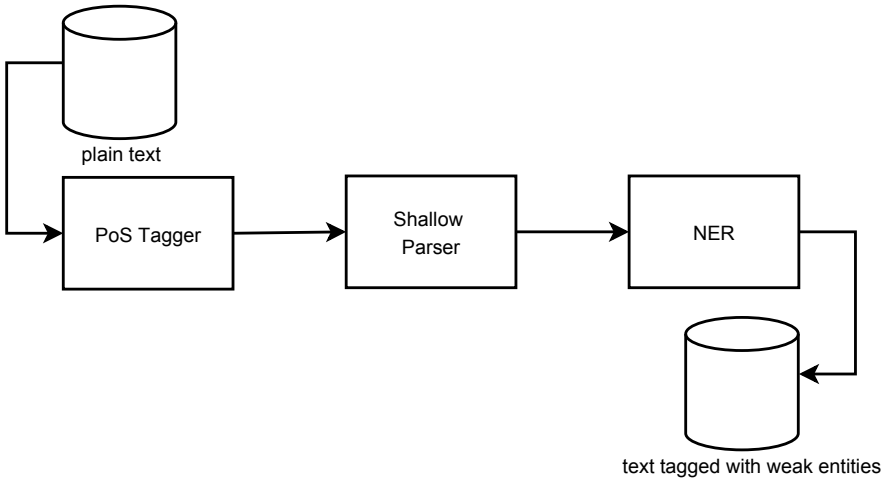


Figure 1: System architecture

The first column indicates the word, the second its lemma, the third its part of speech, the fourth the type of the syntagma it belongs to, the fifth the number of the syntagma within the sentence and finally the sixth the strong named entity category.

With this information the system performs basically two tasks. Firsty, it recognizes syntagmas being weak entities. Secondly, from the extracted syntagmas in the previous step it applies a set of heuristics to smooth their boundaries.

In the first step we study the syntagmas in which a trigger is found. For this search we use the information provided by the PoS tagger. For every common noun the lemma is searched against the trigger gazetteers. If is found in any gazetteer then we look for a strong entity. If any strong entity is found within the syntagma, then we consider it a weak entity and so it will be treated in the next step. Sytagmas in which a trigger is present but no strong entity is found will be also considered in the following step.

In the second step some heuristics are performed to the weak entities recognized in the first step in order to smooth their boundaries. These are the following:

- In some cases the syntagma is a prepositional one, and thus it is introduced by a preposition, which is not part of the weak entity. Therefore, if the syntagma is prepositional we take away the first tokens being prepositions. As an example, the string: '<E class="PER" type="WEAK"> de la presidenta de <E class="ORG" type="STRONG"> Torrecomercios </ENT> </ENT>' would be converted to 'de <E class="PER" type="WEAK">la presidenta de <E class="ORG" type="STRONG"> Torrecomercios </ENT> </ENT>' as 'de' is a simple preoposition in Spanish.

- Due to mistakes made by the shallow parser, often a syntagma corresponding to a weak entity is splitted in two, one containing the trigger word and the other containing a strong entity. Thus, if no strong entity is found in a syntagma where a trigger is present, then we look for a strong entity in the following one. Consider as an example the following input:

```
  el el DA0MS0 sn 3 O
  internacional internacional
AQ0CS0 sn 3 O
  Frank Frank NP00000 sn 4 B-PER
  Farina Farina NP00000 sn 4 I-PER
```

  This string of text should be a syntagma that moreover corresponds to a weak entity having a trigger word (internacional) and a strong entity (Frank Farina). However, the shallow parser has identified it as two different syntagmas, one having the trigger and the other one the strong entity.

4 Evaluation

In order to evaluate our system we have used the Spanish test data which is part of the CoNLL-2002 corpus (Conll-2002 02). We have not used the train part of the corpus as our system does

	Precision	Recall	F
NER	42.00	33.87	37.50
NER+p	66.00	53.23	58.93
NER+s	44.83	41.94	43.33
NER+p+s	65.52	61.29	63.33

Table 1: Results

not need to carry out any learning preprocess. Besides, this corpus was originally tagged with strong entities. Thus, we tagged the test data for Spanish with weak entities. The type of weak entities considered are the same that this corpus had initially tagged as strong ones: person, location, organization and miscellaneous.

We have made several experiments, applying different smooth heuristics from the described in the previous section. This way, in the first experiment (NER) no smooth is applied, in the second (NER+p) the introducing prepositions are treated, in the third (NER+s) syntagmas are jointed and in the fourth (NER+p+s) both smooth processes are applied.

The results for all these experiments can be viewed in table 1. For each experiment several measures are presented: precision, recall and F-value.

From the results obtained, we found that the simple heuristics that have been applied increase the performance notably. Specially the treatmens of prepositions, which in comparison to the plain model obtained an increment of 24% regarding precision and 20% about recall. The model that applies both heuristics obtained the best results, but for the precision obtains a slightly worst result (-0.48%) than the model that only applies prepositions treatment.

5 Conclusions and future work

In this article, a system that deals with weak named entities using morphologic and syntactic features has been presented. In general our conclusions are positive because we have developed a novel approach for a task that has not yet been treated into detail, our system is simple and thus efficient and still obtains satisfactory results. It should be emphasized also that the results were increased notably by applying simple heuristics.

There are some aspects which we think could improve the performance of this system. Thus, here are introduced and proposed as future lines of research.

The first is about PoS. For this research we have used a general PoS tagger which obtains state-of-the-art results. However, we only need to distinguish between three categories: common noun, simple preposition and other. Due to the fact that as the number of categories decreases the result is better, we think that using a customized PoS tagger could increase the general performance of the system as the mistakes made by the PoS tagger will decrease.

Secondly, in this work we have just considered the first level syntagmas but in some cases the shallow parser provides more levels of syntagmas. Our system could benefit from the greater detail these syntagmas provide. Specially, taking into account these syntagmas would help to the detection of the real boundaries of the weak entities.

Finally, it would be interesting to build a complete system for NEs (both strong and weak) by using a state-of-the-art system for the strong ones and the present system for the weak entities. This way, we could study how well the later performs when applying it in a real scenario. We could therefore know how the performance obtained by the strong NEs system afects the performance of the weak NEs system.

References

(Arévalo Rodríguez *et al.* 04) Montserrat Arévalo Rodríguez, Montserrat Civit Torruella, and Maria Antónia Martí Antonín. MICE. A module for Named Entities Recognition and Classification. *International Journal of Corpus Linguistics*, 9(1):53–68, 2004.

(Carreras *et al.* 02) Xavier Carreras, Lluís Márques, and Lluís and Padró. Named Entity Extraction using Adaboost. In *Proceedings of CoNLL-2002*, pages 167–170, 2002.

(Carreras *et al.* 04) X. Carreras, I. Chao, L. Padró, and M. Padró. Freeling: An Open-Source Suite of Language Analyzers. In *Proceedings of the 4th LREC Conference*, 2004.

(Chinchor 98) N. Chinchor. Overview of MUC-7. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, 1998.

(Conll-2002 02) Conll-2002. Language-Independent Named Entity Recognition in Conll-2002. 2002.

(Toral 05) Antonio Toral. DRAMNERI: a free knowledge based tool to Named Entity Recognition. In *Proceedings of the 1st Free Software Technologies Conference*, pages 27–32, 2005.

EAGLES compliant tagset for the morphosyntactic tagging of Esperanto *

Antonio Toral Sergio Ferrández and **Andrés Montoyo**
Departamento de Lenguajes y Sistemas Informáticos
University of Alicante
Carretera San Vicente S/N, Alicante 03690, Spain
{atoral,sferrandez,montoyo}@dlsi.ua.es

Abstract

This paper presents the first stage of a research related to automatic morphosyntactic annotation in Esperanto. We present and justify a tagset which fulfils the EAGLES standard. This standard allows us to map our tagset with the tagset developed for other languages. In future studies, an automatic tagger and a corpus will be developed using the proposed tagset.

1 Introduction

Esperanto is an international language that was created by Dr. Zamenhoff at the end of the 19th century. Its main purpose is to provide a neutral tool that allows the different countries of the world to communicate between each other on equal terms. Because of its characteristics (easy learning, neutrality, and so on), UNESCO has adopted two resolutions in favour of this language (1954 and 1985) in which it calls on member states and international organizations to promote the teaching of Esperanto in schools and its use in international affairs.

This paper presents the first part of our work regarding the automatic morphosyntactic annotation for Esperanto. Thus, we have developed a set of tags that will allow us to classify words. This study is to be continued with the development of automatic annotators that will follow the classification presented here.

The features of Esperanto regarding word formation facilitate its automatic morphosyntactic annotation. For example, the word endings tell the part-of-speech (PoS) in most of the cases. This way, the word ending $-o$ tells that the word is a noun, $-a$ is the ending for an adjective and so on. Besides, the absence of exceptions at the morphologic level reduces the ambiguity problems for annotation.

* This research is part of a project which is being evaluated by the Esperantic Studies Foundation in order to provide funding. We thank Carlo Minnaja, Pau Climent and Hèctor Alòs for their help.

Tagging is the task that consists of classifying the words in a natural language text with respect to a given criterion. Different kinds of tagging can be distinguished according to the criterion employed: syntactic word class, word sense, syntactic parsing, etc.

The aim of a PoS tagger is to assign to each of the words in a text a tag belonging to a tagset that has been previously defined. The PoS tagger should take into account the context of the word to perform the classification.

PoS tagging is an important processing step for most of the Natural Language Processing tasks, such as Information Extraction, Question Answering, Machine Translation, etc. On the other hand, PoS tagging is also used to generate annotated corpora using human supervision.

The tagset presented in this paper has been developed following the EAGLES standard (Leech & Wilson 99) for morphosyntactic annotation. This standard was initially developed in order to provide morphosyntactic annotation guides for the languages of the European Union.

This paper is organized as follows: in the next section the state of the art about PoS tagging, the EAGLES standard and previous works about PoS tagging in Esperanto are described. Section three explains and discuss the developed tagset. Finally, conclusions and future work proposals are presented in section four.

2 State of the art

The standard EAGLES provides a set of features for annotation, being some of them mandatory, some others recommended and the rest optional. These characteristics are codified in an intermediate tagset using numerical values that show the different attributes that this standard provides. The way this numerical values are transformed to final morphosyntactic tags for a given language is left to the user. The only condition is that a matching algorithm can be established to link

each final tagset with the intermediate provided.

The purpose of the intermediate tagset is to allow to establish automatic relations among different final tagsets using this standard. This feature is one of the most important elements of the EAGLES standard.

EAGLES tags are defined as a set of morphosyntactic attributes (for example, *Number* is an attribute that can take the values *Singular* or *Plural*). These attributes are sorted in a hierarchical structure in the intermediate tagset.

PoS of words is the mandatory characteristic that any tagset must fulfil in order to be compatible with the EAGLES standard. The guides of annotation of this standard suggest thirteen different types of PoS (*noun, verb, adjective, etc*). The recommended and optional attributes are different depending on the PoS of the word. For example, the first recommended attribute for verbs (*verb*) is *Person* which can have different values: *First*, *Second* and *Third*.

Our decision to use the EAGLES standard is due mainly to two reasons. Firstly, it is a flexible and broadly used standard. Secondly, the attributes proposed in this standard allow its application to Esperanto, as they are able to represent the characteristics of this language.

Regarding PoS automatic annotation, different approaches have been suggested to solve this problem. The most important have been those based in knowledge (they use a set of explicit rules) (Brill 93) and those based in learning algorithms (Brants 00; Padró & Padró 04).

The precision achieved by these systems is around 95% for the systems using learning algorithms (for example, Freeling (Carreras *et al.* 04) for Spanish and TreeTagger for English (Schmid 95)). On the other hand, implementations based in knowledge (for example, MACO (Atserias *et al.* 98) for Spanish) have a precision around 97%.

Regarding PoS automatic annotation in Esperanto, the research carried out by Minnaja and Paccagnella (Minnaja & Paccagnella 00) should be emphasized. These authors developed a custom tagset for PoS annotation (therefore not based in any standard) and a PoS tagger. Their paper does not clarify the approach used by their annotator. With respect to its precision, they claim that it is over 99%, but no details about the corpus (size, generality) are given.

Our research is related to this one and will study in depth the following aspects:

- Develop a tagset following the EAGLES standard. This will allow to make future linkings with other languages for which a tagset following this standard has been developed.
- Develop an automatic tagger.
- Develop a corpus in a supervised way by using the previously developed annotator. This corpus will be used to evaluate our system and will be made publically available.

This is the first paper to carry out these objectives and covers the first aspect. The three other tasks will be treated in future works.

3 Tagset

As it has been commented, this article presents and discuss the elaboration of a tagset for the PoS tagging of Esperanto following the EAGLES standard. The tagset developed is attached in the appendix 1. The tags included are explained and discussed in this section.

The mandatory attributes for EAGLES are the parts of speech. The thirteen categories provided suit the parts of speech defined in the Esperanto language.

The recommended attributes consist of the common grammatical descriptions (i.e. gender, number) for each of the categories defined as mandatory. Just a subset of the recommended attributes has been necessary to represent these aspects for Esperanto.

- EAGLES considers several values for the attribute case. Because Esperanto only haves the accusative case we only have needed to consider this value.
- Gender and number attributes are not necessary for verb or adjective, because these variations are only present in nouns.
- Equally, the attribute person is not considered for the verb, as the verb is invariable according to the person value.
- We neither need the attribute grade for the adjectives, because the comparative and superlative are compounded by combining the positive adjective with particles which indicate the grade.

- Regarding pronouns, we only need to consider two types (personals and posesives). The impersonal form has been considered as belonging to the third singular person. Reflexive pronouns are considered as normal pronouns.

- Due to the fact that Esperanto has only one article (*la*) none of the attributes that EAGLES suggests for this PoS are used.

Finally, EAGLES provides special extensions with attributes that generally are not included when developing a tagset but could be interesting for non general tagsets. EAGLES also provides specific attributes for each of the languages that it was initially created for. None of these attributes is useful for us.

However, Esperanto has a special attribute that is not present in any of the languages that EAGLES was initially created for. Therefore, no attribute is included in this standard to deal with this fact. This attribute indicates direction in adverbs. The following example describes it:

La hundo kuras tie (The dog runs there)
La hundo kuras tien (The dog runs towards there (not being there at the beginning of the action))

Therefore, we have added tags to take into account this fact. Nevertheless, in order to maintain compatibility with the standard, the intermediate tagset (the one used to make mappings to the tagsets of other languages) does not consider this feature.

4 Conclusions and future research

In this paper we have presented a standard compliant tagset for the morphosyntactic annotation of texts in Esperanto. This will be useful as a starting point to build natural language processing systems for this language.

We can conclude that the EAGLES standard provides a hierarchical organized set of attributes that can represent well enough the morphosyntactic features of Esperanto. The only exception has been the direction adverbs, a specific feature of this language not considered in this standard.

Due to the morphosyntactic features of Esperanto, it has been possible to develop a tagset of small size (86 tags) comparing it to tagsets developed for other languages (i.e. 114 tags for English (Leech & Wilson 99), 274 for Italian (Leech & Wilson 99) or 280 for Urdu (Hardie 03)). This will have an important impact for the later computational treatment, making it simpler and making it likely to obtain better results, both regarding effectiveness and efficiency.

Our future works will focus on the development of a system of PoS tagging for Esperanto. In order to do that, a PoS tagger will be built. Besides, a supervised corpus will be developed. This corpus will be a valuable knowledge base for the development of NLP systems for Esperanto.

References

(Atserias *et al.* 98) J. Atserias, J. Carmona, I. Castellón, S. Cervell, M. Civit, L. Márquez, M.A. Martí, L. Padró, R.Placer, H. Rodríguez, M. Taulé, and J. Turmo. Morphosyntactic Analysis and Parsing of Unrestricted Spanish Text. *First International Conference on Language Resources and Evaluation, LREC'98*, pages 1267–1272, 1998.

(Brants 00) T. Brants. Tnt- a statistical part-of-speech tagger. *Proceedings of the 6rd Conference on Applied Natural Language Procesing, ANLP*, pages 224 – 231, 2000.

(Brill 93) E. Brill. A corpus-based Approach to Language Learning. 1993.

(Carreras *et al.* 04) X. Carreras, I. Chao, L. Padró, and M. Padró. Freeling: An open-source suite of language analyzers. *Proceedings of the 4th International Conference on Languge Resources and Evaluation, LREC'04*, pages 1364 – 1371, 2004.

(Hardie 03) A. Hardie. Developing a tagset for automated part-of-speech tagging in Urdu. *Proceedings of the Corpus Linguistics 2003 conference*, 16, 2003.

(Leech & Wilson 99) G. Leech and A. Wilson. Recommendations for the Morphosyntactic Annotation of Corpora. *EAGLES Report EAG-TCWG-MAC/R*, 1999.

(Minnaja & Paccagnella 00) C. Minnaja and L. Paccagnella. A Part-of-Speech Tagger for Esperanto oriented to MT. *International Conference MT 2000 - Machine Translation and multilingual Applications in the new Millennium*, pages 13.1–13.5, 2000.

(Padró & Padró 04) M. Padró and L. Padró. Developing Competitive HMM PoS Taggers Using Small Training Corpora. *ESPAÑA for NATURAL LANGUAGE PROCESSING, EsTAL*, pages 127 – 136, 2004.

(Schmid 95) H. Schmid. TreeTagger — a language independent part-of-speech tagger. *Institut fur Maschinelle Sprachverarbeitung, Universitat Stuttgart*, 1995.

A Appendix 1: Tagset

This appendix includes a table with the tagset designed. For each tag we include the final tag code, the tag description, an example word and the intermediate tag code.

Tag	Description	Example(s)	Intermediate tag
NCMS	Noun common singular masculine	knabo	N1110
NCMSA	Noun common singular masculine accusative	knabon	N1114
NCMP	Noun common plural masculine	knaboj	N1120
NCMPA	Noun common plural masculine accusative	knabojn	N1124
NCFS	Noun common singular femenine	knabino	N1210
NCFSA	Noun common singular femenine accusative	knabinon	N1214
NCFP	Noun common plural femenine	knabinoj	N1220
NCFPA	Noun common plural femenine accusative	knabinojn	N1224
NCNS	Noun common singular neuter	domo	N1310
NCNSA	Noun common singular neuter accusative	domon	N1314
NCNP	Noun common plural neuter	domoj	N1320
NCNPA	Noun common plural neuter accusative	domojn	N1324
NP	Noun proper	Karlo	N2000
NPA	Noun proper accusative	Karlon	N2004
VP	Verb indicative present	amas	V00001100
VF	Verb indicative future	amos	V00001300
VPA	Verb indicative past	amis	V00001400
VIM	Verb imperative	amu	V00003000
VC	Verb conditional	amus	V00004000
VIN	Verb infinitive	ami	V00005000
VPTPA	Verb participle present active	amanta	V00006110
VPTFA	Verb participle future active	amonta	V00006310
VPTPAA	Verb participle past active	aminta	V00006410
VPTPP	Verb participle present passive	amata	V00006120
VPTFP	Verb participle future passive	amota	V00006320
VPTPAP	Verb participle past passive	amita	V00006420
VGPA	Verb gerund present active	amante	V00007110
VGFA	Verb gerund future active	amonte	V00007310
VGPAA	Verb gerund past active	aminte	V00007410
VGPP	Verb gerund present passive	amate	V00007120
VGFP	Verb gerund future passive	amote	V00007320
VGPAP	Verb gerund past passive	amite	V00007420
AJS	Adjetive singular	bela	AJ0010
AJSA	Adjetive singular accusative	belan	AJ0014
AJP	Adjetive plural	belaj	AJ0020
AJPA	Adjetive plural accusative	belajn	AJ0024
AT	Article	la	AT0000
AV	Adverb	tie	AV0
AVD	Adverb direction	tien	AV0
AP	Preposition	kun	AP1
CC	Conjunction coordinating	kaj	C1
CS	Conjunction subordinating	kvankam	C2
NUMC	Numeral cardinal	unu	N10000
NUMCA	Numeral ordinal	unua	N20000
NUMO	Numeral cardinal accusative	unun	N10040
NUMOA	Numeral ordinal accusative	unuan	N20040

Tag	Description	Example(s)	Intermediate tag
PDP1S	Pronoun personal 1st pers. singular	mi	PD10100150
PDP2S	Pronoun personal 2nd pers. singular	vi	PD20100150
PDP3SM	Pronoun personal 3rd pers. sing. masc.	li	PD31100150
PDP3SF	Pronoun personal 3rd pers. sing. fem.	si	PD32100150
PDP3SN	Pronoun personal 3rd pers. sing. neuter	gxi	PD33100150
PDP1P	Pronoun personal 1st pers. plural	ni	PD10200150
PDP2P	Pronoun personal 2nd pers. plural	vi	PD20200150
PDP3P	Pronoun personal 3rd pers. plural	ili	PD30200150
PDP1SA	Pron. pers. 1st pers. singular accusative	min	PD10104150
PDP2SA	Pron. pers. 2nd pers. singular accusative	vin	PD20104150
PDP3SMA	Pron. pers. 3rd pers. sing. masc. accusative	lin	PD31104150
PDP3SFA	Pron. pers. 3rd pers. sing. fem. accusative	sin	PD32104150
PDP3SNA	Pron. pers. 3rd pers. sing. neuter accusative	gxin	PD33104150
PDP1PA	Pronoun pers. 1st pers. plural accusative	nin	PD10204150
PDP2PA	Pronoun pers. 2nd pers. plural accusative	vin	PD20204150
PDP3PA	Pronoun pers. 3rd pers. plural accusative	ilin	PD30204150
PDPO1S	Pronoun posesive 1st pers. singular	mia	PD10100130
PDPO2S	Pronoun posesive 2nd pers. singular	via	PD20100130
PDPO3SM	Pronoun posesive 3rd pers. sing. masc.	lia	PD31100130
PDPO3SF	Pronoun posesive 3rd pers. sing. fem.	sia	PD32100130
PDPO3SN	Pronoun posesive 3rd pers. sing. neuter	gxia	PD33100130
PDPO1P	Pronoun posesive 1st pers. plural	nia	PD10200130
PDPO2P	Pronoun posesive 2nd pers. plural	via	PD20200130
PDPO3P	Pronoun posesive 3rd pers. plural	ilia	PD30200130
PDPO1SA	Pos. pron. 1st pers. singular accusative	mian	PD10104130
PDPO2SA	Pos. pron. 2nd pers. singular accusative	vian	PD20104130
PDPO3SMA	Pos. pron. 3rd pers. sing. masc. accusative	lian	PD31104130
PDPO3SFA	Pos. pron. 3rd pers. sing. fem. accusative	sian	PD32104130
PDPO3SNA	Pos. pron. 3rd pers. sing. neuter accusative	gxian	PD33104130
PDPO1PA	Pos. pron. 1st pers. plural accusative	nian	PD10204130
PDPO2PA	Pos. pron. 2nd pers. plural accusative	vian	PD20204130
PDPO3PA	Pos. pron. 3rd pers. plural accusative	ilian	PD30204130
I	Interjection	aj	I
U	Particles	ne, cxu	U
RFW	Foreign words	show	R100
RSY	Symbols	$	R300
PUE	Punctuation sentence-final	., ?, !	PU1
PUB	Punctuation sentence-medial	,, ;, :, -	PU2
PUL	Punctuation left-parentheical	(, {, [	PU3
PUR	Punctuation right-parentheical	), },]	PU4

Table 1: Tagset

Automatic Acquisition of Expressions Representing Preparation and Utilization of an Object

Kentaro Torisawa
Japan Advanced Institute of Science and Technology
1-1 Asahidai, Nomi-shi, Ishikawa-ken, 923-1211 JAPAN
torisawa@jaist.ac.jp

Abstract

This paper proposes an automatic acquisition method for *preparation roles (PRs)* and *utilization roles (URs)*, which are analogues of agentive and telic roles in Generative Lexicon Theory. URs roughly express the purpose and function of a given *object*, and are defined as paraphrases of expressions such as "using an *object*" or "enjoying an *object*." A PR of an object is defined as an expression referring to a part of the preparation process to achieve the URs of an object. We regard "reading a book" as a UR of the book, and regard "buying a book" or "opening a book" as PRs. We expect that the acquired roles are useful in various inferences, such as plan recognition, by intelligent agents.

1 Introduction

The objective of this work is to automatically acquire particular semantic relations between verbs, which we call *preparation-utilization pairs (PUPs)*, for Japanese. They are defined for a noun, and are regarded as analogues of agentive roles and telic roles in Generative Lexicon Theory (Pustejovsky 95). For instance, the pair of "writing a book" and "reading a book" is a PUP for the noun "book." We refer to the first item ("writing a book") in the pair as the *preparation role (PR)* and to the second ("reading a book") as the *utilization role (UR)*.

The difference between PUPs and agentive-telic pairs is I) that URs are defined in terms of paraphrases while telic roles are simply defined as the purpose and function of a given object, and II) that PRs contain a wider range of expressions than agentive roles. We define URs of X as *principal* and *direct* paraphrases of the expressions "using X" or "enjoying X." By *principal* paraphrases of "using X" or "enjoying X," we mean the paraphrases that include only the expressions referring to the *normal* and *usual* manners of using or enjoying X. In other words, the expressions such as "beating someone with a book" or "presenting a book to someone" are not included in the principal paraphrases of "using the book." In addition, by *direct* paraphrases of "using X" or "enjoying X," we mean the expressions that *directly* refer to the event of "using X" or "enjoying X." Note that it may be possible to say that "buying beer" is a paraphrase of "enjoying beer" because buying beer is usually followed by enjoying it. But we do not include such a paraphrase to a *direct paraphrase* because buying beer and enjoying beer directly refer to two distinct events. We adopted this definition in the hope of reducing the ambiguity inherent in the definition of telic roles, i.e., the purpose and function of a given object.

A PR of X is defined as an expression referring to a part of the preparation process to achieve a UR of X. For instance, the PRs of "a book" can include "buying it," "writing it," and "publishing it." They can be seen as a part of the preparation process for "reading the book." (Note that we do not demand that a PR and a UR in a PUP must be *executed* by the same person.) An important point is that PRs can contain a significantly wider range of expressions than agentive roles, which are the expressions referring to the origin of an entity, or its coming into being. For instance, "writing a book" might be of both agentive and a PR for the book. But the PR "buying a book" would be difficult to be regarded as an agentive role of the book. Our contention is that PRs can be more useful in NLP tasks than more restrictive agentive roles. By consulting PUPs, NLP systems can guess that not only a writing-a-book event (agentive) but also a buying-a-book event (PR) have happened if a reading-a-book event (telic/UR) occurs.

Our motivation to acquire PUPs is mainly to enable plan recognition (Carberry 90). Consider an agent that can guide users through the WWW. When a user gives a name of a book to the agent, it finds the pages referring to the book. If the agent knows PUPs, it may be able to classify the pages according to the PRs and the URs of the book. The pages should be categorized into the pages about the "writing" process of the book, the pages where one can "buy" the book, and the pages about reader impressions after "reading" the book. Accordingly, the user may be able to buy the book after reading the impressions of other readers just by clicking the links and without typing "review" or "buy." Note that such a process can be seen as recognition of a plan related to books.

As a similar work, Inui proposed a method to extract *means* relations (Inui 04). There have also been attempts to capture typical temporal orders between events (Fujiki *et al.* 03; Chklovski & Pantel 04). In addition, there are numerous researches on acquisition of telic and agentive roles (Pustejovsky *et al.* 93; Lapata & Lasacarides 03; Boni & Manandhar 02; Yamada & Baldwin 04). Note that these methods for acquisition of telic and agentive roles were designed for languages other than Japanese and we could not compare our method with most of them directly. But as a basic approach for UR acquisition, we take an approach similar to that of Lapata and Lascarides's.

2 Acquisition Method for PUPs

Our method consists of the following steps.

Step 1 Unsupervised acquisition of URs.

Step 2 Acquisition of PUPs including the URs of nouns acquired in Step 1. This is done by supervised learning with a support vector machine (SVM) (Vapnik 98).

As mentioned, our algorithm was designed for Japanese. The major difference from English to be stressed here is that, unlike English, the position of an NP does not play a large role in determining a semantic role of an NP in Japanese. Instead, *postpositions*, which correspond to prepositions in English, mark the NPs and plays a crucial role in determining their semantic roles. Consider the following examples. Note that a postposition marks the NP in its left adjacent position and P stands for a postposition.

- `Tom` `ga` `beer` `wo` `nomu.`
  noun/Tom P noun/beer P verb/drink
  (Tom drinks beer.)

- `beer` `wo` `Tom` `ga` `nomu.`
  noun/beer P noun/Tom P verb/drink
  (Tom drinks beer.)

In spite of the change of word order, the meanings of the two sentences are unchanged since the postpositions `ga`, which specify the agent of the event generally, and `wo`, which mark themes in the events, respectively mark `Tom` and `beer` in both sentences.

In the following, we call a pair $\langle p, v\rangle$ an *argument position* where p is a postposition and v is a verb, and assume that a PR (and a UR) of a noun n can be expressed by an argument position such that if n occupies the position, the resulting natural language expression represents a PR (and a UR). For instance, consider the argument position $\langle$`wo`(postposition), `nomu`(drink)$\rangle$. If "beer" occupies the position, then the resulting Japanese expression "`beer wo`(postposition) `nomu`(drink)," which is translated into "drinking beer," expresses a UR of "beer." Then the UR is denoted as an the argument position $\langle$`wo`, `nomu`($drink$)$\rangle$.

2.1 Step 1: Acquisition of URs

Our method produces an argument position as a UR for a given noun n based on the following assumptions. Assume that v is a verb and p is a postposition.

Assumption 1-1: If n marked by p often appears with v, $\langle p, v\rangle$ is a good candidate of a UR.

Assumption 1-2: If first-person pronouns such as "I" often occupy an agent role of v, v is a good candidate of a verb in a UR.

Assumption 1-3: The postposition "`de`", which specifies instrument or locative semantic roles, is a good candidate of a postposition in a UR.

Assumption 1-1 is based on the intuition that a large number of references to a noun happen when talking about its utilization, and that co-occurrences of the noun and URs are frequently observed. The second is introduced to avoid "corpus-dependent" URs. For instance, newspapers include many events unusual in our daily lives, such as traffic accidents and murders. As a result, argument positions used to describe such unusual events tend to be produced as URs according to Assumption 1-1. For instance, in Japanese newspaper articles, a car often appears as a criminals' vehicle for getaways, and, from the first assumption, "getaway with a car" can be produced as a UR. This may be difficult to regard as a proper UR. On the other hand, Assumption 1-2 is based on the intuitions that the first-person pronouns often refer to *normal people* and that they co-occur with the verbs describing *usual* events frequently. In short, we expect that we can prevent the verbs describing unusual events from being produced by considering the co-occurrences with the first-person pronouns.

Assumption 1-3 is related to Japanese syntax. We regard the expressions "going (somewhere) by car" and "drinking (something) in a pub" as URs for "car" and "pub." In their Japanese translations, "`kuruma`(car) `de`(postposition) `iku`(go)" and "`pabu`(pub) `de`(postposition) `nomu`(drink)," the car and the pub are marked by the same postposition "`de`." In general, "`de`" marks the instrument or locative roles. Assumption 1-3 was made based on the observation that if a given noun occupies the instrument role or the locative role of a verb, the verb is a good candidate for a UR.

We define the score $Uscore(n, p', v')$, which embodies our assumptions, over argument positions $\langle p', v'\rangle$ and n. The Step 1 procedure produces the argument position denoted by $U(n) = argmax_{\langle v', p'\rangle \in V \times A}\{Uscore(n, p', v')\}$, as a UR. Here, A is a set of postpositions. V is a set of verbs, which can be a verb in possible URs. As verbs in V, we used only the verbs which appeared in our corpus with the verbal suffix "`tai`," which can be translated to "want," to guarantee the verbs in the set expresses *intentional actions*. We also manually removed from V the 31 verbs that can never be URs/PRs, such as "become," and the nine verbs that mean literally "using," "enjoying" and "preparing."

The score $Uscore$ is given as follows.

$$Uscore(n, p', v') = P(n, p', v')P(S|AP, v')Bias(p')/P(n)$$

$P(n, p', v')$ is the co-occurrence probability of the verb v' and n marked by the postposition p'. This reflects Assumption 1-1. As for $P(S|AP, v)$, S denotes a set of first-person pronouns, and AP is the set of postpositions which can mark agent roles. We used 14 pronouns as S, and $AP = \{$`ga`, `ha`$\}$. Thus, $P(S|AP, v)$ represents the probabilities that the first-person pronouns occupy the agent role of v, given v and the agent role. This captures Assumption 1-2. Note that the term $P(n, p', v')P(S|AP, v')$ is an approximation of $P(S, AP, n, p', v')$, the probability of a sentence such that v' is its head, n marked by p' is an argument of v', and a first-person pronoun occupies the agent role of v. $P(n)$ is the occurrence probability of n. Note that this probability does not affect the value of $U(n)$. But we took this probability into account for *fair* comparison of the $Uscore$ values for distinct nouns. The third term $Bias(p')$ denotes the bias based on Assumption 1-3 concerning `de`, and it is defined below.

$$Bias(p') = \begin{cases} 25 & p' = \texttt{de} \\ 1 & otherwise \end{cases}$$

The bias of 25 was determined empirically from experimental results on our development set.

2.2 Step 2: Acquisition of PUPs

2.2.1 Assumptions

In Step 2, our procedure acquires PRs, given candidates of URs obtained in Step 1. We developed the procedure based on the following assumptions.

Assumption 2-1: A PR is likely to co-occur with a given noun n frequently.

Assumption 2-2: A PR candidate for a noun n is likely to be a proper PR if the candidate is a proper PR for another noun n'. This tendency becomes stronger when n' has the same URs as n. In other words, a proper PUP for n' is also likely to be a proper one for n.

Assumption 2-3: A PR and a UR in a PUP are likely to appear in particular linguistic patterns.

Based on Assumption 2-1, we restrict the candidates of PRs to the argument positions frequently co-occurring with n. As for Assumption 2-2, we found that there are some argument positions that are likely to be PRs. The examples include "buying X" and "making X." Moreover, if two objects have the same UR, they are likely to have a common PR. Assume that the pair of "buying X" and "drinking X" is a PUP for "beer." Then, "buying X" is likely to be a PR for other nouns, such as "whiskey," that have "drinking X" as their URs.

As the linguistic patterns in Assumption 2-3, we assumed I) patterns expressing that an event expressed by a verb temporally precedes another event described by another verb, and II) patterns expressing that an action described by a verb is a means of another action represented by another verb. The following is a translation of the Japanese expressions fitting our patterns. The verbs that can be the ones in a PR and a UR are marked by *prepv* and *utilv* respectively.

Coordinated Sentence (CS) He $\underline{\text{bought}}_{prepv}$ a book *and* $\underline{\text{read}}_{utilv}$ it.

Relative Clauses (RC) He $\underline{\text{read}}_{utilv}$ a book *that* he $\underline{\text{bought}}_{prepv}$.

Conjunction Tame (CT) He $\underline{\text{bought}}_{prepv}$ a book *for the purpose of* $\underline{\text{reading}}_{utilv}$ it.

These expressions can be regarded as evidence supporting that the pair of "buying X" and "reading X" is a PUP for "book." Note that they have already been used in acquiring semantic relations (Fujiki *et al.* 03; Inui 04). In the following, we use co-occurrence frequencies of verbs in the patterns. They are defined over two verbs *prepv* and *utilv*, which can be verbs of a PR and a UR respectively, and are denoted by $f_{CS}(prepv, utilv)$, $f_{RC}(prepv, utilv)$ and $f_{CT}(prepv, utilv)$ for each pattern.

In the following, we describe Step 2, which is divided into two parts, in more detail. The first part is a process to collect candidates of PRs that may constitute a PUP with a UR obtained in Step 1. The second part is a procedure to select only proper PUPs from the pairs of the PR candidate and the UR candidate obtained in the first part.

2.2.2 Collecting Candidates of PRs

We collect the candidates of PRs to Assumptions 2-1 and 2-3. We assume that an argument position $Util = \langle p_{util}, utilv \rangle$ have been generated in Step 1 as a candidate of a UR for a noun n. First, our procedure collects the set of all the argument positions co-occurring with n, which we denote by $PC(n)$. We assume that $PC(n)$ is ranked by the co-occurrence frequencies between the argument positions and n. This step is done for finding the PR candidates according to Assumption 2-1.

The next step is done for collecting the PR candidates according to Assumption 2-3. For each pattern $X \in \{\text{CS}, \text{RC}, \text{CT}\}$, we extract the top five argument positions in the rank of $PC(n)$ such that the position $Prep = \langle p_{prep}, prepv \rangle$ contains a verb $prepv$ that co-occurs with the verb $utilv$ in $Util$ in the pattern X, i.e., $f_X(prepv, utilv) > 0$. In other words, we extract the five argument positions that co-occur $Util$ in the pattern X and that most frequently co-occur with n, for each pattern. The union of the top five items for the three patterns and the top five argument positions in the original $PC(n)$ is treated as a set of candidates for the PRs. We denote this candidate set by $C(n)$.

2.2.3 Finding Proper PUPs

Now, we can move to the second part of Step 2. Our procedure judges if the pair of a PR candidate in $C(n)$ and $Util$ can constitute a PUP by using an SVM. This judgment is done according to Assumption 2-2.

A learning process of SVMs is to determine a decision function that maps a feature vector, which represents some properties of the objects to be classified, to a real number. After a decision function is determined, the SVM classifies a given object according to whether the value of the function is positive or negative. Thus, the SVM was basically designed as a binary classifier. But we use it in a slightly different way. Our algorithm computes the value of a decision function for a candidate of a PUP, and uses it as a score indicating the likeliness that the PUP candidate is proper one. In other words, we assume that the larger the value of the decision function becomes, the more likely the PUP candidate will be the proper one. The final output of our procedure is the PUP candidates that have score values larger than a given threshold.

More precisely, our procedure judges if the pair of candidates in $C(n)$ and $Util$ can constitute a PUP by performing the following substeps.

- For each argument position $Prep$ in $C(n)$, compute the value of the decision function in the SVM for the PUP candidate $\langle Prep, Util \rangle$ and assume that the value is the score for the candidate.
- If the PUP candidate $\langle Prep, Util \rangle$ was labeled as a proper PUP for another noun $n' (\neq n)$ in the training set, add a certain large constant to the score computed in the previous step.
- Produce the PUP candidates that has score values larger than a given threshold as proper PUPs.

In the first substep, the feature vector given to the SVM is generated as follows. We assign a unique integer to each argument position that appears as a candidate of a PR in a training set or a test set. We

regard the integer as an identifier of the candidate. Similarly, each candidate of a UR is given a unique integer/identifier. Then, in the feature vector, the two feature values corresponding to the identifier of a PR candidate and the identifier of a UR candidate are set to 1. The other feature values are zero. This feature vector is used to train the SVMs so that they can memorize the proper PRs and URs in a training set, and reflects Assumption 2-2. Note that one may expect higher precisions can be achieved by by adding various frequencies such as f_{CS} to a feature vector. But this was not the case, as we discuss later.

The second substep also embodies Assumption 2-2. It awards a bonus to the PUPs that were regarded as proper ones in a training set for some noun n'. The bonus value was set to a large value, 10^6 in our experiments, so that such PUPs have the largest score values among all the PUP candidates.

We used the *TinySVM*[1] as an implementation of SVMs. We selected the RBF kernel function, based on the experimental results on our learning set.

3 Experiments

3.1 Estimation of Probabilities

We parsed 35 years of newspaper articles (Yomiuri 87-01, Mainichi 91-99 Nikkei 90-00, 3.24GB in total) and 92.6GB of HTML documents downloaded from the WWW by using a parser (Kanayama *et al.* 00) and extracted the word frequencies. The probabilities used in our experiments were obtained through the maximum likelihood estimation from the frequencies.

3.2 Experiments on Acquisition of URs

First, we evaluated the performance of Step 1 for acquiring URs. We restricted the object to which we applied our algorithm to *artifacts*, since the PUPs may not be defined for non-artifacts. We picked up 2,766 nouns referring to artifacts that appeared more than 500 times in 33 years of newspapers, by consulting a thesaurus (Ikehara *et al.* 99). Then, as a development set, we randomly extracted 300 occurrences of the artifact words (226 distinct words) in randomly selected newspaper articles.

As a test set for evaluating the URs produced by our method, we randomly picked up 200 distinct artifact words that did not occur in the development set but appear in randomly selected articles, which were not used in estimating word frequencies. Our procedure produced 200 candidates of URs for this test set. We asked three human subjects to judge if the UR candidates were proper or not. More precisely, the subjects checked if a given UR candidate for X was a *principal* and *direct* paraphrase of either of the Japanese expressions "X `wo riyousuru`(utilize)," "X `wo tsukau`(use)," "X `wo mochiiru`(use)," "X `wo katsuyousuru`(utilize)," or "X `wo tanoshimu`(enjoy)."

We assumed that only the URs that were judged as proper by all the three subjects were regarded as *acceptable*. The 200 URs produced by our algorithm contained 127 acceptable ones (= 63.5%) in total. The kappa statistic for assessing the agreement between the judgments on the URs was 0.57, which indicated *moderate* agreement according to Landis and Koch ,1977.

[1]Available from http://chasen.org/ taku/

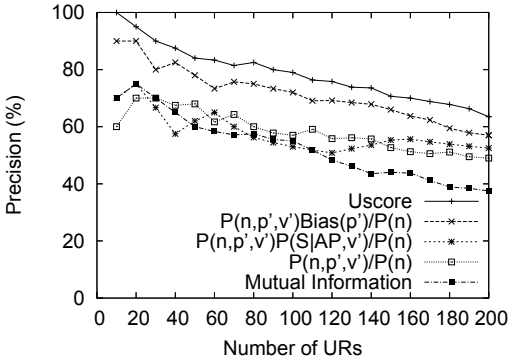


Figure 1: Precisions of the URs

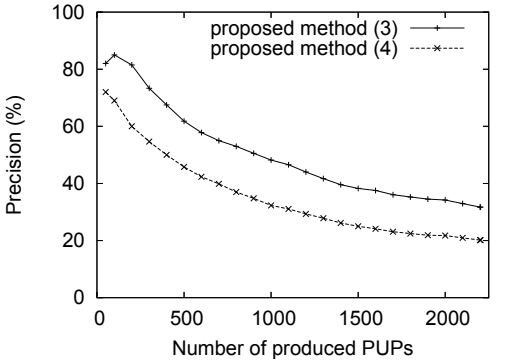


Figure 2: Precisions of the PUPs

The precision curve for the obtained URs are presented in Figure 1. The URs were sorted by their score $Uscore$, and each point plots the precision of the top N elements in the sorted URs. The graph also shows the precisions of alternative methods, which include mutual information between a noun and an argument position (Pustejovsky *et al.* 93) and the scores obtained by removing some terms from $Uscore$. The fact that our method outperformed all the alternatives suggests that, at least, the results are not trivial and each term in $Uscore$ contributes to the precisions.

3.3 Experiments on Acquisition of PUPs

To evaluate our method's ability to obtain PUPs, we made a training set for the SVM from the 226 nouns in our development set and the 200 nouns in the test set for evaluating the UR acquisition. First, we computed candidates of PRs, which were denoted by $C(n)$ for a noun n in the previous section, and assumed that the pair of an item in $C(n)$ and a UR for n computed in Step 1 is a candidate of PUPs. The generated candidates were then labeled by three human subjects. This resulted in 4,570 labeled PUP candidates. The kappa was 0.605, which suggested *good* agreement. Note that only the PUP candidates that were judged as proper by all the three subjects were regarded as *positive examples* in the training set. The 4,570 candidates included 1,106 *positive examples* (=24.2%). Next, as a test set for PUP acquisition, we randomly picked up

200 distinct artifact nouns that did not occur in the training set, and produced 2,204 PUP candidates. The labeling of the PUP candidates was done by four subjects this time. The kappa was 0.61, which indicated *good* agreement.

Figure 2 plots the precisions when we set the threshold so that our procedure produces the top N PUPs in the ranking by our score values. 'Proposed method (3)' refers to the precisions obtained by our method when we assume that the acceptable PUPs are only those regarded as proper by at least three of the four subjects. 'Proposed method (4)' indicates the precisions when the acceptable PUPs are assumed to be only the ones judged as proper by all the four subjects. For the top 200 PUPs, the precision is 82% for 'proposed method (3).' The PUPs regarded as proper covered 56 nouns. For the top 400 PUPs, the precision is 68% for 'proposed method (3).' The proper PUPs covered 87 nouns. Some of these PUPs are listed in Figure 3. Recall that our procedure awards a bonus to the proper PUPs appearing in a training set. Actually, the top 255 PUP candidates were such PUPs and they included 192 PUPs that were regarded as proper ones by three of the four subjects.

Let us examine some other learning schemes. We checked if the precisions could be improved by adding the co-occurrence frequencies in the linguistic patterns (f_{CS}, f_{RC} and f_{CT}), the co-occurrence frequencies between given nouns and PR candidates, and $Uscore$ to the feature vectors. But we could not observe the improvement. We also tried to use the feature vectors that consisted of only the co-occurrence frequencies, but, again, the improvement was not observed.Note that, for each type of co-occurrence frequencies, we observed the tendency that the larger frequency a PUP candidate has, the more likely it is to be proper one. (These tendencies indicate the validity of Assumptions 2-1 and 2-3.) But, with considering the above experimental results using the SVMs, such positive correlations were not so strong that the co-occurrences can improve our PUP acquisition method at least.

rank /sbjs	PR	UR
67/4	suupaa ni iku (go to a supermarket)	suupaa de kau (buy (something) at the supermarket)
68/3	syohan wo syuppansuru (publish the first edition)	syohan de yomu (read (something) in the first edition)
69/4	daizu wo ueru (plant beans)	daizu de tsukuru (make (something) from the beans)
122/4	waapro wo kau (buy a word processor)	waapro de kaku (write (something) with the word processor)
126/3	douga wo torikomu (download/read an animation)	douga de miru (watch (something) in the animation)
127/3	buraunkan ni utsusu (produce (some image) on a TV tube)	buraunkan de miru (watch (something) in the TV tube)
300/4	flute wo kau (buy the flute)	flute de ensousuru (play the flute)
301/4	senkou wo kau (buy a joss stick)	senkou wo ageru (offer the joss stick to a god)
315/4	kamera wo kamaeru (have a camera at the ready)	kamera de toru (take (pictures) with the camera)
316/0	sashie wo tsukeru (insert an illustration)	sashie de yomu (read (something) with/in the illustration)

Figure 3: Examples of the produced PUPs

4 Conclusion

We presented a method to acquire the pairs of the expressions representing preparation and utilization of a given object by using various co-occurrence frequencies and a supervised learning method. For instance, the method could recognize that the expression "play the flute" represented the utilization of the flute and that "buying the flute" was a preparation for playing the flute.

References

(Barzilay & Lee 03) R. Barzilay and L. Lee. Learning to paraphrase:an unsupervised approach using multiple-sequence alignment. In *Proc. of HLT-NAACL 2003*, pages 16–23, 2003.

(Boni & Manandhar 02) M. De Boni and S. Manandhar. Automated discovery of telic relations for wordnet. In *Proc. of the first International WordNet Conference*, 2002.

(Carberry 90) S. Carberry. *Plan Recognition in Natural Language Dialogue*. MIT Press, 1990.

(Chklovski & Pantel 04) T. Chklovski and P. Pantel. Verbocean: Mining the web for fine-grained semantic verb relations. In *Proceedings of EMNLP-04*, 2004.

(Ikehara *et al.* 99) S. Ikehara et al. *Goi-Taikei–CDROM*. Iwanami Shoten, 1999. in Japanese.

(Fujiki *et al.* 03) T. Fujiki, H. Namba, and M. Okumura. Automatic acquisition of script knowledge from text collection. In *Proceedings of The Research Note Sessions of EACL'03*, 2003.

(Inui 04) T. Inui. *Acquiring Causal Knowledge from Text Using Connective markeers*. Unpublished PhD thesis, NAIST, Japan, 2004. NAIST-IS-DT0161005.

(Kanayama *et al.* 00) H. Kanayama, K. Torisawa, Y. Mitsuishi, and J. Tsujii. A hybrid Japanese parser with hand-crafted grammar and statistics. In *Proceedings of COLING 2000*, pages 411–417, 2000.

(Landis & Koch 77) J. R. Landis and G. G. Koch. The measurement of observer agreement for categorial data. *Biometrics*, 33:159–174, 1977.

(Lapata & Lasacarides 03) M. Lapata and A. Lasacarides. A probabistic account of logical metonymy. *Computational Linguistics*, 29(2):263–317, 2003.

(Pustejovsky 95) J. Pustejovsky. *The Generative Lexicon*. MIT Press, 1995.

(Pustejovsky *et al.* 93) J. Pustejovsky, P. Anick, and S. Bergler. Lexical semantic techniques for corpus analysis. *Computational Linguistics*, 19(2):221–358, 1993.

(Vapnik 98) V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

(Yamada & Baldwin 04) I. Yamada and T. Baldwin. Automatic discovery of telic and agentive roles from corpus data. In *Proceedings of PACLIC 18*, 2004.

Knowledge-Poor Approach to Dependency Parsing: Dependency Parsing based on Morpho-Syntactic Information

Julia Trushkina*
Center for Text Technology
North-West University
Potchefstroom 2531, South Africa
20215770@puknet.puk.ac.za

Abstract

The current paper presents a knowledge-poor rule-based dependency parser for German (GRIP). Unlike existing dependency parsers, GRIP parser does not rely on complex resources such as subcategorization information and lexical-semantic information and reaches state-of-the-art performance based on morpho-syntactic characteristics and linear order of tokens in a sentence. Techniques employed in the development of the parser are described in the paper and a detailed evaluation together with an error analysis is presented.

1 Motivation

Many natural language applications benefit from a deep analysis of the text. Thus, in current state-of-the-art machine translation and question answering systems, parsing has become a common module contributing to the successful realization of the task. Parsing has also found successful applications in Information Extraction, Speech Recognition and Text Summarization, among other areas. Lately, dependency parsing models which provide deep analysis in compact and explicit form are receiving an increasing attention and interest (Nivre *et al.* 04; Oflazer 99; Yamada & Matsumoto 03). Constraint-based and rule-based models are particularly popular in dependency parsing (Bröker *et al.* 94; Duchier & Debusmann 01; Oflazer 99; Schröder *et al.* 00; Tapanainen & Järvinen 97) due to high performance demonstrated by the models. To achieve such high performance, most standard dependency parsing models rely in a significant way on subcategorization information, such as verbal subcategorization frames, in the parsing process. Thus, German dependency formalisms such as Topological Dependency Grammar (Duchier & Debusmann 01), Weighted Constraint Dependency Grammar (Schröder *et al.* 00) and Concurrent Lexicalized Dependency Parser (Bröker *et al.* 94), incorporate such information in the lexical entries of tokens and employ valency constraints to ensure the correct assignment of arguments to verbs. Subcategorization information significantly simplifies the parsing task, since necessary clues about obligatory and possible dependency relations of tokens are provided to the parser. However, development of a broad-coverage lexicon which contains subcategorization and lexical-semantic information is a time consuming project[1]. Therefore, such lexicons are unavailable for many languages.

The current paper explores possibilities to build a state-of-the-art dependency parser based on widely available resources. The paper presents a rule-based dependency parser which reduces usage of subcategorization and lexical-semantic information to minimum and, instead, employs morpho-syntactic characteristics of tokens and a linear order of tokens in a sentence as the main information source on which the parsing process is based. Since morpho-syntactic taggers are more widely available and easier to build than lexical databases, a dependency parser based on morpho-syntactic information represents a suitable alternative for the parsing of resource-poor languages. Moreover, the parser does not rely on the use of annotated corpora, since rule-based formalisms do not require training data.

2 Introduction

The GRIP parser is a robust deterministic rule-based parser implemented in the XIP system (Aït-Mokhtar & Chanod 97). The parser is a part of the GRIP system (Trushkina 04) and is composed of two modules: a chunker, which provides shallow constituency analysis for German sentences, and a dependency module, which establishes dependency relations between tokens in the input.

* The research reported in the article has been conveyed while the author was employed at the Seminar für Sprachwissenschaft, Tübingen University.

[1] Thus, a project on development of three large lexical databases at the CELEX Centre for Lexical Information (Baayen *et al.* 93) was on-going for fourteen years.

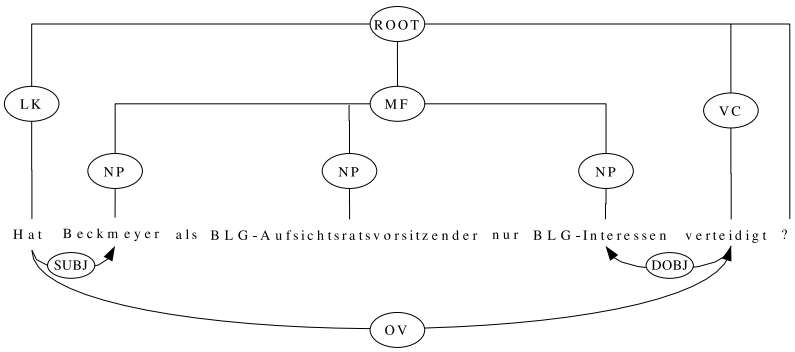


Figure 1: Analysis provided by the GRIP parser

The ultimate goal of the parser is the assignment of dependency structures to German sentences. In the current version, the parser concentrates on the annotation of the frame of a sentence: the parser identifies the main element of the sentence and its arguments, i.e. a verbal group and its complements. Figure 1 exemplifies kind of analysis provided by the parser.[2]

Constituency analysis plays a supporting role in the dependency analysis. By grouping lexical tokens in phrases, it pre-defines the possible domains of dependencies, which significantly simplifies the process of dependency assignment. Thus, for example, a direct object relation can possibly be established between any verb and any noun in accusative case. With a preprocessing constituency analysis which identifies phrases and topological fields, the search space is easily restricted to the heads of nominal phrases in the initial and the middle fields relevant for the verb. This excludes from consideration all nouns in other clauses, nouns occurring in prepositional and adjectival phrases and non-head nouns in nominal phrases.

3 Constituency analysis

3.1 Constituents inventory of GRIP

The output structures of the chunker are based on the TüBa-D/Z treebank structures (Sem03) with minor modifications determined by the purpose of the chunker. Since the chunker aims at providing a basis for the dependency module and does not intend to annotate deep relations, the types of structures that the chunker outputs are flattened in comparison to the TüBa-D/Z annotation scheme. Thus, adjectival, adverbial and infinitival phrases, as well as recursive noun phrases and coordinated topological fields, are not analyzed. Table 1 lists phrasal and topological field constituents annotated by the chunker.

category	description
`NP`	non-recursive noun phrase
`PP`	prepositional phrase
`AP`	adjectival phrase
`VF`	initial field
`LK`	left sentence bracket
`MF`	middle field
`VC`	verb complex
`NF`	final field
`CF`	complementizer field
`KOORD`	field for coordinating particles
`PARORD`	field for non-coordinating particles

Table 1: Constituents annotated by the GRIP parser

3.2 GRIP chunking rules

GRIP chunking rules represent constraints on part-of-speech categories of the tokens, on limited lexical information and on a linear order of tokens in the input string. The rules identify a set of nodes to be combined under the same mother node (`list_of_lexical_nodes` field) and specify a context in which the creation of such new structure is valid (`left_context` and `right_context`) or conditions under which the rule applies (`conditions_on_lexical_nodes`):

(1) new_node -> |left_context| list_of_lexical_nodes |right_context|.

or

new_node -> list_of_lexical_nodes, conditions_on_lexical_nodes.

[2]The categories used in the analysis are: ROOT for 'root node', LK for 'left sentence bracket', MF for 'middle field', VC for 'verbal complex', NP for 'noun phrase', SUBJ for 'subject', DOBJ for 'direct object', OV for 'verbal object'.

The context can extend as far as sentence boundaries but in practice is usually limited to a small window of adjacent tokens. The conditions can be imposed on the linear order of tokens and on the values of tokens, such as identity of the case values. The application of rules is deterministic: once a node has been created, it is not reconsidered on the later stages of analysis.

The chunking grammar consists of the following components: (1) a preprocessing component, (2) a component for annotation of phrasal nodes, and (3) a component for annotation of topological fields.

The preprocessing component is used for the correction of POS categories of input tokens, for grouping tokens under a common mother node with an appropriate POS category and for the preliminary structuring of set phrases into phrasal nodes. An example rule (2) of the preprocessing component is designed for identification of nouns which include a truncated part in parentheses, such as "*(Musik-)Geschichte*" ("*history (of music)*"). Using features *first* and *last*, the rule specifies the first and the last elements of the node sequence to which it applies. Additionally, a linear order of intermediate elements is determined, which restricts the area of rule application to sequences in which a truncated element precedes a right parenthesis. If the constraints on the order of tokens are satisfied, the tokens are combined under a common mother node with the category `NOUN`.

(2) noun -> punct[lpar,first], trunc#1, punct#2[rpar], noun[last], where (#1<#2).

The proper chunking is performed by the component for annotation of phrasal nodes. The component consists of constraint rules for the annotation of adjectival phrases, noun phrases and prepositional phrases. Apart from annotation of simple phrases consisting of standard elements, such as a prepositional phrase "*in einer internen Kontrolle*" ("*in an internal control*"), the chunker provides annotation of recursive phrases, such as a phrase presented in example (3):

(3) in der am vergangenen Montag
'in the on last Monday
abgesegneten rot-grünen Neufassung
approved red-green new version"
in the red-green version approved last Monday

The annotation of recursive phrases is ensured by the repetitive statement of rules.

After the application of the phrasal nodes annotation rules, an input string receives an analysis which includes marking of adjectival phrases, noun phrases and prepositional phrases. The following component provides further annotation of the string in terms of topological field categories (Höhle 85). The fields are annotated in the following order: CF, VC, LK, KOORD, PARORD, NF, VF, MF.[3]

This order simplifies the process of annotation, since identification of consequent fields can rely on previously annotated categories. For example, in complicated cases, the correct assignment of verbal complexes requires reference to a complementizer field. The annotation of the left bracket of a sentence (LK) is considerably simplified if verbal complexes have already been recognized: in this case, all finite verbs which have not been assigned a VC mother node receive a left bracket analysis.

After the first round of the annotation of topological fields, subordinate clauses (SCL) are recognized as sequences of topological fields. The annotation of subordinate clauses allows for the easier recognition of recursive topological fields, such as final fields which are represented by a clause, or middle fields with an embedded clause. The repetitive application of rules guarantees the correct annotation of recursive topological fields.

In total, GRIP constituency module comprises 1328 rules, which include 77 preprocessing rules, 464 rules for the annotation of phrasal nodes and 787 rules for the annotation of topological fields and subordinate clauses.

3.3 Evaluation

The performance of the GRIP chunker has been evaluated against previously unseen treebank data. The test data used in the evaluation of the chunker comprises 12 020 tokens. The average sentence length in the data set is 14.9 tokens per sentence.

The evaluation of the chunker performance is based on the phrase boundaries. Metrics of precision, recall and f-measure, both labeled and un-

[3] See Table 1 on the previous page for an explanation of the labels.

Brackets		Labeled			Unlabeled			Speed	
gold	**test**	**Recall**	**Prec.**	**F-meas.**	**Recall**	**Prec.**	**F-meas.**	**tokens/sec**	**sec/sent**
13.9	13.6	95.31	96.43	95.87	95.71	96.78	96.24	418.52	0.04

Table 2: Evaluation of the GRIP chunker

labeled, have been used. For labeled metrics, not only correct spanning of a constituent is required, but also a correct labeling of the constituent.

Table 2 presents results of the experiments with the chunker when correct part-of-speech tags are provided in the input. The table additionally provides an average number of constituents in gold and test data (first two columns, "*Brackets gold*" and "*Brackets test*"). Evaluation of the speed of the chunker is presented in columns "*tokens/sec*" and "*sec/sent*". The first metric estimates number of tokens analyzed by the chunker per second. The second metric evaluates the speed of the chunker in terms of sentences and represents the amount of time which is required by the chunker for the analysis of one sentence.

The most common types of errors made by the chunker can be grouped under four categories: (1) clause boundaries errors; (2) coordination errors; (3) errors due to complex sentence structures; (4) errors due to conscious differences in annotation style.

The first type of errors concerns erroneous annotation of topological fields. After a complementizer field CF and sentence brackets LK and VC are annotated, other topological fields are recognized as sequences of nodes between the brackets. However, embedded clauses considerably complicate the annotation of fields. A failure to identify clause boundaries leads to erroneous analysis of several fields, and ultimately results in additional wrong annotation of dependencies.

The second type of errors concerns coordination constructions. The errors arise if possible structural ambiguity prevents a chunker from grouping nodes correctly. Such errors are specific for phrasal nodes.

Errors of the third type arise in sentences with a complex structure and/or unusual phenomena. Texts of a newspaper style contain many sentences with heavily embedded clauses, parenthetical constructions, unexpected punctuation and other phenomena difficult for automatic processing.

The last type of errors includes errors which are due to conscious discrepancies of the annotation style of the chunker and the treebank. They mainly concern treatment of nodes that are unattached in the treebank, such as discourse markers. The bracketed format of the GRIP output does not allow for unattached nodes. Therefore, a discourse marker is included in the structure of the sentence. Such discrepancies between the annotation styles lead to a decreased number in the evaluation.

The rule-based GRIP chunker outperforms constituency parsers for German which employ other techniques. Thus, (Dubey & Keller 03) report an accuracy of 74% of labeled recall and precision for a constituency parser based on probabilistic context-free grammars; the memory-based parser presented in (Kübler 03) achieves an accuracy of 84.78% (F-measure). The higher results of the GRIP parser can be attributed to three factors: (1) correct part-of-speech analyses have been provided in the input of the parser; (2) rule-based methods normally provide better performance; (3) the scope of constituency annotation of the parsers differs. To estimate the influence of the first factor, an additional experiment has been performed to evaluate the chunker on automatically tagged data.[4] The evaluation has demonstrated an accuracy of 92.34% (F-measure).

4 Dependency assignment

4.1 Dependency inventory of GRIP

The following dependencies are annotated by the parser: subject relation (SUBJ) including expletive and sentential subjects; direct object (DOBJ), indirect object (OD), genitive object (OG), sentential object (OS), verbal object (OV), predicative object (PRED) and a separable verbal particle (VPT). Additionally, the GRIP analysis identifies the root of the dependency tree and marks it with label ROOT.

4.2 GRIP dependency rules

The GRIP dependency module assumes a prechunked input in the bracketed format, as provided by the GRIP chunker. Nodes in the input

[4] The analyses have been provided by the GRIP morpho-syntactic tagger (Trushkina & Hinrichs 04).

are associated with relevant (possibly ambiguous) morphological and categorial information. Dependency annotation also relies on restricted information contained in the GRIP lexicon: when appropriate, the lexicon assigns one or more of the features listed in Table 3 to high frequency verbs. The lexicon has been compiled based on the training data and contains approximately 150 verbs. In the GRIP system, dependency relations are annotated sequentially, so that annotation on later stages can rely on previously assigned dependencies. Apart from the set of dependency relations listed above, the auxiliary dependencies APP for apposition and KONJ for conjunction are assigned by the parser. The order of dependency annotation is the following: ROOT, APP, KONJ, SUBJ, OV, VPT, DOBJ, PRED, OD, OS, OG.

Feature	**Example**
ditransitive	*fragen* ("to ask")
with genitive object	*bedürfen* ("to require")
reflexive	*sich bedienen* ("to help oneself")
separable	*aussehen* ("to look")
performative	*berichten* ("to report")
with predicative object	*bleiben* ("to remain")

Table 3: Features assigned to verbs in the GRIP lexicon

The general format of dependency rules is the following:

(4) |pattern| if <conditions> <dependency_terms>.

The `pattern` field combines a filter and a context fields. It specifies a node sequence and associates one or more nodes with variables. Reference to the feature values of nodes and exploration of the inner structure of nodes is possible.

The `dependency_terms` field defines a new dependency to be created. It consists of a name for dependency relation and an n-ary set of variables.

The `conditions` field is an optional field that represents a Boolean expression over dependencies. In this field, the existence of other dependency relations and their inter-connections can be checked. A dependency rule can be used for modifying or deleting a previously defined dependency relation. In this case, the relation to be modified or to be deleted is marked in the `conditions` field and the `dependency_terms` field determines whether the relation is to be deleted (with a $\sim$ sign) or to be renamed (with a new dependency term).

In total, the dependency parser of GRIP comprises 1176 rules. Morphological information, such as case information, represents a necessary basis for assignment of grammatical relations in German. Thus, if a clause contains a transitive verb, and a nominative and an accusative NP, it is safe to assume that a subject relation holds between the verb and the nominative NP and a direct object relation holds between the verb and the accusative NP. However, in real texts, the linkage between a case value and a function of a token is not always straightforward. Consider, for example, the sentence in (5):

(5) Julianne Köhler aber ist als sture,
Julianne Köhler however is as stubborn,
treue Musterdeutsche eine Entdeckung.
loyal model German a discovery.
'However, as a stubborn loyal model German, Julianne Köhler is a discovery.'

The sentence contains three nominative NPs, but only one of them is a subject of the verb: "*Julianne Köhler*". The noun phrase "*eine Entdeckung*" plays the role of a predicative object, whereas the noun phrase "*als sture, treue Musterdeutsche*" is a subject modifier.

For efficient and accurate dependency assignment, the following strategy is undertaken in the grammar: first, a dependency relation is established between any two nodes that can be connected by the relation. At this stage, the conditions only have reference to the following information: (1) the categorial values of the nodes; (2) the case value of the nodes (optionally); (3) what other nodes occur in the same clause.

For the example sentence (5), subject relations would be established between the verb and every nominative noun. For example, rule (6) establishes a subject-verbs relation between a nominative noun or a pronoun in an initial field (VF) and a verb in a left sentence bracket (LK):

```
(6) | VF{ ?*,                                          [1]
        NP{?*,                                        [2]
            noun#1[nom];                              [3]
            pron#1[attr:~,nom]}},                     [4]
    ?* [lk:~,vf:~,vc:~,sent:~,paren:~,                [5]
        spec:~],                                      [6]
    LK{ ?*,                                           [7]
        verb#2[fin]} |                                [8]
    if  (~ SUBJ(#1,#2))                               [9]

    SUBJ(#1,#2).                                      [10]
```

In this rule, a context for the rule application is described between the pipe lines (lines [1]-[8]). It includes a sequence of nodes which starts with an initial topological field (VF) (lines [1]-[4]) and ends with a left sentence bracket (LK) (lines [7]-[8]). Between these two topological fields, any number of nodes can occur, unless the nodes bear features of topological fields LK, VF or VC, or features of either sentence-final punctuation, or parentheses, or a hyphen (lines [5]-[6]). The internal structure of the nodes VF and LK is explored and variables are assigned to the nominative head of a noun phrase and to a finite verb. Thus, this rule states constraints on the context, on the internal structure of the nodes and on features of lexical nodes. Moreover, constraints on previously established relations are imposed: nodes #1 (a nominative head of the NP in the initial field) and #2 (a finite verb occurring in the left sentence bracket) are required not to stand in a subject relation with each other (line [9]). If all the constraints are satisfied, a subject relation is established between nodes #1 and #2 (line [10]).

After establishing relations on general basis, more specific constraints on the context and the feature values of the tokens are formulated in the grammar. These constraints aim at resolving conflicts which arise after the application of general rules. Thus, application of general rules to sentence (5) results in multiple subjects for the verb "*ist*". The following constraints resolve the conflict by eliminating or renaming previously established dependencies: First, since the first NP in sentence (5) agrees with the verb in number, all subject relations which involve other nominative NPs in the same sentence are renamed into predicative object relations. Next, the comparative NP "*als sture, treue Musterdeutsche*" is renamed from predicative object to subject modifier, since non-comparative predicative objects are preferred over comparative predicative objects. The constraints are formulated based on the analysis of the parser development data.

The strategy of dividing the annotation process into two stages (general rules and conflict solving constraints) allows for the minimization of the set of dependency rules involved in parsing and for taking maximum advantage of the constraint-based nature of GRIP.

The strategy of establishing relations with general rules and the consequent elimination or renaming of relations with more specific constraints is used for the annotation of all dependencies in GRIP.

4.3 Evaluation

The performance of the dependency module run on the data with correct morpho-syntactic tags is presented in Table 4. Prior shallow constituency analysis of the data is provided to the dependency module by the GRIP chunker.

Evaluation of speed of the dependency module demonstrated that the module analyzes 713.94 tokens per second and requires 0.02 second on average for the analysis of one sentence.

Below, common errors which were identified by the manual analysis of the data output by the parser are described.

Complex sentence structure often leads to a failure to correctly identify the relation. Such complicated structures involve complex named entities, such as the citations included in a statement; occurrence of sentence-final punctuation inside a clause, such as occurrence of an exclamation mark in sentences; or complex embedded constructions.

Another source of errors is failure to annotate relations which involve tokens without any morphological features, such as foreign material tokens, cardinals or prepositional phrases.

A prominent group of errors concerns annotation of subjects and direct objects which represent apposition terms. According to the annotation style of the parser, if an apposition group participates in a dependency relation, then all terms of apposition are assigned the dependency relation in question. However, in some cases correct identification of the apposition construction is complicated by the complex structure of the sentence. In such cases, only one of the apposition terms is assigned a dependency relation, since multiple subjects are ungrammatical in German, whereas double direct object occurs only with ditransitive verbs.

Errors are also caused by incorrect chunking analysis provided for the dependency parser and by confusion of relations. A frequent error in the annotation of sentential object occurs when a parser is unable to recognize the coordination of clauses without a conjunction. Another common error in the annotation of sentential objects, as well as in the annotation of the main element of the sentence concerns the treatment of discourse

Dep.	Labeled			Unlabeled		
label	Recall	Prec.	F-meas.	Recall	Prec.	F-meas.
total	94.91	94.55	94.73	95.95	95.57	95.76
ROOT	98.12	97.71	97.91	98.16	97.76	97.96
SUBJ	96.28	95.50	95.89	96.36	95.56	95.96
DOBJ	93.08	91.38	92.22	93.08	91.38	92.22
OD	83.69	83.15	83.42	83.69	83.15	83.42
OG	100	100	100	100	100	100
OS	70.74	71.27	71.00	71.27	71.63	71.45
OV	97.29	97.11	97.20	97.29	97.11	97.20
PRED	69.58	70.83	70.20	69.58	70.83	70.20
VPT	98.46	98.46	98.46	98.46	98.46	98.46

Table 4: Evaluation of the GRIP dependency parser

markers. They represent unattached nodes in the treebank but are part of the structure provided by GRIP.

5 Comparison to other parsers

The results achieved by the GRIP dependency parser are considerably higher than the results reported for other dependency parsers: thus, (Müller 04) reports an F-score of 82.49% for a German finite-state dependency parser; a memory-based dependency parser for Swedish described in (Nivre *et al.* 04) achieves an accuracy of 81.70%; a statistical parser of (Collins *et al.* 99) provides an accuracy of 80.00% for dependency annotation of Czech; (Yamada & Matsumoto 03) report an unlabeled precision of 90% for an English statistical dependency parser.[5] When applied to parsing of automatically tagged data, the GRIP parser demonstrates performance which is comparable to but is still higher than the performance of other dependency parsers: 85.55% (F-measure).

6 Conclusion

The current paper has presented a rule-based dependency parser for German (GRIP). Unlike existing dependency parsers, GRIP parser does not rely on complex resources such as subcategorization information and lexical-semantic information and reaches state-of-the-art performance based on morpho-syntactic characteristics and linear order of tokens in a sentence.

References

(Aït-Mokhtar & Chanod 97) Salah Aït-Mokhtar and Jean-Pierre Chanod. Incremental finite-state parsing. In *Proceedings of ANLP'97*, pages 72–79, Washington, D.C., 1997.

(Baayen *et al.* 93) Harald Baayen, Richard Piepenbrock, and H. van Rijn. The CELEX lexical data base on CDROM. Linguistic Data Consortium, Philadelphia, PA, 1993.

(Bröker *et al.* 94) Norbert Bröker, Udo Hahn, and Susanne Schacht. Concurrent lexicalized dependency parsing: the ParseTalk model. In *Proceedings of COLING'94*, Kyoto, Japan, 1994.

(Collins *et al.* 99) Michael Collins, Jan Hajič, Lance Ramshaw, and Christoph Tillmann. A statistical parser for Czech. In *Proceedings of ACL'99*, pages 505–512, College Park, Maryland, 1999.

(Dubey & Keller 03) Amit Dubey and Frank Keller. Probabilistic parsing for German using sister-head dependencies. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL'03)*, pages 96–103, Sapporo, Japan, 2003.

(Duchier & Debusmann 01) Denys Duchier and Ralph Debusmann. Topological dependency trees: A constraint-based account of linear precedence. In *Proceedings of ACL'2001*, Toulouse, France, 2001.

(Höhle 85) Tilman Höhle. Der Begriff "Mittelfeld", Anmerkungen über die Theorie der topologischen Felder. In *Akten des Siebten Internationalen Germanistenkongresses 1985*, pages 329–340, Göttingen, 1985.

(Kübler 03) Sandra Kübler. Parsing without grammar–using complete trees instead. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing, RANLP 2003*, Borovets, Bulgaria, 2003. to appear.

(Müller 04) Frank Müller. Annotating grammatical functions in German using Finite-State Cascades. In *Proceedings of the Twentieth International Conference on Computational Linguistics (COLING 2004)*, Geneva, Switzerland, 2004.

(Nivre *et al.* 04) Joakim Nivre, Johan Hall, and Jens Nilsson. Memory-based dependency parsing. In *Proceedings of CoNLL-2004*, pages 49–56, Boston, MA, USA, 2004.

(Oflazer 99) Kemal Oflazer. Dependency parsing with an extended finite state aproach. In *Proceedings of ACL 1999*, 1999.

(Schröder *et al.* 00) Ingo Schröder, Wolfgang Menzel, Kilian Foth, and Michael Schulz. Modeling dependency grammar with restricted constraints. *Traitement Automatique des Langues*, 1:97–126, 2000.

(Sem03) Seminar für Sprachwissenschaft, Universität Tübingen, Tübingen, Germany. *Stylebook for the Tübingen Treebank of Written German (TüBa-D/Z)*, 2003.

(Tapanainen & Järvinen 97) Pasi Tapanainen and Timo Järvinen. A non-projective dependency parser. In *Proceedings of the 5th Conference on Applied Natural Language Processing*, Washington, D.C., 1997.

(Trushkina & Hinrichs 04) Julia Trushkina and Erhard W. Hinrichs. A hybrid model for morpho-syntactic annotation of German with a large tagset. In *Proceedings of EMNLP'2004*, pages 238–246, Barcelona, Spain, 2004.

(Trushkina 04) Julia Trushkina. *Morpho-syntactic annotation and dependency parsing of German*. Unpublished PhD thesis, Tübingen, Germany, 2004.

(Yamada & Matsumoto 03) Hiroyasu Yamada and Yuji Matsumoto. Statistical dependency analysis with support vector machines. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT'03)*, pages 195–206, 2003.

[5] Unfortunately, no evaluation has been provided for other German dependency formalisms (Duchier & Debusmann 01; Schröder *et al.* 00; Bröker *et al.* 94).

Context-based ranking of suggestions for spelling correction

Julia Trushkina
Center for Text Technology
North-West University
Potchefstroom 2531, South Africa
20215770@puknet.puk.ac.za

Abstract

Most spell checkers provide suggestions for correction of misspelled words. The suggestions are usually ranked based on the similarity between a correction candidate and a misspelled word, as well as on the frequency of the correction candidate. The current paper presents a context-based method for ranking candidate correction suggestions. The method uses probabilistic context-free grammars to identify the correction suggestion that results in a word sequence with the most probable parse. The evaluation of the method on real data demonstrates considerable improvement of the ranking results over the results produced by non-context-based methods.

1 Introduction

A spelling checker usually provides a list of suggestions for correction of a misspelled word. The suggestions are normally ranked so that more probable suggestions are presented at the beginning of a list. Two criteria are used for ranking of suggestions:

- similarity between a suggested correction candidate and the misspelled word, and
- frequency of a suggested correction candidate (more frequently used words are given a higher priority).

For example, for a misspelled word *"howse"*, the `ispell` checker, an interactive spell-checking program for Unix, provides the following list of correction suggestions:[1]

"hose, hows, House, horse, house, hoes, Hosea, how's, hawser, hoarse, horsey, houser, Howe, Ho's, hows, hews, hors, hoers, Howie, dowse".

Most spelling checkers operate in an interactive mode: a user is prompted to choose a proper correction suggestion from a list, which is usually easily done given the context. Consider the following possible contexts of the misspelled word *"howse"*:

- *"the howse down the street"*;
- *"the White howse"*;
- *"the howse that won the race"*.

Given the context, different suggestions are likely to be chosen for correction of the misspelled word: *"house"*, *"House"* and *"horse"*, respectively.

Integration of a module for context-based ranking of correction suggestions would be beneficial for spelling checkers in two ways:

1. The process of spelling checking and correction can be made fully automatic: the spelling checker would identify the best correction candidate in the given context and replace the misspelled word with it. Such an automatic spelling corrector can be used as a preprocessor for other natural language processing applications which depend on spell-corrected input, such as grammar correction, tagging and parsing, information extraction, etc.

2. In the interactive mode, a user would profit from a better ranked list of correction suggestions. This is particularly important for short misspelled words, since for them, rather long lists of correction candidates are often provided. Consider, for example, a list proposed by `ispell` for correction of the string *"mak"*:

   "Mack, Maj, Mark, mack, make, mark, Mk, mask, Mal, MA, ma, AK, MAG, Mac, Mag, mac, mag, Mae, Mai, Mao, Mar, Max, May, mar, maw, max, may, Mab, Man, Zak, mad, man, map, mas, mat, oak, yak."

[1] The `ispell` checker has been chosen as a reference point in this paper due to its high performance and free availability.

Ranking of suggestions according to probability of their occurrence in the given context could therefore significantly simplify and speed up the interactive spelling correction process.

This paper presents a method for context-based ranking of spelling correction suggestions. The method uses probabilistic context-free grammars (PCFGs) for identification of most probable suggestions given the context. Section 2 provides a short introduction to probabilistic context-free grammars. Section 3 describes the general idea of PCFG-based ranking of suggestions, while sections 4 and 5 present implementation and evaluation of the method. The results of the experiments with the PCFG-based ranking method are discussed in section 6.

2 Probabilistic context-free grammars

A probabilistic context-free grammar[2] is an extension of a context-free grammar, in which each rule is associated with a probability. PCFGs are normally used for parsing[3]: a PCFG parser generates all possible analyses for an input string and calculates probabilities of the analyses as a product of the probabilities of all rules applied in the parse. The parse with the highest probability is output as the PCFG analysis of an input string. Since the calculation of the probabilities of all possible parses for a sentence is very inefficient, the most likely parse of a sentence is calculated by the Viterbi algorithm, a dynamic programming algorithm first described in (Viterbi 67). Below, the term "Viterbi parse" is used for the most probable parse of an input string.

3 PCFG-based ranking

The PCFG-based ranking method is based on the assumption that a correct word string is more likely than a string with erroneous usage of words. Thus, for example, the string *"the horse that won the race"* appears to be more probable than the string *"the hose that won the race"* or the string *"the how's that won the race"*. Given this assumption, ranking of correction suggestions is reduced to comparison of probabilities of sentences in which a misspelled word is replaced by different correction suggestions. For example, sentence 1 instantiates suggestion 1 in the position of a misspelled word, sentence 2 instantiates suggestions 2 in the position of a misspelled word, etc. A suggestion whose corresponding sentence has the highest probability is ranked first in the resulting list of suggestions, while a suggestion used in the least probable sentence is placed last in the resulting list of suggestions.

A probability of a string can be calculated in different ways:

- as a frequency of the string in a large corpus;
- as a frequency of a corresponding sequence of part-of-speech (POS) tags in a large corpus;
- as a frequency of a substring which contains the suggestion;
- as a product of probabilities of word or POS trigrams of a string.

The first method requires a very large corpus and is inapplicable to comparison of long strings, since the frequency of a string of considerable length in any corpus vanishes. Thus, a chance of encountering the string *"the horse that won the race"* in a corpus is rather low. The second method does not take into account co-occurrence of lexical tokens, which is an important source of information for identification of a proper spelling correction. Using substrings instead of full sentences limits the context, which reduces the advantages of a context-based ranking method. The same is true for the trigram method of calculating the probability of a string.

A PCFG provides a suitable alternative to calculation of sentence probabilities: namely, calculation of probabilities of Viterbi parses of sentences. Here, the first assumption is extended to the assumption that the Viterbi parse of a correct sentence has a higher probability than the Viterbi parse for a sentence with wrong word usage. Consider, for example, strings *"the horse that won the race"* (string 1) and *"the how's that won the race"* (string 2) together with their corresponding Viterbi parses presented in Figures 1 and 2 below. The parses are produced by the PCFG parser LoPar (Schmid 00) trained on the British National Corpus (Aston & Burnard 98). For the sake of readability of the analyses, the parses and node labels in the figures are considerably simplified. The node labels used in the

[2]For a more detailed introduction in PCFGs, see (Manning & Schütze 99).

[3]PCFGs can also be used for chunking and tagging, see (Schmid 00) and (Hinrichs & Trushkina 04).

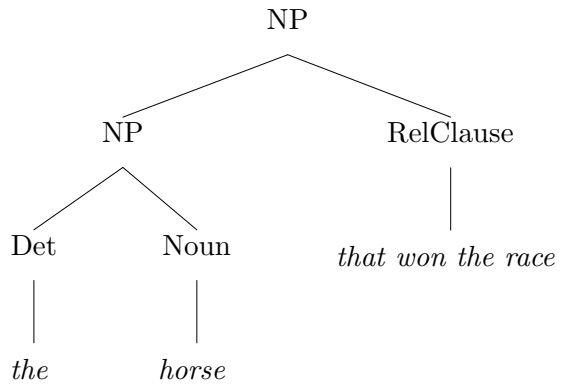


Figure 1: The Viterbi parse for the string *"the horse that won the race"*

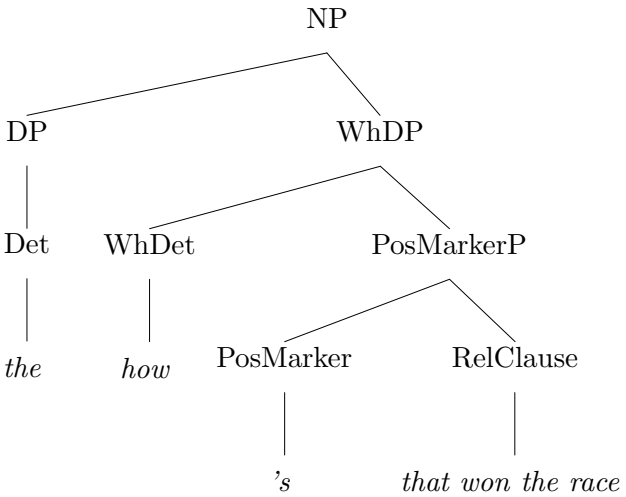


Figure 2: The Viterbi parse for the string *"the how's that won the race"*

figures are: `NP` for a noun phrase, `Det` for a determiner, `RelClause` for a relative clause, `DP` for a determiner phrase, `WhDet` for a wh-determiner, `WhDP` for a wh-determiner phrase, `PosMarkerP` for a possessive marker phrase, and `PosMarker` for a possessive marker.

The probability of the Viterbi parse of string 1 is much higher than the probability of the Viterbi parse of string 2 (-43.4523 vs. -52.4941[4]). This means that string 1 has a more probable structure and a more probable word combination than string 2, which in its case implies that string 1 is more likely to occur in a corpus than string 2.

The strings *"the horse that won the race"* (string 1) and *"the hose that won the race"* (string 3) have the same Viterbi analyses (the analysis presented in Figure 1). However, since lexical probabilities contribute to the resulting probability of a parse, probabilities of Viterbi parses for strings 1 and 3 differ (-43.4523 vs. -45.8858). Therefore, the method can be equally well used for comparison of strings which have the same POS analyses.

Comparison of sentences via their Viterbi parses has the following advantages:

1. the context of the whole sentence can be taken into account, which is crucial in some cases;

2. the probability of a parse is calculated as a product of probabilities of parts of the parse (i.e. rules), which reduces the data sparseness problem.

Below, the implementation of the method is described and the evaluation of the method is provided. Section 4.1 discusses the preparation of the input to the PCFG-based ranking procedure: how the suggestions for spelling correction are generated, which subset of them is presented to the PCFG-based ranking module, and why only a subset of suggestions is used.

4 Implementation

4.1 Generation of suggestions for spelling correction

The process of spelling checking in the implemented system starts by looking up words of an input text in a large dictionary. Words not found in the dictionary are assumed to be misspelled. For each misspelled word, a set of correction suggestions is generated with the following procedures:

1. Insertion procedure: if insertion of one letter, a space or a dash results in a valid word[5], add the resulting word to the set of correction suggestions.

2. Deletion procedure: if deletion of a character results in a valid word, add the resulting word to the set of correction suggestions.

3. Substitution procedure: if substitution of a character by a letter, a space or a dash results in a valid word, add the resulting word to the set of correction suggestions.

[4]LoPar outputs best parse probabilities as logarithms of probability scores.

[5]I.e. a word present in the dictionary.

1.	Keep the first letter (in upper case)	
2.	Replace a vowel in the initial position by a dollar sign ($)	
3.	Replace the following letters with hyphens:	
	A, E, I, O, R, U, Y, H, W	
4.	Replace other letters by numbers as follows:	
	B, F, P, V	1
	C, G, J, K, Q, S, X, Z	2
	D, T	3
	L	4
	M, N	5
5.	Delete adjacent repeats of a number	
6.	Delete the hyphens	
7.	Keep first three numbers or pad out with zeros	

Table 1: The original SOUNDEX algorithm

4. Phonetic procedure: based on the SOUNDEX algorithm (Knuth 73; Davidson 62), find all phonetic equivalents of a misspelled word and add them to the set of correction suggestions.

   The underlying idea of the algorithm originates from the phonetic classification of human speech sounds into bilabial, labiodental, dental, alveolar, velar and glottal sounds. The algorithm defines confusion sets of letters based on the phonetic classification, assigns a single marker to each confusion set and substitutes each letter of a word, except for the initial letter, with a marker of its confusion set. The original algorithm was slightly modified. The full modified SOUNDEX algorithm is presented in Table 1. With the modified algorithm, a SOUNDEX key is calculated for each word. Words which share a SOUNDEX key are assumed to represent phonetic equivalents of each other.

5. Concatenation procedure: for each sequence of two misspelled words, check whether a concatenation of the two words represents a valid word-form. In case of a positive outcome, add the concatenated word to the list of correction suggestions.

After a set of correction suggestions is generated, the suggestions are ranked based on (a) their similarity to the misspelled word and (b) on their frequency in the British National Corpus (BNC). The ranking is performed with a trigram method described below.

Firstly, lists of letter trigrams[6] are compiled for the misspelled word and for each correction suggestion. To ensure equal weighing of edge and middle letters of a word, two word-boundary signs are added to the beginning and the end of each word. Secondly, a trigram similarity score is calculated for each pair <misspelled word, correction suggestion>. Lastly, correction suggestions are rated according to the obtained trigram similarity scores. If two suggestions have the same trigram score, the suggestion with a higher BNC frequency is given a priority.

The trigram similarity score is calculated as follows:

1. For each shared trigram, add one point.

   For example, words *"Leave"* and *"live"* share trigrams `ve#` and `e##`[7]. The number of points for shared trigrams equals two.

2. Calculate the maximum number of points that the pair could have received. This number equals the number of points obtained by comparing the longer word of the pair to itself.

   For the pair *"Leave"* and *"live"*, the maximum number of points equals the number of trigrams of the word *"Leave"*: 7 (trigrams `##L`, `#Le`, `Lea`, `eav`, `ave`, `ve#` and `e##`).

3. Calculate the relative value of the obtained trigram points as compared to the maximum possible number of points.

   For instance, in this case: $2/7 = 28.57\%$.

[6] I.e. lists of all sequences of three letters.

[7] The hash sign represents a word-boundary.

4. Repeat the procedure for a case-insensitive trigram comparison, giving half a point for each shared trigram.

   Shared case-insensitive trigrams of words *"Leave"* and *"live"* are `##l`, `ve#` and `e##`, which produces 1.5 points. The maximum number of points for case-insensitive trigrams equals 3.5 (half point for each trigram of the longer word). The relative value of the obtained trigram points is 42.86%.

5. Add the two obtained relative values.

   The resulting trigram similarity score for the words *"Leave"* and *"live"* is 71.43%.

With the described trigram algorithm, similarities between a misspelled word and its correction suggestions are computed and the suggestions are ranked according to the obtained trigram scores. Thus, correction suggestions for the string *"howse"* are ranked as follows:

"hows, how's, house, horse, hose, hoarse, hoes, hawser, horsey, houser, hors, hoers, dowse, Howe, hews, Howie, House, Hosea, Ho's".

4.2 Restricting the set of suggestions

The analysis of rated correction suggestions on a development data set[8] has demonstrated that the correct suggestion is present among first three suggestions in 79.59% of cases, while the average length of suggestions lists exceeds two hundred words. Since the PCFG-based ranking is a time-consuming process, the list of suggestion corrections has been restricted to:

1. first three suggestions of the list; plus
2. all suggestions with a trigram score above 90%; plus
3. a suggestion whose BNC frequency is the highest among the first ten suggestions in the list.

An analysis of the resulting sets of correction suggestions in the development set has shown that a proper correction suggestion is contained in the set in 98.88% of cases. Two further restrictions are imposed on the suggestions:

1. the similarity trigram score of suggestions should be above 75%;
2. the difference in the length of a misspelled word and a correction suggestion should not exceed three letters.

The average length of resulting correction lists equals four words.

Apart from speed optimization of the context-based ranking method, the restriction of suggestion sets aims at improving the performance of the method. Since a PCFG is able to compare the similarity of a misspelled word and its correction, a preprocessing step should be made to ensure that suggestions which do not bear high similarity to the misspelled word are excluded from consideration.

4.3 Ranking procedure

A set of sentences in which misspelled words are replaced by their correction suggestions is generated. For example, for a sentence that contains two misspelled words with suggestion lists of lengths 2 and 4, respectively, a set of 8 sentences (2x4) is generated. For each sentence, a Viterbi parse is provided by the PCFG parser LoPar trained on the British National Corpus. LoPar additionally produces probabilities of the Viterbi parse. A sentence that has the highest probability of a Viterbi parse is considered to be a sentence with proper corrections of the misspelled words. The correction suggestions used in the sentence are ranked first in the corresponding suggestion lists.

4.4 Overview of the entire spelling correction process

Below, a summary of the entire process of spelling correction implemented in the system is presented. Given an input string, the system goes through the following steps:

- Identification of misspelled words

  My voice is **howse**.

- Generation of correction suggestions

  howse -> *hose, hows, House, horse, house, hoes, Hosea, how's, hawser, hoarse, horsey, houser, Howe, Ho's, hows, hews, hors, hoers, Howie, dowse*

[8] A development data set consists of 20 000 words extracted from the Tswana Learner English Corpus described in section 5.

- Similarity-based ranking of suggestions

  hows, how's, house, horse, hose, hoarse, hoes, hawser, horsey, houser, hors, hoers, dowse, Howe, hews, Howie, House, Hosea, Ho's

- Restriction of suggestions

  hows, how's, house, horse, hose, hoarse

- Identification of the best suggestion given the context

  howse -> *hoarse*

- (Optional) correction of misspelled words

  My voice is **hoarse**.

5 Evaluation

The PCFG-based ranking method has been evaluated on data extracted from the Tswana Learner English Corpus (TLE corpus). The TLE corpus is a collection of argumentative essays of Tswana learners of English compiled at the North-West University. The TLE corpus consists of 200 000 words, of which 50 000 words have been manually error-tagged and provided correct word-forms.

A spelling checking procedure described in the previous section has been performed on a test set of 22 225 words extracted from the TLE corpus. A test set for the experiments with the PCFG-based ranking module, called *the PCFG test set* below, has been compiled in the following way: If a sentence contains a misspelled word which has been successfully identified by the spelling checking procedure, include the sentence in the PCFG test set, unless a proper correction suggestion is not present in the list of correction suggestions. The resulting PCFG test set contains 167 sentences with 190 misspelled words.

The baseline for the evaluation of the ranking method is the performance of the trigram ranking method described in section 4. The baseline represents a number of proper correction suggestions ranked first in the suggestions lists in the input to the PCFG-based ranking module. The baseline is presented in Table 2, line 1 ("baseline"). For the comparison with a well-known spell checker, the performance of the `ispell` checker on the same data is presented in line 2 of Table 2 ("ispell"). Line 3 of the table ("PCFG") represents the performance of the PCFG-based ranking module. As the table shows, application of the PCFG-based ranking module provides a considerable improvement of results and leads to a significant error rate reduction of 47.89%.

	CORRECT	WRONG
baseline	62.83%	37.17%
ispell	56.02%	43.98%
PCFG	80.63%	19.37%

Table 2: Performance of the PCFG-based ranking method as compared to the performance of non-context-based methods

6 Discussion

The evaluation has demonstrated that the PCFG-based ranking method leads to a significant improvement of ranking results. However, remaining 19.37% of errors represent too large an error rate to use the method for fully automatic error correction purposes.

A qualitative analysis of the incorrect suggestions has demonstrated that the errors are mostly caused by a large difference in the frequencies of correction suggestions. Thus, although the string *"to be taught how they should behave"* seems more probable than the string *"to be thought how they should behave"*, the word *"thought"* is much more frequent than the word *"taught"* (56 883 occurrences of the word *"taught"* vs. 3 908 occurrences of the word *"taught"* in the BNC). This difference in word frequencies resulted in a higher probability of the Viterbi parse of the string *"to be thought how they should behave"* and, therefore, the word *"thought"* has been erroneously suggested as the best candidate for correction of a misspelled word *"tought"*.

To reduce the error rate of the PCFG-based ranking module, some normalization of lexical token frequencies should be introduced into the method. Investigation of this problem represents one of the areas for future research.

A further improvement of the PCFG-based ranking module performance can be introduced by using a lexicalized version of a PCFG parser, which would give more weight to associations between lexemes.

Another interesting direction for future research is to explore the use of collocations for ranking the correction suggestions (Bolshakov & Gelbukh 00) and to compare the syntactic- and

lexical-based approaches to identification of the best correction suggestion.

7 Conclusion

This paper has presented a novel method for context-based ranking of suggestions for spelling correction. With the ranking procedure, a set of correction suggestions for a misspelled word is reorganized so that the most probable suggestions given the global context of the misspelled word are placed at the top of the suggestion list. The method is based on the use of probabilistic context-free grammars (PCFGs). The evaluation of the method on data extracted from the Tswana Learner English Corpus has demonstrated that the method provides a significant improvement of 47.89% error rate reduction as compared to the results of non-context-based ranking methods.

Acknowledgments

I would like to thank Gerhard van Huyssteen and the anonymous reviewers for their valuable comments and suggestions on the final version of the paper.

References

(Aston & Burnard 98) Guy Aston and Lou Burnard. *The BNC Handbook*. Edinburgh UP, Edinburgh, 1998.

(Bolshakov & Gelbukh 00) Igor A. Bolshakov and Alexander F. Gelbukh. A very large database of collocations and semantic links. In *Proceedings of NLDB'2000: 5th International Conference on Applications of Natural Language to Information Systems*, pages 103–114, Versailles, France, 2000. Springer-Verlag.

(Davidson 62) Leon Davidson. Retrieval of misspelled names in an airlines passenger record system. *Communications of the ACM*, 5:169–171, 1962.

(Hinrichs & Trushkina 04) Erhard W. Hinrichs and Julia Trushkina. Treebank transformations for performance optimizations of a PCFG-based tagger. In *Proceedings of the International Conference on Linguistic Evidence*, pages 66–70, Tübingen, Germany, 2004.

(Knuth 73) Don Knuth. *The Art of Computer Programming*. Addison-Wesley, 1973.

(Manning & Schütze 99) Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, 1999.

(Schmid 00) Helmut Schmid. Lopar: Design and implementation. Technical Report 149, IMS Stuttgart, 2000. Arbeitspapiere des Sonderforschungsbereiches 340.

(Viterbi 67) A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, IT-13:260–269, 1967.

A Research Taxonomy for Latent Semantic Analysis-Based Educational Applications

Debra Trusso Haley, Pete Thomas, Anne DeRoeck, Marian Petre
The Computing Research Centre
The Open University
Walton Hall, Milton Keynes MK7 6AA UK
[D.T.Haley, P.G.Thomas, A.DeRoeck, M. Petre] at open.ac.uk

Abstract

The paper presents a taxonomy that summarises and highlights the major research into Latent Semantic Analysis (LSA) based educational applications. The taxonomy identifies five main research themes and emphasises the point that even after more than 15 years of research, much is left to be discovered to bring the LSA theory to maturity. The paper provides a framework for LSA researchers to publish their results in a format that is comprehensive, relatively compact, and useful to other researchers.

1 Introduction

The major contribution of this paper is a taxonomy resulting from an in-depth, systematic review of the literature concerning latent semantic analysis (LSA) research in the domain of educational applications. The taxonomy presents the key points from a representative sample of the literature. Researchers and developers implementing LSA-based educational applications will benefit by studying the taxonomy because it brings to one place the techniques and evidence reported in the vast LSA literature.

We realized the need for a taxonomy while building an LSA-based assessment system for use in computer science courses. Although our original assessment results were encouraging, they were not good enough for the intended task of summative assessment (Thomas, Haley, et al. '04). We conducted a comprehensive, in-depth literature review to find techniques to improve our system. The taxonomy documents our findings and supports the insights gained by studying the literature.

There exists a great deal of literature on LSA. Some of it involves educational uses (Steinhart '01), some concentrates on LSA theory (Landauer & Dumais '97), and some of the newer articles[1] suggest uses of LSA that go beyond analysing prose (Marcus, Sergeyev, et al. '04, Quesada, Kintsch, et al. '01).

The literature demonstrated that others were having difficulty matching the results reported by the original LSA researchers. We found a lot of ambiguity in various critical implementation details (e.g. weighting function used) as well as unreported details. We speculate that the conflicting or unavailable information explains at least some of the inability to match the success of the original researchers.

This paper is not an LSA tutorial. Readers desiring a basic introduction to LSA should consult the references section.

Section 2 explains the taxonomy, section 3 discusses insights gained by studying the taxonomy, and section 4 concludes with a suggestion for other LSA researchers.

Space limitation preclude presenting the taxonomy. See the Open University Technical Report 2005/09 at http://computing-reports.open.ac.uk/ for the full, six page taxonomy.

2 About the taxonomy

2.1. Scope of the taxonomy

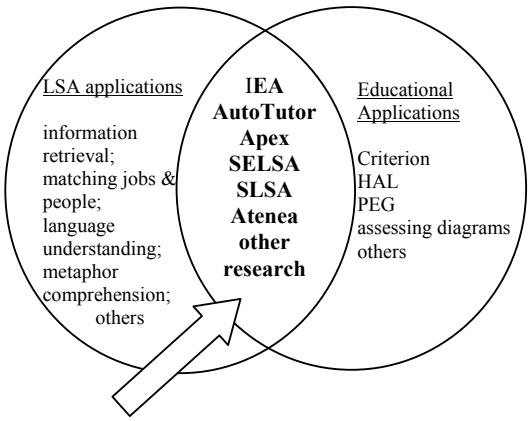


Figure 1. Scope of the Taxonomy – the intersection of LSA and educational applications

[1] To avoid confusion, we refer to papers in the literature as *articles*. *Paper* refers to this paper, which includes a taxonomy.

The taxonomy summarises and highlights important details from the LSA literature. Because the literature is extensive and our interest is in the assessment of essays and related artefacts, the taxonomy includes only those LSA research efforts that overlap with educational applications. Therefore, LSA research into such areas as information retrieval (Nakov, Valchanova, et al. '03) and metaphor comprehension (Lemaire & Bianco '03) do not appear in the taxonomy. Similarly, the taxonomy ignores various non-LSA techniques that have been used to assess essays (Burgess, Livesay, et al. '98, Burstein, Chodorow, et al. '03) and diagrams (Anderson & McCartney '03, Thomas, Waugh, et al. in press).

The next subsections discuss the rationale for choosing certain articles over others and the meaning of the headings in the taxonomy.

2.2. Method for choosing articles

The literature review found 150 articles of interest to researchers in the field of LSA-based educational applications. In order to limit this collection to a more reasonable sample, we constructed a citer – citee matrix of articles. That is, each cell entry (i,j) was non blank if article *i* cited article *j*. The articles ranged in date from perhaps the first LSA published article (Furnas, Deerwester, et al. '88), to one published in May 2005 (Perez, Gliozzo, et al. '05). We found the twenty most-cited articles and placed them, along with the remaining 130 articles, in the categories shown in Table 1.

Type of Article	Number in Lit Review	Number in Taxonomy
most cited	20	13
LSA and ed. applications	43	15
LSA but not ed. apps.	13	0
LSI	11	0
theoretical / mathematical	11	0
reviews / summaries	11	0
ed. apps. but not LSA	41	0
Total	**150**	**28**

Table 1. Categories of articles in the literature review and those that were selected for the taxonomy

We chose the twenty most-cited articles for the taxonomy. Some of these most-cited articles were early works explaining the basic theory of Latent Semantic Indexing (LSI).[2] Although not strictly in the scope of the intersection of LSA and educational applications, some of these articles appear in the taxonomy because of their seminal nature. Next, we added articles from the category that combined educational applications with LSA that were of particular interest, either because of a novel domain or technique, or an important result. Finally, we decided to reject certain heavily cited articles because they present no new information pertinent to the taxonomy. This left us with 28 articles in the taxonomy.

2.3. The taxonomy categories

The taxonomy organises the articles involving LSA and educational applications research into three main categories: an *Overview*, *Technical Details*, and *Evaluation*. Figures 2, 3, and 4 show the headings and sub-headings. Most of the headings are self-explanatory; some clarifications are noted in the figures.

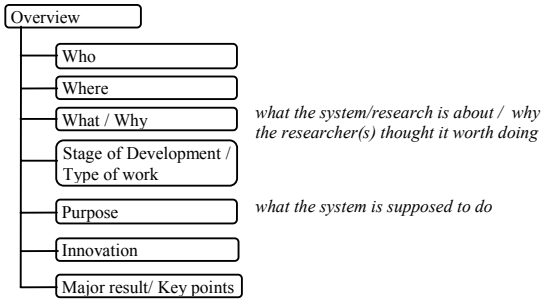


Figure 2. Category A: Overview

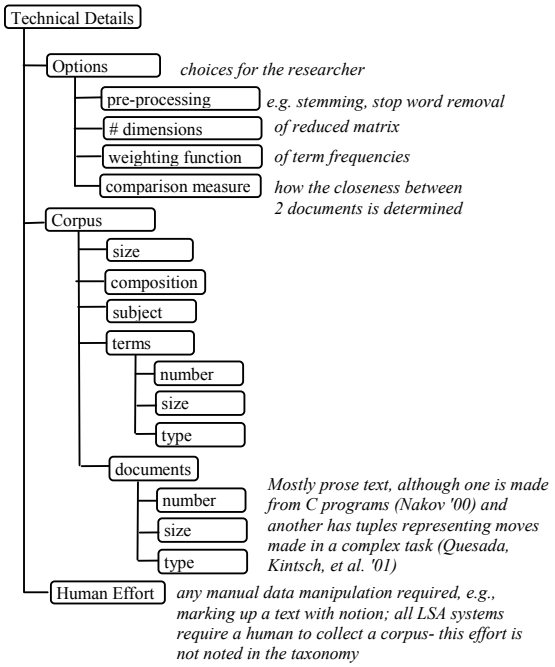


Figure 3. Category B: Technical Details

[2] Researchers trying to improve information retrieval produced the LSI theory. Later, they found that LSI could be useful to analyse text and created the term LSA to describe LSI when used for this additional area.

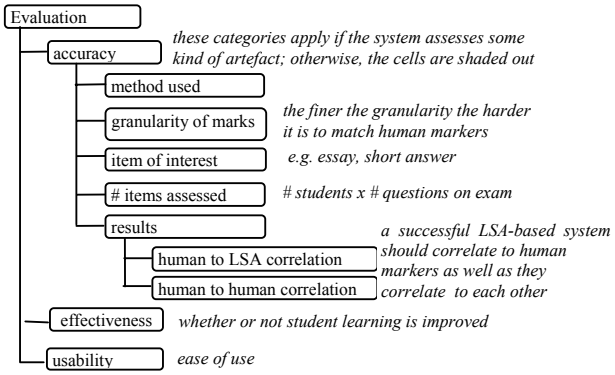


Figure 4. Category C: Evaluation

When looking at the taxonomy, the reader should keep a few points in mind. First, each line presents the data relating to one study. However, one article can report on several studies. In this case, several lines are used for a single article. The cells that would otherwise contain identical information are merged. Second, the shaded cells indicate that the data item is not relevant for the article being categorised. Third, blank cells indicate that we were unable to locate the relevant information in the article.[3] Fourth, the information in the cells was summarised or taken directly from the articles. Thus, the *Reference* column on the far left holds the citation for the information on the entire row. [4]

Organising a huge amount of information in a small space is not easy. The taxonomy in the technical report (http://computing-reports.open.ac.uk) is based on an elegant solution in (Price, Baecker, et al. '93).

3 Discussion

This section discusses the insights revealed by the taxonomy. Sections 3.1 and 3.2 describe what can be found in the literature, and section 3.3 highlights some of the gaps in the literature.

3.1. Main research themes

A great deal of literature exists about LSA and about educational applications. Even the intersection of these two areas contains many articles. However, the taxonomy reveals five main research themes:

- seminal literature describing the new technique named LSI, which was later renamed to LSA
- attempts to reproduce the results reported in the seminal literature, which for the most part failed to achieve the earlier results
- attempts to improve LSA by adding syntax information
- applications that analyse non-prose text.
- attempts to improve LSA by experimenting with corpus size and composition, weighting functions, similarity measures, number of dimensions in the reduced LSA matrix, and various pre-processing techniques – exactly those items in Category B1 of the taxonomy

3.2. Diversity in the research

The taxonomy reveals a great deal of variety in the research. Researchers work in North America, Europe, and Asia on both deployed applications and continuing research. They use a wide variety of options for pre-processing techniques, number of dimensions in the reduced matrix, weighting functions, and composition and size of corpus. They use English, French, Spanish and Bulgarian corpora. The researchers report their evaluation methods with different specificity.

3.3. Gaps in the literature

The great variety of techniques used by researchers mentioned in the previous section leads to difficulty in comparing the results. Other researchers need to know all of the details to fully evaluate and compare reported results.

Much information is missing on page 2 of the taxonomy – *Category B: Technical Details*. These missing data concern the choices researchers must make when they implement their systems. Page 3 of the taxonomy, *Category C: Evaluation*, shows that some researchers have not evaluated the effectiveness or usability of their deployed systems.

The *Method used* subheading under *Accuracy* in *Category C* is a major area for gaps. Although many researchers report correlations between LSA and human graders, they usually do not mention whether they are using the Pearson, Spearman, or Kendall's tau correlation measure.

The existence of the blank cells in the taxonomy is troubling. They imply that researchers often neglect to report critical information, perhaps due to an oversight or page length restrictions. Nevertheless, the ability to reproduce results would be enhanced if more researchers provided more detailed data regarding their LSA implementations.

[3] Please send any corrections to the first author, who will gladly update the taxonomy.

[4] The *Reference* column contains a pointer to the references section at the end of this paper. Each reference contains a code at the end that corresponds to the entry in the *Reference* column. The entries are of the form *xxxnn* where *xxx* are the initials of up to three of the authors. If capitalised, they represent different authors; if the first is capitalised and the second two are lower case, the article has one author. *nn* is the 2-digit year of publication.

4 Conclusions

We hope that future LSA researchers will keep the taxonomy in mind when presenting their work. Using it will serve two main purposes. First, it will be easier to compare various research results. Second, it will ensure that all relevant details are provided in published articles, which will lead to improved understanding and the continued development and refinement of LSA.

The variability in the results documented in the taxonomy shows that LSA is still something of an art. More than 15 years after its invention, the research issues suggested by (Furnas, Deerwester, et al. '88) are still very much open.

Acknowledgements

The work reported in this study was partially supported by the European Community under the Innovation Society Technologies (IST) programme of the 6th Framework Programme for RTD - project ELeGI, contract IST-002205. This document does not represent the opinion of the European Community, and the European Community is not responsible for any use that might be made of data appearing therein.

References

(Anderson & McCartney '03) M. Anderson & R. McCartney, Diagram processing: Computing with Diagrams. Artificial Intelligence, vol. 145, pp. 181-226, 2003 [AM03].

(Bassu & Behrens '03) D. Bassu & C. Behrens. Distributed LSI: Scalable concept-based information retrieval with high semantic resolution. In Proceedings of Text Mining 2003, a workshop held in conjunction with the Third SIAM Int'l Conference on Data Mining. pp, San Francisco, 2003 [BB03].

(Berry, Dumais, et al. '95) M. W. Berry, S. T. Dumais & G. W. O'Brien, Using linear algebra for intelligent information retrieval. SIAM Review 37, vol. 4, pp. 573-595, 1995 [BDO95].

(Burgess, Livesay, et al. '98) C. Burgess, K. Livesay & K. Lund, Explorations in context space: Words, sentences, discourse. Discourse Processes, vol. 25, pp. 211-257, 1998 [BLL98].

(Burstein, Chodorow, et al. '03) J. Burstein, M. Chodorow & C. Leacock. Criterion Online Essay Evaluation: An Application for Automated Evaluation of Student Essays. In Proc. of the Fifteenth Annual Conference on Innovative Applications of Artificial Intelligence. pp, Acapulco, Mexico, 2003 [BCL03].

(Deerwester, Dumais, et al. '90) S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer & R. Harshman, Indexing by Latent Semantic Analysis. Journal of the American Society for Information Science, vol. 41, pp. 391-407, 1990 [DDF90].

(Dumais '91) S. T. Dumais, Improving the retrieval of information from external sources. Behavioral Research Methods, Instruments & Computers, vol. 23, pp. 229-236, 1991 [Dum91].

(Foltz, Britt, et al. '96) P. W. Foltz, M. A. Britt & C. A. Perfetti. Reasoning from multiple texts: An automatic analysis of readers' situation models. In 18th Annual Cognitive Science Conference. pp 110-115, NJ, 1996 [FBP96].

(Foltz, Kintsch, et al. '98) P. W. Foltz, W. Kintsch & T. K. Landauer, The Measurement of Textural Coherence with Latent Semantic Analysis. Discourse Process, vol. 25, pp. 285-307, 1998 [FKL98].

(Foltz, Laham, et al. '99) P. W. Foltz, D. Laham & T. K. Landauer. Automated Essay Scoring: Applications to Educational Technology. In Proceedings of EdMedia '99. pp, 1999 [FLL99].

(Franceschetti, Karnavat, et al. '01) D. R. Franceschetti, A. Karnavat, J. Marineau, G. L. McCallie, B. A. Olde, B. L. Terry & A. C. Graesser. Development of Physics Text Corpora for Latent Semantic Analysis. In Proc. of the 23rd Annual conference of the Cognitive Science Society. pp, 2001 [FKM01].

(Furnas, Deerwester, et al. '88) G. W. Furnas, S. Deerwester, S. T. Dumais, T. K. Landauer, R. A. Harshman, L. A. Streeter & K. E. Lochbaum. Information retrieval using a singular value decomposition model of latent semantic structure. In Proc. of 11th annual int'l ACM SIGIR conference on Research and development in information retrieval. pp 465-480, 1988 [FDD88].

(Kanejiya, Kumar, et al. '03) D. Kanejiya, A. Kumar & S. Prasad. Automatic Evaluation of Students' Answers using Syntactically Enhanced LSA. In Building Educational Applications Using Natural Language Processing, Proc. of the HLT-NAACL 2003 Workshop. pp 53-60, 2003 [KKP03].

(Kintsch, Steinhart, et al. '00) E. Kintsch, D. Steinhart, G. Stahl, C. Matthews & R. Lamb, Developing summarization skills through the use of LSA-based feedback. Interactive Learning Environments. [Special Issue, J. Psotka, guest editor], vol. 8, pp. 87-109, 2000 [KSS00].

(Landauer '02) T. K. Landauer. On the computational basis of learning and cognition: Arguments from LSA. In The Psychology of Learning and Motivation. edited by B. Ross, 41, pp 43-84, New York, 2002 [Lan02b]

(Landauer & Dumais '97) T. K. Landauer & S. T. Dumais, A solution to Plato's problem: the Latent Semantic Analysis theory of acquisition, induction and representation of knowledge. Psychological Review, vol. 104, pp. 211-240, 1997 [LD97].

(Landauer, Foltz, et al. '98) T. K. Landauer, P. W. Foltz & D. Laham, An introduction to Latent Semantic Analysis. Discourse Processes, vol. 25, pp. 259-284, 1998 [LFL98].

(Landauer, Laham, et al. '97) T. K. Landauer, D. Laham, B. Rehder & M. E. Schreiner. How Well Can Passage Meaning be Derived without Using Word Order? A Comparison of Latent Semantic Analysis and Humans. In Proceedings of the 19th Annual Meeting of the Cognitive Science Society. pp 412-417, 1997 [LLR97].

(Lemaire & Bianco '03) B. Lemaire & M. Bianco. Contextual effects on metaphor comprehension: Experiment and simulation. In Proceedings of the 5th Int'l Conference on Cognitive Modeling (ICCM'2003). pp, Bamberg, Germany, 2003 [LB03].

(Lemaire & Dessus '01) B. Lemaire & P. Dessus, A system to assess the semantic content of student essays. J. of Educational Computing Research, vol. 24, pp. 305-320, 2001 [LD01].

(Marcus, Sergeyev, et al. '04) A. Marcus, A. Sergeyev, V. Rajlich & J. I. Maletic. An Information Retrieval Approach to Concept Location in Source Code. In Proceedings of the 11th IEEE Working Conference on Reverse Engineering. pp 214-223, Delft, The Netherlands, 2004 [MSR04].

(Nakov '00) P. Nakov. Latent Semantic Analysis of Textual Data. In Proceedings of the Int'l Conference on Computer Systems and Technologies. pp, Sofia, Bulgaria, 2000 [Nak00b].

(Nakov, Popova, et al. '01) P. Nakov, A. Popova & P. Mateev. Weight functions impact on LSA performance. In Proc. of the EuroConference Recent Advances in Natural Language Processing (RANLP'01). pp, Tzigov Chark, Bulgaria, 2001 [NPM01].

(Nakov, Valchanova, et al. '03) P. Nakov, E. Valchanova & G. Angelova. Towards Deeper Understanding of the LSA Performance. In Proc. of Recent Advances in Natural Language Processing. pp 311-318, Borovetz, Bulgaria, 2003 [NVA03].

(Olde, Franceschetti, et al. '02) B. A. Olde, D. R. Franceschetti, A. Karnavat & A. C. Graesser. The Right Stuff: Do you need to sanitize your corpus when using Latent Semantic Analysis? In Proceedings of the 24th Annual Meeting of the Cognitive Science Society. pp 708-713, Fairfax, 2002 [OFK02].

(Perez, Gliozzo, et al. '05) D. Perez, A. Gliozzo, C. Strapparava, E. Alfonseca, P. Rodriquez & B. Magnini. Automatic Assessment of Students' free-text Answers underpinned by the combination of a Bleu-inspired algorithm and LSA. In Proceedings of the 18th Int'l FLAIRS Conference. pp, Clearwater Beach, Florida, 2005 [PGS05].

(Price, Baecker, et al. '93) B. A. Price, R. M. Baecker & I. S. Small, A Principled Taxonomy of Software Visualization. Journal of Visual Languages and Computing, vol. 4, pp. 211-266, 1993 [PBS93].

(Quesada, Kintsch, et al. '01) J. Quesada, W. Kintsch & E. Gomez, A computational theory of complex problem solving using the vector space model (part 1): Latent Semantic Analysis, through the path of thousands of ants. Cognitive Research with Microworlds, vol. 43-84, pp. 117-131, 2001 [QKG01a].

(Rehder, Schreiner, et al. '98) B. Rehder, M. E. Schreiner, M. B. W. Wolfe, D. Laham, T. K. Landauer & W. Kintsch, Using Latent Semantic Analysis to assess knowledge: some technical considerations. Discourse Process, vol. 25, pp. 337-354, 1998 [RSW98].

(Steinhart '01) D. J. Steinhart, Summary Street: An intelligent tutoring system for improving student writing through the use of Latent Semantic Analysis. Unpublished PhD Thesis, Department of Psychology, University of Colorado, Boulder, 2001 [Ste01].

(Thomas, Haley, et al. '04) P. Thomas, D. Haley, A. De Roeck & M. Petre. E-Assessment using Latent Semantic Analysis in the Computer Science Domain: A Pilot Study. In Proc. of the eLearning for Computational Linguistics and Computational Linguistics for eLearning Workshop at COLING 2004. pp 38-44, Geneva, 2004 [THD04].

(Thomas, Waugh, et al. in press) P. Thomas, K. Waugh & N. Smith. Experiments in the automatic marking of ER-Diagrams. In Proc. of ITiCSE 05. pp, Lisbon, Portugal, in press [TWS05].

(Wiemer-Hastings '00) P. Wiemer-Hastings. Adding syntactic information to LSA. In 22nd Annual Conference of the Cognitive Science Society. pp 989-993, 2000 [Wie00].

(Wiemer-Hastings & Graesser '00) P. Wiemer-Hastings & A. C. Graesser, Select-a-Kibitzer: A computer tool that gives meaningful feedback on student compositions. Interactive Learning Environments, vol. 8, pp. 149-169, 2000 [WG00].

(Wiemer-Hastings, Wiemer-Hastings, et al. '99) P. Wiemer-Hastings, K. Wiemer-Hastings & A. C. Graesser. Improving an intelligent tutor's comprehension of students with Latent Semantic Analysis. In Artificial Intelligence in Education. pp, Amsterdam, 1999 [WWG99].

(Wiemer-Hastings & Zipitria '01) P. Wiemer-Hastings & I. Zipitria. Rules for Syntax, Vectors for Semantics. In Proc. of the 23rd Cognitive Science Conference. pp, 2001 [WZ01].

(Wolfe, Schreiner, et al. '98) M. B. W. Wolfe, M. E. Schreiner, B. Rehder, D. Laham, P. W. Foltz, W. Kintsch & T. K. Landauer, Learning from text: Matching readers and texts by Latent Semantic Analysis. Discourse Processes, vol. 25, pp. 309-336, 1998 [WSR98].

Finite-State Morphology of Estonian: Two-Levelness Extended

Heli UIBO
Institute of Computer Science
University of Tartu
J. Liivi 2
Tartu 50409, Estonia
Heli.Uibo@ut.ee

Abstract

The paper is concentrated on modeling the Estonian morphology in the framework of two-level morphology model. The result is a consistent description of Estonian morphology, which consists of a network of lexicons (root lexicons cover 2500 most frequent word roots) and two-level rules. The main rule set contains 45 rules, which describe various stem changes. The subset of rules dealing with stem internal changes is applied separately as well. For modeling the derivation process a new solution has been found – to extend the two-levelness into the upper side of the morphological transducer (to the lemmas). It has been shown that finite-state methods are applicable and sufficient for describing Estonian inflectional processes, but word formation rules, especially compounding, require more investigation.

1 Introduction

During the last 25 years the finite-state approach has been the most fruitful one in the field of computational morphology. Although there exist two computerized descriptions of the Estonian morphology (Viks 00; Kaalep 00) it is worth to try to apply finite-state techniques to the Estonian morphology, to make the results comparable to those of other languages.

It is important that a finite-state transducer is bidirectional in its nature, as it describes a regular relation, or a correspondence between two languages. In the simplest case the morphological transducer is a lexical transducer, on the upper side of which are primary forms concatenated with appropriate morphological information and on the lower side – word forms. Each path from the initial state to a final state represents a mapping between a word form and its morphological reading. The morphological analysis can then be understood as the "lookup" operation in the lexical transducer, whereas synthesis – the "lookdown" operation (Beesley & Karttunen 03). The lexical transducer can be composed with rule transducer(s) that convert lexical representation to surface representation, using either two-level (Koskenniemi 83) or replace rules (Karttunen 95).

2 Finite-state morphology of Estonian

2.1 Overview

Estonian is a highly inflected language – grammatical meanings are expressed by grammatical formatives which are affixed to the stem instead of using prepositions. According to more detailed analysis the stem consists of word root and derivational affixes and formative – of features and endings.

The morphological word classes in Estonian:

- nouns (can be declined)
- verbs (can be conjugated)
- indeclinables (remain unchanged)

Nouns have 14-15 cases in singular and plural, there are often parallel forms in plural. Verbs have four moods (indicative, conditional, imperative, quotative), four tenses (present, imperfect, present perfect and past perfect), two modes (personal and impersonal), two voices (affirmative and negative), three persons and two numbers (singular and plural). Derivation is mostly done by affixing:

kiire (Adj) 'quick' *kiire*|*sti* (Adv) 'quickly'
õppi|*ma* (V) 'to learn' *õppi*|*mine* (N) 'learning'

For compounding the concatenation of stems is used. The pre-components of compound nouns can be either in singular nominative, singular genitive and in some cases in plural genitive case. Only the last component is declinable.

Example:
piiri + valve + väe + osa = piirivalveväeosa
border guard power part = 'troup of border guards'
sg gen sg gen sg gen sg nom

There generally exist two different processes in natural language morphology:

1. morphotactics – how to combine word forms from morphemes
   a) concatenative processes (prefixation and suffixation, compounding)
   b) non-concatenative processes (reduplication, infixation, interdigitation)
2. phonological alternations (examples from Estonian)
   a) assimilation (*hi**nd**:hi**nn**a* 'price' sg nom : gen)
   b) insertion (*jooksma:jooks**e**v* 'to run' : 'running')
   c) deletion (*numb**e**r:numbri* 'digit' sg nom : gen)
   d) gemination (*tu**b**a:tu**pp**a* 'room' sg nom : adit)

It has been shown in (Beesley & Karttunen 00) that concatenation, composition and iteration are sufficient means for describing the morphology of languages with concatenative morphological processes. The Estonian morphotactics does not make use of productive non-concatenative processes, thus, theoretically, no problems should occur by the modeling the Estonian morphology by finite-state methods.

The morphological description of Estonian has been built up by the author, lead by the principles of the two-level morphology model (Koskenniemi 83). The two-levelness means that the lexical representations of morphemes are maintained in the lexicons and the task of two-level rules is to "translate" the lexical forms into the surface forms and vice versa. The lexical forms may contain information about the phoneme alternations, about the structure of the word form (morpheme boundaries) etc.

The model is language-independent, but for the different languages the balance between rules and lexicons can be different. The network of lexicons is good for agglutinating languages like Finnish (Koskenniemi 83), Turkish (Oflazer 94) and Swahili (Hurskainen 95), where word forms are built by concatenation of morphemes. Two-level rules are convenient to handle single phoneme alternations. If the stem variants differ more from each other (e.g. *pidu:peo* ('party' sg nom : sg gen) then the stem change can be handled analytically (cf. section 2.4). The Estonian language is both agglutinative and flective. For instance, the word form *hammastega* 'with teeth' is built from the morphemes *hammas+te+ga* and stem flexion rules determine that the stem variant is *hammas* but not *hamba*.

The morphological phenomena occurring in the Estonian language have been divided between rules and lexicons as follows:

- The rules of phonotactics, different stem flexion types and morphological distribution have been formalized as two-level rules.
- The rules of morphotactics have been described in the network of lexicons.
- The stem final alternations have been divided between lexicons and rules. Most of the alternations concerning only one grapheme have been formalized as rules. Handling the change of a whole segment by two-level rules requires several rules to be coordinated (Trosterud & Uibo 05) and therefore, the stem final changes like *hobune : hobuse : hobust* are handled by continuation lexicons.

2.2 The network of lexicons

The network of lexicons was designed after the morphological classification by Ülle Viks (Viks 92), which is based on pattern recognition. It is compact and oriented for automatic morphological analysis. It contains 38 inflection types – 26 for nouns and 12 for verbs. 84 words (including most of the pronouns) are handled as exceptions. We have additionally splitted some noun types according to the stem final vowel.

Each inflection type has been modeled as a number of linked lexicons. The first group generates stem variants (lexicon 28 in Figure 2), the second group locates the stem variants in paradigm (lexicons TP_28at and TP_28an) and the third builds the base forms and their analogy forms (lexicons An_ma … An_takse). This kind of structure has been inspired by (Viks 92).

The paradigms of all the noun and verb inflection types have been described in the network of lexicons. Comparison of adjectives, productive derivation and compounding have also been implemented, using continuation lexicons. The word formation rules are too general yet. Nevertheless, the problem is application-dependent. For information retrieval, the problem of overgeneration is of less importance than for spelling check (Uibo 02).

2.3 Problems with lexicons

The network of lexicons seems to be a powerful tool: following the links between different lexicons word roots, derivation suffixes, inflectional features and endings can be combined into grammatical word forms. However, a number of problems occured in practice:

- As there are many inflection types in Estonian, the number of continuation lexicons is also

high (164) and the network of lexicons becomes difficult to manage. But the number could be even bigger if we did not use two-level rules for handling stem internal changes (Trosterud & Uibo 05).

- Using word lists does not fit into the model, however it is needed to constrain the overgenerating derivation and compounding.
- The principle that the rules of morphotactics and the distribution of stem variants are described by lexicons and the phonological relations of stem variants are formalized as two-level rules cannot always be followed. Stem final alternations have often become individual properties of a word and are not predictable by phonological rules.
- The network of lexicons would be best readable if for each morpheme there is exactly one lexicon. In the existing network of lexicons the morphemes are often splitted that cuts the readability down.

2.4 Rules

The majority of two-level rules handle stem flexion and phonotactics. The most interesting inflection type from the point of view of phonological changes is characterized by weakening consonant gradation – the deletion of *b*, *d*, *g* or *s* – and also changes in the immediate neighbourhood of the disappeared consonant – the lowering of the surrounding vowels.

Example list of words belonging to the type:

*ma**d**u : mao*	*si**g**a : sea*	*u**b**a : oa*
*lu**g**u : loo*	*kä**s**i : käe*	*sü**s**i : söe*

There should be a rule for handling the deletion ($ is the weak grade marker):

```
SC:0 <=> Vok: _ Vok: %$:;
```

And another rule for vowel lowering:

```
HVow:LVow<=>Bgn_LV:  StemVow:  %$:  ;
                Bgn Vow: LV: _ %$: ;
where     HVow in (u ü i)
          LVow in (o ö e)
matched ;
```

The last rule has two contexts: the lowering can occur both in the right (*mad**u** : mao*) and in the left context (*si**g**a : se**a***). In (Uibo 00) the stem flexion types and the discovery process of rules have been discussed in details. Figure 1 gives an overview of the whole rule set.

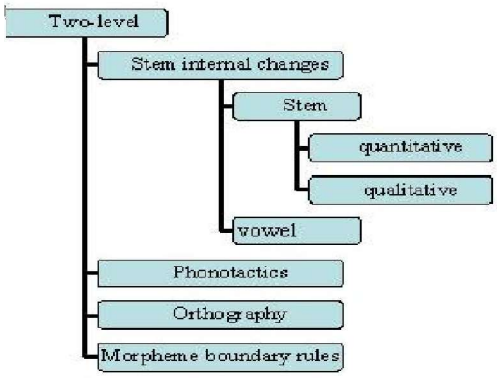


Figure 1: Two-level rules for Estonian

3 A new approach to word formation modeling – two-levelness extended

All Estonian verbs are subject to productive derivation processes resulting in the word forms exemplified in Table 1.

Deriv. suffix	Example *(lugema)*	Translation (*to read*)	Word class
-ja	**luge**ja	reader (person)	Subst
-mine	**luge**mine	reading (process)	Subst
-v	**luge**v	reading	Adj
-tav	**loe**tav	being read	Adj
-nud	**luge**nud	having read	Adj
-tud	**loe**tud	read (finished)	Adj
-nu	**luge**nu	one who has read	Subst
-tu	**loe**tu	one that has been read	Subst

Table 1: Productive derivation from verbs

Modeling the productive derivation from verbs with **weakening consonant gradation**, i.e. verbs for which the primary form (supine) is in the strong grade but some inflected forms in the weak grade, we have run into a serious problem. Namely, the information for the derived word form, outputted during the analysis, should contain the derived primary form, which can be in the weak grade (*loetav, loetud, loetu*).

The lexical transducer picks up the strong-grade stem and the word class V (verb), but it may be a derived word with a weak lemma. The initial solution was to include the weakening verbs into root lexicons three times – into the root lexicon of verbs and into the root lexicon of verbal derivatives in both strong and weak grade (Figure 2).

```
LEXICON Verb  ! Root lexicon
  lugema+V : luGe 28;

LEXICON 28  ! Building of stem variants
        TP_28at;
: $     TP_28an;

! Distribution of stem variants in the paradigm
LEXICON TP_28at     ! luge+…
 An_ma;
 Am_mata;
 An_v;
 An_sin;
 An_sime;
 An_da;
 An_ge;
 Ja_mine;

LEXICON  TP_28an ! loe+…
 An_b;
 An_me;
 An_tud;
 An_takse;

! Base forms and their analogy forms.

LEXICON An_ma
 ma+V+sup+ill :                 ma        GI;
 ma+V+quot+pres+ps :            vat       GI;

LEXICON An_v
 ma+V+partic+pres+ps :          v         GI;
 v+A+pos+sg+nom+partic :        v         02_A;
 …

LEXICON An_takse
 ma+V+indic+pres+imps+af :      takse   GI;

! Productive derivation
LEXICON Verb-Deriv
  loe Partic/N-N;
  luge Partic/N-T;

LEXICON Partic/N-N
  tav+A :                       tav       A_02_A;
  tav+S :                       tav       Axx;
  tud+A+Sg+N :                  +tud      #;
  tu+S :                        tu        01;

LEXICON Partic/N-T
  v+A  :        v         A_02_A;
  nud+A  :      nud       #;
  nu+S  :       nu        01;
                          Ja_mine;
LEXICON Ja_mine
 ja+S :         +ja       01;
 mine :         +m        12_nE-SE-S;
 mata+A :       +mata     #;
```

Figure 2: Derivation from verbs: a storage consuming solution

Finally we have found a helpful solution to the weak grade verb derivatives problem: **to extend the two-levelness to the upper side of the lexical transducer** (to the lemma). The solution has been implemented as sketched on Figure 3.

```
LEXICON Verb
  luGe 28;

! inflection like in figure 2; skipped
! productive derivation
LEXICON 28_deriv
 ja+S          :         +ja       Szz;
mine+S         :         +mine     Sqq;
v+A            :         +v        Aww;
$tav+A         :         @+tav     Aww;
+nud+G         :         +nud      #;
$tud+G         :         $+tud     #;
nu+S           :         +nu       Scc;
$tu+S          :         $+tu      Sdd;

LEXICON Substantiiv
 Scc;
  ...
LEXICON Adjektiiv
Aww;
```

Figure 3: Derivation modeling: a better solution

As a result, the productive verb derivatives do not require three, but only one record in the root lexicon. To get the lemma in the correct surface form, stem flexion rules have to be applied onto the upper side of the lexical transducer. The resulting morphological transducer of Estonian can be formulated as follows:

$$((\text{LexiconFST})^{-1} \circ \text{RulesFST}_1)^{-1} \circ \text{RulesFST}$$

Here LexiconFST is the lexical transducer, RulesFST is the rule transducer (the intersection of all two-level rules) and RulesFST_1 is the intersection of consonant gradation rules. The operations used are composition and inversion.

The percentage of verbs is about 15 % among the 10 000 most frequent words of written Estonian (Kaalep & Muischnek 02). Thus, after the extension of two-levelness the number of records in root lexicons will decrease ca 23 %.

4 Implementation

The rules and lexicons are compiled into finite-state transducers using the Xerox finite-state tools *twolc* (Karttunen & Beesley 92) and *lexc* (Karttunen 93).

In the course of the project some additional tools have been developed:

- A tool for automatic updating of root lexicons (generates the lexical representation and detects the inflection type);
- A tool for testing the morphological analyzer on correctly tagged corpus. The program lists the words tagged correctly and incorrectly as well as unknown words.

The testing and lexicon extending cycle will go on, as the present coverage of the lexicon is about 30 % only.

5 Conclusion and perspectives

The finite-state approach has been resulted in a consistent description of the Estonian morphology, consisting of a network of lexicons and two rule sets. 45 rules that handle stem flexion, phonotactics, orthography and morphophonological distribution. A subset of stem flexion rules is used separately as well. The root lexicons contain 2500 most frequent words, based on the frequency dictionary of Estonian (Kaalep & Muischnek 02). There are 164 continuation lexicons which describe stem final changes, noun declination, verb conjugation, derivation and compounding.

A new solution has been proposed for modeling derivation: two-levelness has been partly extended to the upper side of the lexical transducer – to the lexical representations of the lemmas of forms productively derivable from the verb stems. The proposed approach may be applied for other languages where the word stems change in the course of derivation.

It has been shown that two-level representation is useful for the description of the stem internal changes, especially because the stem flexion does not depend on the phonological shape of a stem in the contemporary Estonian any more. The network of lexicons, combined with rules, having effect on morpheme boundaries, naturally describe the morphotactic processes. Lexicons are also good for describing non-phonological stem end alternations.

However, some open problems remain to be solved for the Estonian finite-state morphology:

- To increase the coverage of root lexicons.
- To guess the analysis of unknown words. The idea is to include a regular expression (e.g. $CVVC^{+}V$) in the root lexicon for each productive inflection type.
- To constrain the overgeneration of compound words by semantic constraints.
- To include the finite-state morphological component into practical applications. The most interesting idea in this perspective is to work on fuzzy information retrieval that is tolerant to misspellings and typos.

6 Acknowledgements

The research on finite-state morphology of Estonian has been supported by the Estonian Science Foundation grant No. 4605. Our thanks also go to Kimmo Koskenniemi, Lauri Karttunen, Kenneth Beesley and Trond Trosterud for encouraging discussions.

References

(Beesley & Karttunen 00) K. Beesley, L. Karttunen. *Finite-State Non-Concatenative Morphotactics*. In "Proceedings of SIGPHON-2000" 5th Workshop of the ACL Special Interest Group in Computational Phonology, Centre Universitaire, Luxembourg. 1-12.

(Beesley & Karttunen 03) K. Beesley, L. Karttunen. *Finite State Morphology*. CSLI Studies in Computational Linguistics. CSLI Publications. Stanford, USA 2003.

(Kaalep & Muischnek 02) H.-J. Kaalep, K. Muischnek. *Eesti keele sagedussõnastik*. (*The frequency dictionary of written Estonian*). University of Tartu Press, Tartu 2002.

(Hurskainen 95) A. Hurskainen. *Information Retrieval and Two-Directional Word Formation.* Nordic Journal of African Studies 4 (2): 81-92 (1995).

(Karttunen 93) L. Karttunen. *Finite-State Lexicon Compiler.* Technical Report. ISTL-NLTT-1993-04-02. April 1993. Xerox Palo Alto Research Centre. Palo Alto, California.

(Karttunen 95) L. Karttunen. *The Replace Operator*. Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics. ACL-95, pp. 16-23, Boston, Massachusetts.

(Karttunen & Beesley 92) L. Karttunen, K. Beesley. *Two-Level Rule Compiler*. Technical Report. ISTL-92-2. Xerox Palo Alto Research Centre. Palo Alto, California.

(Koskenniemi 83) K. Koskenniemi. *Two-level Morphology*: *A General Computational Model for Word-Form Recognition and Production*. University of Helsinki, Dept of General Linguistics. Publications No. 11. Helsinki 1983.

(Oflazer 94) K. Oflazer. *Two-level Description of Turkish Morphology*, Literary and Linguistic Computing, Vol. 9, No:2 (1994).

(Trosterud & Uibo 05) T. Trosterud, H. Uibo. *Consonant gradation in Estonian and Sámi: two-level solutions.* Festschrift in honor of Professor Kimmo Koskenniemi's 60th anniversary. CSLI Publications 2005.

(Uibo 00) H. Uibo. *Kahetasemeline morfoloogiamudel eesti keele arvutimorfoloogia alusena*. (*Two-level morphology model as a basis for computational morphology of Estonian*) In "Arvutuslingvistikalt inimesele." Publications of the Department of General Linguistics, University of Tartu No. 1, pp. 37-72. Tartu 2000.

(Uibo 02) H. Uibo. *Experimental Two-Level Morphology of Estonian.* In "LREC 2002. Third International Conference on Language Resources and Evaluation." Las Palmas de Gran Canaria, Spain. Proceedings. Vol. III. pp. 1012 – 1015.

(Viks 92) Ü. Viks. *A Concise Morphological Dictionary of Estonian I.* Introduction & Grammar. Tallinn 1992.

(Viks 00) Ü. Viks. *Eesti keele avatud morfoloogiamudel.* (*Open morphology model of Estonian*). In "Arvutuslingvistikalt inimesele." Publications of the Department of General Linguistics, University of Tartu No. 1, pp. 9-36. Tartu 2000.

Knowledge Acquisition for Fine-grained Named Entity Classification

Olga Uryupina
Computational Linguistics, Saarland University
Saarbruecken 66041, Germany
ourioupi@coli.uni-sb.de

Abstract

This paper presents a novel web-based bootstrapping approach to the lexicon acquisition for fine-grained NE classification. We evaluate the algorithm performance and its impact on a context-based NE subclassification system motivated by (Fleischman & Hovy 02), achieving 30-75% error reduction on a small-scale corpus. We also show that different NE classes (PERSON vs. LOCATION) rely on different classification clues — lexical or syntactic knowledge.

1 Introduction

Named Entity Recognition (NER) is an important preprocessing step for a variety of NLP tasks such as Information Extraction, Question Answering or Coreference Resolution. Most state-of-the-art NER systems support primarily coarse-grained classification: for example, the MUC scheme distinguishes only among PERSON, ORGANIZATION, LOCATION, TIME, MONEY, and PERCENTAGE.

It has been shown that NLP systems could benefit from more fine-grained NER. For example, Srihari & Li (99) have added new classes (e.g., PRODUCT) and subclasses (e.g., SCHOOL as a subtype of ORGANIZATION). Using such a tagset, they achieved the best score in TREC-8 QA.

One of the main problems for successful NE classification is the lack of lexical knowledge about names (gazetteers). Collecting gazetteers by hand is a very time-consuming task, and the work that has already been done for coarse classes cannot be re-used for finer ones. In the present study we explore possibilities to extract relevant lexical information automatically and incorporate it into a fine-grained NER system. We currently focus on two major NE classes: LOCATION and PERSON. In future, we plan to extend our approach to cover other MUC NE classes.

The main contributions of our study are twofold. First, we propose a novel web-based bootstrapping algorithm for acquiring lexical knowledge for subclassifying NEs. Second, we combine our automatically acquired gazetteers with the features of (Fleischman & Hovy 02) to build a resolution system, proposing an algorithm for generating training data. We believe that our approach helps relieve knowledge acquisition bottleneck for NE classification making it more portable.

2 Relation to previous work

Although several high-quality coarse-grained NER systems have been proposed in the literature, we are aware of only very few approaches supporting fine-grained classes (see (Fleischman & Hovy 02; Evans 03) and references therein for detailed discussion).

Our bootstrapping algorithm is based on counter-training (Yangarber 03), but has important differences. Counter-training is a technique that deals with the increasing amount of noise during bootstrapping process: different classes are processed simultaneously and, thus, constrain each other. Counter-training alone can not deal with such a noisy dataset as the Internet. So, we have developed an additional control strategy: we use the information from a machine learner to re-score and discard patterns and items.

Fleischman and Hovy (02) propose an algorithm for fine-grained NER with context features. However, their experiments show that even humans cannot reliably subcategorize NEs based solely on the context (when the name is encrypted). We use their features to model NEs' contexts, but, in addition, we incorporate automatically acquired lexical information and syntactic heuristics. As section 4.3 shows, these improve the system's performance on our corpus.

3 Bootstrapping NE Gazetteers

In this section we present our bootstrapping approach for extracting NE gazetteers from the web data. The algorithm is shown on Figure 1 and described below.

Manually compiled gazetteers provide high-quality data. Unfortunately, these resources have some drawbacks. First, some items can simply be missing. For example, most World atlases do not list small ISLANDs. Second, adjusting such resources to specific domains, for example, introducing new classes, might be very time-consuming. Finally, for some classes, for example, STUDENT, it seems to be unrealistic to maintain an extensive gazetteer: by the time we need to create such a resource it will be out-dated.

We propose a bootstrapping approach to acquire gazetteers automatically. Unlike other bootstrapping algorithms for lexicon acquisition (Thelen & Riloff

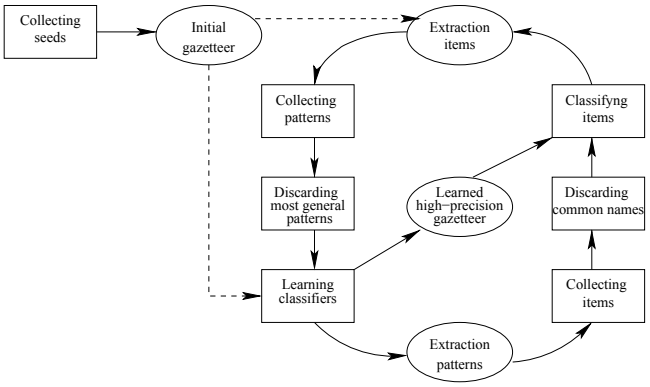


Figure 1: Bootstrapping NE Gazetteers

02; Yangarber 03), our system relies on Web data. This allows us to account for low frequency NEs not represented in a standard size corpora. But it also aggravates the noise problem crucial for bootstrapping approaches. We propose a two-step rescoring strategy to deal with noisy data: we first use counter-training to compile a short list of bootstrapped patterns and then run a machine learner to restrain it even further.

3.1 Collecting seed data

Usually bootstrapping algorithms (Thelen & Riloff 02) rely on very few manually selected seed items to start the processing. In our case, we want to run a machine learner on these data, so we need more items.

For LOCATIONs, we have collected our seeds manually, sampling them randomly from three World Atlases and adding several well-known names to get a more balanced distribution. Following the major classes of the atlases, we have classified the items manually into CITY, COUNTRY, REGION, RIVER, ISLAND, or MOUNTAIN. For bootstrapping, we sampled randomly 100 items of each class from this gazetteer. According to the MUC-7 definition of a LOCATION, we have added the TERRITORY (continents, "Europe", and "Asia") and PLANET classes. Having full gazetteers for these two classes, we do not apply bootstrapping to them.

The PERSON class is different: it does not support a unique subclassification. Thus, PERSONs can be subdivided into groups based on their profession, gender, place of birth, age, etc. Unlike for LOCATIONs, these sub-classes are not necessary stable – for example, PERSONs often change their professions. This makes the manual seed selection unfeasible for the PERSON class. We have investigated possibilities to collect seeds automatically, using high-precision corpus-based methods.

In some supportive contexts it is easy to classify a proper name even without any lexical information:

> ... said Lt. Greg Geisen, a Navy spokesman at the Pentagon. [...] Geisen said.

The apposition tells us that *Greg Geisen* is a *spokesman*, and the ontology links *spokesman* to *spokesperson* to *person*.

We have used parsed texts from the MUC-7 IE corpus to mine seeds: with a simple regexp matcher, we identify appositive constructions linking an NE and a common name.[1] We associate WordNet hypernyms of the common name with the NE. Having pruned under-represented nodes, we have the following PERSON subclasses: *director* (48 seeds), *executive* (40), *spokesperson* (56), *worker* (54), *person_other* (138).

s_1 scoring	s_2 scoring	Extraction patterns
of X	X island	X island
the X	island of X	and X islands
X and	X islands	insel X
X the	island X	
to X	islands X	

Table 1: 5 Best patterns for ISLAND at different bootstrapping steps

3.2 Bootstrapping Algorithm

In this subsection we go through a bootstrapping loop, describing our re-scoring strategies, using the ISLAND class as an example. Table 1 shows the best patterns suggested by the system after different steps at the first bootstrapping iteration.

We start with the seed lists. As the first step, we process each list individually. For each name on a list we download 500 web pages and extract patterns: contexts up to 2 words to the left and 2 words to the

[1] The same idea proved to be helpful in the experiment of (Phillips & Riloff 02) on semantic lexicon acquisition.

right of the name. We score a pattern p for a class cl:

$$s_1(p, cl) = \frac{\sum_{w \in lex(cl)} N(w, p)}{|lex(cl)|},$$

where $lex(cl)$ is a current lexicon extracted for the class cl (at the first bootstrapping iteration $lex(cl)$ is the seed set for cl), and $N(w,p)$ is a number of occurrences of the phrase w in the context p in all the pages the system has downloaded so far. After this step we've got 27190 patterns for ISLANDs. The best ones, according to the s_1 scoring function, are "of X" and "the X" (Table 1, first column).

Obviously, we should penalize too general patterns, such as "of X". First, we apply counter-training, rescoring all the patterns:

$$s_2(p, cl) = s_1(p, cl) - \sum_{c \neq cl} s_1(p, c).$$

Patterns with negative scores are discarded, resulting in much more specific lists. Thus, for ISLANDs we have 250 patterns, the best being "X island" and "island of X" (Table 1, second column).

After re-scoring we have more relevant patterns, but still not all of them can be successfully used for bootstrapping. So, as a third step, we apply machine learning to produce classifiers and select the most relevant patterns: we chose the 10 best patterns after the s_2 re-scoring and use our seed sets (520 items for LOCATIONs and 336 for PERSONs) to make AltaVista queries ("Sicily island", "island of Sicily", ...) and get the corresponding counts (number of webpages in English worldwide). We normalize them by the count for the name alone. Feature vectors of both normalized and raw counts are sent to a machine learner.

For learning, we use Ripper (Cohen 95), an information gain-based rule induction system: first, its output is easily interpretable and provides extraction patterns for bootstrapping; second, the classifier selects only very few features, which is crucial for time-consuming web-based processing.

We run Ripper with three settings for the *Loss Ratio* parameter , obtaining high-recall, high-precision, and high-accuracy classifications. We take patterns from the high-recall classifier (Table 1, third column) and extract from the web NEs used in those constructions. Then we use the high-precision classifier to double-check extracted items. Finally, we have 5 (for PERSONs) or 6 (for LOCATIONs) new lists of proper names. They are added to the temporary lexicons for each class.

The temporary lexicon and the high-accuracy classifier constitute the system's output at each bootstrapping iteration. To get a gazetteer entry for an entity, the lexicons can be used for a quick look-up; if the entity is not present in the lexicons, we construct AltaVista queries and run the classifiers.

When the classifiers are produced and the lexicon is updated, the bootstrapping process starts again: we download more pages, extract more patterns, get new learning data, produce new classifiers, extract new items, update the lexicons, and so on.

4 Experiments

4.1 Data

We use the MUC-7 corpus, a dataset designed to test algorithms for different NLP tasks: IE, (coarse) NER, and Coreference. We rely on the IE and Coreference subcorpora for tuning the parameters of the bootstrapping and name-matching modules and to generate training instances. This will be explained in detail below. We parsed the corpus (Charniak 00), and extracted NEs (Curran & Clark 03).

For testing, we have selected randomly 20 texts from the NE corpus and manually reclassified all the LOCATIONs (259 items) and PERSONs (153) into the fine-grained categories introduced in Section 3.1: CITY (61), COUNTRY (119), REGION (7), TERRITORY (29), PLANET (19), LOC_OTHER (24); DIRECTOR (11), EXECUTIVE (24), WORKER (29), SPOKESPERSON (16), PERS_OTHER (73).

4.2 Experiment 1: Evaluating Gazetteers' Performance

In this experiment we compare our bootstrapped classifiers to the seed gazetteer. If a system suggests several classes, we chose the majority class.

Table 2 shows the F-scores for bootstrapped classifiers and original gazetteers (there were no RIVER, ISLAND, or MOUNTAIN items in both the gold standard and the system's output; REGION was underrepresented in training/test data and therefore always suggested incorrectly). Bootstrapping significantly improves (χ^2-test applied to confusion matrices for each class, $p < 0.05$) the performance for the LOCATION subclasses (recall that we do not bootstrap PLANET and TERRITORY). PERSON subclasses show only slight improvement. Overall, the error drops by 3.3% for PERSONs and 24.2% for LOCATIONs.

Table 2 clearly suggests that LOCATIONs are much easier for our approach than PERSONs: the overall accuracy of the bootstrapping approach is

	seed gazetteer	bootstrapped classifier
director	15.4	15.4
executive	59.5	63.2
worker	0.0	0.0
spokesperson	47.6	54.6
pers_other	70.9	71.6
city	77.1	**85.0**
country	81.6	**85.8**
territory	97.3	97.3
planet	100	100
loc_other	39.0	41.5

Table 2: Performance of the original and bootstrapped gazetteers (F-measure), significant improvement shown in boldface

	gazetteer sampling	syntax sampling
director	16.67	30.8
executive	15.5	**45.2**
worker	0.0	0.0
spokesperson	10.5	29.0
pers_other	65.5	67.9
city	17.14	N/A
country	64.71	N/A
territory	0.00	N/A
planet	61.9	N/A
loc_other	8.45	N/A

Table 3: The system's performance (F-measure) for different sampling strategies, significant improvement shown in boldface

80.7% for LOCATIONs and only 56.2% for PERSONs. LOCATIONs are usually well-known names assumed familiar to the reader, whereas PERSONs are unknown and introduced via explanatory descriptions. This decreases the importance of a gazetteer and increases the role of syntactic features for PERSONs. We address this issue in Experiment 2.

4.3 Experiment 2: Integrating the Bootstrapping Approach into a Fine-grained NE Classification System

Experiment 1 shows that bootstrapping improves the lexicon. In Experiment 2 we incorporate our acquired gazetteer into a fine-grained NE classification system, a simplified version of (Fleischman & Hovy 02), to see whether it helps in a real application.

Features. We combine our features with the topic signature and word frequency counts (Fleischman & Hovy 02). Altogether, we have 104/80 features for LOCATION/PERSON: 8/5 gazetteer-based, 8/5 syntactic, 80/50 word frequencies, and 8/5 topic signatures. The former two groups are described below. The latter represent shallow context information. We point the reader to (Fleischman & Hovy 02) for details. The features are sent to Ripper to obtain two classifiers: one for PERSON and one for LOCATION.

Gazetteer-based features provide lexical information for well-known names. For each NE and each category, we run our bootstrapped high-accuracy classifier (Section 3) to obtain a binary value. For example, "Mars" is represented as [+*planet*, +*city*, −*territory*, −*region*, −*country*, −*island*, −*mountain*, −*river*].

Syntax-based features help to process unknown names, exploiting the fact that such entities are often introduced with explanatory descriptions. However, subsequent mentions are usually shorter and can be used without any supportive context.

We extract syntactic features in two steps. First, we assign feature values to proper names participating in appositive constructions. Second, we run a name-matching algorithm to account for subsequent mentions. In our example, we assign the *spokesperson* label to "Greg Geisen" at the first step, and then to "Geisen" at the second one. We represent this information by a set of binary features: [+*spokesperson*,−*executive*,−*worker*,−*director*, −*person other*]. They encode the same knowledge, as the gazetteer features, mined in a different way.

Generating training data. To train our system and compute values for context-based features, we need preclassified instances. As our data consist of raw texts, we have explored possibilities to generate train instances automatically: we use either the gazetteer or syntactic features to automatically classify unlabelled NEs, thus, exploiting data redundancy.

Although this strategy helps us to generate training instances automatically, it has some weaknesses: it is only applicable when the chosen feature group provides a very high-precision classification, we can not use the same features for sampling and in the main learning algorithm, and the resulting distribution is very biased. Improving the sample strategy is a topic of our future research.

Results. We address several issues in our evaluation. First, we want to find the most reliable sampling strategy. As we have seen in the first experiment, the gazetteer performance on PERSONs is not very promising, so, the syntax-based sampling scheme might be more suitable. Second, we want

	baseline	Preprocessing	
		gazetteer	syntax
PERSON, syntax-based sampling			
director	30.8	28.6	**90.0**
executive	45.2	**66.7**	55.6
worker	0.0	0.0	**22.2**
spokesperson	38.1	**69.2**	**82.8**
pers_other	67.9	72.9	**75.7**
LOCATION, gazetteer-based sampling			
city	17.4	**84.3**	17.4
country	64.7	**87.8**	64.7
territory	0.0	**97.3**	0.0
planet	61.9	**100.0**	61.9
loc_other	8.45	**41.5**	8.45

Table 4: Performance for lexical/syntactic information added (F-measure), significant improvement shown in boldface

to investigate the impact of gazetteers and syntactic heuristics on the overall resolution accuracy.

Table 3 shows the performance for different sampling strategies, using only the word frequency and topic signature features. For PERSONs, we created two train sets using different sampling strategies. LOCATIONs do not appear in appositive constructions in our corpus, so, only gazetteer-based sampling was investigated. We see again that LOCATIONs and PERSONs are very different: syntax-based sampling outperforms gazetteer sampling for PERSON (11.7% error reduction) but is not applicable to LOCATIONs.

Table 4 shows the importance of different knowledge sources: we add lexical and syntactic information to the baseline (Table 3). We cannot use the same features for generating training data and for learning. So, we separately create a preprocessing classifier from these features to mark up our test data in the same way as with the training set (for example, "If an NE is marked as *city* in the gazetteer, classify it as *city*"). We first apply the pre-processing module, and then use the baseline for remaining (PERSON_OTHER or LOCATION_OTHER) items.

As Table 4 shows, syntax-based (PERSON) and gazetteer-based (LOCATION) preprocessing boosts the system's performance: the error goes down by 31.3% and 75%. Our corpus is not large enough for a very shallow approach of the baseline: for example, we have 3000 PERSONs instances compared to the 25K corpus of (Fleischman & Hovy 02). We plan larger-scale experiments on the corpus of (Fleischman & Hovy 02). In a small-scale domain-specific application, however, lexical and syntactic features show a clear improvement over a context-based approach.

5 Conclusion

We have presented an algorithm for automatic fine-grained NE classification. As our experiments show, both syntactic and lexical knowledge are very helpful for the task. Unlike several other approaches, we do not use hand-crafted gazetteers, proposing a novel algorithm to extract them automatically from the web. The empirical evaluation supports our hypothesis that bootstrapped gazetteers are reliable enough for successful subclassification of LOCATIONs.

Our evaluation shows that different coarse NE types require different information for automatic subclassification. We can use syntactic (for the first mention) or coreference (for subsequent mentions) information to assign fine-grained classes to a PERSON name. LOCATIONs, on the contrary, are normally assumed to be in the reader's knowledge base and, thus, require a gazetteer. This picture, of course, is a bit simplistic: for example, names of celebrities can be used without any supportive context. We plan to further investigate the interaction between our syntactic and lexical features, applying co-training. Future directions of this work will also concentrate on the remaining MUC-7 types, especially ORGANIZATION.

References

(Charniak 00) E. Charniak. A maximum-entropy-inspired parser. In *Proceedings of the 1st NAACL*, 2000.

(Cohen 95) W. Cohen. Fast effective rule induction. In *Proceedings of the 12th ICML*, 1995.

(Curran & Clark 03) J.R. Curran and S. Clark. Language independent NER using a maximum entropy tagger. In *Proceedings of the 7th CoNLL*, 2003.

(Evans 03) R. Evans. A framework for named entity recognition in the open domain. In *Proceedings of RANLP*, 2003.

(Fleischman & Hovy 02) M. Fleischman and E. Hovy. Fine grained classification of named entities. In *Proceedings of the 19th COLING*, 2002.

(Phillips & Riloff 02) W. Phillips and E. Riloff. Exploiting strong syntactic heuristics and co-training to learn semantic lexicons. In *Proceedings of EMNLP02*, 2002.

(Thelen & Riloff 02) M. Thelen and E. Riloff. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *Proceedings of EMNLP02*, 2002.

(Yangarber 03) R. Yangarber. Counter-training in discovery of semantic patterns. In *Proceedings of ACL*, 2003.

Parallel corpora for medium density languages

Dániel Varga
Media Research and Education Center
Stoczek u. 2
H-1111 Budapest
daniel@mokk.bme.hu

Péter Halácsy
MREC
Stoczek u. 2
H-1111 Budapest
halacsy@mokk.bme.hu

András Kornai
MetaCarta Inc.
875 Massachusetts Avenue
Cambridge MA 02139
andras@kornai.com

Viktor Nagy
Institute of Linguistics
Benczúr u 33
H-1399 Budapest
nagyv@nytud.hu

László Németh
MREC
Stoczek u. 2
H-1111 Budapest
nemeth@mokk.bme.hu

Viktor Trón
U of Edinburgh
2 Buccleuch Place
EH8 9LW Edinburgh
v.tron@ed.ac.uk

Abstract

The choice of natural language technology appropriate for a given language is greatly impacted by *density* (availability of digitally stored material). More than half of the world speaks medium density languages, yet many of the methods appropriate for high or low density languages yield suboptimal results when applied to the medium density case. In this paper we describe a general methodology for rapidly collecting, building, and aligning parallel corpora for medium density languages, illustrating our main points on the case of Hungarian, Romanian, and Slovenian. We also describe and evaluate the hybrid sentence alignment method we are using.

0 Introduction

There are only a dozen large languages with a hundred million speakers or more, accounting for about 40% of the world population, and there are over 5,000 small languages with less than half a million speakers, accounting for about 4% (Grimes 2003). In this paper we discuss some ideas about how to build parallel corpora for the five hundred or so medium density languages that lie between these two extremes based on our experience building a 50M word sentence-aligned Hungarian-English parallel corpus. Throughout the paper we illustrate our strategy mainly on Hungarian (14m speakers), also mentioning Romanian (26m speakers), and Slovenian (2m speakers), but we emphasize that the key factor leading the success of our method, a vigorous culture of native language use and (digital) literacy, is by no means restricted to Central European languages. Needless to say, the density of a language (the availability of digitally stored material) is predicted only imperfectly by the population of speakers: major Prakrit or Han dialects, with tens, sometimes hundreds, of million speakers, are low density, while minor populations, such as the Inuktitut, can attain high levels of digital literacy given the political will and a conscious Hansard-building effort (Martin et al 2003). With this caveat, population (or better, GDP) is a very good approximation for density, on a par with web size.

The rest of the paper is structured as follows. In Section 1 we describe our methods of corpus collection and preparation. Our hybrid sentence-level aligner is discussed in Section 2. Evaluation is the subject of Section 3.

1 Collecting and preparing the corpus

Starting with Resnik (1998), mining the web for parallel corpora has emerged as a major technique, and between English and another high density language, such as Chinese, the results are very encouraging (Chen and Nie 2000, Resnik and Smith 2003). However, when no highly bilingual domain (like `.hk` for Chinese or `.ca` for French) exists, or when the other language is much lower density, the actual number of automatically detectable parallel pages is considerably smaller: for example, Resnik and Smith find less than 2,000 English-Arabic parallel pages for a total of 2.3m words.

For medium density languages parallel web pages turn out to be a surprisingly minor source of parallel texts. Even in cases where the population and the economy is sizeable, and a significant monolingual corpus can be collected by crawling, mechanically detectable parallel or bilingual web pages exist only in surprisingly small numbers. For example a 1.5 billion word corpus of Hungarian (Halácsy et al 2004), with 3.5 million unique pages, yielded only 270,000 words (535 pages), and a 200m word corpus of Slovenian (202,000 pages) yielded only 13,000 words (42 pages) using URL parallelism as the primary matching criterion as in PTMiner (Chen and Nie 2000).

Some indication of this problem can already be seen in the low number, 2,491, of English–French pages by Resnik and Smith (2003), who discuss the is-

sue under the heading "Too little data" (p 374). Since by GDP France is about 21 times the size of Hungary, and 66 times the size of Slovenia, we expect that an effort similar to ours would yield a quite respectable English-French parallel corpus, perhaps 5-6 m words for French, consistent with the growth of `.fr` since 1998. However, for medium density languages, even if we extrapolate optimistically for the next 5-10 years, the yield can not be expected to be significant.

Web pages are undoubtedly valuable for a diversity of styles and contents that is greater than what could be expected from any single source, but a few hundred web pages alone fall short of a sensible parallel corpus. Therefore, one needs to resort to other sources, many of them impossible to find by mechanical URL comparison, and often not even accessible without going through dedicated query interfaces. We discuss the nature of these resources using Hungarian as our primary example.

Literary texts The Hungarian National Library maintains a large public domain digital archive *Magyar Elektronikus Könyvtár* 'Hungarian Electronic Library' `mek.oszk.hu/indexeng.phtml` with many classical texts. Comparison with the Project Gutenberg archives at `www.gutenberg.org` yielded well over a hundred parallel texts by authors ranging from Jane Austen to Tolstoy. Equally importantly, many works still under copyright were provided by their publishers under the standard research exemption clause. While we can't publish most of these texts in either language, we publish the aligned sentence pairs alphabetically sorted. This "shuffling" somewhat limits usability inasmuch as higher than sentence-level text layout becomes inaccessible, but at the same time makes it prohibitively hard to reconstruct the original texts and contravene the copyright. Since shuffling nips copyright issues in the bud, it simplifies the complex task of disseminating aligned corpora considerably.

Religious texts The entire Bible has been translated to over 400 languages and dialects, and many religious texts from the Bhagavad Gita to the Book of Mormon enjoy nearly as broad currency. The Catholic Church makes a special effort to have papal edicts translated to other languages from the original Latin (see `www.vatican.va/archive`).[1]

[1]It has often been noted that archaic biblical texts offer little help in translating e.g. newswire text. The situation can be greatly improved by using a contemporary English translation (as opposed to the King James Version).

International Law From the Universal Declaration of Human Rights (`www.unhchr.ch/udhr`) to the Geneva Convention many important legal documents have been translated to hundreds of languages and dialects. Those working on the languages of the European Union have long availed themselves of the CELEX database.

Movie captioning Large mega-productions are often dubbed, but smaller releases will generally have only captioning, often available for research purposes. For cult movies there is also a vigorous subgenre of amateur translations by movie buffs.

Software internationalization Multilingual software documentation is increasingly becoming available, particularly for open source packages such as KDE, Gnome, OpenOffice, Mozilla, the GNU tools, etc (Tiedemann and Nygaard 2004).

Bilingual magazines Both frequent flyer magazines and national business magazines are often published with English articles in parallel. Many magazines from Scientific American to National Geographic have editions in other languages, and in many countries there exist magazines with complete mirror translations (for instance, *Diplomacy and Trade Magazine* publishes every article both in Hungarian and English).

Annual reports, corporate home pages Large companies will often publish their annual reports in English as well. These are usually more strictly parallel than the rest of their web pages.

There is no denying that the identification of such resources, negotiating for their release, downloading, format conversion, and character-set normalization remain labor-intensive steps, with good opportunities for automation only at the final stages. But such an effort leverages exactly the strengths of medium density languages: the existence of a joint cultural heritage both secular and religious, of national institutions dedicated to the preservation and fostering of culture, of multinational movements (particularly open source) and multinational corporations with a notable national presence, and of a rising tide of global business and cultural practices. Altogether, the effort pays off by yielding a corpus that is two-three orders of magnitude larger, and covering a much wider range of jargons, styles, and genres, than what could be expected from parallel web pages alone. Table 1 summarizes the different types of texts and their sizes in our Hungarian-English parallel corpus.

source	docs	E words (m)	H words (m)
Literary	156	14.6	11.5
Legal	10374	24.1	18.3
Captioning	437	2.5	1.9
Sw docs	187	0.8	0.7
Magazines	107	0.3	0.3
Business	19	0.5	0.4
Religious	122	2.3	2.0
Web	435	0.3	0.2
Total	**11550**	**44.6**	**34.6**

Table 1: Distribution of text types in the Hungarian–English parallel corpus

In addition to the texts, we identified other significant lexical resources, such as public domain glossaries specifically prepared for EU law, Microsoft software, Linux, and other particular domains and most importantly, a large (over 254,000 records) general-purpose bilingual dictionary manually created over many years by Attila Vonyó. Since there is no guarantee that such materials are available for other languages, in the next section we describe a sentence-alignment algorithm which does not rely on the existence of such bilingual dictionaries, but can take advantage of it if it is available.

After some elementary format-detection and conversion routines such as `catdoc` and `pdftotext` which are standard in the open source world, we have a corpus of raw text consisting of assumed parallel documents. While the texts themselves were collected and converted predominantly manually, the aligned bicorpus is derived by entirely automatic methods. Due to the manual effort, parallelism is nearly perfect, therefore the size of the raw corpus of collected texts is not significantly different from the size of the useful (aligned) data.

The first steps of our corpus preparation pipeline are tokenizers performing sentence and paragraph boundary detection and word tokenization. These are relatively simple flex programs (along the lines of Mikheev 2002) both for English and Hungarian. For languages with more complex morphology such as Hungarian, it makes sense to conflate by stemming morphological variants of a lexeme before the texts are passed to the aligner. We used `hunmorph`, a language-independent word analysis toolkit (Trón et al 2005) both for Hungarian and English.

The most important ingredient of the pipeline is of course automatic sentence alignment which we carried out using our own algorithm and software `hunalign`, described in detail in the next section.

2 Sentence level alignment

There are three main approaches to the problem of corpus alignment at the sentence level: length-based (Brown et al 1991, Gale and Church 1991), dictionary- or translation based (Chen 1993, Melamed 1996, Moore 2002), and partial similarity-based (Simard and Plamondon 1998). This last method in itself may work well for Indo-European languages (probably better between English and Romanian than English and Slovenian), but for Hungarian the lack of etymological relation suggests that the number of cognates will be low. Even where the cognate relationship is clear, as in *computer/kompjúter, strike/sztrájk* etc., the differences in orthography make it hard to gain traction by this method. Therefore, we chose to concentrate on the dictionary and length-based methods, and designed a hybrid algorithm, `hunalign`, that successfully amalgamates the two.

In the first step of the alignment algorithm, a crude translation of the source text is produced by converting each word token into the dictionary translation that has the highest frequency in the target corpus, or to itself in case of lookup failure.

This pseudo target language text is then compared against the actual target text on a sentence by sentence basis. The similarity score between a source and a target sentence consists of two major components: token-based and length-based. The dominant term of the token-based score is the number of shared words in the two sentences, normalized with the larger token count of the two sentences. A separate reward term is added if the proportion of shared numerical tokens is sufficiently high in the two sentences (especially useful for the alignment of legal texts).

For the length-based component, the character counts of the original texts are incremented by one, and the score is based on the ratio of longer to shorter. The relative weight of the two components was set so as to maximize precision on the Hungarian–English training corpus, but seems a sensible choice for other languages as well. Paragraph boundary markers are treated as sentences with special scoring: the similarity of two paragraph-boundaries is a high constant, the similarity of a paragraph-boundary to a real sentence is minus infinity, so as to make paragraph boundaries pair up.

The similarity score is calculated for every sentence pair around the diagonal of the alignment matrix (at least a 500-sentence neighborhood is calculated or all sentences closer than 10% of the longer text). This is justified by the observation that the beginning and

the end of the texts are considered aligned and that the sentence ratio in the parallel text represents the average one-to-many assignment ratio of alignment segments, from which no significant deviations are expected. We find that 10% is high enough to produce reassuringly high recall figures even in the case of faulty parallelism such as long surplus chapters.

Once the similarity matrix is obtained for the relevant sentence pairs, the optimal alignment trail is selected by dynamic programming, going through the matrix with various penalties assigned to skipping and coalescing sentences. The score of skipping is a fixed parameter, learnt on our training corpus while the score of coalescing is the sum of the minimum of the two token-based scores and the length-based score of the concatenation of the two sentences. For performance reasons, the dynamic programming algorithm does not take into account the possibility of more than two sentences matching one sentence. After the optimal alignment path is found, a postprocessing step iteratively coalesces a neighboring pair of one-to-many and zero-to-one segments wherever the resulting new segment has a better character-length ratio than the starting one. With this method, any one-to-many segments can be discovered.

The hybrid algorithm presented above remains completely meaningful even in the total absence of a dictionary. In this case, the crude translation will be just the source language text, and sentence-level similarity falls back to surface identity of words.After this first phase a simple dictionary can be bootstrapped on the initial alignment. From this alignment, the second phase of the algorithm collects one-to-one alignments with a score above a fixed threshold. Based only on all one-to-one segments, cooccurrences of every source-target token pair are calculated. These, when normalized with the maximum of the two tokens' frequency yield an association measure. Word pairs with association higher than 0.5 but are are used as a dictionary.

Our algorithm is similar in spirit to that of Moore (2002) in that they both combine the length-based method with some kind of translation-based similarity. In what follows we discuss how Moore's algorithm differs from ours.

Moore's algorithm has three phases. First, an initial alignment is computed based only on sentence length similarity. Next, an IBM 'Model I' translation model (Brown et al 1993) is trained on a set of likely matching sentence pairs based on the first phase. Finally, similarity is calculated using this translation model, combined with sentence length similarity. The output alignment is calculated using this complex similarity score. Computation of similarity using Model I is rather slow, so only alignments close to the initially found alignment are considered, thus restricting the search space drastically.

Our simpler method using a dictionary-based crude translation model instead of a full IBM translation model has the very important advantage that it can exploit a bilingual lexicon, if one is available, and tune it according to frequencies in the target corpus or even enhance it with extra local dictionary bootstrapped from an initial phase. Moore's method offers no such way to tune a preexisting language model. This limitation is a real one when the corpus, unlike the news and Hansard corpora more familiar to those working on high density languages, is composed of very short and heterogeneous pieces. In such cases, as in web corpora, movie captions, or heterogeneous legal texts, average-based models are actually not close to any specific text, so Moore's workaround of building language models based on 10,000 sentence subcorpora has little traction.

On top of this, our translation similarity score is very fast to calculate, so the dictionary-based method can be used already in the first phase where a much bigger search space can be traversed. If the lexicon resource is good enough for the text, this first phase already gives excellent alignment results.

Maximizing alignment recall in the presence of noisy sentence segmentation is an important issue, particularly as language density generally correlates with the sophistication of NLP tools, and thus lower density implies poorer sentence boundary detection. From this perspective, the focus of Moore's algorithm on one-to-one alignments is less than optimal, since excluding one-to-many and many-to-many alignments may result in losing substantial amounts of aligned material if the two languages have different sentence structuring conventions.

While speed is often considered a mundane issue, `hunalign`, written in C++, is at least an order of magnitude faster than Moore's implementation (written in Perl), and the increase in speed can be leveraged in many ways during the building of a parallel corpus with tens of thousands of documents. First, rapid alignment allows for more efficient filtering of texts with low confidence alignments, which usually point to faulty parallelism such as mixed order of chapters (as we encountered in *Arabian Nights* and many other anthologies), missing appendices, extensive extra editorial headers (typical of Project Gutenberg),

comments, different prefaces in the source texts etc. Once detected automatically, most cases of faulty parallelism can be repaired and the texts realigned. Second, debugging and fine-tuning lower-level text processing steps (such as the sentence segmentation and tokenization steps) may require several runs of alignment in order to monitor the impact of certain changes on the quality of alignment. This makes speed an important issue. Interestingly, runtime complexity of Moore's program seems to be very sensitive to the faults in parallelism. Adding a 300 word surplus preface to one side of *1984* but not the other slows down this program by a factor of five, while it has no detectable impact on `hunalign`.

Finally, Moore's aligner, while open source and clearly licensed for research, is not free software. In particular, parallel corpora aligned with it can not be made freely available for commercial purposes. Since we wanted to make sure that our corpus is available for any purpose, including commercial use, Moore's aligner program was not a viable choice.

3 Evaluation

In this section we describe our attempts to assess the quality of our parallel corpus by evaluating the performance of the sentence aligner on texts for which manually produced alignment is available. We also compare our algorithm to Moore's (2002) method.

Evaluation shows `hunalign` has very high performance: generally it aligns incorrectly at most a handful of sentences. As measured by Moore's method of counting only on one-to-one sentence-pairs, precision and recall figures in the high nineties are common. But these figures are overly optimistic because they hide one-to-many and many-to-many errors, which actually outnumber the one-to-one errors. In *1984,* for example, 285 of the 6732 English sentences or about 4.3% do not map on a unique Hungarian, and 716 or 10.6% do not map on a unique Romanian sentence – similar proportions are found in other alignments, both manual and automatic.

To take these errors into account, we used a slightly different figure of merit, defined as follows. The alignment trail of a text can be represented by a ladder, i.e. an array of pairs of sentence boundaries: rung (i, j) is present in the ladder iff the first i sentences on the left correspond to the first j sentences on the right. Precision and recall values are calculated by comparing the predicted and actual rungs of the ladder: we will refer to this as the *complete rung* count as opposed to the *one-to-one* count. In general, complete rung figures of merit tend to be lower than one-to-one figures of merit, since the task of getting them right is more ambitious: it is precisely around the one-to-many and many-to-one segments of the text that the alignment algorithms tend to stumble.

Table 3 presents precision and recall figures based on all the rungs of the entire ladder against the manual alignment of the Hungarian version of Orwell's 1984 (Dimitrova et al 1998).

condition	precision	recall
id	34.30	34.56
id+swr	74.57	75.24
len	97.58	97.55
len+id	97.65	97.42
len+id+swr	97.93	97.80
dic	97.30	97.08
len+dic-stem	98.86	98.88
len+dic	99.34	99.34
len+boot	99.12	99.18

Table 3: Performance of the sentence-level aligner under various conditions

If length-based scoring is switched off and we only run the first phase without a dictionary, the system reduces to a purely identity based method we denote by *id*. This will still often produce positive results since proper nouns and numerals will "translate" to themselves. With no other steps taken, on *1984 id* yields 34.30% precision at 34.56% recall. By the simple expedient of stopword removal, *swr*, the numbers improve dramatically, to 74.57% precision at 75.24% recall. This is due to the existence of short strings which happen to have very high frequency in both languages (the two predominant false cognates in the Hungarian-English case were *a* 'the' and *is* 'too').

Using the length-based heuristic *len* instead of the identity heuristic is better, yielding 97.58% precision at 97.55% recall. Combining this with the identity method does not yield significant improvement (97.65% precision at 97.55% recall). If, on top of this, we also perform stopword removal, both precision (97.93%) and recall (97.80) improve.

Given the availability of a large Hungarian-English dictionary by A. Vonyó, we also established a baseline for a version of the algorithm that makes use of this resource. Since the aligner does not deal with multiword tokens, entries such as *Nemzeti Bank* 'National Bank' are eliminated, reducing the dictionary to about 120k records. In order to harmonize the dictionary entries with the lemmas of the stemmer, the dictionary is also stemmed with the same tool as the texts. Using this dictionary (denoted by *dic* in the Table) without the

length-based correction results in slightly worse performance than identity and length combined with stop word removal.

If the translation-method with the Vonyó dictionary is combined with the length-based method (*len+dic*), we obtain the highest scores 99.34% precision at 99.34% recall on rungs (99.41% precision and 99.40% recall on one-to-one sentence-pairs). In order to test the impact of stemming we let the algorithm run on the non-stemmed text with a non-stemmed dictionary (*len+dic-stem*). This established that stemming has indeed a substantial beneficial effect, although without it we still get better results than any of the non-hybrid cases.

Given that the dictionary-free length-based alignment is comparable to the one obtained with a large dictionary, it is natural to ask how the algorithm would perform with a bootstrapped dictionary as described in Section 2. With no initial dictionary but using this automatically bootstrapped dictionary in the second alignment pass, the algorithm yielded results (*len+boot*), which are, for all intents and purposes, just as good as the ones obtained from combining the length-based method with our large existing bilingual dictionary (*len+dic*). This is shown in the last two lines of Table 3. Since this method is so successful, we implemented it as a mode of operation of `hunalign`.

To summarize our results so far, the pure sentence length-based method does as well in the absence of a dictionary as the pure matching-based method does with a large dictionary. Combining the two is ideal, but this route is not available for the many medium density languages for which bilingual dictionaries are not freely avaliable. However, a core dictionary can automatically be created based on the dictionary-free alignment, and using this bootstrapped dictionary in combination with length-based alignment in the second pass is just as good as using a human-built dictionary for this purpose. In other words, the lack of a high-quality bilingual dictionary is no impediment to aligning the parallel corpus at the sentence level.

While we believe that an evaluation based on all the rungs of the ladder gives a more realistic measure of alignment performance, for the sake of correct comparison with Moore's method, we present some results using the one-to-one alignments metric. Table 4 summarizes results on Orwell's *1984* for Hungarian–English (*1984-HE-S*, stemmed and *1984-HE-U*, unstemmed), Romanian–English (*1984-RE-U*, unstemmed), as well as on Steinbeck's *Cup of Gold* for Hungarian–English (*CoG-HE-S*, 80k words, stemmed) using `hunalign` (with bootstrapped dictionary, no further tuning and omitting paragraph information) and Moore's (2002) algorithm (with the default values).

task	`hunalign`		`Moore '02`	
	prec	**rec**	**prec**	**rec**
1984-HE-S	99.22	99.24	99.42	98.56
1984-HE-U	98.88	99.05	99.24	97.39
1984-RE-U	97.10	97.98	97.55	96.14
CoG-HE-S	97.03	98.44	96.45	97.53

Table 4: Comparison of `hunalign` and Moore's (2002) algorithm on three texts. Performance figures are based on one-to-one alignments only.

In order to be able to compare the Hungarian and Romanian results for *1984*, we provide the Hungarian case for the unstemmed *1984*. One can see that both algorithms show a drop of performance. This makes it clear that the drop in quality from Hungarian–English to Romanian–English can not be attributed to the fact that we tuned our system on the Hungarian case. As mentioned earlier, the Romanian translation has 716 non-one-to-one segments compared to the Hungarian translation's 285. Given both algorithm's preference to globally diagonal and locally one-to-one alignments, this difference in one-to-one alignments is likely to render the Romanian–English alignment a harder task.

In order to sensibly compare our results with that of Moore's, paragraph information was not exploited. `huntoken`, the sentence tokenizer we use is able to identify paragraph boundaries which are then used by the aligner. Experiments showed that paragraph information can substantially improve alignment scores: measured on the Hungarian–English alignment of Steinbeck's 'Cup of Gold', the number of incorrect alignments drop from 148 to 115.[2] Therefore the figures shown in Table 4 are in no way absolute best bisentence scores for the texts in question.

4 Conclusion

In the past ten years, much has been written on bringing modern language technology to bear on low density languages. At the same time, the bulk of commercial research and product development, understandably, concentrated on high density languages. To a

[2] Although paragraph identification itself contains a lot of errors, improvement may be due to the fact that paragraphs, however faulty, are consistent in terms of alignment. The details of this and the question of exploiting higher-level layout information is left for future research.

surprising extent this left the medium density languages, spoken by over half of humanity, underresearched. In this paper we attempted to address this issue by proposing a methodology that does not shy away from manual labor as far as the data collection step is concerned. Harvesting web pages and automatically detecting parallels turns out to yield only a meager slice of the available data: in the case of Hungarian, less than 1%. Instead, we proposed several other sources of parallel texts based on our experience with creating a 50 million word Hungarian–English parallel corpus.

Once the data is collected and formatted manually, the subsequent steps can be almost entirely automated. Here we have demonstrated that our hybrid alignment technique is capable of efficiently generating very high quality sentence alignments with excellent recall figures, which helps to get the maximum out of small corpora. Even in the absence of any language resources, alignment quality is very high, but if stemmers or bilingual dictionaries are available, our aligner can take advantage of them.

Acknowledgements

The project is supported by an ITEM 2003 grant from the Hungarian Ministry of Informatics. We are grateful to Magyar Telecom for hardware and logistical support. We are also indebted to Tamás Váradi, and the whole Corpus Linguistics Department at the Institute of Linguistics, Hungarian Academy of Sciences, for the 1984 corpus and joint work on the Steinbeck text, and to Attila Vonyó for his Hungarian-English dictionary.

References

(Brown *et al.* 91) Peter Brown, Jennifer Lai, and Robert Mercer. Aligning sentences in parallel corpora. In *Proceedings of ACL29*, pages 169–176, 1991.

(Brown *et al.* 93) Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993.

(Chen & Nie 00) Jiang Chen and Jian-Yun Nie. Automatic construction of parallel english-chinese corpus for cross-language information retrieval. In *Proceedings of the sixth conference on Applied natural language processing*, pages 21–28, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

(Chen 93) Stanley F. Chen. Aligning sentences in bilingual corpora using lexical information. In *Proceedings of the 31st conference on Association for Computational Linguistics*, pages 9–16, Morristown, NJ, USA, 1993. Association for Computational Linguistics.

(Dimitrova *et al.* 98) Ludmila Dimitrova, Tomaz Erjavec, Nancy Ide, Heiki Jaan Kaalep, Vladimir Petkevic, and Dan Tufis. Multext-east: Parallel and comparable corpora and lexicons for six central and eastern european languages. In Christian Boitet and Pete Whitelock, editors, *Proceedings of the Thirty-Sixth Annual Meeting of the Association for Computational Linguistics and Seventeenth International Conference on Computational Linguistics*, pages 315–319, San Francisco, California, 1998. Morgan Kaufmann Publishers.

(Gale & Church 91) William A. Gale and Kenneth Ward Church. A program for aligning sentences in bilingual corpora. In *Meeting of the Association for Computational Linguistics*, pages 177–184, 1991.

(Grimes 03) Barbara Grimes. *The Ethnologue (14th Edition)*. SIL, 2003.

(Halácsy *et al.* 04) Péter Halácsy, András Kornai, László Németh, András Rung, István Szakadát, and Viktor Trón. Creating open language resources for Hungarian. In *Proceedings of Language Resources and Evaluation Conference (LREC04)*. European Language Resources Association, 2004.

(Martin *et al.* 03) Joel Martin, Howard Johnson, Benoit Farley, and Anna Maclachlan. Aligning and using an english-inuktitut parallel corpus. In *HLT-NAACL Workshop: Building and Using Parallel Texts*, pages 115–118, Edmonton, Alberta, Canada, May 31 2003. Association for Computational Linguistics.

(Melamed 00) I. Dan Melamed. Models of translational equivalence among words. *Computational Linguistics*, 26(2):221–249, 2000.

(Mikheev 00) Andrei Mikheev. Periods, capitalized words, etc. *Computational Linguistics*, 28(3):289–318, 2000.

(Moore 02) Robert C. Moore. Fast and accurate sentence alignment of bilingual corpora. In *Proc 5th AMTA Conf: Machine Translation: From Research to Real Users*, pages 135–244, Langhorne, PA, 2002. Springer.

(Resnik & Smith 03) Philip Resnik and Noah Smith. The web as a parallel corpus. *Computational Linguistics*, 29(3):349–380, 2003.

(Resnik 98) Philip Resnik. Parallel strands: A preliminary investigation into mining the web for bilingual text. In D. Farwell, L. Gerber, and E. Hovy, editors, *Machine Translation and the Information Soup: Third Conference of the Association for Machine Translation in the Americas*, Langhorne, PA, 1998. Springer.

(Simard *et al.* 98) Simard, Michel, and Pierre Plamondon. Bilingual sentence alignment: Balancing robustness and accuracy. In *Machine Translation*, volume Volume 13, no. 1, pages 59–80, 1998.

(Tiedemann & Nygaard 04) Jörg Tiedemann and Lars Nygaard. The opus corpus - parallel and free. In *Proceedings of LREC'04*, volume IV, pages 1183–1186, Lisbon, 2004.

(Trón *et al.* 05) Viktor Trón, György Gyepesi, Péter Halácsy, András Kornai, László Németh, and Dániel Varga. Hunmorph: open source word analysis. In *Proceeding of ACL*, 2005. paper presented at the ACL05 Software Workshop.

Temporally Ordering Event Instances in Natural Language Texts

Argyrios Vasilakopoulos and **William J. Black**
School of Informatics
Faculty of Humanities
The University of Manchester
Sackville Street
Manchester M60 1QD, England
{mcaikav2@co.umist.ac.uk, william.black@manchester.ac.uk}

Abstract

Discovering temporal relations between event instances in free (natural language) texts is an important Information Extraction task, essential for a number of higher-level analyses such as question answering and text summarization. In this paper we present an approach to automatically order event instances in natural language texts using machine learning. We describe an architecture designed to analyse row text and the respective implementing system. We also provide our initial results for the event temporal ordering task evaluated on the TIMEBANK corpus.

1 Introduction

Temporal information is necessary when it comes to describe text structure (monologue and dialogue) and analysis tasks such as question answring and text summarization are hugely dependent on this information type. Following the paradigm of Information Extraction (IE), we consider that it is possible to extract temporal elements in text and the relations between them without necessarily performing a full syntactic and semantic analysis of the text. Traditionally, IE has been defined as a set of five analysis steps (see (Chinchor & Robinson 97)[1]) where temporal information is extracted, without constituting a separate task. However, recent research on Temporal Information Extraction (TIE) considers the task of recognition and use of temporal information as a completely autonomous analysis. To go further, accordiing to (Huang *et al.* 03), TIE can be broadely decomposed in three main subtasks:

- *Temporal Information Representation*
- *Time Expression Resolution*
- *Event Temporal Anchoring and Ordering*

[1]Information Extraction steps according to MUC are: *Named Entity Recognition, Coreference Resolution, Template Element Extraction, Template Relation* and *Scenario Template.*

In this paper, we present an approach to deal with the recognition and extraction of temporal information from unstructured data (text). In more detail, the layout of this paper is the following: We initially enumerate most major recent approaches on the three categories of TIE research. Next, we describe our theoretical background, the annotation schema we are based on and our temporal model. In the following two sections we describe the system architecture and the design of the temporal ordering module in some extend respectively. Next, we discuss our initial results on the TIMEBANK temporally annotated corpus and, finally, we outline our conclusions and indicate new ways that future research could follow.

2 Previous Work on Event Annotation

For the representation of temporal information various researchers have proposed different corpus-based and abstract representation schemes. Among the relative merits of the two types of schemes corpus-based approaches are easier to understand and apply, while abstract representations are more difficult to automatically construct but offer wider capabilities for inferencing. Examples of abstract representations are *Discourse Representation Theory* (DRT) (Kamp & Reyle 93), *Dynamic Aspect Trees* (DATs) (Meulen 95) and *Language Neutral Representation* (LNS) (Campbell *et al.* 02) structures. In parallel, several corpus-based representation approaches are being currently proposed. Initially, a Time Expression (TIMEX) recognition task was defined on the basis of MUC Guidelines (Chinchor & Robinson 97). An extension to MUC was then the TIDES representation scheme which refers to TIMEXes as separate *objects* with specific temporal value (Ferro *et al.* 01). Independently from that, STAG (Setzer 01) is another braoder time representation scheme that also encodes events,

times and temporal relations among them. Temporal relation encoding is also the target of (Katz & Arosio 01). Combining all the strenghts of the above schemata, TimeML was proposed after the TERQAS Workshop[2] in 2002 (Sauri *et al.* 04). TimeML can represent events, times and temporal relations conveniently and is the most complete corpus-based time representation scheme to our knowledge so far.

Based on the proposed annotation (representation) schemata, research has been focussed on the recognition of TIMEX elements and their assignment on the respective event instances. Individual researchers have manually annotated corpora of varying sizes according to specific formats and thereby tested different approaches. Most trials have been based on the use of knowledge bases and hand-crafted linguistic rules, (created following textual examples), which target at extracting the temporal expressions with their appropriate features (temporal meaning)- see (Wilson *et al.* 01) and (Saquete *et al.* 02). A few approaches also employ Machine Learning Techniques, as in (Mani & Wilson 00), (Jahn *et al.* 04). However, as stated in (Ahn *et al.* 05), although ML-baed approaches seem to perform well in identifying the boundaries of the time expressions, they are outperformed by rule-based ones when it comes to extract the respective temporal meaning.

Work on extracting temporal order remains at an exploratory stage. Current related research (see (Mani & Wilson 00), (Filatova & Hovy 01), (Li *et al.* 01)) is based on the extraction of time expressions with appropriate temporal values and their assignment to event instances, which indicates the temporal anchoring, and implicit temporal ordering, of events. The above approaches are dependent on extended knowledge bases (temporal lexica) and heuristic rules, which, based on extracted lexical and contextual clues, are used to recognize the temporal semantic representation of discourse. In some cases, machine learning is also employed for learning models for extracting temporal relations from appropriately annotated training data (Mani *et al.* 03).

3 Theoretical Background

Based on the TimeML specification, we are targeting at extracting the TimeML defined elements, namely, < EVENT >, < TIMEX3 > and < SIGNAL > with certain specific features. For the exact feature set we have chosen, refer to sections 3.1, 5. For a complete description of TimeML the reader is directed to (Sauri *et al.* 04). In the remainder of this section we provide a brief description of the three different elements and the various sets of temporal relations we have been experimenting with.

3.1 Events, Times and Signals

According to most researchers, *eventualities* consist of *facts* (or *states*) and *events* (Mani 03). States are situations that hold true over a long timespan (normally longer than the timespan covered by the document) and events cause the alteration of states and leading to new ones (Moens & Steedman 88). To go further, events are further classified according to their *aspect*. Philosophical and linguistic studies have sought to elaborate this classification since (Vendler 67), (Dowty 86), (Nakhimovsky 88), (Mourelatos 78), (Halliday 85). (Nakhimovsky 88), describes three different aspectual values [3] related to each event in a clause: TimeML uses the *grammatical aspect* for the event denoting verbs. Additional features for events are the *tense* and *polarity*[4]. For TIMEX elements, although accompanied by certain features that indicate their actual temporal values, we only need to specify their exact type: whether each extracted expression is a *date* or *time*. Finally, the SIGNAL elements indicate connective words that potentially provide clues for the existence of a temporal relation. There are no specific attributes for these elements.

3.2 Temporal Relations

In TimeML temporal relations are indicated using the < TLINK > element. The number of relations at the number of 13, is sometimes considered to be too detailed to be used for describing natural language texts (Setzer 01). For this reason, we have been experimenting with three different relation sets consisting of 13, 11 and 4 respectively. The minimal set of four relations has been tested for comparison with similar work (Mani *et al.* 03). The three relation sets are the following:

- *Set A (13 Relations)*: before, after, includes, is_included, during, simultaneous, iafter, ibefore,

[2] http://www.timeml.org/terqas/

[3] According to Nakhimovsky, aspect can be further decomposed into: a) *grammatical aspect*, b) *aspectual class of the event* and c) *aspectual perspective of the sentence*.

[4] Indicates whether the container clause is in the affirmative or negative form.

identity, begins, ends, begun_by, ended_by.

- *Set B (11 Relations)*: before, after, includes, is_included, overlaps, overlapped_by, equals, begins, ends, begun_by, ended_by.
- *Set C (4 Relations)*: before, after, overlaps, equals.

3.3 Transitivity

TimeML TLINKs mark temporal relations between any two events and/or times. Our work, as based on both the TLINKs and work described in (Katz & Arosio 01), targets at extracting relations that refer to adjacent events and times in the text. TimeML annotation can be however considered as broader in terms of being able to mark up relations between any two events and times, no matter what their absolute position in the text is. In this respect, in order to be able to tell the temporal relation that holds between any specific, non-neighbouring events (constituents of the relation) in the same document, we need to somehow "infer" the relation in question from similar temporal relations between the two constituents and other events/times. To do this, and following Allen's paradigm (Allen 83) we define the *Transitivity* function which produces a new temporal relation for a pair of relations between events/times that have a common participant. The values of the transitivity function are calculated according to table 1. The question marks indicate the existence of more than one relation for the respective pair of relations. The transitivity table for the minimal set of relations is chosen to be presented here due to its reduced size. Respective tables for the other sets of relations follow the same idea.

R1 vs. R2	before	after	overlaps	equals
before	before	?	?	before
after	?	after	?	after
overlaps	?	?	?	overlaps
equals	before	after	overlaps	equals

Table 1: Transitivity Table for 4 relations

4 System Arhitecture

In Figure 1 we provide a schematic view of the architecture of the system we have implemented. In brief, we work on texts in the *Common Annotation Scheme* (CAS) format (Rinaldi *et al.* 03), which was developed for the EU-funded Parmenides project[5]. Through a pipeline

[5]http://www.crim.co.umist.ac.uk/parmenides

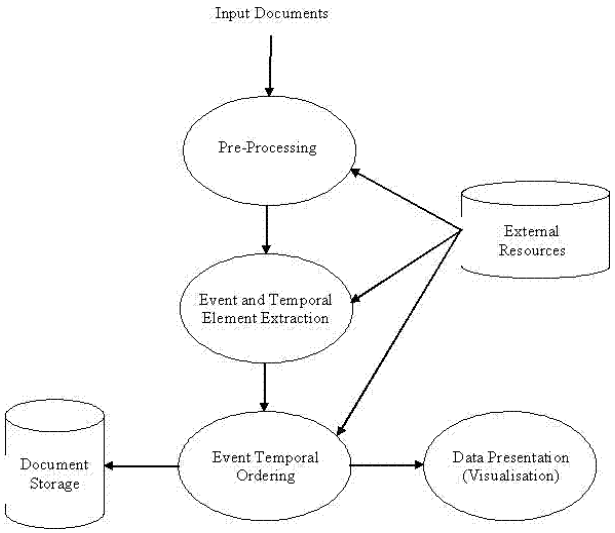


Figure 1: A schematic view of the System Architecture

of NLP and IE analysis stages we initially extract events, times and signals from NL texts using the Cafetiere Environment (see (Vasilakopoulos *et al.* 04)), and we pass the results to the Temporal Ordering Module described in the next section. The results can be saved to an appropriate document repository or be viewed through the Relations Viewer window of the Cafetiere GUI.

5 Temporal Ordering Module

The Temporal Ordering module runs after the basic NLP and IE tasks. The basic idea of the approach is that we consider the text to be a sequence of events, times and signals (discarding everything else in between these elements). A schematic, abstract view of a random text is depicted in Figure 2.

In this figure, event, time and signal expressions are marked up. It is evident that between any two adjacent (according to a certain window) event/time expressions there is exactly one temporal relation that holds. For the specific example, between *05/04/2005* and *Today* there is an *identity relation*, between *Today* and *announced* an *includes* relation and between *announced* and *addition* an *after* relation. To go further, by transitivity we can also say that between *05/04/2005* and *announced* there is also an *includes* relation and also that between *Today*, *05/04/2005* and *addition* there is also an *after* temporal relation (please refer to transitivity table 1).

So, if we now assume that between any two ad-

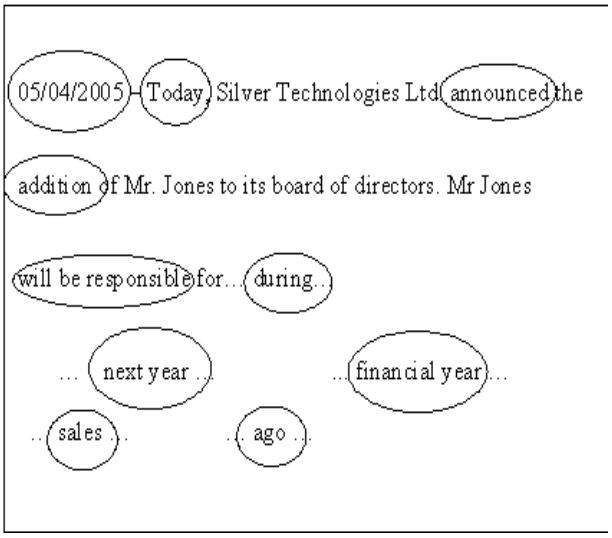


Figure 2: Temporal Elements in a sample text

jacent events/times there is always not more than one temporal relation that holds, we can then consider that all such event/time pairs can be categorized according to the existing temporal relation to a set of categories, which corresponds to the set of categories defined in our temporal model. In this respect, the problem of temporal ordering is transformed to a classification problem, for which we could build appropriate classifiers using standard data mining techniques. In this case, the event/time pairs will constitute the training instances and their TimeML defined features, with additional contextual cues , the instance features for the classifier. Moreover, an implementation of the transitive closure algorithm will produce the temporal relations acquired by transitivity.

In a more consize view, our training instances' structure is:

```
features of Left Context
SOURCE
features of Middle Context
TARGET
features of Right Context
Classification Value (Temporal Relation)
```

For the three context spans we include certain features for the temporal elements contained there according to a specific window size. The features as such are included in table 2.

The virtue of this approach relates to the fact that it targets directly to the extraction of temporal expressions without requiring as a prerequisite the recognition of the temporal meaning of the time expressions involved. This contradicts to the standard way of approaching the ordering task, which is the initial assignment of a timestamp to every event and the placement of the latter on an absolute timeline. Requiring recognizing only the textual span of the time expressions in a document is a task much easier than to also try to figure out their hidden temporal meanings as results indicate in (Puscasu 04).

6 Results

The system we have implemented has been evaluated on the TIMEBANK corpus[6]. The corpus consists of 186 documents from various sources and there are 8255 event instances, 1456 time expressions, 2160 signal words and 5899 TLINK elements marked up by human annotators[7]. Results of the first IE step are obtained using two different approaches: a rule-based and a machine learning based (TBL) one. For more information on this part please refer to (Vasilakopoulos & Black 05). Regarding the actual experiments on ordering, we have been tested the use of variable feature sets, relation (classification) sets and distance values between the participants of the relations in the training corpus. The training algorithms we have used are implemented in the WEKA[8] software (Witten & Frank 00). We have chosen a major classifier (ZeroR) to achive baseline performance and three representative algorithms: a) Naive Bayes (Statistical), b) K* (Memory-based) and c) C4.5 (decision tree). The tree algorithms have been trained on the same training corpora and evaluated using the stratified 10-fold cross validation method offered by the tool.

From Table 3 can see that the best performance is achieved by the K* memory-based algorithm (65.93%) when the minimal set of relations and only 16 out of 40 features are used. Regarding the TimeML defined relations (13) the performance is again the best for the K* algorithm (55.45%), while for the *optimal* set of 11 relations the respective percentage is 59.52% (and still the best of all).

By considering the various setups and window sizes (*rows*) of the learning algorithms as instances and the different algorithms as the cases (*columns*) the instances are applied on we can

[6]The official site for TIMEBANK is at http:// www.cs.brandeis.edu/ jamesp/arda/time/timebank.html

[7]During our experiments we have corrected a few documents and for this reason our statistics differ slightly from the official ones.

[8]The weka open source software can be found at http://www.cs.waikato.ac.nz/ml/weka/

FEATURE	MEANING
Signal before	This is the signal word that is found before the first/second relation participant
Event	A normalized form of the source/target (if not a time expression)
Event Class	The TimeML-defined event classification
Event Tense	The TimeML-defined event verb tense
Event Aspect	The TimeML-defined aspectual class for the event verb
Event Polarity	Affirmative or negative mode for the event verb
Time Expression	A normalized form of the source/target (if not an event)
Function In Document	TimeML-defined feature
Temporal Function	TimeML-defined feature
Signal After	This is the signal word that is found after the first/second relation participant
Punctuation	Boolean feature indicating the existence of a fullstop between the source/target.
Signal Between	This is the signal word that is found between the two relation participants
Double Quotes	Boolean feature indicating whether the source/target is enclosed in double quotes.

Table 2: Feature set for the temporal elements.

Algorithm	4 Features	7 Features	11 Features	16 Features	38 Features	40 Features
			13 Relations			
ZeroR	17.96-21.36	17.96-21.36	17.96-21.36	17.96-21.36	17.96-21.36	17.96-21.36
Naive Bayes	40.16-43.17	43.59-45.48	46.48-49.54	45.55-48.51	41.78-45.78	41.59-44.20
K*	42.13-46.00	48.25-50.52	52.13-54.70	53.14-55.45	44.24-47.80	44.20-47.82
C4.5	37.21-37.92	38.80-44.93	40.30-49.67	40.97-50.30	46.13-50.15	46.17-50.15
			11 Relations			
ZeroR	20.90-23.31	20.90-23.31	20.90-23.31	20.90-23.31	20.90-23.31	20.90-23.31
Naive Bayes	44.24-47.18	47.91-49.88	50.86-53.06	50.03-51.73	46.29-50.24	46.16-50.33
K*	45.18-49.00	52.19-53.13	55.71-58.34	56.70-59.52	49.14-58.70	47.83-51.64
C4.5	40.74-43.29	42.72-48.73	44.56-52.67	46.04-52.82	50.64-53.21	50.68-53.21
			4 Relations			
ZeroR	41.96-49.30	41.96-49.30	41.96-49.30	41.96-49.30	41.96-49.30	41.96-49.30
Naive Bayes	53.47-57.46	54.65-60.13	56.03-60.92	54.37-58.65	53.52-58.83	53.50-58.71
K*	55.51-57.89	60.54-61.98	63.26-65.66	63.35-65.93	58.29-61.11	58.33-61.18
C4.5	49.79-52.66	50.96-58.53	52.32-62.44	52.36-62.29	58.25-62.50	58.27-62.20

Table 3: Classifier performances according to variable parametres (ranges indicate upper and lower performance bounds for 4 different window sizes).

test the statistical relevance amongst the various performances[9]. The results are not normally distributed so, using a non-parametric statistical test (Friedman's ANOVA - *chi square*=245.086, $p<0.05$) verify significant differences amongst the algorithms' performances. Additional Wilcoxon signed-ranks tests show significant differences amongst all algorithm pairs except for the Decision Tree/Naive Bayes pair.

In a similar way, we use a combination of Kruskal-Wallis and Wilcoxon rank-sum and Mann-Whitney tests or Friedman's ANOVA and Wilcoxon signed-ranks tests to show that the window size (chi-square=42.171, $p<0.05$) and relation set (Win2/$H(2)$=67.951, $p<0.05$, Win6/$H(2)$=68.667, $p<0.05$, WinALL/$H(2)$=45.662, $p<0.05$) size significantly affect the classification result while the feature set size does not seem to have significant differences (Win2/$H(8)$=8.162,$p>0.05$, Win6/$H(8)$=6.002, $p>0.05$, WinALL/$H(8)$=11.952, $p>0.05$). However, the results using the minimal 4-member set significantly differ from the ones obtained when the 16-member fearure set is employed.

In general, it is not the case that the more features we use the better the performance will be. The results obtained show that approximately half of the initially considered set of features yields the best results. Another outcome is the fact that when the feature set is small then all algorithms tend to perform better when a small window is considered and when the feature set grows the algorithm performance ameliorates for larger windows. However, we believe that this is due to the extended number of instances available with larger windows rather than the descriptive power of the larger feature set. In any case, more investigation in the actual effect of variable feature sets and windows is required.

7 Conclusions and Future Work

In this paper we explored the use of machine learning in the automatic induction of temporal relations between temporal elements in natural

[9]All statistical tests have been performed using the SPSS environment.

language texts. We have presented a relevant architecture and the exact design of the temporal ordering module we have implemented based on the WEKA toolkit. Based on this temporal module, we have been experimenting with the use of variable feature sets, classification values (temporal relations) and various dataset subsets containing relations referring to temporal elements of various distances between each other. As expected, all the above tunings have proven to significantly affect the performance of all the algorithms we have tested. The results as such, although not straightforwardly comparable with results reported in relevant work, seem to be of the same magnitude. In any case, our plans for future work include the exploration of the involvement of knowledge base information in the features for the machine learning algorithms, as well as the use of more training data for the learning of better classifier models.

References

(Ahn *et al.* 05) D. Ahn, S.F. Adafre, and M. de Rijke. Extracting Temporal Information from Open Domain Text: A Comparative Exploration. In *5th Dutch-Belgian Information Retrieval Workshop (DIR 2005)*, 2005.

(Allen 83) James F. Allen. Maintaining Knowledge About Temporal Intervals. *Communications of the ACM*, 26(1):832–843, 1983.

(Campbell *et al.* 02) Richard Campbell, Takako Aikawa, Zixin Jiang, Carmen Lozano, Maite Melero, and Andi Wu. A Language-Neutral Representation of Temporal Information. In *LREC 2002: Annotation Standards for Temporal Information in Natural Language*, Las Palmas, Canary Islands, Spain, 2002.

(Chinchor & Robinson 97) N. Chinchor and P. Robinson. MUC-7 Information Extraction Task Definition (version 5.1), 1997.

(Dowty 86) D. Dowty. The Effects of Aspectual Class on the Temporal Structure of Discourse: Semantic or Pragmatics? *Linguistics and Philosophy*, 9:37–61, 1986.

(Ferro *et al.* 01) L. Ferro, I. Mani, B. Sundheim, and G. Wilson. TIDES Temporal Annotation Guidelines (version 1.0.2). In *MITRE Technical Report*, 2001.

(Filatova & Hovy 01) E. Filatova and E. Hovy. Assigning Time-Stamps to Event Clauses. In *The 2001 ACL Workshop on Temporal and Spatial Information Processing*, Toulouse, France, 2001.

(Halliday 85) M.A.K. Halliday. *An Introduction to Functional Grammar*. Edward Arnold, Great Britain, 1985.

(Huang *et al.* 03) Z. Huang, K.F. Wong, W. Li, D. Song, and P Bruza. Back to the Future: A Logical Framework for Temporal Information Representation and Inferencing from Financial News. In *IEEE NLP/KE 2003 Conference*, pages 95–101, IEEE Press, 2003.

(Jahn *et al.* 04) S.B. Jahn, J Baldwin, and I. Mani. Automatic TIMEX2 Tagging of Korean News. *ACM Transactions on Asian Language Information Processing*, 3(1):51–65, 2004.

(Kamp & Reyle 93) H. Kamp and U. Reyle. *From Discourse to Logic Introduction to Modeltheoretic Semantics of Natural Language (Discourse Representation Theory)*. Kluwer, Dordrecht, 1993.

(Katz & Arosio 01) G. Katz and F. Arosio. The Annotation of Temporal Information in Natural Language Sentences. In *In Proceedings of ACL-EACL 2001, Workshop for Temporal and Spatial Information Processing*, pages 104–111, Toulouse, France, 2001.

(Li *et al.* 01) W. Li, K.F. Wong, and C. Yuan. A Model for Processing Temporal References in Chinese. 2001.

(Mani & Wilson 00) I. Mani and G. Wilson. Robust Temporal Processing of News. In *The 38th Meeting of the Association of Computational Linguistics (ACL 2000)*, pages 69–76, New Jersey, 2000.

(Mani 03) I. Mani. Recent Developments in Temporal Information Extraction. In *In Proceedings of the Conference on Recent Advances in NLP (RANLP 2003)*, Borovets, Bulgaria, 2003.

(Mani *et al.* 03) I. Mani, B. Schiffman, and J. Zhang. Inferring Temporal Ordering of Events in News. In *Proceedings of Human Language Technology Conference (HLT-NAACL'03)*, pages 55–57, Edmonton, Canada, 2003.

(Meulen 95) T. Meulen. *Representing Time in Natural Language: The Dynamic Interpretation of Tense and Aspect*. The MIT Press, Massachusetts, 1995.

(Moens & Steedman 88) M. Moens and M. Steedman. Temporal Ontology and Temporal Reference. *Computational Linguistics*, 14(2):15–28, 1988.

(Mourelatos 78) A.P.D. Mourelatos. Events, Processes, and States. *Linguistics and Philosphy*, 2:415–434, 1978.

(Nakhimovsky 88) A. Nakhimovsky. Aspect, Aspectual Class, and the Temporal Structure of Narrative. *Computational Linguistics*, 14(2):29–43, 1988.

(Puscasu 04) G. Puscasu. A Framework for Temporal Resolution. In *In Proceedings of LREC 2004*, pages 1901–1904, Lisbon, Portugal, 2004.

(Rinaldi *et al.* 03) F. Rinaldi, J. Dowdall, M. Hess, J. Ellman, G.P. Zarri, A. Persidis, L. Bernard, and H. Karanikas. Multilayer Annotations in Parmenides. In *In Proceedings of the K-CAP2003 Workshop on Knowledge Markup and Semantic Annotation*, Sanibel, Florida, USA, 2003.

(Saquete *et al.* 02) E. Saquete, P. Martinez-Barco, and R. Munoz. Recognizing and Tagging Temporal Expressions in Spanish. In *Workshop on Annotation Standards for Temporal Information in Natural Language (LREC)*, pages 44–51, Las Palmas, Canary Islands, Spain, 2002.

(Sauri *et al.* 04) R. Sauri, J. Littman, B. Knippen, R. Gaizauskas, A. Setzer, and J. Pustejovsky. TimeML Annotation Guidelines, 2004.

(Setzer 01) A. Setzer. *Information In Newswire Articles: An Annotation Scheme and Corpus Study*. Phd Thesis, Sheffield, 2001.

(Vasilakopoulos & Black 05) A. Vasilakopoulos and W. J. Black. A Comparison Between a Rule-Based and a TBL-Based Approach for Temporal Element Extraction. In *To Appear. Workshop on Text Mining Research, Practice and Opportunities.*, Borovets, Bulgaria, 2005.

(Vasilakopoulos *et al.* 04) A. Vasilakopoulos, M. Bersani, and W. J. Black. A Suite of Tools for Marking Up Temporal Text Mining Scenarios. In *In Proceedings of LREC 2004*, pages 807–810, Lisbon, Portugal, 2004.

(Vendler 67) Z. Vendler. *Linguistics in Philosophy*. Cornell University Press, New York, 1967.

(Wilson *et al.* 01) G. Wilson, I. Mani, B. Sundheim, and L. Fero. A Multilingual Approach to Annotating and Extracting Temporal Information. In *Workshop for Temporal and Spatial Information Processing (EACL-ACL 2001)*, Toulouse, France, 2001.

(Witten & Frank 00) I.H. Witten and E. Frank. *Data Mining*. Morgan Kaufmann, San Francisco, California, 2000.

Description of the guidelines for the syntactico-semantic annotation of a corpus in Spanish

Vázquez, Gloria* **Fernández-Montraveta, Ana**** **Alonso, Laura†**
*Dept. of English and Linguistics, Universitat de Lleida, Spain
gvazquez@dal.udl.es
**Dept. of English and German Philology, U. Autònoma de Barcelona, Spain
ana.fernandez@uab.es
† Department of Linguistics, Universitat de Barcelona, Spain
lalonso@ub.edu

Abstract

The aim of the SenSem project[1] is to build a databank that reflects the syntactic and semantic behavior of Spanish verbs. This databank will eventually consist of a verbal lexicon linked to a significant number of examples from corpus. These examples are being manually analyzed following the guidelines presented here.

1 The SenSem project

The final aim of the SenSem project is to create a databank (a lexicon linked to manually analyzed corpus examples) that reflects the syntactic and semantic behavior of the verbs selected. As the initial phase of the project, a reference corpus for Spanish annotated with syntactico-semantic information is being constructed.

A major problem in the SenSem project has been to bridge the gap between traditional grammatical concepts and the actual phenomena found in a corpus from real language. The final goal of the guidelines used is to bring the theoretical insights to the annotation of the actual examples found in corpus and to provide annotators with procedures as objective as possible to deal with phenomena found in corpus.

Each sentence is linked to the verb sense it exemplifies. Each verb sense is in turn associated with its Aktionsart and its argument structure in the form of a semantic role list. So sentences inherit this information from the sense.

Participants in the sentence are annotated with respect to their syntactic function, the semantic role they hold with respect to the verb and their argument or adjunct status, along the lines of the Spanish FrameNet (Subirats and Sato 2004) and ADESE (García de Miguel and Comesaña 2004). The argument head is marked, together with any metaphorical usages.

SenSem differs from other projects that treat syntactico-semantic annotation in that sentences are also tagged with respect to their semantics, both aspectual and construction.

Annotators have been trained and provided with an annotation manual. Once the sentences have been annotated, the annotation methodology requires they be validated by other linguists in order to detect possible errors and provide a more uniform treatment of problematic cases (Alonso et al. 2005).

As a result of this project, a corpus of approximately 1,000,000 words will be created, containing 100 sentences for each of the 250 most frequent verbs of Spanish. These sentences have been randomly selected from a corpus of approximately 13,000,000 words of the electronic versions of different newspapers. The journalistic register provides a high number of examples and reflects standard language usage, but a future development of this project will take into account the need to diversify the corpus. Also, we want to apply mechanisms that automatically increase the number of sentences per verb.

2 Sentence-level tagging

Two kinds of sentential semantics have been distinguished: one which concerns the aspectual information expressed in the sentence (Section 2.1), and another which specifies the semantics of its construction (Section 2.2).

[1] Databank Sentential Semantics: "Creación de una Base de Datos de Semántica Oracional". MCyT (BFF2003-06456).

2.1 Aspectual semantics

Following traditional proposals in aspectual research (Comrie 1976, Vendler 1957, Pustejovsky 1995), we distinguish between three types of classes:

- Events, those actions in which the logical culmination is implied. Verbs such as *put* or *finish* are considered events.

  ...El diálogo **acabará** hoy…
  *...The conversations **will finish** today...*

- Processes, those actions that do not have an implicit limit; they are dynamic actions that take place over a stretch of time with the same properties at any interval. Verbs such as *eat* or *live* express a process.

  …cuando le preguntaron de qué **había vivido** hasta aquel momento...
  *…when he was asked what he **had been living on** until then...*

- State denote relationships between an entity and a quality, or between an entity and a context or between two entities. Verbs such as *consist* or *come close* (where movement is not implied) are considered states.

  ...El gasto de personal **se acerca** a los 2.990 millones de euros...
  *...Personnel expenses **come close** to 2,990 million euros...*

As we have previously mentioned, lexical aspect is indicated for every lexical item in the lexical database. When a sense is chosen for a verb, the information regarding its Aktionsart is automatically assigned. Annotators can adjust it if they consider that the contextual elements modify the verb's aspectuality. We must take into account that we are annotating sentences and, therefore, some participants in the action might alter the Aktionsart.

For example, some processes are limited, that is to say they express an event when they are modified by a "bounded" object. For example, a verb such as *write*, which is lexically a process, gives an eventive reading when uttered in a verbal phrase such as *write a letter*.

Sometimes, it is the semantic type of one of the arguments that changes the lexical aspectual information. This is the case of procedural movement verbs which lexically are processes but that can be limited when the destination of the movement is expressed. When a verb like *walk* is realized together with the goal of the movement, it conveys an event instead of a process (*walk to the fence*).

Verb tense can also change the aspect of a sentence. Nevertheless, we do not consider tense as an element to take into account when analyzing the aspectuality of the sentence since we believe it should be considered at a different level. The only exception to this is the use of present to express a habitual reading (Section 2.2).

2.2 Construction semantics

We believe syntactic configurations always convey a meaning which is different to the meaning expressed by the same elements arranged differently. A speaker of a language always chooses a particular arrangement of elements for communicative purposes (Goldberg 1995).

In order to describe this level of sentential meaning, various labels related to focalization of arguments, reference binding and aspectuality are provided, as we will see next.

On the one hand, we have distinguished constructions according to which element constitutes the focus of communication. First, we have considered *anticausative* constructions. In Spanish an anticausative construction is typically a pronominal structure in which the participant upon which communicative intention falls is the entity undertaking the action and not the cause that has triggered it.

… las perspectivas que se le **abren** a Catalunya tras la llegada del PSOE al Gobierno...
*... the political horizon **opened up** in Catalunya by the installment of the PSOE political party in government...*

Secondly, we also include passive constructions and we account for both pronominal and syntactic passive constructions. They have been grouped together under the *antiagentive* tag. It is the equivalent to an anticausative construction but instead of a cause we have an agent as the element that starts the actions:

...En el peor de los casos **se construirán** o rehabilitarán en Barcelona un total de 65.000 pisos…
*...At the very least, 65.000 apartments **will be built** or rehabilitated in Barcelona…*

If the action is neither an agentive nor a causative structure, then we use the *passive* tag to indicate that the logical subject of the sentence is no longer the grammatical focus and that the logical object is acting as the functional subject of the sentence.

> ...Hasta el 40 % hay familias que se lo pueden permitir, pero cuando **se supera** este porcentaje,…
> *...Some families can afford up to 40%, but **past** this level...*

The last tag used to refer to the communicative focus is the *impersonal* tag. Whenever a sentence does not present a functional subject, the sentence is tagged as impersonal.[2]

> ...En este restaurante **se come** barato...
> *...In this restaurant one can **eat** cheap...*

On the other hand, some properties affecting reference binding are explicitly tagged, namely *reflexivity* and *reciprocity*.

> Piloto y copiloto **se cambiaron** el sitio...
> *The pilot and the copilot **exchanged** places....*

In relation to aspectuality, two specific states are distinguished: *habitual* and *middle*. The first term refers to those actions that are not truly a state, in that they do not describe a relation. However, they do not refer to a particular real-world action.

> …Wimbledon siempre **cierra** sus puertas en el primer domingo del torneo...
> *…Wimbledon always **closes** its doors the first Sunday of the tournament...*

Middle constructions are states that give information about how an entity's characteristic can be modified, such as "Este material se dobla con facilidad" –This material bends easily.[3]

Finally, we use two more categories to account for those structures expressing an *indirect cause* and *dative of interest*. We have an instance of indirect cause in those cases in which the syntactic agent is not the real, direct agent of the action.

> ..., el Gobierno también **construyó** el puente sobre el Duero...
> *..., the Government also **built** the bridge over the Duero river....*

The dative of interest includes sentences such as the following in which the indirect pronoun is used to express a possessive relation of the speaker with the object of the sentence.

> …se me ha detenido el motor…
> *… the motor died on me…*

3 Constituent-level tagging

Those constituents of the sentence that are directly dependent on the verb are assigned an interpretation at various syntactic and semantic levels. First, we determine whether a constituent is an argument or an adjunct (Section 3.1). Arguments are further labeled with respect to syntactic category (Section 3.2), syntactic function (Section 3.3) and semantic role (Section 3.4).

3.1 Arguments and adjuncts

Constituents are either arguments or adjuncts depending on whether they are required or not by the verb semantics. Some arguments are optional:

> *Maria has eaten bread - Maria has eaten*
> *He has arrived from Paris - He has arrived*

Adjuncts usually express aspects related to contextual references. Typically, the aspects that can be conveyed by such constituents are the expression of place, purpose, manner, and so on. However, this is not always true the other way around. Some verbs require the expression of these types of aspects that are compulsory because of their semantics. Consider these examples:

Arguments:
- He is feeling *well* – manner
- He lives *in Barcelona* – place
- It started *at 8:00 AM* – time
- He uses it *for writing* – purpose

Adjuncts
- Today, I worked *pretty well* – manner
- I bought it *in Barcelona* – place
- He had dinner *at 8:00 PM* – time
- He came here *to sell it* – purpose

[2] Here we are not making reference to typical cases of subject elision in Spanish. It is important to remember that subject elision in Spanish does not imply defocalization or its disappearance as a function.

[3] We have not found any constructions of this type in the corpus so we use an invented example here.

In the annotation, arguments and adjuncts are treated differently. Adjuncts are simply tagged as such without any further analysis.

3.2 Syntactic categories

Each constituent is assigned a syntagmatic category: *prepositional phrase, relative clause,* etc.

We have created categories such as *reported speech*, *comparative phrase* and *reduced clause*. Even though these categories are not traditional syntactic categories, we have considered it necessary to create them in order to adequately solve the tagging of some segments.

As a category, reported speech is very useful for labelling such complements, which are common in journalistic discourse.

> ...aunque sólo se han alcanzado récords en Lleida, destaca Antoni Gázquez.
> *...although records have only been reached in Lleida, highlights Antoni Gázquez.*

In the 'comparative phrase' label, we join together the two elements of a comparison into a single tag.

> Esta cuestión afecta más a mi padre que a mi madre.
> *This matter affects my father more than my mother.*

As for the tag 'reduced clause', we unify as an only constituent two separate constituents. Consider the example:

> ...Carod consideró normal echar de menos el cargo...
> *...Carod considered it normal to miss his* position...

Considerar has two complements, an adjective and an infinitive clause that can be converted into a single completive clause: *Carod considered that it was normal to miss his position*. With the aim of standardizing the treatment of both types of construction, we label the two complements of the former as a reduced clause.

3.3 Syntactic functions

Each constituent is also assigned a syntactic function. In addition to traditional functions such as *subject*, *predicative*, *attributive*, etc., we have distinguished three different kinds of *prepositional object* –PO– (all of which are used when annotating arguments):

- PO-1: The argument is required by the verb to form a grammatical sentence; even though it is not a prepositional verb, the verb does require a prepositional phrase to be syntactically realized. Sometimes more than one preposition is allowed; e.g. *ir a, hasta – go to, go until (you get to)*.
- PO-2: The preposition dominating the argument is determined by the verb; e.g. *acostumbrase a –get used to–, reírse de –laugh at*.
- PO-3: The complement is included in the subcategorization frame of the verb, but it is not necessarily compulsory as it is in the case of PO-1; e.g. the verb *correr –run–* can be used with or without complements and it accepts prepositions such as *a –to–* or *hasta –until (you get to)*.

3.4 Semantic roles

Each argument is assigned a semantic role. Our inventory maintains the majority of the well-established semantic roles, such as *cause*, *agent*, *theme* and *destination*. Other tags are newer and have been created in order to solve the problems that have appeared. Some of these tags are: *initiator, indirect cause, resulting state theme, initial state theme, affected theme, substitution, comparative,* and *quality.*

The role *initiator* is used to label those cases in which the promoter of the action is neither a cause nor an agent nor an experiencer, as in the case of the verb *lose*.

Indirect cause[4] is represented by verbs such as *formar –muster–,* in which the syntactic subject may not be the direct agent but rather the instigator of the action; in fact, the true agent is the object.

> ...el sargento **formó** a los reclutas para pasar revista...
> *...the sergeant* ***mustered*** *the recruits in order to pass review...*

The themes *resulting-state* and *initial-state* are required to annotate the complements of verbs such as *convertir –convert:*

> El mago **ha convertido** el pañuelo en una paloma.

[4] We have distinguished between an indirect cause at the constituent level and another at the sentence level.

*The magician **has turned** the handkerchief into a dove*

The role *affected them* is very useful as it serves to differentiate objects whose properties are modified in order to achieve the action.

...las entidades y los feriantes han **acabado** contentos...
*... the organizers and the fair show stand sponsors **are pleased** with the result ...*

The term *substitution* is used to tag arguments such as *por ti* –for you– in a sentence such as "He hablado por ti" –I spoke on your behalf–. C*ompany* is a role used in cases such as "Está con Luisa" –He's with Luisa–. Lastly, the role that identifies an object as part of attributive sentences is tagged *quality*.

...en el que Capella **ha actuado** como detective**.**
*... in which Capella **has acted** as a detective.*

Besides, we have further used two mechanisms to account for specific semantic relations between verbs and arguments. We have foreseen the possibility of double-tagging an argument using tags as *ag_exp* and *ag_t-des* when we want to express that an argument is both an agent and an experiencer or an agent and a moved-theme.

We also use more generalizing tags such as *ag/caus* or *circ*. The former is used for those verbs that can be either agentive or causative (*romper –break–*). The latter expresses circumstances of the action which are diverse in nature, such as time ("The fire started at 10") and place ("The fire started in the forest").

The semantic head of each argument constituent is also signaled. These heads will constitute the set of words required to acquire the selection restrictions of a given verb.

To avoid interference with the information provided at this level, whenever a metaphorical or metonymical complement is observed, it is marked as not to be taken into account in this process.

...Documentos TV **celebra** hoy sus 800 programas...
*...the show "Documentos TV" today **celebrates** its 800th program...*
Moreover,

4 Conclusions

We have presented a description of annotation guidelines designed to bring a theoretical perspective to the annotation of actual corpus examples. To our knowledge, no comparable guidelines have ever been made public for Spanish.

The guidelines described are flexible and they are being progressively enriched as new phenomena arise.

References

(Alonso et al. 2005) L. Alonso, J.A. Capilla, I. Castellón, A. Fernández, G. Vázquez. The SenSem Project: syntactico-semantic annotation of sentences in Spanish. RANLP 2005.

(Carreras et al. 2004) X. Carreras, I. Chao, L. Padró and M. Padró, FreeLing: An Open-Source Suite of Language Analyzers. Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC'04), Lisbon, Portugal, 2004.

(Comrie 1976) B. Comrie, Aspect. Cambridge University Press, 1976.

CoNLL. 2005. http://cnts.uia.ac.be/conll/

(García de Miguel & Comesaña 2004) J. M. García de Miguel and S. Comesaña, *Verbs of Cognition in Spanish: Constructional Schemas and Reference Points* in A. Silva, A. Torres, M. Gonçalves (eds.) *Linguagem, Cultura e Cogniçao: Estudos de Linguística Cognitiva.* Almedina, 2004, pp. 399-420.

(Goldberg 1995) A. Goldberg, *Constructions: a construction grammar approach to argument structure.* University of Chicago Press, 1995.

(Pustejovsky 1995), J. Pustejovsky, *Generative Lexicon*. Cambridge University Press, 1995.

Senseval. http://www.senseval.org/

(Subirats & Sato 2004) C. Subirats and H. Sato, *Spanish FrameNet and FrameSQL. Proceedings of 4th International Conference on Language Resources and Evaluation, Workshop on Building Lexical Resources from Semantically Annotated Corpora*, Lisbon, Portugal, 24-30 May, 2004, 2004.

(Vendler 1957) Z. Vendler, 1957, *Verbs and Times.* Philosophical Review 56, 1957, pp. 143-160.

Exploring Features to Identify Semantic Nearest Neighbours: A Case Study on German Particle Verbs

Sabine Schulte im Walde
Computational Linguistics, Saarland University
Saarbrücken, Germany
schulte@coli.uni-sb.de

Abstract

This paper addresses the influence of specific factors in feature selection, in the context of empirical studies on lexical verb semantics. We identify the semantic nearest neighbours of German particle verbs, based on distributional similarity and standard similarity measures, with a focus on features at the syntax-semantics interface. Varying the gold standard explores the types of similarities between the particle verbs and their nearest neighbours. Finally, we apply a Latent Semantic Analysis to check the effect of dimensionality on the semantic choices.

1 Introduction

German particle verbs represent a challenge for statistical NLP: They show specific patterns of behaviour at the syntax-semantics interface, and the semantic relation to their base verbs (transparency vs. opaqueness) is largely non-deterministic. We are interested in automatically inducing semantic classes for German particle verbs to determine the semantically most similar verb groups and predict the compositionality. This paper presents a preliminary step on this path: A complex analysis such as classification requires the definition of multiple parameters, of which the choice of suitable distributional features is a crucial part and should be addressed on a simplified level. In this context, we present an exploration of features to describe German particle verbs. The simplified NLP task for applying the features is to identify the semantic nearest neighbours of the particle verbs, i.e. to identify the German verbs which are semantically most similar. We specifically address the influence of three factors in feature exploration that are important in the context of distributional similarity and have not yet been raised. Future work on classification will capitalise on our insights.

First Issue. We are interested in exploring *the importance of feature selection with respect to a considerable sub-class of verbs*, and choose German particle verbs for a case study. Earlier work concerned with the distributional similarity of verbs such as (McCarthy *et al.* 03; Weeds *et al.* 04) uses standard features (e.g. grammatical dependency relations) and concentrates on the influence of similarity measures. Approaches which address feature selection with respect to semantic classes of verbs such as (Joanis & Stevenson 03; Schulte im Walde 03) explore features for verbs in general to induce classes; so far, only (Merlo & Stevenson 01) address the issue of verb subclasses, and identify semantic role features to distinguish intransitive verb classes.

Second issue. The evaluation of semantic similarity depends on the definition of a gold standard. However, available resources differ strongly in the types of semantic relations and the number of their instantiations. Previous work has ignored the influence of these evaluation parameters. We vary the gold standard (i) since it allows us to *assess the types of semantic relations* between the particle verbs and their nearest neighbours; and (ii) to get an intuition about the *influence of the gold standard size*.

Third issue. We apply a Latent Semantic Analysis (LSA) to our feature choice, to *explore whether a dimensionality reduction improves the results* by filtering the relevant information from the feature vectors, or makes the results worse by losing relevant information as provided by the feature vectors. LSA was designed to approach synonymy and polysemy of high-dimensional words (Deerwester *et al.* 90), and has been applied successfully to NLP semantic tasks such as measuring word similarity (Landauer & Dumais 97) and particle verb compositionality (Baldwin *et al.* 03). We investigate the difference of high- vs. low-dimensional vectors for our semantic task. Reaching an identical or better result with a reduced number of features would allow us to cut down on the time demands for complex NLP tasks.

2 German Particle Verbs

German particle verbs are productive compositions of a base verb and a prefix particle, whose part of speech varies between open-class nouns, adjectives, and verbs, and closed-class prepositions and adverbs. This work concentrates on prepositional particle verbs, such as *ab-holen*, *anfangen*, *ein-führen*. Particle verb senses may be transparent (i.e. compositional) or opaque (i.e. non-compositional) with respect to their base verbs. For example, *ab-holen* 'fetch' is transparent with respect to its base verb *holen* 'fetch', *anfangen* 'begin' is opaque with respect to *fangen* 'catch', and *ein-setzen* has both transparent (e.g. 'insert') and opaque (e.g. 'begin') verb senses with respect to *setzen* 'put/sit (down)'.

German particle verbs may change the syntactic behaviour of their base verbs: the particle can saturate or add an argument to the base verb's argument structure, cf. example (1) from (Lüdeling 01). Theoretical investigations (Stiebels 96) and corpus-based work (Aldinger 04) demonstrate that those changes are quite regular.

(1) Sie *lächelt.*
'She smiles.'
*Sie *lächelt* [$_{NP_{acc}}$ ihre Mutter].
'Sie smiles her mother.'
Sie *lächelt* [$_{NP_{acc}}$ ihre Mutter] *an.*
'Sie smiles her mother at.'

Even though German particle verbs constitute a significant part of the verb lexicon, recent work is mostly devoted to theoretical investigations. To my knowledge, so far only (Aldinger 04) and (Schulte im Walde 04) have addressed German particle verbs from a corpus-based perspective: (Aldinger 04) defines alternation patterns for subcategorisation frames of particle and base verbs; (Schulte im Walde 04) describes the automatic identification and quantitative analysis of German particle verbs. This work relies on the data by (Schulte im Walde 04) and explores features at the syntax-semantics interface to identify the semantically most similar verbs of German particle verbs, a preliminary step towards determining transparency/opaqueness.

Syntax-Semantics Interface Previous work on empirical verb semantics has shown that distributional similarity which models verb behaviour (mainly with reference to subcategorisation, partly including selectional preferences) is a useful indicator of semantic classes, e.g. (Merlo & Stevenson 01; Joanis & Stevenson 03; Korhonen *et al.* 03; Schulte im Walde 03). The underlying hypothesis is that to a certain extent, the lexical meaning of a verb determines its behaviour, particularly with respect to the choice of its arguments, cf. (Levin 93). To check on the behaviour-meaning relationship for the specific case of particle verbs, we use the following distributions to describe verbs.

(1) *syntax* – syntactic frame types
(2) *syntax-pp* – syntactic frame types + PPs
(3) *pref:frame-noun* – selectional preferences; nouns with reference to frame type and slot
(4) *pref:noun* – selectional preferences; nouns without reference to frame type and slot

With descriptions (1) and (2) we follow previous work and assume syntactic frames and prepositional phrases as useful indicators of verb behaviour to induce semantic similarity. Descriptions (3) and (4) take a step away and refer to specific definitions of selectional preferences.

Quantitative Verb Descriptions The quantitative data are from a statistical grammar (Schulte im Walde 03), whose parameters were estimated in an unsupervised training, using 35 million words of a German newspaper corpus. The subcategorisation information was evaluated against dictionary entries, to ensure reliability.

(1) Subcategorisation Frames: The verbs are described by probability distributions over 38 frame types. Possible arguments in the frames are nominative (n), dative (d) and accusative (a) noun phrases, reflexive pronouns (r), prepositional phrases (p), expletive *es* (x), non-finite clauses (i), finite clauses (s), copula constructions (k). For example, the frame type 'nai' indicates the subcategorisation of the obligatory nominative NP (the subject of the clause), an accusative NP (the direct object) and a non-finite clause.

(2) Subcategorisation Frames + PPs: In addition to the syntactic frame information, the frame types distinguish prepositional phrase types by distributing the probability mass of pp-frames over prepositional phrases, according to their corpus frequencies. We consider the 30 most frequent PPs, referred to by case and preposition such as 'Dat.mit', 'Akk.für'. For example, the refined frame type 'nap:Dat.mit' indicates a nominative and an accusative NP, plus a PP with the prepositional head *mit*, requiring dative case.

(3/4) Selectional Preferences: The grammar provides selectional preference information on a fine-grained level: it specifies argument realisations by their lexical heads, with reference to a specific verb-frame-slot combination. For example, the most frequent nominal heads subcategorised in the transitive frame 'na' by the verb *einsetzen* 'insert, start' are for the nominative slot *Polizei* 'police', *Regierung* 'government', *Wehr* 'army', *Bahn* 'railway services', and for the accusative slot *Gas* 'gas', *Mittel* 'means', *Kommission* 'committee', *Waffe* 'weapon'. Our distributions restrict the selectional preferences to frames which are 'relevant' for particle verbs: particle verbs do not show the same diversity of frame usage as non-prefixed verbs but rather focus on intransitive and transitive variants, including adjuncts, cf. (Aldinger 04; Schulte im Walde 04). We construct an *intransitive frame set* where we consider the nominative NPs in the frame types 'n' and 'np', and a *transitive frame set* where we consider the accusative NPs in the frame types 'na', 'nap', 'nad', 'nai', 'nas'. The frame sets therefore include the original frame types 'n' (intransitive) and 'na' (transitive), plus frame types which are their potential extensions, i.e. which add an argument/adjunct to the frame. The distributions *pref:frame-noun* and *pref:noun* refer to the probabilities of nouns in these frame types; the former distribution does encode the reference of the nouns to the specific frame and slot, the latter does not, i.e. frequencies of identical nouns in different frame types and positions are merged and then transfered to probabilities. The underlying assumption for this rather crude simplification refers to the observation that the selectional preferences of particle verbs overlap with those of semantically similar verbs, but not necessarily in identical frames (Schulte im Walde 04). Finally, we define frequency cut-offs, to investigate the influence of the number and frequency range of nouns. The cut-offs are induced from the statistical grammar, referring to the total frequencies of the nouns in the training corpus.

3 Gold Standard Resources

A gold standard in our nearest neighbour classification is applied to two tasks: (1) as source for nearest neighbour candidates, i.e. to define a set of verbs among which the nearest neighbours are chosen, and (2) to evaluate the chosen neighbours on the existence and the type of semantic relation with respect to the particle verbs. Varying the gold standard allows us to assess different types of semantic relations between the particle verbs and their nearest neighbours, and to explore the experiment setup with respect to the size of the gold standard.

GermaNet (GN) (Kunze 00) is the German version of *WordNet* (Fellbaum 98), a lexical semantic taxonomy which organises nouns, verbs, adjectives and adverbs into classes of synonyms, and connects the classes by paradigmatic relations such as antonymy, hypernymy, meronymy, etc. We extracted all particle verbs from GermaNet, a total of 1,856 verbs; for 605 of them GN provides synonyms, for 113 antonyms, and for 1,138 hypernyms. As candidate verbs we extracted all verbs related to any of the particle verbs, a total of 2,338. For comparing different sizes of verb sets, we created a reduced set of particle and candidate verbs (GN-red), by randomly extracting 25 particle verbs each with antonymy, synonymy, and direct and indirect hypernymy relations. We obtained 95 particle and 613 candidate verbs.

Dictionary (DIC): We use one out of numerous monolingual print dictionaries defining synonyms and antonyms (Bulitta & Bulitta 03), and manually copied all synonyms and antonyms for particle verbs which also appeared with a minimum frequency of 500 in the grammar model. This provides us with a total of 63 particle verbs (referring to 18 different base verbs) and 1,645 candidate verbs.

Human Associations (Assoc): In a set of two online web experiments (Melinger & Schulte im Walde 05), we obtained human associations on particle verbs. In the experiments, we asked German native speakers to list spontaneous associations. Each participant provided associations for 50/55 verbs, the total number of verbs in the experiments was 330/100. In the first experiment, 36 particle verbs were included in the 330 verbs; in the second experiment, 76 out of 100 verbs were particle verbs. Each verb was given associations by 46–54 (exp1) and 32–34 (exp2) participants. We use all associated verbs from the experiment as candidates.

Table 1 shows for each gold standard resource the number of particle verbs (pv), the number of candidate verbs (cand), the average number of candidate verbs with a semantic relation to a par-

	pv	cand	avg rel	baseline
GN	1,856	2,338	10	0.43%
GN-red	95	613	12	1.93%
DIC	63	1,645	47	2.84%
Assoc1	36	623	25	4.01%
Assoc2	76	1,040	19	1.84%

Table 1: Verbs and baseline

ticle verb (avg rel), and the average number of related verbs in relation to the number of candidate verbs. The last column represents the baseline for the experiments, since it is the chance of 'guessing' a related verb. Note that the baselines are very low because of the large number of candidate verbs.[1]

4 Semantic Nearest Neighbours

The experiments explore the semantic nearest neighbours of the German particle verbs in the following way. The particle verbs and their candidates are instantiated by probability distributions based on the feature descriptions, and for each particle verb the nearest neighbour is determined. Semantic similarity is calculated by the distance measure *skew divergence*, cf. Equation (3), a variant of the Kullback-Leibler (KL) divergence, cf. Equation (2). The skew divergence measures the distance between the particle verbs v_1 and the candidate verbs v_2 and determines the closest verb. It has been shown an effective measure for distributional similarity (Lee 01). As compared to KL, it tolerates zero values in the distributions, because it smoothes the distances by a weighted average of the two distributions compared. The weight w is set to 0.9.

$$d(v_1, v_2) = D(p \mid\mid q) = \sum_i p_i \, log \, \frac{p_i}{q_i} \qquad (2)$$

$$d(v_1, v_2) = D(p \mid\mid w * q + (1 - w) * p) \qquad (3)$$

A nearest neighbour is correct if it bears a semantic relation to the particle verb, according to the gold standard. The success of the experiments is measured by precision, the number of correct neighbours in relation to the total number of guesses, i.e. the number of particle verbs in the gold standard. Table 2 presents precision results for the different kinds of distributions. The numbers of features are given in italics. The *pref* distributions refer to the *intransitive frame set* and the *transitive frame set*, and to noun cut-offs of 10, 100, 500 and 1,000. Considering higher cut-offs than 1,000 resulted in lower precision results than in the presented table. The best number per gold standard is printed in bold.

The precision results might appear quite low at first sight; but relating them to the respective baselines (between 0.43% and 4.01%) demonstrates the success of the higher table scores. The syntactic behaviour by itself (distribution: *syntax*) is not much help for identifying semantic nearest neighbours; additional prepositional information improves the results (distribution: *syntax-pp*) only slightly. This insight is especially interesting because it is specific for particle verbs; comparable experiments on non-prefixed verbs demonstrated that *syntax-pp* information is a very useful hint for semantic verb similarity, sometimes even better than selectional preference information, cf. (Joanis & Stevenson 03; Schulte im Walde 03). For the particle verbs, the most successful distributions are clearly the nominal preferences (distributions: *pref:frame-noun* and *pref:noun*), with only slight differences between the cut-offs. Interestingly, the differences between *pref:frame-noun* (with reference to the frame) and *pref:noun* (without reference to the frame) are also minimal.

For *DIC* and *Assoc1*, the differences between the *syntax* and the *pref* variants are significant,[2] while the differences within those groups are not. For the other resources, none of the differences are significant. We conclude that the relevant information in the distributions are the nouns; the references to the argument structure (and, therefore, the functions of the nouns) are of minor importance. Triggered by the observation that the nouns play such a major role in the verb descriptions, we performed a follow-up experiment where we created verb distributions that used all nouns in the window of the respective verbs, disregarding the noun function completely. We used windows of 5, 20 and 50 words to the left and the right of the verbs, and noun frequency cut-offs as before, 10, 100, 500 and 1,000. None of the window distributions reached the results as based on the *pref* distributions; summarising, the relation of the nouns to the verbs is of minor importance (as we said above), but yet it plays a role that only

[1] We realise that our baseline is generous, but it is sufficient, since the baseline is not crucial for our exploration.

[2] All significance tests have been performed with $\chi^2, df = 1, \alpha = 0.05$.

	syntax	syntax-pp	pref:frame-noun				pref:noun			
			10	100	500	1000	10	100	500	1000
	38	*183*	*81,710*	*51,092*	*22,314*	*13,570*	*14,371*	*5,989*	*2,072*	*1,170*
GN	2.13	3.11	8.11	8.77	7.87	7.38	**9.67**	9.59	9.26	8.11
GN-red	6.32	9.47	17.89	17.89	15.79	12.63	**20.00**	**20.00**	**20.00**	15.79
DIC	6.35	12.70	33.33	34.92	**36.51**	34.92	31.75	31.75	33.33	31.75
Assoc1	16.76	22.22	50.00	50.00	50.00	47.22	50.00	52.78	**55.56**	50.00
Assoc2	9.21	11.84	**21.05**	**21.05**	**21.05**	19.74	18.42	15.79	15.79	19.74

Table 2: Precision of *skew* nearest neighbours

nouns with specific functions are included in the distributions. In addition, varying the frequency cut-offs for nouns illustrates that using very high or very low cut-offs (referring to using most vs. only high-frequent nouns) tends to be less successful than keeping to a medium range.

The results with *syntax* and *syntax-pp* show that the syntax-semantics mapping hypothesis does not apply to particle verbs as it does to verbs in general, and we provide the following explanation. *Transparent particle verbs* are semantically similar to their base verbs, but nevertheless do not necessarily agree with them in their syntactic behaviour. (Recall that German particle verbs may change the syntactic behaviour of their base verbs, cf. Section 2.) And since we know that semantically similar non-prefixed verbs show agreement in their behaviour to a large extent, we assume that the frame mismatch transfers from the base verbs to other verbs in their respective semantic class. This means that a syntactic description of transparent particle verbs and semantically similar verbs is not expected to show strong overlap. As a follow-up step on this insight, future work will implement Aldinger's alternation patterns for subcategorisation frames of particle verbs and their base verbs, and investigate whether the syntactic features are more helpful when they include the regular mappings of typical frames. For *opaque particle verbs*, we cannot make strong statements. Since they compositionally represent idioms, we assume that they undergo the syntax-semantic relationship, i.e. that they behave similarly as semantically similar verbs. For both particle verb categories, there is general agreement in the selectional preferences of particle verbs and verbs in the same semantic class, as the *pref* results illustrate.

Comparing the results with respect to the gold standard resources, we observe strong differences; for *Assoc1* we obtain significantly better results than for all other resources except *DIC*. *GN* is significantly worse than most other resources. The differences illustrate the difficulty of the task; it is easier to 'guess' a correct nearest neighbour for *DIC* and *Assoc1* than for the other resources, especially *GN*, cf. Table 1. This has to do with the size of the resources and also with their 'generosity' of providing related verbs. Furthermore, the semantic nearest neighbours allow us to investigate the kinds of semantic relations which are detected. In the GermaNet results, the hypernyms dominate the relations: the neighbours in the best results include 72/68% hypernyms, 23/21% synonyms, and 2/0% antonyms; in some cases the neighbours are defined in GermaNet as both synonyms and hypernyms (e.g. *anfeuern* 'shout encouragement'–*animieren* 'animate' where *animieren* can be a synonym or a hypernym). The fact that the hypernyms dominate the results is not surprising, because they represent 44% of the current GN relations (as compared to 10% synonyms and 1% antonyms), but the proportion is even stronger than in GN. This means that our distributional similarity corresponds rather to the GermaNet hypernym than the GermaNet synonym/antonym definitions. In the dictionary results, we encounter more balanced proportions: 43% synonyms vs. 48% antonyms, plus 2 cases defining a synonymous and antonymous relation at the same time. Still, as compared to 51% and 49% of all encoded relations representing synonyms/antonyms, the proportion of antonyms in our results is slightly stronger than for synonyms. Finally, the human associations demonstrate a more variable picture of semantic verb relations: we find a large number of synonyms or near-synonyms such as *abhalten–veranstalten* 'arrange, organise', *zunehmen–ansteigen* 'increase'; antonyms such as *aufhören–anfangen* 'stop' vs. 'begin', *einpacken–auspacken* 'pack' vs. 'unpack'; but only a few hypernyms such as *aufbrechen–öffnen* 'break open' vs. 'open', *einschärfen–mitteilen* 'inculcate' and 'inform'. In addition, we find verb pairs with backward presupposition, such as *abstürzen–fliegen* 'crash (with re-

spect to a plane)' and 'fly', causal relations such as *einbrocken–auslöffeln* 'get into/out of trouble', *einstürzen–renovieren* 'collapse' and 'renovate', and verbs referring to temporally related script-based events, such as *einschenken–trinken* 'pour' and 'drink', and *umbringen–sterben* 'kill' and 'die'. The examples show that semantic similarity as based on our distributional similarity refers to a variety of semantic relations, which are not covered by the standard manual resources. Future work will address the question of which kinds of features/distributions are associated with which kinds of relations.

5 Latent Semantic Analysis

In a final step of feature exploration, we apply a Latent Semantic Analysis (LSA) to the feature distributions, and then identify the nearest neighbours on basis of the LSA matrix. LSA is a technique for dimensionality reduction which was introduced by (Deerwester *et al.* 90) to address the synonymy and polysemy of high-dimensional word vectors. It performs a Singular-Value Decomposition on high-dimensional vectors: The original *object* × *feature* matrix $M_{o \times f}$ is represented as the product of three matrixes $O_{o \times k} * S_{k \times k} * F_{k \times f}$, with the diagonal of S as the linearly independent singular vectors. Choosing k considerably smaller than the original number of dimensions f, the matrix O represents a dimensionality reduction of M, approximating a least squares best fit to M. The optimal number of dimensions varies, depending on the task.

The goal of applying LSA to our data is twofold: (i) to explore whether a dimensionality reduction improves the results by using relevant information from the feature vectors, or makes the results worse by losing relevant information provided by the vectors; (ii) reaching an identical or better result with a reduced number of features cuts down on time demands for NLP tasks. As basis for LSA, we use the most successful verb-feature combination from our experiments, with *Assoc1* as gold standard and *pref:noun* and cut-off 500 as feature set. The verb-noun matrix has $623 \times 2,072$ dimensions. As matrix values we use (a) the original verb-noun co-occurrence frequencies f_{vn}, (b) the frequencies transformed to their logarithm: $f_{vn}^{log} = log(f_{vn} + 1)$, and (c) weighted by their *idf (inverse document frequency)* value: $f_{vn}^{idf} = f_{vn} * log(N/n)$, with N the total number

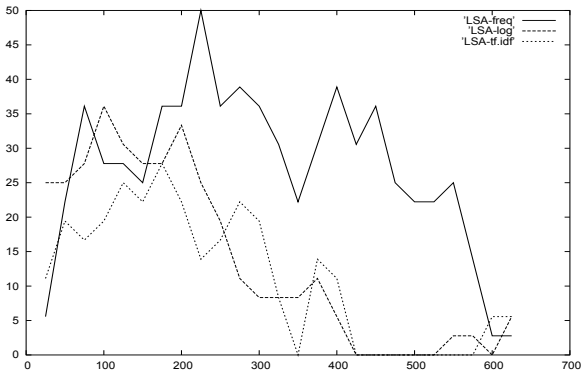


Figure 1: Precision for varying dimensions

of features, and n the number of features a verb co-occurs with. The transformations (b) and (c) are common matrix transformations in LSA, cf. (Deerwester *et al.* 90; Manning & Schütze 99). LSA is applied to the three matrixes, and the feature dimensions are systematically reduced to $k = 25, 50, ..., 2050$. Since the lower-dimensional vectors are not probability distributions, we cannot apply the *skew divergence*; we use the *cosine* of the vectors' angle, another standard measure. For comparison reasons, our previous experiments were repeated with the *cosine*; the precision for *Assoc1/pref:noun500* is 38.89%, non-significantly worse than the *skew divergence* result (55.56%).

Figure 1 shows the precision results of identifying the semantic nearest neighbours with the LSA matrices (a) LSA-freq, (b) LSA-log, and (c) LSA-tf.idf. LSA does improve the results on semantic neighbourhood, but only when performed on the original frequencies, and only with specific dimensionality (225 dimensions). That LSA is most successful on the original frequencies is surprising, since previous work emphasised the importance of feature weighting for LSA, e.g. (Landauer & Dumais 97). The improvement is non-significant. In addition, even the best results with the *cosine* measure for reduced dimensionality are still below the results as obtained with the *skew divergence* for the original probability vectors.

Summarising, in our task of identifying semantic nearest neighbours on the basis of specific verb-noun relations, the task precision suffers from reducing the matrix information by LSA. Only when using the original frequencies and with certain dimensionality, the task-relevant information is preserved. However, for the purpose of time-saving experiments, a single specific reduction is sufficient. In conclusion, it is advisable to apply LSA (and invest the time to find the op-

timal dimensions) only in cases where succeeding experiments profit from the reduced number of features.

6 Summary

In this paper, we addressed the influence of three factors in feature exploration that are important in the context of distributional semantic similarity. In a case study on German particle verbs the task was to determine their semantic nearest neighbours. First, we showed that the effect of features at the syntax-semantics interface differs for particle verbs as compared to the standard case of non-prefixed verbs. In accordance with theoretical observations, the relevant information in the distributions are the nouns; the references to the argument structure (and, therefore, the functions of the nouns) are of minor importance. Our results illustrate the importance of feature selection with respect to a specific set of data and the task. Second, we varied the gold standard in the evaluation of the nearest neighbours, to check the dependencies on the various types of similarities and the number of correct solutions. We demonstrated that the precision is related to the number of correct choices, which shows how much the size of the gold standard influences the success. Our best result was a precision rate of 55.56%, as compared to a baseline of 4.01%. This result was obtained on a gold standard of human associations in web experiments. It outperforms precision values for gold standard resources encoding only synonymy, antonymy and hypernymy, and illustrates that semantic similarity as based on our distributional similarity refers to a variety of semantic relations, such as temporal and causal relations, which are not covered by the standard manual resources. Finally, a dimensionality reduction by LSA reduced the features to an optimised number of dimensions. In contrast to previous work, we demonstrated that LSA on the original frequency distribution is more appropriate for our data and task than using the weighted versions. But only specific lower-dimensional representations outperform the high-dimensional representations, so it is advisable to apply LSA only in cases where succeeding experiments profit from the reduced number of features. In future work we will investigate which of our insights transfer from the case study to the general case of German verbs.

References

(Aldinger 04) Nadine Aldinger. Towards a Dynamic Lexicon: Predicting the Syntactic Argument Structure of Complex Verbs. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, 2004.

(Baldwin *et al.* 03) Timothy Baldwin, Colin Bannard, Takaaki Tanaka, and Dominic Widdows. An Empirical Model of Multiword Expression Decomposability. In *Proceedings of the ACL-2003 Workshop on Multiword Expressions*, 2003.

(Bulitta & Bulitta 03) Erich Bulitta and Hildegard Bulitta. *Wörterbuch der Synonyme und Antonyme.* Fischer Taschenbuch Verlag, 2003.

(Deerwester *et al.* 90) Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society of Information Science*, 1990.

(Fellbaum 98) Christiane Fellbaum, editor. *WordNet – An Electronic Lexical Database.* MIT Press, 1998.

(Joanis & Stevenson 03) Eric Joanis and Suzanne Stevenson. A General Feature Space for Automatic Verb Classification. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics*, 2003.

(Korhonen *et al.* 03) Anna Korhonen, Yuval Krymolowski, and Zvika Marx. Clustering Polysemic Subcategorization Frame Distributions Semantically. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, 2003.

(Kunze 00) Claudia Kunze. Extension and Use of GermaNet, a Lexical-Semantic Database. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation*, 2000.

(Landauer & Dumais 97) Thomas K. Landauer and Susan T. Dumais. A Solution to Plato's Problem: the Latent Semantic Analysis Theory of Acquisition, Induction and Representation of Knowledge. *Psychological Review*, 1997.

(Lee 01) Lillian Lee. On the Effectiveness of the Skew Divergence for Statistical Language Analysis. *Artificial Intelligence and Statistics*, 2001.

(Levin 93) Beth Levin. *English Verb Classes and Alternations.* The University of Chicago Press, 1993.

(Lüdeling 01) Anke Lüdeling. *On German Particle Verbs and Similar Constructions in German.* CSLI Publications, 2001.

(Manning & Schütze 99) Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing.* MIT Press, 1999.

(McCarthy *et al.* 03) Diana McCarthy, Bill Keller, and John Carroll. Detecting a Continuum of Compositionality in Phrasal Verbs. In *Proceedings of the ACL-SIGLEX Workshop on Multiword Expressions*, 2003.

(Melinger & Schulte im Walde 05) Alissa Melinger and Sabine Schulte im Walde. Evaluating the Relationships Instantiated by Semantic Associates of Verbs. In *Proceedings of the 27th Annual Conference of the Cognitive Science Society*, 2005.

(Merlo & Stevenson 01) Paola Merlo and Suzanne Stevenson. Automatic Verb Classification Based on Statistical Distributions of Argument Structure. *Computational Linguistics*, 2001.

(Schulte im Walde 03) Sabine Schulte im Walde. *Experiments on the Automatic Induction of German Semantic Verb Classes.* Unpublished PhD thesis, Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart, 2003.

(Schulte im Walde 04) Sabine Schulte im Walde. Identification, Quantitative Description, and Preliminary Distributional Analysis of German Particle Verbs. In *Proceedings of the COLING Workshop on Enhancing and Using Electronic Dictionaries*, 2004.

(Stiebels 96) Barbara Stiebels. *Lexikalische Argumente und Adjunkte. Zum semantischen Beitrag von verbalen Präfixen und Partikeln.* Akademie Verlag, 1996.

(Weeds *et al.* 04) Julie Weeds, David Weir, and Diana McCarthy. Characterising Measures of Lexical Distributional Similarity. In *Proceedings of the 20th International Conference of Computational Linguistics*, 2004.

Noun Phrases and Named Entities in Biomedical Texts: Does Domain Change Without Retraining Matter?

Joachim Wermter [a] **Udo Hahn** [a] **Juliane Fluck** [b]

[a] Jena University Language and Information Engineering (JULIE) Lab, D-07743 Jena, Germany

http://www.coling.uni-jena.de

[b] Fraunhofer Institute – SCAI Bioinformatics, D-53754 Sankt Augustin, Germany

http://www.scai.fraunhofer.de/bio.html

Abstract

Many text mining and information extraction systems rely on basic syntactic routines such as part-of-speech tagging or phrase chunking. In the biomedical domain, such systems, due to the lack of sufficient biomedical training data, often make use of NLP tools trained and evaluated on newspaper-language data only. Scientific texts in the life sciences, however, differ markedly from general language in the structure and complexity of noun phrases. Therefore, we tested the effects this domain change has on the performance of three machine-learning-based base NP chunking systems. Originally trained on PENN TREEBANK newspaper tagging and chunking annotations, we ran these systems on the GENIA treebank which contains annotations for MEDLINE biomedical abstracts. We, first, observed a significant over-all loss in chunking performance (on the order of 5 percentage points F-score) and, second, found (with the exception of the SVM-based system) no significant difference between the performance of the alternative systems. The performance loss can partly be remedied by few biomedical domain-specific adaptations. Third, we show that base NP chunking is beneficial for the crucial task of biological named entity recognition and that, much to our surprise, all systems perform extremely well in recognizing the named entity parts of base NPs, despite the domain change involved.

1 Introduction

In the life sciences domain, a large fraction of information is only available in form of unstructured free text, such as medical narratives, technical reports or scientific articles. By now, the sheer volume of this literature makes it almost impossible for biologists, clinical researchers and medical professionals to retrieve all relevant information on a specific topic and to keep up with current research. Effective named entity recognition and subsequent information extraction is therefore a major challenge in molecular biology and genome-based clinical research.

Fortunately, the field of human language technology (HLT) makes available various tools for text mining in order to automatically extract relevant information locked in free text. Their benefits are to filter out relevant information, to extract structured knowledge from large, unstructured text collections, or to support database curators and providers in locating the crucial information and extracting it for database updates.

Many HLT applications distinguish different levels of text analysis. At the basic processing level, part-of-speech (POS) tagging accounts for the assignment of part-of-speech tags to text tokens. A subsequent step focuses on the identification of structurally related groups of words, noun phrases (NPs) mostly, a task usually referred to as (base NP) chunking.

In the biomedical domain, base NP chunking may not only serve as a necessary preprocessing step for higher-level syntactic analysis such as (partial or shallow) parsing, but it may also be worthwhile to investigate whether it could be a beneficial preprocessing step to the vital task of biomedical named entity recognition (NER) as well. Currently, only few biomedical named entity recognizers employ linguistic preprocessing beyond the POS tagging stage (e.g., (Litrán *et al.* 04; Park *et al.* 04; Finkel *et al.* 04)). Although it seems plausible to assume that most biomedical named entities, such as protein, gene or cell names, are linguistically expressed within base noun phrases, this linguistic intuition has not been investigated yet in depth and thus calls for a thorough empirical examination.

Due to the lack of sufficient syntactically annotated biomedical training data most text mining systems in the life sciences domain (e.g., (Pustejovsky *et al.* 02; Narayanaswamy *et al.* 03; Saric *et al.* 04) have to make direct use of NLP tools for POS tagging and chunking, which were developed for general-purpose language studies. Hence their performance on biomedical language (or any other domain-specific sublanguage) has not been evaluated up until now. Moreover, although work has been done on full parsing methods in the biomedical domain (Yakushiji *et al.* 01; McDonald *et al.* 04), the resource-intensive nature of biomedical text processing might require the use of syntactic processing tools operating at a shallower level of analysis for the sake of robustness and efficiency. The question thus arises whether such tools – and, if so, which ones – are portable to the biomed-

ical domain without a considerable performance loss. Results on POS tagging indicate already that different methods vary as for performance loss when domains and text genres change (Campbell & Johnson 01; Wermter & Hahn 04).

2 Methods

In this paper, we focus on the exemplary evaluation of noun phrase chunking in the biomedical domain and examine three general-purpose chunkers which rely on statistical machine learning techniques and are trained on a common newspaper-language corpus: YAMCHA (Kudo & Matsumoto 01), a kernel-based support vector machine system, TBL (Ramshaw & Marcus 95), a base NP chunking tool that learns transformation rules, and BOSS, a statistical chunking tool developed at Jena University.

2.1 The Training and Test Environments

All three chunking tools were trained on the standard data set for base NP[1] chunking, *viz.* Sections 15-18 of the *Wall Street Journal* part of the PENN TREEBANK (Marcus *et al.* 93). This benchmark set amounts to 211,727 tokens which are POS-tagged (Brill 95) with the PENN TREEBANK (PTB) tagset and chunk-annotated using the standard Inside/Outside (or IOB)[2] chunk representation. This was first introduced by (Ramshaw & Marcus 95) and since then canonically applied to base NP chunking. Typically, ML-based chunking systems make use of the available types of linguistic information (i.e., word and POS information) in the training corpus in order to estimate their model parameters.

The test set on which we evaluated the different systems was derived from the Beta version of the GENIA treebank,[3] a subset of the GENIA corpus (Ohta *et al.* 02), which comprises 200 syntactically annotated MEDLINE abstracts from the molecular biology domain. Although GENIA is POS-tagged using the PTB tagset, its POS-annotation scheme had to be changed (and is thus different to the PTB scheme) to account for various properties specific to text from the molecular biology domain (Tateisi & Tsujii 04). Among these are (non-proper) names beginning with capital letters (e.g., "*NFAT*", "*RelB*"), chemical and numeric expressions including non-alphanumeric characters such as commas, parentheses, or hyphens (e.g., "*beta-(1,3)-glucan*"), participles of unfamiliar verbs describing domain-specific events, and fragments of words (e.g., "*up- and downregulate*").

In conformance to already established evaluation metrics (Sang & Buchholz 00), the GENIA treebank was automatically converted to the IOB-format (cf. Table 1). We thus obtained a test set which runs about 34,000 tokens in size.

Tokens	POS tag	Base NP	Named Entity (NE)
a	DT	I-NP	O
mechanism	NN	I-NP	O
that	WDT	B-NP	O
increases	VBZ	O	O
NF-kappa	NN	I-NP	B-protein
B/I	NN	I-NP	I-protein
kappa	NN	I-NP	I-protein
B	NN	I-NP	I-protein
dissociation	NN	I-NP	O
without	IN	O	O
affecting	VBG	O	O
the	DT	I-NP	O
NF-kappa	NN	I-NP	B-protein
B	NN	I-NP	I-protein
translocation	NN	I-NP	O
step	NN	I-NP	O

Table 1: The standard IOB NP chunk tag annotation together with the aligned IOB NE annotation in the GENIA corpus

2.2 Base NP Chunking and NE Recognition

Base NP chunking may also be a beneficial linguistic preprocessing step to recognize named entities (NEs), a particularly hard and important task in the biomedical domain. In order to see how NP chunking and NE recognition are interrelated, we took the version of the GENIA Named Entity corpus which was used for the JNLPBA Bio-Entity recognition task (Kim *et al.* 04) and selected those abstracts which are also contained in the GENIA treebank. By doing so, we were able to align the IOB base NP annotation of the treebank with the IOB bio-entity annotation of the NE corpus,[4] which is shown in Table 1.[5]

[1] *Base NPs* are defined as non-recursive noun phrases ending after their nominal head and excluding any type of postmodification (e.g., prepositional phrases, attributes, appositions). Base NP recognition is a standard task used to compare different HLT methods. This definition of *Base NPs*, however, is often not consistently obeyed in syntactic corpus annotations. Thus, an expression such as *"Mucopolysaccharidosis Type IV"* may be annotated as a base NP as well.

[2] I = current token is inside of a chunk, O = current token is outside of any chunk, B = current token is the beginning of a chunk immediately following another chunk.

[3] `http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/topics/Corpus/GTB.html`

[4] There is no direct link in the GENIA corpus between the NE annotation, on the one hand, and the syntactic annotation, on the other hand. This is why we had to go through this rather elaborate process in aligning/linking both types of information.

[5] Besides *proteins*, there are four other types of named entities annotated in this version of the GENIA Named Entity corpus, viz. *DNA*, *RNA*, *cell types* and *cell lines*.

Default	PTB corpus			GENIA corpus		
	Recall	Precision	F-score	Recall	Precision	F-score
YAMCHA	94.29	94.15	94.22	89.00	89.30	89.15
BOSS	89.92	90.10	90.01	86.46	86.84	86.65
TBL	92.27	91.80	92.03	86.31	85.49	85.90
BOSS$_{Par}$				87.25	89.19	88.21

Table 2: Benchmark results of the different systems as default. BOSS$_{Par}$ uses a pattern which recognizes NP-internal parentheses

2.3 The Machine-Learning-based Chunkers

YAMCHA (Kudo & Matsumoto 01) is an open source text chunker based on so-called Support Vector Machines (SVMs). Typically, SVMs are binary classifiers and thus must be extended to multi-class classifiers to classify three (as in the case for NP chunking with (I,O,B)) or more classes (see (Vapnik 98) for the underlying statistical learning theory). Typically, they map their n-dimensional input space into a high-dimensional feature space in which a linear classifier is then constructed. Generally, this approach requires considerable computational resources. Hence, various methods are employed by YAMCHA to reduce the training costs incurred by this approach (see (Kudo & Matsumoto 01) for details).

TBL – Transformation-based error-driven learning (Ramshaw & Marcus 95) starts with a training corpus specifying the correct values for the linguistic features of interest, a baseline heuristic for predicting initial values for these features, and a set of rule templates that determine a space of possible transformational rules. Model learning is achieved by iteratively testing and improving hypotheses using the rule templates. TBL turned out to be one of the standard systems used for base NP chunking.

BOSS – the chunking system developed at Jena University's Language and Information Engineering (JULIE) Lab predicts borders of noun phrases (beginning and end points) based on statistical criteria.[6] These predictions are estimated by combining the observed probabilities of NP borders and NP POS patterns in a training corpus. The challenge is to pair the predicted borders in an 'optimal' way so that non-overlapping phrases are identified. BOSS, in analogy to (Muñoz *et al.* 99), finds the pairing with the maximal value using a shortest-path algorithm. At its current development stage, BOSS is comparatively knowledge-poor as it only uses POS information from the training corpus, whereas both YAMCHA and TBL also integrate lexical and word feature information.

[6] Viewing noun phrase recognition as a border finding problem was first introduced by (Church 88).

3 Experiments and Results

In evaluating the performance of the chunkers on the GENIA test set, we ran three experiments. The first one used all systems in their default configuration leaving the parameters from their PENN TREEBANK training unchanged. We also performed an error analysis to investigate which linguistic properties of base NPs proved to be particularly troublesome for the default chunkers on the GENIA test set. In the second experiment, we made an in-domain adaptation to BOSS (yielding BOSS$_{Par}$)— a simple bio-domain-specific pattern was introduced, which allows to recognize NPs with internal parentheses such as in "*interleukin 2 (IL-2) activation*". In the third experiment, we examined which proportion of biological NEs in GENIA are contained within base NPs and how the different systems fared in recognizing these NE constituent parts of base NPs.

3.1 Evaluation of the Different Systems

The three chunking tools were all trained and tested on the same PENN TREEBANK (PTB) general-language newspaper corpus data set.[7] Table 2 contains the performance figures of the three default systems on the GENIA corpus plus the result of the adapted BOSS$_{Par}$ system. All systems made use of the manually annotated POS tag information in both the PTB and the GENIA corpus, i.e., no prior automatic POS tagging was performed.

As far as the default systems are concerned, the YAMCHA kernel-based support vector machine performs best on both corpora, but loses approximately 5 percentage points of performance (from an F-score of 94.22% to 89.15%) on the GENIA corpus. The TBL method, which performs second best on the PENN TREEBANK corpus (F-score: 92.03), performs worst on the biomedical corpus (with a F-score of 85.9%). Of all ML-based systems, the BOSS system has the lowest performance on the PENN TREEBANK corpus but faces the least loss (only 3.36 percentage points)

[7] The results for YAMCHA and TBL are reported in (Kudo & Matsumoto 01) and (Ramshaw & Marcus 95), respectively.

on the GENIA corpus, on which it performs second best. Its comparatively low performance on PTB and its smaller loss on GENIA may be explained by the fact that it only utilizes POS information for chunking but no lexical information like the two other chunking systems do.

3.2 Overall Error Analysis on GENIA

For error analysis, the false negative hits and the false positive hits were sorted with the help of the positional IOB chunk tag information. The hits were then compared in a pair- and n-wise fashion between the different systems and thus allowed them to be compared as to whether they assign the same erroneous IOB chunk tag to the same token, i.e., their common mistakes could be identifi ed.

For the false negative (FN) hits (cf. Figure 1), the YAMCHA system had the highest number of mistakes (735), whereas the other two systems had quite similar lower error rates. The proportion of common mistakes between all three systems is 53.2% according to the system with the lowest error rate (BOSS), and varies between 63.9% and 68.8% on pairwise comparisons. According to the system with highest error rate (YAMCHA), 39.5% of all mistakes are common to the three chunkers, whereas the overlap on a two-system comparison basis ranges from 47.3% to 66.8%.

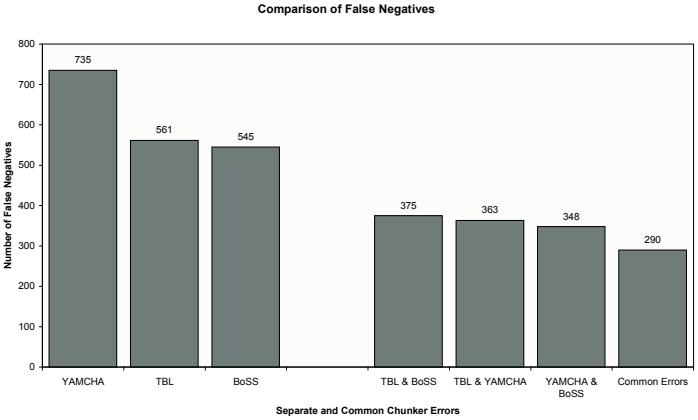


Figure 1: False Negative (FN) errors based on positional IOB chunk tag information

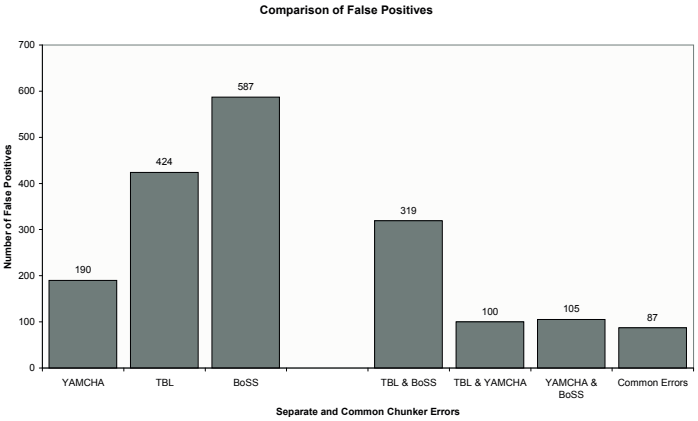


Figure 2: False Positive (FP) errors based on positional IOB chunk tag information

For the false positive (FP) rates (cf. Figure 2), the BOSS system came up with the highest number of errors (587) followed by the TBL system (424). On this dimension, YAMCHA performs by far the best with the lowest error rate (190). In particular, the error overlap between BOSS and TBL is very high (75%) in comparison to 53% and 55% error overlap from these systems to YAMCHA.

3.3 Error Type Analysis on GENIA

Although the false negative/positive error rates shed some light on the *overall* performance of each system, they alone do not *explain* the performance on the GENIA corpus. Therefore, we tried to identify the most common *error types* across the different systems by looking at the part of speech and the context of each false negative/positive hit (see Tables 3 and 4 for their distribution).

There were certain linguistic constructions around which error types could be established for FNs (i.e., tokens that were not recognized as part of an NP) as well as for FPs (i.e., tokens that were erroneously identifi ed as part of an NP). The following list enumerates the most salient ones:[8]

- NPs with coordinated/enumerated elements **(Coord)**, e.g.,
  FN: *new DNA binding proteins of 85, 75 and* 54 kDa*
  FP: *Cyclosporin A and FK506 inhibit T- and B-cell activation and* other processes*

- NPs with internal parenthesized/bracketed elements **(Par)**, e.g.,
  FN: *chloramphenicol acetyl-transferase (CAT)* gene expression*
  FP: *human immunodeficiency virus type 1 (HIV-1)**

- NPs with verbal forms in prenominal adjective function, **(Verbal)**, e.g.,
  FN: *from resting* and induced* ML-1 cells*
  FP: *a specific target termed* TAR*

[8] The underscored items marked with an asterisk (*) are misclassified as an FN or FP by some or all systems as for their correct IOB chunk tag.

Method	Coord	Par	Verbal	Adv	Adj
YAMCHA	52.0 (382)	21.6 (159)	8.3 (61)	3.3 (24)	3.3 (24)
BOSS	30.5 (166)	34.7 (189)	12.1 (66)	2.2 (12)	5.0 (27)
TBL	37.4 (210)	23.4 (131)	14.8 (83)	3.7 (21)	7.8 (44)

Table 3: Distribution of error types (in %, with absolute numbers in parentheses) for false negatives

Method	Coord	Par	Verbal	Adv	Adj
YAMCHA	40.5 (77)	9.5 (18)	4.7 (9)	6.3 (12)	20.5 (39)
BOSS	52.3 (307)	1.2 (7)	8.5 (50)	6.3 (37)	11.8 (69)
TBL	60.1 (255)	6.8 (29)	9.2 (39)	3.8 (16)	9.2 (39)

Table 4: Distribution of error types (in %, with absolute numbers in parentheses) for false positives

- NPs with adverbs modifying prenominal elements **(Adv)**, e.g.,
  FN: *abnormally* low plasma cysteine levels*
  FP: *Together* these results constitute...*

- Adjectives **(Adj)** in various functions, e.g.,
  FN: *the expression of endogenous AP-1 regulated* genes*
  FP: *16 patients, aged* 16-27 years,...*

In terms of the error type distribution, the most frequent FN and FP errors occur with coordination elements. It is a dominant error source for all systems (YAMCHA: 52% FN and 40.5% FP; BOSS: 52.3% FP), except for BOSS, for which parenthesized elements are the most common FN error type (34.7%). The fact that the BOSS system very often erroneously recognizes coordinative elements[9] as part of an NP must be attributed to the fact that it does not utilize any lexical information. As for FNs, verbal forms in prenominal adjective functions as well as noun phrases with parenthesized/bracketed elements are other common error sources for all default systems. In particular, the latter error source is particularly characteristic of the biomedical domain.

Such elements, however, can be recognized in a straightforward way by checking whether the opening parenthesis is directly preceded and the closing one directly followed by an NP (i.e., by a chunk I-tag). We thus examined in an exemplary way whether such a heuristic adaptation facilitating the recognition of these types of NPs would lead to any performance increase on the BOSS system. As can be seen in our results in Table 2, BOSS_{Par} indeed increased its base NP recognition performance by a 1.6% F-score. In particular, this heuristic performed a boost on its precision value by 2.3%.

[9] This mistake is also responsible for its overall high number of FP errors.

3.4 Recognizing NE Parts of Base NPs

In order to assess the impact of base NP chunking on biological NE recognition, we first examined which proportion of the annotated biological NEs are actually contained within base NPs in the GENIA test set. The total number of NEs (in this scenario, we considered names for proteins, DNA, RNA, cell types, and cell lines only) amounts to 3,065. As can be seen from the second row of Table 5, almost all of these NEs are contained within base NPs, *viz.* 99.1% (3,037 out of 3,065). This is already good evidence for the usefulness of base NP chunking for biological NER because, by first identifying base NP chunks, the search space for named entities is considerably reduced. Furthermore, a manual inspection of the cases in which NEs are not contained within base NPs revealed that this is mainly due to two factors:

- Boundary marking errors in the GENIA NE annotations. For example, in the utterance "*in the [TCR alpha beta lineage ,]*$_{cell-type}$ *however, ...*", the first comma after the word *lineage* is erroneously marked as part of the NE.

- NE type/boundary annotation errors. For example, complex NPs (e.g., *[interferon-alpha in U937 cells]*$_{cell-line}$) are erroneously marked as one NE, although they are constituted of two distinct NE types (e.g., *[interferon-alpha]*$_{protein}$ *in [U937 cells]*$_{cell-line}$).

Accordingly, the base NP chunking performance of the three systems we dealt with may also be reexamined from the perspective of biological NER. For this purpose, we checked how many of the NEs inside base NPs (labeled as NE_{baseNP}) were correctly

	YAMCHA	TBL	BOSS	BOSS_{Par}
NE_{baseNP} recognized	98.0% ($\frac{2977}{3037}$)	97.6% ($\frac{2965}{3037}$)	97.7% ($\frac{2968}{3037}$)	99.2% ($\frac{3012}{3037}$)
Total proportion of NE_{baseNP}	99.1% ($\frac{3037}{3065}$)			

Table 5: Accuracy of the chunking systems in recognizing the NE part of base NPs in the GENIA test set (first row). Proportion of NEs actually contained within base NPs (second row).

recognized by the chunking systems. We quantified this in the following way: An NE was correctly recognized if the part of the base NP containing it was also recognized by the system, no matter whether any preceding or following material not belonging to the NE (adverbs, adjectives, coordination markers, etc.) was recognized or not. These results are given in the first row of Table 5.

First, it can be seen that all default systems perform very well in recognizing the NE part of base NPs, with YAMCHA hitting 98% of these parts, BOSS is only 0.3 points less (97.7%), TBL only 0.4 points (97.6%). The domain-adapted BOSS_{Par} outperforms all of them (even YAMCHA) by recognizing 99.2% of the NE parts of base NPs. This result is noteworthy given the superior performance of YAMCHA on the general base NP chunking task.

4 Discussion and Conclusions

We evaluated the performance of three machine-learning-based systems which perform noun phrase recognition on a biomedical text corpus (GENIA). All three systems were trained on the PENN TREEBANK newspaper corpus. The F-score performance on the newspaper corpus ranges between 90% and 94% and drops down to ranges between 85% and 89% on the biomedical corpus. Porting chunkers to the life sciences domain, therefore, implies a substantial loss of performance.

Furthermore, the drop of performance is system-dependent. The kernel-based support vector machine system, YAMCHA, performs best on both corpora but still loses 5 points on the GENIA corpus. The performance loss for TBL is even higher (over 6 points). By contrast, the border-finding chunking tool BOSS only loses 3.36 points.

Despite of these differences, we postulate that standard ML approaches (with systems trained on a newspaper corpora) yield comparable performance results. This holds true unless support vector machines come into play. Though they grant a considerable performance boost, their application in large-scale systems is hard to envisage given their immense resource consumption requirements.[10] This is a crucial counterargument for their usability in the biomedical domain, which requires cheap computations on very large data sets. From an HLT perspective, one should also bear in mind that we deal here with the rather basic preprocessing step of NP chunking, and have not even touched upon subsequent in-depth text processing and mining tasks, which tend to grow in their computational load.

For error analysis, false negative and false positive matches were compared. Although the individual systems' false negative and false positive hits do not directly correspond to their final performance, they do have an effect on it. For YAMCHA, e.g., the high number of false negative hits is leveled out by its very low number of false positive hits.

An analysis of error types showed that coordination turned out as the most common error class. This comes as no surprise, since this linguistic pattern was not only reported to be problematic for NP chunking tasks (Ramshaw & Marcus 95), but also for more expressive higher-level formalisms such a as full-sentence parsing. Another prominent error class also reported in the literature is the recognition of verbal elements inside NPs. A more domain-specific error source came from NP-internal parentheses, which is a feature specific to the biomedical domain. In a follow-up experiment, however, a domain-specific adaption specifically targeted the recognition of noun phrases with such internal parenthesized/bracketed elements. A straightforward heuristic solution for the BOSS system led to a noticeable performance increase.[11]

Although the performance of base NP chunking systems ported to other domains is already important from a linguistic processing view, it gains additional relevance from an application-oriented perspective, *viz.* the crucial task of biological named entity recognition. We showed that almost all biological named entities (99.1%) are contained within base NPs. Thus, recognizing base NPs may be a beneficial linguistic preprocessing step to bio-entity recognition because it considerably reduces the search space. This finding is particularly interesting in light of the fact that most current biological named entity recognizers, in terms of linguistic processing, do not go beyond the POS-tagging stage. (Of course, future work will have to test whether base NP chunking actually boosts the recognition rate of biological NEs.)

In addition, we could also show that all chunking systems though ported from the general-language newspaper domain perform very well in recognizing

[10] Training a support vector machine leads to a quadratic optimization problem with bound constraints and one linear equality constraint. For large learning tasks with many training examples, off-the-shelf optimization techniques for general quadratic programs then quickly become intractable in their space and time requirements. Still, several heuristic solutions addressing this problem have already been developed, e.g., by (Kaufman 99; Joachims 99).

[11] These results are also in line with our previous studies (Wermter & Hahn 04; Hahn & Wermter 04) which examined the portability of POS taggers to the biomedical domain.

the parts of base NPs which contain domain-specific (viz., biological) named entities. Interestingly, the superior performance of the SVM-based YAMCHA chunker on the general linguistic base NP chunking task is not reflected in the recognition of the domain-specific NE part of base NP chunks, in which it is outperformed by the domain-adapted BOSS chunker.

Therefore, our findings are crucial with respect to the fast re-usability of such systems for biomedical text mining applications and tasks, such as named entity recognition, especially in the light of insufficient in-domain (i.e., biomedical) training resources.

Acknowledgements. This work was partly supported by the European Network of Excellence "Semantic Mining in Biomedicine" (NoE 507505).

References

(Brill 95) Eric Brill. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565, 1995.

(Campbell & Johnson 01) David A. Campbell and Stephen B. Johnson. Comparing syntactic complexity in medical and non-medical corpora. In Suzanne Bakken, editor, *AMIA 2001 – Proceedings of the Annual Symposium of the American Medical Informatics Association. A Medical Informatics Odyssey: Visions of the Future and Lessons from the Past*, pages 90–94. Washington, D.C., November 3-7, 2001. Philadelphia, PA: Hanley & Belfus, 2001.

(Church 88) Kenneth W. Church. A stochastic parts program and noun phrase parser for unrestricted text. In *ANLP 1988 – Proceedings of the 2nd Conference on Applied Natural Language Processing*, pages 136–143. Austin, TX, USA, 9-12 February 1988. Association for Computational Linguistics, 1988.

(Finkel *et al.* 04) Jenny Finkel, Shipra Dingare, Huy Nguyen, Malvina Nissim, Christopher Manning, and Gail Sinclair. Exploiting context for biomedical entity recognition: From syntaxt to the Web. In Nigel Collier, Patrick Ruch, and Adeline Nazarenko, editors, *JNLPBA – Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, pages 88–91. Geneva, Switzerland, August 28-29, 2004, 2004.

(Hahn & Wermter 04) Udo Hahn and Joachim Wermter. High-performance tagging on medical texts. In *COLING Geneva 2004 – Proceedings of the 20th International Conference on Computational Linguistics*, volume 2, pages 973–979. Geneva, Switzerland, August 23-27, 2004. Association for Computational Linguistics, 2004.

(Joachims 99) Thorsten Joachims. Making large-scale support vector machine learning practical. In Bernhard Schölkopf, Chris Burges, and Alex J. Smola, editors, *Advances in Kernel Methods. Support Vector Learning*, pages 169–184. Cambridge, MA: MIT Press, 1999.

(Kaufman 99) Linda Kaufman. Solving the quadratic programming problem arising in support vector classification. In Bernhard Schölkopf, Chris Burges, and Alex J. Smola, editors, *Advances in Kernel Methods. Support Vector Learning*, pages 147–167. Cambridge, MA: MIT Press, 1999.

(Kim *et al.* 04) Jin-Dong Kim, Tomoko Ohta, Yoshimasa Tsuruoka, Yuka Tateisi, and Nigel Collier. Introduction to the Bio-Entity Recognition Task at JNLPBA. In Nigel Collier, Patrick Ruch, and Adeline Nazarenko, editors, *JNLPBA – Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, pages 70–75. Geneva, Switzerland, August 28-29, 2004, 2004.

(Kudo & Matsumoto 01) Taku Kudo and Yuji Matsumoto. Chunking with support vector machines. In *NAACL'01, Language Technologies 2001 – Proceedings of the 2nd Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 192–199. Pittsburgh, PA, USA, June 2-7, 2001. San Francisco, CA: Morgan Kaufmann, 2001.

(Litrán *et al.* 04) José Carlos Clemente Litrán, Kenji Satou, and Kentaro Torisawa. Improving the identification of non-anaphoric 'it' using support vector machines. In Nigel Collier, Patrick Ruch, and Adeline Nazarenko, editors, *JNLPBA – Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, pages 58–61. Geneva, Switzerland, August 28-29, 2004, 2004.

(Marcus *et al.* 93) Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The PENN TREEBANK. *Computational Linguistics*, 19(2):313–330, 1993.

(McDonald *et al.* 04) Daniel M. McDonald, Hsinchun Chen, Hua Su, and Byron B. Marshall. Extracting gene pathway relations using a hybrid grammar: The Arizona Relation Parser. *Bioinformatics*, 20(18):3370–3378, 2004.

(Muñoz *et al.* 99) Marcia Muñoz, Vasin Punyakanok, Dan Roth, and Dav Zimak. A learning approach to shallow parsing. In Pascale Fung and Joe Zhou, editors, *EMNLP/VLC'99 – Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 169–178. College Park, MD, USA, June 21-22, 1999. Association for Computational Linguistics, 1999.

(Narayanaswamy *et al.* 03) Meenakshi Narayanaswamy, K.E. Ravikumar, and K. Vijay-Shanker. A biological named entity recognizer. In Russ B. Altman, A. Keith Dunker, Lawrence Hunter, Tiffany A. Jung, and Teri E. Klein, editors, *PSB 2003 – Proceedings of the Pacific Symposium on Biocomputing 2003*, pages 427–438. Kauai, Hawaii, USA, January 3-7, 2003. Singapore: World Scientific Publishing, 2003.

(Ohta *et al.* 02) Tomoko Ohta, Yuka Tateisi, and Jin-Dong Kim. The GENIA corpus: An annotated research abstract corpus in molecular biology domain. In M. Marcus, editor, *HLT 2002 – Human Language Technology Conference. Proceedings of the 2nd International Conference on Human Language Technology Research*, pages 82–86. San Diego, Cal., USA, March 24-27, 2002. San Francisco, CA: Morgan Kaufmann, 2002.

(Park *et al.* 04) Kyung-Mi Park, Seon-Ho Kim, , Ki-Joong Lee, Do-Gil Lee, and Hae-Chang Rim. Incorporating lexical knowledge into biomedical NE recognition. In Nigel Collier, Patrick Ruch, and Adeline Nazarenko, editors, *JNLPBA – Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, pages 76–79. Geneva, Switzerland, August 28-29, 2004, 2004.

(Pustejovsky *et al.* 02) James Pustejovsky, José Castaño, Jason Zhang, Maciej Kotecki, and Brent Cochran. Robust relational parsing over biomedical literature: Extracting inhibit relations. In Russ B. Altman, A. Keith Dunker, Lawrence Hunter, Kevin Lauderdale, and Teri E. Klein, editors, *PSB 2002 – Proceedings of the Pacific Symposium on Biocomputing 2002*, pages 362–373. Kauai, Hawaii, USA, January 3-7, 2002. Singapore: World Scientific Publishing, 2002.

(Ramshaw & Marcus 95) Lance Ramshaw and Mitchell P. Marcus. Text chunking using transformation-based learning. In *Proceedings of the 3rd ACL Workshop on Very Large Corpora*, pages 82–94. Cambridge, MA, USA, June 30, 1995. Association for Computational Linguistics, 1995.

(Sang & Buchholz 00) Erik F. Tjong Kim Sang and Sabine Buchholz. Introduction to the CoNLL-2000 shared task: Chunking. In Claire Cardie, Walter Daelemans, Claire Nédellec, and Erik F. Tjong Kim Sang, editors, *Proceedings of the 4th Conference on Computational Language Learning (CoNLL-2000) and the 2nd Learning Language in Logic Workshop (LLL-2000)*, pages 127–132. Lisbon, Portugal, 13-14 September 2000. Association for Computational Linguistics, 2000.

(Saric *et al.* 04) Jasmin Saric, Lars J. Jensen, Rossitza Ouzounova, Isabel Rojas, and Peer Bork. Extracting regulatory gene expressions expression networks from PubMed. In *ACL'04 – Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*. Barcelona, Spain, July 21-26, 2004. San Francisco, CA: Morgan Kaufmann, 2004.

(Tateisi & Tsujii 04) Yuka Tateisi and Jun'ichi Tsujii. Part-of-speech annotation of biology research abstracts. In *LREC 2004 – Proceedings of the 4th International Conference on Language Resources and Evaluation. In Memory of Antonio Zampolli. Vol. 4*, pages 1267–1270. Lisbon, Portugal, 26-28 May 2004. Paris: European Language Resources Association (ELRA), 2004.

(Vapnik 98) Vladimir N. Vapnik. *Statistical Learning Theory*. New York: Wiley, 1998.

(Wermter & Hahn 04) Joachim Wermter and Udo Hahn. Really, is medical sublanguage that different? Experimental counter-evidence from tagging medical and newspaper corpora. In Marius Fieschi, Enrico Coiera, and Yu-Chan Jack Li, editors, *MEDINFO 2004 – Proceedings of the 11th World Congress on Medical Informatics. Vol. 1*, number 107 in Studies in Health Technology and Informatics, pages 560–564. San Francisco, CA, USA, September 7-11, 2004. Amsterdam: IOS Press, 2004.

(Yakushiji *et al.* 01) Akane Yakushiji, Yuka Tateisi, Yusuke Miyao, and Jun'ichi Tsujii. Event extraction from biomedical papers using a full parser. In Russ B. Altman, A. Keith Dunker, Lawrence Hunter, Kevin Lauderdale, and Teri E. Klein, editors, *PSB 2001 – Proceedings of the 6th Pacific Symposium on Biocomputing*, pages 408–419. Maui, Hawaii, USA. January 3-7, 2001. Singapore: World Scientific Publishing, 2001.

Knowledge based Feature Engineering using Text Sense Representation Trees

Ronald Winnemöller
Regional Computer Centre, University of Hamburg
20146 Hamburg, Germany
ronald.winnemoeller@rrz.uni-hamburg.de

Abstract

In this paper, we present a novel knowledge based approach to feature engineering. In our case, features are not just entities of a "bag of words" but rather predefined nodes of a large directory-like structure.

These features are weighted according to our notion of "text sense", the number of occurrences within the directory. The gathered weights can be used straightforwardly for feature selection, thus reducing text model size while maintaining a high degree of performance.

Therefore, through application of a controlled feature set we are able to implement an efficient, task independent and computationally inexpensive strategy for feature management We will also introduce further means of feature engineering and present our evaluation results based on text classification experiments.

1 Introduction

Modern text based systems that employ sophisticated techniques, e.g. modern content management systems and university e-learning environments usually require some understanding of the state of affairs of the environment that is targeted by the system, cf. (Allen 95; Cole *et al.* 95).

One important aspect of text understanding is concerned with knowledge about the context in which particular words, phrases, sentences and whole texts are used. For example, the word *bank* can be used in a financial context (*bank loan*) and a geographical context (*sand bank*). Lexico-syntactical analysis alone will not suffice to distinguish between these meanings. Dictionary based approaches on the other hand might leave out important aspects as pointed out 1997 by Kilgarriff (Kilgarriff 97).

In this paper, we will show that the TSR Tree methodology introduced 2004 by Winnemoeller (Winnemöller 04) is well suited to represent text senses in a highly controlled scheme. We use Text Classification[1] as evaluation scheme of text sense representation appropriateness because – as proposed by Schütze and Pedersen (Schutze & Pedersen 95) – classification performance can be seen as indicator for the effectiveness of a sense or meaning representation.

The contributions of this paper include our presentation of how TSR trees are suited for solving feature selection issues in a text classification context[2].

After a short introduction to the concept of representing text sense by TSR Trees, we will discuss related scientific work and then present our own experimental findings. Finally, we will summarize our conclusions and propose future work.

2 Text Sense Representation Trees

Within this paper, we define the notion "*word sense*" as "the ways a word is used in its context" (Frege, cf. (Frege 92)[3] and S. G. Pulman (Cohen 96, section 3.5)) - "*context*" being the surrounding text. We will further define "*Text sense*" by extending the word sense definition as "the ways a text is used in context".

Text Sense Representation Trees provide a methodology to represent text senses in a uniform and generally applicable way while being constructed in a fully automatic fashion prior to their use. The basic underlying data structure, the TSR Tree, consists of a tree hierarchy of labeled and weighted nodes. The TSR trees are constructed from a web directory such as the Open Directory Project (Netscape Inc. 04)[4].

[1] In the context of this paper, we will define Text Classification informally as the task of assigning text documents to a set of predefined class labels through a process of textual analysis.

[2] For more in-depth explanations of the TSR Tree construction and analysis process please consult the introductory article 2004 by Winnemöller (Winnemöller 04)

[3] Frege's definition for sense is rather a truth function of text meaning. In this sense, the use of "music" in "the music is green" is senseful whereas "the frog is green" is correct.

[4] Using our Java servlet based prototype to construct the complete TSR Tree database produces approx. 600 MB data within approx. 4 hours on a Celeron 1.4 MHz machine with 256 MB RAM.

The construction procedure can be described briefly and informally as follows:

1. **Web directory data acquisition** In our case, we retrieved the publicly available ODP RDF-like data dumps, but in principle other hierarchical data sources like Yahoo (Yahoo Inc. 04) or Internet newsgroups can be used as data source as well.

2. **Sense path construction** Every text node of the web directory is analyzed into distinctive terms (usually stemmed words). Each term is then associated with the full entry path. For example, if a term "account" is found in the directory node "/top/business/finance/banking", then a respective path instance consisting of the weighted nodes *top(1), business(1), finance(1), banking(1)* is created. There is no $1:1$ relationship of terms to path instances; instead each path represents an atomic aspect to a particular words sense.

3. **TSR Tree construction from sense paths** All paths associated with a term are merged into a single TSR tree structure: distinct path nodes are appended to the tree, equally labeled nodes increase the respective tree node weight by one. For example, the TSR tree in Figure 1 contains 20 path instances "/business/credits/cashflow".

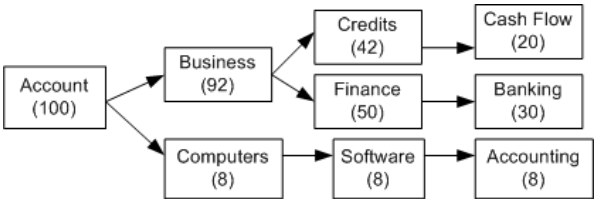


Figure 1: TSR Tree excerpt for the word "account" (numbers are node weights)

The TSR Tree approach includes a number of operations on these data structures in order to provide methods for comparing, merging and manipulating trees:

a) TSR Tree comparison The comparison operations are important to achieve an understanding of "semantical relatedness" of two text fragments[5]. For example, a high relatedness value is found between closely related terms such as "account" and "cash" whereas a low value is found between largely unrelated terms such as "air" and "bezier curve". This behaviour of TSR trees is well suited for tasks that involve "soft" word occurrences – for example, when designing a "fuzzy" text search functionality.

b) TSR Tree merge The merging operation is vital for transition from *word* sense to (arbitrary) *text* sense analysis. By creating the union of a set of TSR trees, it is possible to obtain a TSR tree that describes the set itself rather than any of its individual constituents. For example one may merge the words of a sentence or paragraph to obtain a TSR tree that represents the respective sentence or paragraph.

c) TSR Tree merge A certain kind of "feedback" learning can be used for deriving sense representations for unknown words or to adapt senses to specific texts.

The overall size of the feature set, i.e. the maximum possible number of attribute nodes, is limited in any case by the size of the source ODP directory, but the number of utilized nodes can be reduced substantially: Feature selection is implemented straightforwardly through selecting the n top weighted attribute nodes of (flattened) TSR Trees. This drastically reduces the amount of features while maintaining a high degree of expressiveness.

Another important way of feature selection is the elimination of inexpressive nodes, i.e. those nodes that do not carry distinctive information between individual text classes.

3 Related Work

The current flow of scientific progress related to text sense representation can be separated into *intensionally* and *extensionally* oriented approaches. These notions were borrowed from analytical philosophy and are both explained as follows, even though the TSR tree approach is extensionally oriented.

3.1 Intensional Approaches

In this paper we will define the category of "intensional oriented (text sense representation) approaches" as a set of methodologies that use an intensional definition for words or concepts: an intensional definition includes the *set of the necessary and sufficient properties of the respective word, concept or term in question.* Intensional approaches usually require a high amount of manual work constructing term definition data structures; in turn, they enable automatic reasoning and data oriented computing.

[5]This usually is a single floating point value denoting the "relatedness" between two terms

Prominent examples for intensional oriented text sense representation methodologies include ontologies as defined by Gruber (cf. (Gruber 93)), frame-based techniques, semantic networks and deductive databases such as Cyc (cf. (Guha *et al.* 90)).

3.2 Extensional Approaches

Extensional text sense representation approaches provide the meaning of a term by enumerating a large corpus of textual context of the term in question. Extensional approaches are often highly automaticable and fairly universal where no deductive capabilities are required. In most cases no deep syntactic analysis is necessary[6].

By far the most common and well known representation methods are based on Luhn's term frequency (TF) approach, sometimes elaborated as term frequency - inverse document frequency (TF.IDF), cf. 1975 Salton and Yu (Salton & Yu 75). In a large amount of related literature TF and TF.IDF based techniques form the basis of text classification, cf. for example (McCallum *et al.* 98; Nigam 01) etc.

Other concepts are based on "word vectors", e.g. by Schütze (Schutze & Pedersen 95) and by Yarowsky (Yarowsky 92), respectively. Word vectors are similar to TF or TF.IDF structures but include "concept terms" instead of mere word terms. These concept terms are generated using an external knowledge source, e.g. Roget's Thesaurus or the WordNet database, cf. (Miller *et al.* 05) and (Priss 98). Concept terms represent text meaning rather than statistical representations between single terms (such as TF.IDF). Even though Schütze and Yarowsky report word vectors to be more effective than TF.IDF and comparable methods in terms of accuracy, they also show a number of disadvantages: small differences in meaning cannot be captured appropriately and the quality of thesaurus definitions is essential.

The main difference between TSR and TF.IDF oriented strategies is that in the case of TSR based text processing there is a controlled hierarchical feature set while in the TF.IDF cases there is an open, uncontrolled feature set[7].

[6]sometimes a stemming algorithm is used to reduce complexity, e.g. the Porter stemmer, cf. (Porter 80)

[7]Please note that a controlled feature set does not imply the usual restrictions of using a controlled language environment!

Another closely related approach is presented by Santamaria et al. (cf. (Santamaria *et al.* 03)): they provided a – on the first sight – similar methodology of obtaining sense information by harvesting ODP path and textual information. Santamaria et al. associate WordNet synsets to "ODP directories" (in this paper called "ODP paths") thereby defining a single ODP path as sense representation in respect to the according WordNet data. The TSR tree approach instead uses path instances as components of actual sense representations which makes it independent of lexicographic issues such as stated by Kilgarriff, cf. (Kilgarriff 97), for example the problem of adequate sense definition. Another important difference is that Santamaria et al. stop at creating *word* base sense analysis whereas TSR trees can be created (and used uniformly) in respect to *arbitrary text* fragments.

Mohit and Narayanan presented 2003 an approach to "Semantic Extraction", cf. (Mohit & Narayanan 03), a particular kind of particular on Extraction in which they used FrameNet, cf. (Baker *et al.* 98) and WordNet data to build semantic representations on a restricted feature set.

4 Experiments

The main focus of our experiments was not to show outstanding text classification results in general but rather to see how the TSR tree approach would work out in a text classification context. Thus, the actual accuracy values in section 4.3 should not be read in an "absolute" fashion but rather in relation to the size of feature and training sets.

4.1 Setup

We use the WEKA Machine Leaning Toolkit by Witten and Frank (cf. (Witten & Frank 00)), which is a comprehensible Java tool set for design and execution of standardized machine learning experiments.

We also used our own primitive classifier which computes a "relatedness" value for each pair of classifier TSR Tree and message[8] text tree. Results are then grouped by the respectively highest rated classifier and compared to the actual classes. The advantage of this classification scheme is that it is very close to the TSR tree principle and does

[8]A "message" is a distinct text document instance in our test corpora

not introduce external functionality.

4.2 Data Sets

We compiled three different datasets for experimenting:

1. The (nonstandard) "Small" corpus contains about 150 texts within 5 categories: we chose 3 newsgroups, a selection of the Java documentation files and excerpts from the King James Bible. This corpus contains a relatively small amount of off-topic texts.

2. The "6newsgroups" corpus is a subset of the widely accepted *20newsgroups* corpus (e.g. cf. (McCallum *et al.* 98)) and consists of about 800 messages posted to 6 different newsgroups. This corpus contains a certain noise ratio due to the nature of its data source[9].

3. the "5abstracts" corpus was compiled by us from approximately 1500 abstracts of 5 different scientific domains.This corpus is relatively free of noise.

4.3 Results

For evaluating TSR performance within the WEKA tool set, we applied a 4-fold cross validation procedure using the J48 learning operator (a Java version of the C4.5 decision tree learner)[10].

Dataset	Type	FS	# Feat.	Time	Acc.
small	TSR20	64 K	123	0.38	92 %
small	TSR40	136 K	278	0.22	89 %
small	TSR	1,4 M	3315	2.73	86 %
small	TEXT	1,1 M	3407	2.78	86 %
6ng	TSR20	659 K	352	5.08	64 %
6ng	TSR40	1,5 M	849	16.2	62 %
6ng	TSR	18,9 M	10842	275	64 %
6ng	TEXT	21,9 M	14041	326	72 %
5ab	TSR20	1,4 M	388	8.42	85 %
5ab	TSR40	3,8 M	1089	25.5	86 %
5ab	TSR	36,5 M	10427	213	92 %
5ab	TEXT	45,9 M	14847	273	96 %

Table 1: 4-fold cross validation comparison of TSR and word vector based classification using different degrees of feature selection by node pruning

Table 1 presents our results of evaluating all three corpora using word vectors and TSRs respectively, with a listing of the effects of feature selection on the latter result sets. The table is to be read as follows (For brevity, we will not describe the obvious columns here): 6ng stands for "6newsgroups", 5ab for "5abstracts". The "Type" column denotes the type of test: TSR20 means: TSR testing, selecting only the 20 top weighted features of each message, TEXT means: applying word vector testing. The time is measured in seconds. "Acc." abbreviates "accuracy" and "FS" abbreviates "File Size".

In 2 out of 3 times, the (mature) word vector[11] ML approach results in better accuracy than the TSR tree approach but TSR trees achieve roughly comparable results with much smaller feature sets. This makes them more effective in real world applications.

Classification accuracy does not increase through feature selection in 2 out of 3 cases. These findings are related to the nature of the corpuses: an increase of accuracy exists in the minimum-noise / highest information density corpus. Because the other corpora contain a higher amount of noise, a reduction of feature set size also includes the noise-related features which makes the process less effective.

In general, the best accuracy values for the word vector approach and the TSR Tree approach are approximately equal.

Fold ratio	WV acc.	TSR Tree acc.
2	60,85	68,51
3	66,75	68,64
4	74,58	69,13
10	74,07	71,11

Table 2: k-fold cross validation results for the *6newsgroups* corpus [13]

Table 2 shows that word vector based learning performance[14] increases by the size of the training set for low values of folds while TSR Tree based learning performance remains stable (i.e. is approximately constant). This indicates that the TSR tree methodology is – to a certain degree – largely independent of the actual training set size. Our interpretation is that TSR trees only use the *sense–related* information and not use statistical co-occurrence data (which makes TSR trees a "knowledge based" approach). This interpre-

[9] In this context, *noise* is the amount of "spam" and off-topic mails usually found within any public newsgroup

[10] Comparative tests have shown that many other learners – even SVMs – behave similarly to J48 in terms of relative accuracy measures, we therefore skipped thorough experimenting using other operators

[11] From now on we will use this term equivalently to "term frequency vector"

[14] Applied on the 6newsgroups corpus here, for brevity we will skip the results for the other datasets because the outcome is similar

tation is supported by preliminary experiments where we used only 5-10 manually chosen keywords per message for text classification which resulted in a performance loss of approximately 15%.

5 Conclusions

The results of section 4 underpins our claim that – apart from other advantages such as "fuzzy" text analysis – the TSR tree methodology can be used effectively for text classification problems while maintaining a certain degree of control over the classification feature set. Specifically,

- TSR based feature selection techniques reduce computing time and space requirements by model size reduction while still maintaining a high performance level.
- The maximum number of features is controlled by the underlying web directory data structure size.
- The performance of TSR Trees is mainly based on semantic knowledge, i.e. the amount of global and local context information. This makes the TSR Tree approach largely independent from the need of a large manually prepared training corpus.
- The effectiveness of TSR Trees in terms of classification accuracy is – as shown – approximately equal to the effectiveness of comparable approaches.
- Feature selection leads to more efficient TSR tree base text processing but – in certain cases – may not increase effectiveness as well.

In summary, the TSR Tree approach provides an effective and stable methodology for feature engineering through appropriate text sense representation, especially in situations with very little (local) knowledge available.

Future work will be based on the evaluation of different "relatedness" measures and on the introduction of TSR tree learning capabilities to our experiments and on application of TSR trees on word sense disambiguation in order to use the standard evaluation procedures of Senseval, cf. (Kilgarriff 98).

References

(Allen 95) James Allen. *Natural Language Understanding*. Benjaming/Cummings Publish. Corp, CA, 2 edition, 1995.

(Baker *et al.* 98) Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The berkeley framenet project. In *Proceedings of the 17th international conference on Computational linguistics*, pages 86–90, Morristown, NJ, USA, 1998. Association for Computational Linguistics.

(Cohen 96) William W. Cohen. Learning rules that classify e-mail. In *Proceedings of the 1996 AAAI Spring Symposium on Machine Learning in Information Access*, 1996.

(Cole *et al.* 95) R. Cole, J. Mariani, H. Uszkoreit, A. Zaenen, and V. Zue. Survey of the state of the art in human language technology. http://www.coli.uni-sb.de/ hansu/publ.html, 1995.

(Frege 92) Gotthold W. Frege. über sinn und bedeutung. *Zeitschrift für Philosophie und philosophische Kritik*, pages 25–50, 1892. translated "On sense and meaning.", Reprinted in McGuinness, Brian (ed.) Collected Papers on Mathematics, Logic and Philosophy, 157–177, Oxford: Basil Blackwell (1984).

(Gruber 93) T. R. Gruber. A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2):199–220, 1993.

(Guha *et al.* 90) R. V. Guha, D. B. Lenat, K. Pittman, D. Pratt, and M. Shepherd. Cyc: A midterm report. *Communications of the ACM*, 33(8):30–49, 1990.

(Kilgarriff 97) Adam Kilgarriff. I don't believe in word senses. *Computers and the Humanities*, 31 (2):91–113, 1997.

(Kilgarriff 98) Adam Kilgarriff. SENSEVAL: An exercise in evaluating word sense disambiguation programs. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, pages 581–588, Granada, Spain, 1998.

(McCallum *et al.* 98) Andrew McCallum, Ronald Rosenfeld, Tom M. Mitchell, and Andrew Y. Ng. Improving text classification by shrinkage in a hierarchy of classes. In *ICML*, pages 359–367, 1998.

(Miller *et al.* 05) George A. Miller, Christiane Fellbaum, Randee Tengi, Susanne Wolff, Pamela Wakefield, Helen Langone, and Benjamin Haskell. Wordnet - a lexical database for the english language. http://www.cogsci.princeton.edu/ wn/index.shtml, Jan 2005.

(Mohit & Narayanan 03) Behrang Mohit and Srini Narayanan. Semantic extraction with wide-coverage lexical resources. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 64–66, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

(Netscape Inc. 04) Netscape Inc. Open directory project. http://dmoz.org, 2004.

(Nigam 01) Kamal Nigam. *Using Unlabeled Data to Improve Text Classification*. Unpublished PhD thesis, Computer Science Department, Carnegie Mellon University, Pittsburgh, US, 2001.

(Porter 80) Martin F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

(Priss 98) Uta Priss. *WordNet: An Electronic Lexical Database and Some of its Applications*, chapter The Formalization of WordNet by Methods of Relational Concept Analysis, pages 179–196. MIT press, 1998.

(Salton & Yu 75) G. Salton and C. T. Yu. Effective information retrieval using term accuracy. Technical Report 75-249, Department of Computer Science, Cornell University, Ithaca, New York, Jul 1975.

(Santamaria *et al.* 03) C. Santamaria, J. Gonzalo, and M. F. Verdejo. Automatic association of web directories to word senses. *Computational Linguistics*, 29(3), 2003.

(Schutze & Pedersen 95) H. Schutze and J. Pedersen. Information retrieval based on word senses. In *Proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, Las Vegas, NV, 1995.

(Winnemöller 04) Ronald Winnemöller. Constructing text sense representations. In *Proceedings of the 2nd Workshop on Text Meaning and Interpretation*, Barcelona, Spain, Jul 2004. Association for Computational Linguistics.

(Witten & Frank 00) Ian H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools with Java implementations*. Morgan Kaufmann, San Francisco, 2000.

(Yahoo Inc. 04) Yahoo Inc. Yahoo. http://www.yahoo.com, 01 2004.

(Yarowsky 92) David Yarowsky. Word-sense disambiguation using statistical models of Roget's categories trained on large corpora. In *Proceedings of COLING-92*, pages 454–460, Nantes, France, July 1992.

Handling Corrupt Input in a Domain-specific Spoken Dialogue System*

Wei-Lin Wu, Ru-Zhan Lu, Feng Gao and **Hui Liu**
Department of Computer Science and Engineering
Shanghai Jiao Tong University
200030, China
{wu-wl,lu-rz,gaofeng,liuhui}@cs.sjtu.edu.cn

Abstract

In spoken dialogue systems, the input errors inevitably occur. This paper proposes a mechanism for input error handling, and evaluates this mechanism in the context of public transportation information inquiring domain. This mechanism supports error handling at different levels instead of only regarding error handling as a post-processor for speech recognition. Firstly, error detection and correction is performed in the language analysis component. We combine error similarity-based correction approach and rule-based error correction approach to account for different kinds of errors. Then, the error is directly corrected or provided with the correction hypotheses by the error correction module. Finally, the system confirms these hypotheses to the user at dialogue level. The preliminary evaluation in the context of public transportation information inquiring domain showed that the use of the proposed error handling mechanism could significantly improve the understanding of the queries. We also found that certain dialogues in our system improved effectively because of the use of this error handling mechanism.

1 Introduction

In spoken dialogue systems, different kinds of input errors inevitably occur, such as recognition errors. These errors will make more difficult the communication between the system and the user. It is often said that the major problem in spoken dialogue systems is their inability to detect and correctly handle different types of errors. Therefore, error handling in spoken dialogue applications is crucial for successful interaction (Turunen & Hakulinen 01).

There is considerable research on the error handling in spoken dialogue systems. One typical approach is to view error handling as a post-processor for speech recognition. This kind of methods employs the noisy channel model (Ringger & Allen 96) for speech error correction, or apply the patterns to correct speech errors, which are extracted utilizing statistical information of word co-occurrence (Kaki *et al.* 98) or semantic information obtained from some knowledge bases (Jeong *et al.* 03). This kind of approach makes use of only shallow level knowledge (word co-occurrence or lexical semantic information). Also, it has another negative aspect: correction often set off a cascade of additional errors, which then need to be corrected (Halverson *et al.* 99). In addition, the performances of some of such methods rely heavily on the size of corpus or the set of erroneous sentences relevant to the application. Another direction is the interactive error correction, which views error handling as a part of the dialogue flow (Turunen & Hakulinen 01). This type of approach usually uses context sensitive feedback messages to facilitate the error detection and involves the user in interactive dialogues to recover from errors (Suhm *et al.* 96). The typical interactive error correction strategies include requiring the user to repeat reparandum by respeaking or spelling out loudly, to paraphrase the reparandum, or to choose from a list of alternative words (Suhm *et al.* 96; Ainsworth & Pratt 92; Gorrell 03).

This paper proposes an error handling mechanism trying to combine two strategies above. This mechanism supports error handling at different levels instead of regarding error handling as a post-processor for speech recognition. Firstly, we make use of as much knowledge sources as possible for error detection, such as a domain-specific dictionary, information offered by the language analysis component, and the user correction for system's error (e.g. the user answers "no, ..."). Then, we combine similarity-based error correction approach and rule-based error correction approach to account for different kinds of errors: directly correct the errors or provide the correction hypotheses for them. Finally, the system confirms these hypotheses to the user at dialogue level.

2 The Domain and Input Error Types

2.1 The Public Transportation Information Inquiry Domain

Our works are related to a telephone-based spoken dialogue system for automatic public transportation information inquiries (Mao *et al.* 03; Wu *et al.* 03). The user conducts a mixed-initiative dialogue with the system via telephone, getting access to public transportation information in Shanghai city. Currently, the specific domain covers information on the routes between two locations in Shanghai city by different means of transportation (bus, taxi or bicycle). Since the task of spoken language understanding in our system is to

*This work is supported by National Natural Science Foundation of China (NSFC) (No.60496326) and 863 project of China (No.2001AA114210-11).

extract the information needed to complete the query (Mao *et al.* 03; Wu *et al.* 03), our work concentrates on the repair of key information in the input sentence.

2.2 Input Error Types and Their Characteristics

2.2.1 Input Error Sources

1. Speech recognition errors

   The speech recognition errors include: substituting, inserting or deleting a word in the user utterance by the recognizer, and in addition, partial multiple errors. It should be pointed out that Chinese speech recognition faces more challenges due to its unique characteristics, such as homonyms and tonality problems. In particular, the two factors turn more serious in our domain since there is a large set of entity names (location names, street names and so on), among which many pairs of homonyms occur.

2. User's cognitive errors

   The user's knowledge and language capability restriction also lead to input errors. The user's familiarity with the entity names in our domain and her/his proficiency in pronunciations of Chinese characters combine to have direct influence on the input error rate. For instance, when the user intend to say 番禺路 (a street name), the user may speak "pan yu lu" as "fan yu lu", where 番 has two pinyins[1] "fan" and "pan". At the same time, many entity names in our domain have several aliases and abbreviations. With the principle of least effort in conversation (Clark 96), the user will try to minimize their collaborative effort for achieving their goals and thus prefer to use the abbreviations of entity names.

2.2.2 Non-word Error V.S. Real-word Error

The errors can also be categorized as non-word and real-word errors based on whether the corrupt string is a word or not. This kind of classification contributes a lot in choosing the strategies for the error detection and correction. A direct principle can be applied to distinguish a non-word error with a real-word one: it is a non-word error if the corrupt string is not in the domain-specific dictionary, otherwise a real-word error.

2.2.3 Non-content Error V.S. Content Error

Considering the side-effect of the errors on the understandability of the input sentence, we classify errors into non-content errors and content errors. An error is usually regarded as a content error if it distorts the key information intended by the user, otherwise non-content error. For example, there is little change on the meaning of the input sentence if the Chinese character 请(please) is missed. In contrast, it will drastically decrease the understandability of the input sentence if the word 从(from) is misrecognized as 车(vehicle).

[1]The pinyin is a system of romanization for standard Mandarin, for simplicity, which can be understood as the pronunciation of the Chinese character.

3 Methods for Errors Detection and Correction

The amount of data available in the specific domain is usually limited and then does not license the solution based on the statistical model. Therefore, according to the characteristics of non-word and real-word errors, we employ similarity-based and rule-based error correction methods for the two sorts of errors respectively. The procedure of error correction is interactive with the process of language analysis, such as segmentation, tagging and semantic parsing. Hence we can make use of the information offered by the language analysis.

3.1 Non-word Error Detection and Correction

It is straightforward to detect the non-word errors since the non-word error words will be tagged UNKNOWN (this semantic tag stands for the out-of-vocabulary words) after the input sentence is preprocessed. The continuous non-word errors are clustered into a single error.

The correction strategy for the non-word errors is based on the extended idea of edit distance. With the assumption that the dictionary is complete, when a non-word error is detected, we will try to find the similar legitimate word in the domain-specific dictionary.

The key point is to appropriately define the distance measure between the corrupt string and the corresponding correct string. Here, we extend the notion of MED (minimum edit distance) for Chinese strings so that it not only considers the similarity of the graphemes of two Chinese strings but also reflect how similar their pronunciations are. Assume a Chinese string X_{1m} (i.e. $X_1 \dots X_m$, where X_i is a Chinese character) can be transformed to another Chinese string Y_{1n} through T_i insertions, T_d deletions and T_t transpositions of Chinese characters, we can define the corresponding distance as follows:

$$Dist(T_i, T_d, T_t) = (T_i + T_d + \sum_{j=1}^{T_t} C_t(j))/n \quad (1)$$

In Equation (1), $C_t(j)$ is used to measure the cost of the jth transposition from a Chinese character in X_{1m} to the corresponding one in Y_{1n}. Assume the pinyins of X_j and Y_j are PX_j and PY_j respectively, we can compute $C_t(j)$ as follows: $C_t(j) = MED(PX_j, PY_j)/Len(PX_j)$, where $Len(PX_j)$ is the length of PX_j. Intuitively, $C_t(j)$ represents the similarity between the pronunciations of two Chinese characters involved in the jth transposition. Finally, we

can define the extended MED for Chinese strings as follows:

$$MED_{ch}(X_{1m}, Y_{1n}) = \min_{(T_i, T_d, T_t)} Dist(T_i, T_d, T_t) \quad (2)$$

3.2 Real-word Error Detection and Correction

The real-word errors will lead to the syntactic, semantic or pragmatic inconsistency in the input sentence. These kinds of inconsistency are clues for real-word error detection and correction. Based on the literature of recognition error correction (Kaki *et al.* 98) and the examination on the data collection in our domain, we know that real-word errors occur in regular patterns rather than at random. Therefore, the rule-based approach is applied to deal with the real-word errors. Due to the flexibility of spoken language, the syntactic restriction should be relaxed. Hence, we mainly make use of the semantic and pragmatic restrictions.

The semantic rules for error correction capture the semantic dependency among words. In Example 1, the real-word error 从→ 乘 is caused by a character substitution. This error will result in a semantic contradiction: the semantic class of 乘 is $by (it represents a group of words which mean 'take' and expects the next word to be a vehicle name), which is incompatible with that of 人民广场, $loc (it stands for the location name). This error can be corrected by applying a semantic rule associated with the word 乘 as follows: ^(semcls.$by, lex.乘)+(semcls.$loc) ⇒ (^.change.lex.从, ^.change.semcls.$from). This rule is explained as follows: 乘 is corrected as 从 and its semantic class is changed to $from if the current word is 乘 and the next word is of the semantic class $loc.

请问**乘**人民广场到外滩步行怎么走？ *
请问**从**人民广场到外滩步行怎么走？
How can I walk from the people's square to the Bund?

Example 1: An example sentence with a lexical semantic error[2]

Unfortunately, there are still some real-word errors, which can hardly be corrected directly by rules. In Example 2, there are two location names 徐家汇 and 外滩 which can be explained as the source point. It obviously violates the pragmatic principle. This kind of pragmatic real-word errors can hardly been corrected directly since we can not determine the first 从 (from) or the second 从 is misrecognized. Moreover, if the location name 桂林路 is misrecognized as another location name 桂平路, it is almost impossible to directly detect this error. Therefore, this sort of errors should be handled through the further interaction between the user and the system, which will be discussed in details in the latter section.

步行**从** 徐家汇**从**外滩怎么走？ *
How can I walk **from** Xujiahui **from** the bund?*

Example 2: A pragmatic real-word error which can't be directly corrected by rules.

3.3 Combination of Error Correction and Dialogue Strategy

As stated in the Introduction, correction often set off a cascade of additional errors, which then need to be corrected (Suhm *et al.* 96). Our experiments conducted within the context of transportation information enquiry also showed that it had some side-effects as follows due to the use of error correction (especially the similarity-based correction method):

- The error correction module may misrecognize some out-of-vocabulary words or phrases as recognized errors.
- The error correction module may correct the corrupt string using an irrelevant dictionary entry.

These side-effects will make strange the behavior of the system and take a risk of confusing the user.

We add a filter to avoid the first problem. Since the MED between the out-of-vocabulary words or phrases and the corresponding most similar dictionary entry is often relatively large, we set a threshold of MED to filter those "bogus" errors.

As for the second problem, we combine the error correction with the confirm mechanism to address it. Firstly, the error correction module provides the n-best corrected results for the corrupt text rather than to choose the best one as the final corrected result. Then, the system prompts the user that there may exist some errors in her/his prior query and confirms these candidates to the user. The system can prompt the user with all candidates in one turn and require the user to choose the corrected one from them if the correct one is among the candidates, otherwise speak a new one. Alternatively, the system can also confirms the candidates to the user one by one (i.e. only one candidate is prompted in one turn) in order of from the most possible to the least possible. During the confirmation process, the system needs to keep track of which candidate is denied by the user. This one-by-one confirmation mode is effective in avoiding the recurrence of the recognition errors.

This mechanism of combining the error correction with the confirmation is also capable of handling the real-word errors discussed in Section 3.2. In Dialogue 1, the location name 桂林路("gui lin lu") is misrecognized as 桂平路("gui ping lu"). As can be seen in S3

[2]The first sentence with the asterisk is a corrupt input. The other two sentences stand for the correct Chinese sentence and the corresponding English sentence respectively. The bold font is used to mark the erroneous words in the corrupt sentences and the correct counterpart in the actual sentences.

in Dialogue 1, this real-word error of the speech recognizer recurs (i.e. the speech recognizer misrecognized 桂林路 as 桂平路 again). In our mechanism, the system searches for n-best corrected candidates for the 桂平路 through computing the similarity measure after it has processed the denial of the user in U2. The candidates are ranked in order of from the most possible to the least possible. Then the system confirms the candidate to the user one by one. This is illustrated in Dialogue 2.

U1: "我要去桂林路。"("I want to go to Gui Lin Lu.")
S1: "您要去桂平路，对不对？"("You want to go to Gui Ping Lu?")
U2: "不对。"("No.")
S2: "请问您要到哪里去？" ("Please tell me where will you go?")
U3: "我要去桂林路。" ("I want to go to Gui Lin Lu.")
S3: "您要去桂平路，对不对？" ("You want to go to Gui Ping Lu?")

Dialogue 1: Recurrence of a real-word error of speech recognizer in the system without error handling.

U1: "我要去桂林路。" ("I want to go to Gui Lin Lu.")
S1: "您要去桂平路，对不对？"("You want to go to Gui Ping Lu?")
U2: "不对。"("No.")
S2: "您要去桂林路，对不对？"("You want to go to Gui Lin Lu?")
The system guesses what the user said is 桂林路*(Gui Lin Lu) and confirms it to the user.*
U3: "是的。"("Yes.")

Dialogue 2: Avoiding recurrence of a real-word error through combination of the error correction and the one-by-one confirmation.

4 The Preliminary Evaluation

4.1 Data Collection and Evaluating Method

We collected 122 recognized Chinese queries. The distribution of all kinds of errors described in Section 2 is shown in the Table 1.

	Semantic Error Num.	%
Total queries	122	100.0
Queries with errors	75	61.5
Queries with non-word errors	68	55.7
Queries with real-word errors	7	5.7
Queries with non-content errors	12	10.7
Queries with content errors	62	50.8

Table 1: The distribution of all kinds of errors on the test data set

The effect of error correction on the understandability of the queries can be evaluated using semantic error rate. The semantic representation is formalized as a frame with an internal structure consisting of slot/value pairs. A strict definition of semantic error is given as follows: an error occurs when there is at least one difference between the hypothesized frame and the reference frame. Since the error correction provides possible candidates for the key information, it is no longer considered as an error if the reference slot is among the candidate set offered by the error correction module. Another effect of error handling on the system we want to evaluate is the contribution of the error correction to improve the dialogues. This evaluation can be hardly quantitative. We will illustrate it by giving the example dialogues.

4.2 Experimental Results

Firstly, we carried out the understanding experiments on the data set. Table 2 reports the comparative semantic recognition rate of the two systems without error handling and with error handling. From the evaluation, we can conclude that the proposed error handling mechanism improves significantly the understanding of the query, especially for the corrupt queries but has limited influence on clean queries.

	Semantic Error Num.	%
For clean queries (without error handling)	5	4.1
For corrupt queries (without error handling)	62	50.8
For clean queries (with error handling)	5	4.1
For corrupt queries (with error handling)	33	27.0

Table 2: The comparative results of understanding experiments (sentence level)

U1: "坐公交车怎么去北京路？" ("How can I get to Bei Jing Lu by bus?")
Here, Bei Jing Lu is a transmeridional street in Shanghai city, which is divided into two parts: Bei Jing Dong Lu and Bei Jing Xi Lu. In real life, for brief, people often use "Bei Jing Lu" instead of indicating definitely "Bei Jing Dong Lu" or "Bei Jing Xi Lu".
S1: "请问您要到哪里去？" ("Excuse me, where will you go?")
The system without error correction can't acquire any useful information and then prompt the user to give his/her destination.
U2: "我想去北京路" ("uh, I want to go to Bei Jing Lu.")
The user doesn't realize that the term "Bei Jing Lu" is invalid to the system and reformulates her/his question in a different way.
S2: "请问您的目的地？" ("Where is your destination?")
The system without error handling still can't acquire any useful information and then prompts the user to provide the destination again.

Dialogue 3: Recurrence of a user's cognitive error in the system without error handling.

U1:“坐公交车怎么去北京路？” (“How can I get to Bei Jing Lu by bus?”)
S1: “你的目的地可能不正确，请在下面的地名中选择：北虹路，北京西路，北京东路，北海路，或者重新输入！” (“Your destination may be mistaken or not in my locations database. Please select your destination from the following: Bei Hong Lu, Bei Jing Xi Lu, Bei Jing Dong Lu, Bei Hai Lu, or speak a new one.”)
The system with error handling detects the mistaken destination and then prompts the user to give her/his destination definitely.
U2: “北京西路”(“Bei Jing Xi Lu.”)
The user realizes that the term “Bei Jing Lu” is invalid to the system and chooses her/his destination from the candidates list provided by the system.

Dialogue 4: The system with error handling mechanisms prevents recurrence of a user's cognitive error and moves the dialogue properly.

Two comparative examples of dialogue are shown in Dialogue 3 (without error handling) and 4 (with error handling). Some of the prompts in Dialogue 4 differ from that in Dialogue 3, which make the behaviors of the system easier to be understood by the user. This can prevent the dialogue from getting stuck and make the dialogue smoother.

5 Conclusions

This paper proposed a mechanism for error handling in spoken dialogue systems, which supports error handling at different levels instead of only regarding error handling as a post-processor for speech recognition. Firstly, we make use of as much knowledge sources as possible for error detection, such as a domain-specific dictionary, information offered by the language analysis component, and the user correction for system's error. Then, we combine similarity-based error correction approach and rule-based error correction approach to account for different kinds of errors: directly correct the input errors or provide the correction hypotheses for them. Finally, the system confirms these hypotheses to the user at dialogue level.

From the preliminary comparative experimental results in the context of public transportation information inquiring domain, it can be concluded that the use of the proposed error handling mechanism is likely to make a significant contribution to the performance of spoken language understanding. Though no quantitative results are available yet, our study on the tested dialogues (as Dialog 2 and 4) has shown that dialogues where the certain errors occur in the input sentences have improved significantly. The use of error handling can effectively prevent the dialogues from getting stuck, move dialogue states properly and make the dialogues more natural.

References

(Ainsworth & Pratt 92) W.A. Ainsworth and S.R. Pratt. Feedback strategies for error correction in speech recognition systems. *International Journal of Man-Machine Studies*, 36:833–842, 1992.

(Clark 96) H. H. Clark. *Using language*. Cambridge University Press, 1996.

(Gorrell 03) Genevieve Gorrell. Recognition error handling in spoken dialogue systems. In *The 2nd International Conference on Mobile and Ubiquitous Multimedia*, Norrkoping, Sweden, December 2003.

(Halverson *et al.* 99) C.A. Halverson, D. Horn, C. Karat, and J. Karat. The beauty of errors: Patterns of error correction in desktop speech systems. In *INTERACT'99*, pages 133–140, Edinburgh, Scotland, 1999.

(Jeong *et al.* 03) Minwoo Jeong, Byeongchang Kim, and Gary Geunbae Lee. Semantic oriented error correction for spoken query processing. In *ASRU'03*, US Virgin Island, November 2003.

(Kaki *et al.* 98) Satoshi Kaki, Eiichiro Sumita, and Hitoshi Iida. A method for correcting errors in speech recognition using the statistical features of character co-occurrence. In *ACL-COLING'98*, Montreal, Quebec, Canada, August 1998.

(Mao *et al.* 03) Jia-Ju Mao, Qiu-Lin Chen, Feng Gao, Rong Guo, and Ru-Zhan Lu. Stis: A chinese spoken dialogue system about shanghai transportation information. In *The 6th IEEE Conference on Intelligent Transportation Systems*, Shanghai, China, October 2003.

(Ringger & Allen 96) Eric K. Ringger and James F. Allen. A fertility channel model for post-correction of continuous speech recognition. In *ICSLP'96*, Philadelphia, PA., USA, October 1996.

(Suhm *et al.* 96) B. Suhm, B. Myers, and A. Waibel. Interactive recovery from speech recognition errors in speech user interfaces. In *ICSLP'96*, pages 861–864, Philadelphia, PA., USA, October 1996.

(Turunen & Hakulinen 01) Markku Turunen and Jaakko Hakulinen. Agent-based error handling in spoken dialogue systems. In *EUROSPEECH'01*, pages 2189–2192, Aalborg, Denmark, September 2001.

(Wu *et al.* 03) Wei-Lin Wu, Ru-Zhan Lu, and Zheng Liu. Comparative experiments on task classification for spoken language understanding using naive bayes classifier. In *IEEE NLP-KE'03*, pages 492–497, Bejing, China, October 2003.

Knowledge Acquisition Based on Automatically-Extracted Word Hierarchies from Domain-Specific Texts

Eiko Yamamoto and **Hitoshi Isahara**
Computational Linguistics Group
National Institute of Information and Communications Technology
3-5 Hikari-dai, Seika-cho, Souraku-gun
Kyoto 619-0289, Japan
{eiko,isahara}@nict.go.jp

Abstract

Automatic knowledge acquisition from huge corpora is one of the crucial topics in knowledge information processing. We are investigating a method of automatically constructing semantic hierarchies from corpora. In this paper, we discuss the possibility of extracting domain-specific knowledge from web documents within the medical domain as an example. We tried to acquire knowledge from experimental data using dependency relations between words in a corpus and to evaluate their potential for providing knowledge. We applied the complementary similarity measure (CSM) to determine a hierarchical structure of words in the corpus. We verified that the CSM-based method could extract domain-specific knowledge such as hierarchical semantic relations and causal relations.

1 Introduction

Automatic knowledge acquisition from huge corpora is a critical issue in knowledge information processing. We are investigating a method for automatically constructing semantic hierarchies from corpora. In determining semantic relations among words in corpora, the usual approach is to use patterns such as "a part of," "is a," "such as," and "and" (Hearst 92; Berland & Charniak 99; Caraballo 99). For Japanese documents, a method using collocations retrieved from documents (Nakayama & Matsumoto 97) and a hybrid method that uses both dictionaries and the dependency relations of words taken from documents (Matsumoto et al. 96) have been reported previously. We reported a method for constructing objective hierarchies of abstract nouns (Kanzaki et al. 04), and we also discussed improving that method by using information on the frequency of word collocations (Yamamoto et al. 05).

As we assumed the inclusion relations among word appearance patterns represent hierarchical relations between words, our method to build word hierarchies is based on the inclusion relation of word appearance patterns. We applied the complementary similarity measure (CSM) to determine a hierarchical structure of words in the corpus. The CSM is a similarity measure developed to recognize degraded machine-printed text (Hagita & Sawaki 95).

Although we used corpora on a wide range of topics during our experiments (Kanzaki et al. 04; Yamamoto et al. 05), the results varied based on the input corpora because the method itself is an automatic acquisition of hierarchy from corpora. In other words, we can extract domain-specific word hierarchies from domain-specific corpora. In this paper, we discuss the possibility of extracting domain-specific knowledge from web documents within the medical domain as an example of acquiring knowledge from corpora.

In our experiments, we used two types of experimental data obtained from the web documents. One type was obtained by exploiting the co-occurring relation between nouns in a corpus; the other was obtained by exploiting the dependency relation between nouns and verbs. Finally, we compared our extracted knowledge with the knowledge extracted with the baseline method and the hierarchy in the 2005 MeSH thesaurus[1].

2 Complementary Similarity Measure

As mentioned, we used CSM to estimate the hierarchical relations between word pairs. CSM was developed to recognize degraded machine-printed text and was designed to accommodate heavy noise or graphical designs (Hagita & Sawaki 95).

It has been applied to estimate one-to-many relationships between words from a corpus (Yamamoto & Umemura 02). Considering that the hypernym-hyponym relation is a kind of one-to-many relation, we applied CSM to the extraction of a hierarchy of abstract nouns that co-occur with adjectives in Japanese (Kanzaki et al. 04; Yamamoto et al. 05). We estimated the hierarchi-

[1]The U.S. National Library of Medicine created, maintains, and provides the Medical Subject Headings (MeSH) thesaurus.

cal relations from the inclusion relations between the appearance patterns for two words. An appearance pattern is expressed as an n-dimensional binary feature vector.

Let $F=(f_1,..,f_i,..,f_n)$ and $T=(t_1,..,t_i,..,t_n)$, where f_i and t_i are 0 or 1, be the feature vectors of the appearance patterns for two words. The CSM of F to T is defined as follows:

$$CSM(F,T) = \frac{ad-bc}{\sqrt{(a+c)(b+d)}},$$

$$a=\sum_{i=1}^{n} f_i \cdot t_i, \quad b=\sum_{i=1}^{n} f_i \cdot (1-t_i),$$
$$c=\sum_{i=1}^{n} (1-f_i) \cdot t_i, \quad d=\sum_{i=1}^{n} (1-f_i) \cdot (1-t_i),$$
$$n=a+b+c+d.$$

Note that "$ad-bc$" is symmetric, but "$(a+c)(b+d)$" is asymmetric. Therefore, $CSM(F,T)$ usually differs from $CSM(T,F)$. CSM computes the degree of inclusion of pattern T in pattern F.

If "n" is the number of sentences, "a" indicates the number of sentences in which both words appear and "b" indicates the number of sentences in which only the word corresponding to F appears. In contrast, "c" indicates the number of sentences in which only the word corresponding to T appears and "d" indicates the number of sentences in which neither word appears.

3 The Hierarchy Extraction Process

Our CSM-based method extracts word hierarchies as follows:

1. Compute the degree of inclusion between appearance patterns for each word by using CSM in each direction. The hierarchical relation between two words is determined by the CSM-value between them. If the CSM-value of X to Y is higher than the CSM-value of Y to X, the appearence pattern of word X mostly covers the appearence pattern of word Y, and we consider X as a hypernym of Y and Y as a hyponym of X. Then, we express this relation by a tuple (X, Y).

2. Normalize CSM-values and eliminate tuples with values below a threshold (TH).

3. For each word C,

   (a) Choose the tuple (C, D) with the highest CSM-value. This tuple is placed in the initial hierarchy.

   (b) Choose a tuple (D, E) such that the hyponym E is not contained in the current hierarchy and (D, E) has the highest value among the tuples where the bottom word D of the current hierarchy is a hypernym.

   (c) Connect the hyponym E to D at the bottom of the current hierarchy.

   (d) Choose another tuple (E, F) according to the previous step and repeat the process until no more such tuples can be chosen.

   (e) Choose a tuple (B, C) such that the hypernym B is not contained in the current hierarchy and (B, C) has the highest value among the tuples where the top word C of the current hierarchy is a hyponym.

   (f) Connect the hypernym B in front of C at the top of the current hierarchy.

   (g) Choose another tuple (A, B) according to the previous step and repeat the process until no more such tuples can be chosen.

4. If a short hierarchy exists that is included in a longer hierarchy and the order of the words stays the same, the short one is dropped from the list of hierarchies.

5. If hierarchies exist between which only one or a few words differ, the two hierarchies are merged and the different words are connected based on CSM-value. Suppose there are A-B-C-D-E-F-H and A-B-C-D-E-G-H in the list. If the CSM-value of F to G is higher than both TH and the CSM-value of G to F, the two hierarchies are merged and the resulting hierarchy is A-B-C-D-E-F-G-H. For the opposite relation, it is A-B-C-D-E-G-F-H.

4 Experimental Data

The Japanese language has case-marking particles that provide semantic relations between two elements in a dependency relation. We focused on these particles and then, using these particles as grounds for extraction, extracted two types of data for our experiments, i.e., dependency relations between a noun and a verb and co-occurrence relations among nouns.

First, we parsed sentences with the KNP[2] and

[2] A Japanese parser developed at Kyoto University.

collected from the parsing result dependency relations that match one of the following five patterns of case-marking particles. The five patterns are "A ⟨*no* (of)⟩ B," "A ⟨*wo* (object)⟩ C," "A ⟨*ga* (subject)⟩ C," "A ⟨*ni* (dative)⟩ C," and "A ⟨*ha* (topic)⟩ C," where A and B are nouns including compound words, and C is a verb. ⟨X⟩ is a case-marking particle. Suppose we have a sentence "*Cloe ha Mike ga Judy ni Christmas no gift wo ageta to kiita* (Cloe heard that Mike gave Judy a Christmas gift.)" in a corpus. From this sentence, we can extract five dependency relations between words, such as "*Christmas* ⟨*no*⟩ *gift*," "*gift* ⟨*wo*⟩ *ageta* (gave)," "*Mike* ⟨*ga*⟩ *ageta*," "*Judy* ⟨*ni*⟩ *ageta*" and "*Cloe* ⟨*ha*⟩ *kiita* (heard)." From this set of dependency relations we compiled two types of experimental data. Based on the co-occurrence relations among nouns, nouns followed by the case-marking particles *no*, *wo*, *ga*, *ni* and *ha* and nouns proceeded by *no* are gathered for each sentence. As for data based on the dependency relation between a noun and a verb, nouns with the case-marking particles *wo*, *ga*, *ni* and *ha* are gathered for each verb. We compiled the following five types of experimental data:

- **Co-data**: data based on co-occurrence between nouns.
- **Wo-data**: data based on a dependency relation with the case-marking particle ⟨*wo*⟩.
- **Ga-data**: data based on a dependency relation with the case-marking particle ⟨*ga*⟩.
- **Ni-data**: data based on a dependency relation with the case-marking particle ⟨*ni*⟩.
- **Ha-data**: data based on a dependency relation with the case-marking particle ⟨*ha*⟩.

When we represent Co-data by a binary vector, the number of dimensions is the number of sentences. The element in the vector is 1 if the noun appears in the sentence and 0 if the term does not appear in the sentence. For the other types of data, the number of dimensions is the number of verbs with each case-marking particle. The element in the vector is 1 if the noun depends on the verb with the particle and 0 if the term does not depend on the verb.

In compiling the experimental data, we used sentences collected from Japanese-language web pages related to the medical field. The size of the corpus is 37 Mbytes (10,144 pages). The number of sentences in the corpus is 225,402.

5 Experiment

We acquired term hierarchies as knowledge. First, as shown in processes 1 and 2 in Section 3, we extracted a list of tuples of medical terms. For the medical terms, we used those that are Japanese translations of descriptors in the 2005 MeSH thesaurus. The number of words used as medical terms in this experiment was 2,557. Using the tuples from the list in processes 3 to 5 in Section 3, we created hierarchies of medical terms. To avoid an upsurge in the number of hierarchies extracted, we carefully set the threshold (TH) and chose tuples to build term hierarchies that exceeded the TH. Finally, we selected as knowledge the hierarchies consisting of three or more terms from the extracted hierarchies.

6 Comparison

To evaluate our extracted knowledge, we compared it with the knowledge extracted from the experimental data by the baseline method and the MeSH thesaurus.

6.1 Comparison with the baseline method

We compared the result extracted with the Co-data and the CSM-based method with the output of the baseline method. For the baseline method, we used a typical approach for obtaining knowledge from a corpus in which a list of terms co-occurring in a corpus is extracted as knowledge. We gathered pairs of terms that co-occurred at least twice and used them as baseline knowledge. Of course, the CSM-based method with a sufficiently low TH can extract almost all the tuples extracted by the baseline method. Comparing sorted lists of tuples, i.e., the CSM-based list sorted by the CSM-value of the tuples and the baseline list sorted by the frequency of the pairs, we found more informative tuples near the top of the CSM-based list (see Table 1).

For example, both methods gave the highest score to the tuple ("administration," "treatment"). This indicates that, if the frequency of the tuple is high, the CSM-value of the tuple is also high. Because general terms have the tendency to appear more frequently than do technical terms in corpora, we can see many tuples of general terms near the top of the baseline list.

The tuple ("iron," "transferrin") in the fourth row of Table 1 has a high CSM-value; however,

No.	Tuples
1	(administration, treatment)
2	(daughter, nursery school)
3	(attention, referral)
4	(iron, transferrin)
5	(woods, orangutan)
6	(daughter, son)
7	(role, cytokine)
8	(stroke, epilepsy)
9	(secretion, glucocorticoid)
10	(nature, rights)

Table 1: List of the top 10 tuples extracted from Co-data with the CSM-based method

the tuple does not appear near the top of the baseline list because the frequency of the tuple is low. As shown in this sentence from a medical dictionary, "Iron is taken into the body with the molecule called Transferrin.," this tuple is informative and could only be extracted by CSM. This means that CSM can extract informative tuples even if the frequency of the tuple is low.

Another feature of the CSM-based method is that it can extract not only word pairs but also the hierarchical structure of words. As shown in section 3, the CSM can calculate the inclusive relations between two words and the results can be merged. This feature of CSM is not limited within a sentence. That is, once we obtain two tuples (A, B) and (B, C) determined by the CSM-value, we can connect them and obtain the triples A→B→C, even though A do not co-occur with C in a sentence. However, the baseline method extracts only the co-occurrence relations within a sentence and extracted set of words cannot be merged easily. The CSM-based method can use not only information within a sentence but also information from a wider context. Figure 1 shows some examples of combining tuples by using the CSM-based method.

6.2 Comparison with the MeSH thesaurus

We also compared the CSM-based knowledge with the MeSH Trees in 2005 MeSH thesaurus. The MeSH Trees are hierarchical arrangements of headings with their associated tree numbers. We gathered synonyms or closely related terms of headings which are stored as cross-references in the MeSH thesaurus, and add them to the MeSH Trees. The tree number includes information about the category. The MeSH headings are organized into 15 categories. If CSM-based method can extract the knowledge which

role - cell - tumor suppressing gene - chromosome
  - delayed fertilizations
secretion - gastric acid - gastric mucosa
  - duodenal ulcer
skin - atopic dermatitis - herpes viruses
  - antiviral drugs
fatigue - uterine muscle - pregnancy toxemia
fatigue - stress - duodenal ulcer
water - iron - transferrin - hemochromatosis
water - oxygen - hydrogen - hydrogen ion
person - nicotiana - smoke - oxygen deficiencies
neonate - patent ductus arteriosus
  - necrotizing enterocolitis
data - causation - depression - reduction
  - platelet count - bone marrow examination
data - causation - depression - nutritional status
  - intestinal fistula

Figure 1: Examples of knowledge obtained from Co-data with the CSM-based method

agree with MeSH thesaurus, terms in a CSM-based hierarchy are classified into one category in MeSH Trees. We examined the distribution of terms in MeSH categories for each type of experimental data, e.g., Ga-data (see Table 2). We found that the ratio of the CSM-based hierarchies whose terms were distributed in 1 or 2 MeSH categories was between 32% and 50%, and the ratio of the CSM-based hierarchies whose terms were distributed in 3 or fewer categories was between 52% and 66%. Of the CSM-based knowledge, Ga-data provided the highest agreement ratio. The reason for this seems to be that the subject case represented by the case-marker particle *ga* is more straightforward than the others.

	Co-data	Wo-data	Ga-data	Ni-data	Ha-data
Num. of knowledge	594	194	62	37	85
Distribution in					
1 category	24	35	12	3	6
2 categories	169	42	19	14	26
3 categories	116	34	10	5	14
Ratio in					
1 or 2 categories	.32	.40	.50	.46	.38
3 or fewer	.52	.57	.66	.59	.54

Table 2: Distribution of terms with CSM-based method

Figure 2 shows examples of our hierarchies whose all composed terms are classified into one category in MeSH Trees, i.e., those hierarchies are properly extracted by our method. Figure 3 shows examples whose composed terms are classified into two categories. The underlined terms in Figure 3 are those classified in a different category from others. We examined the differences between results obtained with our method and the MeSH thesaurus.

hand - mouth - ear - finger
skin - abdomen - cervix - cavitas oris - chest
bleb - flatulence - lower back pain
 - ulnar nerve palsies - brain hemorrhage
 - obstructive jaundice
cardiovascular disease - coronary artery disease
 - bronchitis - thrombophlebitides - flatulence
 - hyperuricemia - lower back pain
 - ulnar nerve palsies - brain hemorrhage
 - obstructive jaundice
anemia - emesis - lower back pain
 - ulnar nerve palsies - brain hemorrhage
 - obstructive jaundice
pancreatitides - angina pectoris
 - nephrotic syndrome
 - hypertensive encephalopathy
cephalalgia arthralgia - cyanoses
 - hematuria - purpura - leukopenia - sweating
 - peritoneal effusion
fatigue - stress - duodenal ulcer

Figure 2: Examples in which all terms are classified into one category

ice cream - chocolate - wine
medicine - herbalism - pharmacognosies
ovary - spleen - palpation
variation - cross reactions - outbreaks - secretion
bleeding - pyrexia - hematuria
 - consciousness disorder - vertigo
 - high blood pressure
fatigue - uterine muscle - pregnancy toxemia
fecundability - acrylic resins - cardiotonic drugs
 - vascular prostheses

Figure 3: Examples in which all terms are classified into either of two categories

For example, there is a collection, "ice cream - chocolate - wine" extracted by our method shown in Figure 3. From the standpoint of MeSH thesaurus, those should be divided into two categories, i.e., "ice cream" and "wine" are categorized as foods and "chocolate" is categorized as a material. However, from the viewpoint of natural language processing (NLP), even from the viewpoint of NLP for medical domain, those can be the same category, because they are all edible. This suggests the CSM-based method can extract better semantic relations from corpora than the MeSH thesaurus from the viewpoint of collocation relations, which are useful for NLP.

Moreover, "ovary - spleen - palpation" was extracted by the CSM-based method, reflecting the fact that "Diseases of the ovary and spleen can be diagnosed by palpation." We can interpret this as a causal relation.

We can also find such examples in Figure 1. "data - causation - depression - reduction - platelet count - bone marrow examination" is an example in which the terms are classified into 3 or more different categories. This includes a relation that means, "Bone marrow examination is necessary because bone marrow illnesses can cause depression and reduced platelet count." Our CSM-based method can extract such causal relations.

7 Conclusion

In this paper, we discussed the possibility of extracting domain-specific knowledge by using the example of web documents within the medical domain. We tried to acquire knowledge from experimental data by exploiting the dependency relations between words in the corpus and to evaluate their potential for providing knowledge. We then verified that the CSM-based method could extract domain-specific knowledge such as hierarchical semantic relations and causal relations. In the future, we will compare knowledge extracted from each of the types of experimental data.

References:

(Berland & Charniak 99) M. Berland, E. Charniak, *Finding parts in very large corpora*, In Proceedings of the 37^{th} Annual Meeting of the ACL, pp. 57-64, 1999.

(Caraballo 99) S. A. Caraballo, *Automatic construction of a hypernym-labeled noun hierarchy from text,* In Proceedings of the 37^{th} Annual Meeting of the ACL, pp. 120-126, 1999.

(Hagita & Sawaki 95) N. Hagita, M. Sawaki, *Robust recognition of degraded machine-printed characters using complementary similarity measure and error-correction learning,* In Proceedings of the SPIE – The International Society for Optical Engineering, 2442: pp. 236-244, 1995.

(Hearst 92) M. A. Hearst, *Automatic acquisition of hyponyms from large text corpora*, In Proceedings of the 14^{th} International Conference on Computational Linguistics, pp. 539-545, 1992.

(Kanzaki et al. 04) K. Kanzaki, E. Yamamoto, Q. Ma, and H. Isahara, *Construction of an objective hierarchy of abstract concepts via directional similarity.* In Proceedings of the 20^{th} International Conference on Computational Linguistics, Vol.2, pp. 1147-1153, 2004.

(Matsumoto et al. 96) Y. Matsumoto, S. Sudo, T. Nakayama, and T. Hirao, *Thesaurus construction from multiple language resources,* In IPSJ SIG Notes NL-93, pp. 23-28, 1996.

(Nakayama & Matsumoto 97) T. Nakayama, Y. Matsumoto, *Positioning nouns in a classification-based thesaurus,* In IPSJ SIG Notes NL-120, pp. 103-108, 1997.

(Yamamoto & Umemura 02) E. Yamamoto, K. Umemura, *A similarity measure for estimation of one-to-many relationship in corpus,* In Journal of Natural Language Processing, pp. 45-75, 2002.

(Yamamoto et al. 05) E. Yamamoto, K. Kanzaki and H. Isahara, *Extraction of hierarchies based on inclusion of co-occurring words with frequency information.* In Proceedings of the 19^{th} International Joint Conference on Artificial Intelligence, pp. 1166-1172, 2005.