Proceedings of the

Third Conference on Empirical Methods in Natural Language Processing

Sponsored by

The Association for Computational Linguistics
ACL/SIGDAT

Edited by

Nancy Ide
and
Atro Voutilainen

2 June 1998
Palacio de Exposiciones y Congresos
Granada, Spain

Order additional copies from:

ACL
P.O. Box 6090
Somerset, NJ 08875
USA
1-732-873-3898 (phone)
1-732-873-0014 (fax)
acl@aclweb.org

SPONSORS:

The Association for Computational Linguistics (ACL)
SIGDAT (ACL's SIG for Linguistic Data and Corpus-based Approaches to NLP)

INVITED SPEAKER:

Kevin Knight (USC Information Sciences Institute)

ORGANIZERS:

Nancy Ide (Vassar College), Chair
Atro Voutilainen (University of Helsinki), Co-chair

PROGRAM COMMITTEE:

Steven Abney	AT&T Laboratories-Research, USA
Susan Armstrong	(ISCCO, Switzerland)
Pascale Fung	Hong Kong Univ. of Science and Technology, Hong Kong
Gregory Grefenstette	Xerox Research Centre Europe, France
Eduard Hovy	USC/ISI, USA
Dan Jurafsky	University of Colorado, Boulder, USA
Kimmo Koskenniemi	University of Helsinki, Finland
Hwee Tou Ng	DSO National Laboratories, Singapore
Kemal Oflazer	Bilkent University, Turkey
Peter Schauble	ETH Zürich, Switzerland
Keh Yih Su	Tsing-Hua University, Taiwan
Dan Tufis	Romanian Academy of Sciences, Romania
Evelyne Viegas	New Mexico State University, USA

FURTHER INFORMATION:

Nancy Ide
Department of Computer Science
Vassar College
124 Raymond Avenue
Poughkeepsie, New York 12604-0520 USA
email: ide@cs.vassar.edu

FOREWORD

The Third Conference on Empirical Methods in Natural Language Processing offers a general forum for novel research in corpus-based and statistical natural language processing. This year, EMNLP is held in conjunction with the First International Language Resources and Evaluation Conference in Granada, Spain, which is concerned with existing and required resource development to support language processing work in an increasingly multi-lingual setting. Indeed, the development of natural language applications that handle multi-lingual information is the next major challenge facing the field of computational linguistics.

Given this context, this year's EMNLP conference is focused on work that describes and evaluates the strengths, weaknesses, and recent advances in corpus-based NLP as applied to multi-lingual applications. In particular, many of the papers in this volume consider questions such as the following: how well do techniques for lexical tagging, parsing, anaphora resolution, etc., handle the specific problems of multi-lingual applications? What new methods have been developed to address the deficiencies of existing algorithms for these tasks or to address problems specific to handling multi-lingual applications? What problems still lack an adequate empirical solution? Conversely, how can data-driven NLP methods be improved with the help of multi-lingual data?

It is appropriate that this is the first EMNLP conference to be held outside the U.S. We are very encouraged to see the participation of so many researchers from Europe and Asia, which will result, we hope, in greater communication and collaboration across the international NLP community.

Many people are owed thanks for their contributions to setting up this conference. In particular, Atro Voutilainen, EMNLP3 co-chair, and David Yarowksy, SIGDAT chair, provided continual and indispensable help and support throughout. The EMNLP3 Program Committee enabled us to work within a very brief time frame, by quickly turning around all the reviews for the substantial number of submissions to the conference. Finally, the LREC conference organization committee at the University of Granada, the LREC program organizers at the Istituto di Linguistica Computazionale in Pisa, and the Department of Computer Science at Vassar College provided administrative and organizational support. All of them are responsible for the success of EMNLP3.

Nancy Ide, EMNLP3 Chair
Poughkeepsie, New York
May, 1998

TABLE OF CONTENTS

CONFERENCE PROGRAM

8:45 - 9:00	Welcome
9:00 - 9:30	Breck Baldwin and Thomas S. Morton *Dynamic Coreference-Based Summarization*
9:30 - 10:00	Ruslan Mitkov, Lamia Belguith, and Malgorzata Stys *Multilingual Robust Anaphora Resolution*
10:00 - 10:30	Sotiris Boutsis and Stelios Piperidis *Aligning Clauses in Parallel Texts*
10:30 - 11:00	Michel Simard *Automatic Insertion of Accents in French Text*
11:00 - 11:30	Break
11:30 - 12:00	Glenn Carroll and Mats Rooth *Valence Induction with a Head-Lexicalized PCFG*
12:00 - 12:30	Adam Kilgarriff and Tony Rose *Measures for Corpus Similarity and Homogeneity*
12:00 - 12:30	Rebecca Bruce and Janyce Wiebe *Word-Sense Distinguishability and Inter-Coder Agreement*
1:00 - 2:45	Lunch
2:45 - 3:30	Invited Speaker : Kevin Knight (USC Information Systems Institute) *Statistical Translation: Where It Went*
3:30 - 4:00	Stephen D'Alessio, Keitha Murray, Robert Schiaffino, and Aaron Kershenbaum *Category Levels in Hierarchical Text Categorization*
3:30 - 4:00	Fumiyo Fukumoto and Yoshimi Suzuki *An Empirical Approach to Text Categorization based on Term Weight Learning*
4:30 - 5:00	Break
5:00 - 5:30	Shirai Kiyoaki, Inui Kentaro, Tokunaga Takenobu and Tanaka Hozumi *An Empirical Evaluation on Statistical Parsing of Japanese Sentences Using Lexical Association Statistics*
5:30 - 6:00	Fujio Masakazu and Matsumoto Yuji *Japanese Dependency Structure Analysis based on Lexicalized Statistics*
6:00 - 6:30	Adam Berger and Harry Printz *A Comparison of Criteria for Maximum Entropy / Minimum Divergence Feature Selection*

Dynamic Coreference-Based Summarization

Breck Baldwin
Institute for Research
in Cognitive Science
University of Pennsylvania

Thomas S. Morton
Department of Computer
and Information Science
University of Pennsylvania

{breck,tsmorton}@linc.cis.upenn.edu

Introduction

We have developed a query-sensitive text summarization technology well suited for the task of determining whether a document is relevant to a query. Enough of the document is displayed for the user to determine whether the document should be read in its entirety. Evaluations indicate that summaries are classified for relevance nearly as well as full documents. This approach is based on the concept that a good summary will represent each of the topics in the query and is realized by selecting sentences from the document until all the phrases in the query which are represented in the summary are 'covered.' A phrase in the document is considered to cover a phrase in the query if it is coreferent with it. This approach maximizes the space of entities retained in the summary with minimal redundancy. The software is built upon the CAMP NLP system [2].

Problem Statement

Given the relative immaturity of summarization technologies and their evaluation, it is worthwhile to describe our approach in detail and the problems it is intended to solve. An important aspect of our technique is that we produce sentence extraction summaries which are constructed by selecting sentences from the source document. In addition, our summaries are focused on providing relevant information about a query. We feel that the current state-of-the-art techniques are better equipped to produce high quality query-sensitive summaries than generic summaries. Our goal is to produce 'indicative' summaries [4] which allow a user to determine whether the document is relevant to his or her query. The summary is not intended to replace the document or provide answers to questions directly but may have this effect.

Casting our technology in terms of a product, we see the application as an intermediate step between viewing entire documents and the output of an information retrieval engine. Instead of looking at either headlines or an entire document, the user would look at the summaries of the documents and then decide whether the document merited further reading.

Approach

We conducted a simple experiment with summaries produced in the TIPSTER summarization dry run [6]. For 5 queries with 200 documents each, we took the set of summaries produced by the 6 dry-run participants and retained only those summaries that were true-positives, i.e., the summary was judged 'relevant' and the full document was judged 'relevant'. Over all the queries, at least one of the six systems produced a true-positive summary for 96.6% of the documents, although no individual system performed nearly at that level. This meant that some existing technology produced a correct summary for almost every relevant document. Hence we viewed the problem as one of balancing the capabilities of our system to behave like the amalgamated system implicit in joined output. Based on this result we are confident that this class of summarization is tractable with current technologies and this has strongly motivated our design decisions.

Upon encountering a query like "Reporting on possibility of and search for extra-terrestrial life/intelligence.", we assume that the user has defined a class of actions, ideas, and/or entities that he or she is interested in. The job of an information retrieval engine is to find instantiations of those classes in text documents in some database. We view summarization as an additional step in this process where we attempt to present the user with the smallest collection of sentences in the document that instantiate the user specified classes and do not mislead the user about the overall content of the document. By doing so, we can greatly shorten the amount of the document that

the user must read in order to determine whether the document is relevant for the user's needs.

Just as information retrieval algorithms approximate document relatedness by examining various string matchings between the query and the text, we approximate certain classes of coreference between the query and the text by examining linguistic information. These coreference relations include identity of reference and part-whole relations for nominal and verbal phrases.[1] This moves us a step closer to reasoning at a more appropriate level of generalization, for summarization, which is still technologically feasible. Below are examples indicating the classes of relatedness that we are trying to capture.

The identity relation between the query and the document

Noun phrase coreference is the best understood class of relations that we compute. For example, there is coreference between 'Federal Emergency Management Agency' in the query and the acronym 'FEMA' in the document below:

> *Query:* What is the main function of the **Federal Emergency Management Agency** and the funding level provided to meet emergencies?
>
> *Document:* ... **FEMA** agrees that "fine-tuning" is needed to the 1974 act establishing a coordinated federal program to prepare for and respond to hurricanes, tornadoes, storms and floods. ...

Since these noun phrases refer to the same entity in the world, sentences that mention the organization would be particularly valuable in a summary. This class of coreference can include people, companies and objects such as automobiles or aluminum siding. It need not be restricted to proper nouns as it is possible to refer to an entity using common nouns, i.e. 'the agency' and pronouns.

Identity also holds between events mentioned in the query and document. Sometimes the event that a query describes is the best indicator of what document should be retrieved, and correspondingly what sentences are appropriate for a summary. Consider the following:

> *Query:* A relevant document will provide new theories about **the 1960's assassination** of President Kennedy.
>
> *Document:* ...The House Assassinations Committee concluded in 1978 that Kennedy was "probably" **assassinated** as the result of a conspiracy involving a second gunman, a finding that broke from the Warren Commission's belief that Lee Harvey Oswald acted alone in Dallas on Nov. 22, 1963. ...

The noun phrase 'the 1960's assassination' refers to an event, which is the same as the one referred to in the document with the verb 'assassinated'. Note also that there is coreference between 'President Kennedy' and 'Kennedy' in the document.

The part-whole relation between the query and the document

In addition to the identity relation, phrases in a text which refer to parts of an entity or concept mentioned in the query will likely provide useful information, and therefore should be included in a summary. Finding these relations in in general is beyond the scope of this paper, however, our approximation of a subclass of these relations proved helpful for a number of queries.

A strong example of the part-whole relation occurs when a country is mentioned in the query and a province or city within that country is mentioned in the document. For example:

> *Query:* Document will discuss efforts by the black majority in **South Africa** to **overthrow** domination by the white minority government.
>
> *Document:* About 90 soldiers have been arrested and face possible death sentences stemming from a coup attempt in **Bophuthatswana**, ... Rebel soldiers **staged** the takeover bid Wednesday, **detaining** homeland President Lucas Mangope. ...

Bophuthatswana is inside South Africa, and sentences that mention it are clearly good candidates for inclusion in a summary.

We also consider part-whole relations between events as in the relation between 'overthrow' and 'staged' and 'detained'. Those events are sub-parts of overthrow events, and as such, sentences that contain sub-parts of the events are reasonable candidates for inclusion in summaries.

Implementation

The summarization technique was developed within the CAMP NLP framework. This system provides an integrated environment in which to access many levels of linguistic information as well as world knowledge. Its main components include: named entity recognition, tokenization, sentence detection, part-of-speech tagging, morphological analysis, parsing, argument detection, and coreference resolution. Many of the techniques used for these tasks perform at or near the

[1] It is not clear whether more sophisticated annotations are appropriate for information retrieval, and perhaps more to the point, it is not clear that there are sufficient resources to process 2 GB collections of data.

state of the art and are described in more depth in [12, 9, 8, 7, 5, 1, 2]. The system produces coreference annotated documents which serve as the input to the summarization algorithm.

Relating the query to the document

The relationships discussed previously are approximated via a series of associations between tokens in the query, headline, and the body of the document. Event references are captured by associating verbs or nominalizations in the query with verbs and nominalizations in the document.

Given three verbal forms v_1 in the query, v_2 in the document, and v_3 in the set of all verbal forms, where a verbal form is the morphological root of a verb or the verb root corresponding to a nominalization, v_1 is associated with v_2 if at least one of the following criteria are met:

1. $(v_1 \neq v_2) \wedge p(v_1, v_2)/(p(v_1)p(v_2)) \geq 5$
2. $(v_1 = v_2) \wedge (\exists v_3 \neq v_1 \mid p(v_1, v_3)/p(v_1)p(v_3) \geq 5)$
3. $(v_1 = v_2) \wedge ((subject(v_1) = subject(v_2)) \vee (object(v_1) = object(v_2)))$

Here $p(v_i)$ is the probability that v_i occurs in a document and $p(v_i, v_j)$ is the probability that v_i and v_j occur in the same document. These probabilities are based on frequencies gathered from approximately 45,000 Wall Street Journal articles. Criterion 1 is a measure of mutual information between two verbs. Criterion 2 is used to rule out frequently occurring verbs such as "be" and "make". Criterion 3 allows for verbs which are ruled out by criterion 2 to be associated when additional context is available. This is important since some queries only contain verbal forms which are ruled out by criterion 2.

Relationships between proper nouns are made on the basis of string matches, acronym matching, and dictionary lookup. Acronyms are determined either through a table lookup or an appositive construction occurring in the document which designates the acronym for a specific proper noun. A proper noun in the query is considered associated with a proper noun in the document if it matches the string or acronym of the proper noun in the document or it appears in the definition of the proper noun in the document. A reverse dictionary lookup often allows cities to be associated with the country they are in.

A token in the query which is a lowercase noun or adjective is associated with any token in the document which matches its morphological root and part of speech.

Tokens which occur in the headline are associated with tokens in the document body using the same criteria as the query, with the exclusion of the dictionary lookup. The dictionary lookup was excluded because the headline will likely use the same lexicalization of a proper noun as that used in a document. This is less likely to be the case with the query.

Selecting a sentence

The associations discussed in the previous section are used to rank and select sentences from the document. Every token in the document which is associated with the same token in the query or headline is considered to be in the same coreference chain. A sentence which contains any token in a given coreference chain is said to cover that chain.

The following scores are computed for each sentence in the document:

1. The number of coreference chains from the query which are covered by the sentence and haven't been covered by a previously selected sentence.
2. The number of noun coreference chains from the query which are covered by the sentence and the number of verbal terms in the sentence which are chained to the query.
3. The number of coreference chains from the headline which are covered by the sentence and haven't been covered by a previously selected sentence.
4. The number of noun coreference chains from the headline which are covered by the sentence and the number of verbal terms in the sentence which are chained to the headline.
5. The number of coreference chains which are covered by the sentence and haven't been covered by a previously selected sentence.
6. The number of noun coreference chains which are covered by the sentence.
7. The index of the sentence in the document; sentences are sequentially numbered.

The sentences are sorted based on the above scores, where the *ith* scoring criteria is only considered in case of a tie for all criteria less than i. Scores 1-6 are ranked in descending order while score 7 is ranked in ascending order. The top-ranked sentence is selected, and scores 1, 3, and 5 are recomputed in order to select the next sentence. Selection halts when all coreference chains in the query have been covered and the summary contains at least 4 sentences.

Scores 1 and 2 are used to select sentences which are related to the query. Scores 3 and 4 are motivated by documents which have 1 or 2 sentences which appear

related to the query but if presented alone would give a false impression of the true content of the document. Thus sentences related to the headline are presented to provide additional background. Consider the following example:

Query: What evidence is there of paramilitary activity in the U.S.?

Summary: ...Last month the extremists used rocket-propelled grenades for the first time in three attacks on police and paramilitary units. ...

This sentence was selected because it contains tokens which are in coreference chains with tokens in the query; however, alone it is potentially misleading because the place of the attack is not mentioned. This ambiguity is resolved when the following sentence is selected because it is well associated with the headline.

Summary: ...Sikh militants may have acquired one or two U.S.-made Stinger anti-aircraft missiles and hidden them inside the Golden Temple, the Sikh faith's holiest shrine, Punjab police officials said Saturday....

This provides enough background information for the reader to realize that the para-military activity is not taking place in the U.S. and thus that the document is irrelevant to the query.

Likewise, scores 5 and 6 act similarly to 3 and 4 for documents which do not contain a headline. We found this particularly important for advertisements which often don't state a product or company name in the beginning of the document, but will repeat these names numerous times throughout the document.

Generating the summary

Once sentences have been selected, they are presented in the order they occurred in the document. Pronouns which do not have a referent in the previous sentence of the summary are filled with a more descriptive string whenever a referent can be determined. If space is of concern, prepositional phrases attached to nouns (which are not nominalizations), appositives, conjoined noun phrases and relative clauses are removed, provided they contain no tokens associated with the query or the headline. Since determining pronoun referents and the selection of clauses for removal are subject to errors, filled pronouns are placed in square brackets and removed clauses are replaced with an ellipsis to indicate to the reader that the original text has been modified.

Example summary

An example summary which demonstrates many of the features of our system appears below. It has been constrained to be approximately 10% of the original document length, so it is not representative of the summaries used in the evaluation, but it contains examples of the of both pronoun filling and clause deletion.

The last sentence in the summary was selected first because the tokens "death","sentence", "kill", and "term" were associated with the nominalization "punishment". The stranded pronoun "it" has also been filled. Sentence 2 was selected next because of the match-up between the verb "is" and the object "deterrent" in the document and the query. Finally, the first sentence was chosen because there is another mention of the prison name "Marion" in the document. This summary differs from the one generated when the 10% length constraint is not imposed, because some higher ranked sentences were passed over since their inclusion would have exceeded the length restriction.

Query: Is there data available to suggest that capital punishment is a deterrent to crime?

Summary: "Marion is basically the end of the line," Bogdan said.
... There is no deterrent ... to keep them from doing this again.
Additionally, [the pending Senate bill] would create five new death penalty offenses: murder by a federal inmate serving a life sentence; drug kingpins in a continuing criminal enterprise even if no murders occur; drug kingpins who try to kill to obstruct justice; drug felons who unintentionally kill with aggravated recklessness; and people who kill with a firearm during a violent ... crime.

Evaluation

In order to evaluate our summarization algorithm, we selected 10 unseen queries from the Text REtrieval Conference (TREC) document collection. Summaries were generated for 200 documents, 20 per query, and assessors[2] were asked to make relevance judgments based on the summaries. A document was considered relevant if it contained the information requested in the query or if the assessor believed that the full document would likely contain this information. The relevance judgments were then compared to those made by the TREC assessors using the full document. This comparison places a summary in one of the following categories:

- a = judged relevant, full document is relevant
- b = judged relevant, full document is irrelevant
- c = judged irrelevant, full document is relevant

[2]Each author served as an assessor making judgments for 100 documents across 10 queries.

- d = judged irrelevant, full document is irrelevant

Precision, recall, and accuracy are then computed as follows:

$$\text{precision} = a/(a+b)$$
$$\text{recall} = a/(a+c)$$
$$\text{accuracy} = (a+d)/(a+b+c+d)$$

Compression is computed over the number of non-whitespace characters in the summary and the original document. Here compression is defined as the percentage of the document that was not included in the summary:

$$\text{compression} = \frac{(length_{document} - length_{summary})}{length_{document}}$$

The results from our experiment are shown in the following table:

Precision	82.8%	101/(101+21)
Recall	77.7%	101/(101+29)
Compression	82.8%	(704686-121272)/704686
Accuracy	75.0%	(101+49)/200

A second evaluation on 910 documents was performed for [4]. These results superficially appear significantly worse than those from the initial evaluation however a more careful analysis (provided in the discussion section) shows that they are in fact similar to the results of the previous evaluation.

Precision	80.3%	322/(322+79)
Recall	57.6%	322/(322+237)[f]
Compression	83.0%	
Accuracy	65.3%	(322+272)/910

Discussion

We view the results of the first evaluation as promising in that they compare favorably with inter-assessor consistency using the entire document. [11] reports unanimous relevance judgments by three assessors for 71.7% of the documents. Interpolating this figure to two assessors yields an 80.1% agreement figure. Using summaries which on average are only 17.2% of the original document, our assessors matched the TREC assessors for 75.0% of the documents.

The second evaluation yielded a much lower recall figure while precision remained comparable. This, however, is also the case when the same assessors judgments on the full documents are compared to those of the TREC assessors. These results are as follows:

Precision	83.5%	167/(167+33)
Recall	63.5%	167/(167+96)
Compression	100.0%	
Accuracy	69.3%	(167+124)/420

We view these results as favorable as well since our accuracy is 65.3% using 17.0% of the document on average compared to 69.3% accuracy using the entire document. The discrepancy between the two evaluations appears to be based on the assessors in the second evaluation using a stricter criteria for relevance than that used by the previous evaluation's assessors or the TREC assessors.

It was noted after the first evaluation that different criteria for relevance accounted for some of the disagreement between our assessors and the TREC assessors. Many documents considered relevant were marked as irrelevant due to different notions of relevance and not because the summary failed to provide material on which to base a correct decision. These difficulties only hinder the evaluation of a summary system and not its use in an application, since a user will have a clear idea of his or her intentions when determining a document's relevance.

As we mentioned previously, our approach has been to balance methods of relating the query to sentences in the document. The nearly 100% recall of the dry-run summaries encouraged us, and we even used the output of those summaries to provide a test-bed for evaluating our summaries. Although we never actively sought to emulate aspects of other systems directly, our final algorithm does share some basic ideas and approaches from those systems. Some of the similarities are listed below:

In [3], they eliminate redundant information from summaries by classifying sentences according to Maximal Marginal Relevance (MMR). MMR ranks text chunks according to their dissimilarity to one another. Summaries can then be produced with sentences that are maximally dissimilar, thereby increasing the likelihood that distinguishing information will be in the summary. One can view our coverage requirement for terms in the query as an attempt to pick dissimilar sentences from the document. Instead of MMR, we use the fact that a sentence which does not contain redundantly referring phrases to the query is more highly ranked than a sentence that does.

Our individual sentence scoring algorithm shares some properties with [10]. Their approach includes scores for anaphoric density, string equivalence with the title or headline of a document, and position of the sentence in the document. However, we do not take advantage of overt cues for summary sentences, such as 'in summary' or 'in conclusion', nor do we use temporal information in generating a summary.

Like many systems, we do a form of word expansion in attempting to relate the query to the document. However, the fact that we restrict expansion to proper nouns and verbs and their nominalizations is notable. We found this limited set of expansions restricts the relations between the text and the query well and also fits

within the framework of part-whole relations in coreference. We did not consider part-whole relations for common nouns, because in practice we have not had very good results limiting over-generation in that domain.

Conclusions and Future Work

We have developed and tested a query-sensitive text summarization system that is nearly as effective as full text documents for determining whether a document is relevant to the query. The system uses a limited class of coreference-based relations between the query and the document to select sentences which represent instantiations of entities, events, or concepts articulated in the query. The algorithm is implemented within the CAMP NLP system and utilizes linguistic generalizations like part-of-speech, parsing and predicate-argument structure.

An issue in evaluating our system is that the input data has been selected by an information retrieval engine. As such, we have no data on how well our summaries would work on relevant documents that the information retrieval engine fails to retrieve. These engines tend to select documents based on string matching and we have shown that our summarization technology does an excellent job of summarizing them. However, the information retrieval engine may be acting as an advantageous filter on the space of documents. It would be interesting to do experiments on relevant documents that contain very few string matches with the query.

In the future we hope to improve the accuracy of the coreference relations. Specifically, we will focus on the recognition of events which we believe are very important to a large class of queries.

Acknowledgments

We would like to acknowledge three anonymous reviewers for their helpful comments and Tonia Bleam for providing assessments during the development of this system.

References

[1] Breck Baldwin. CogNIAC: High precision coreference with limited knowledge and linguistic resources. In *Proceedings of the ACL Workshop on Operational Factors in Practical, Robust Anaphora resolution for Unrestricted Texts*, pages 38–45, Madrid, Spain, June 1997.

[2] Breck Baldwin, Christine Doran, Jeffrey C. Reynar, Michael Niv, B. Srinivas, and Mark Wasson. EAGLE: An extensible architecture for general linguistic engineering. In *Proceedings of RIAO-97*, Montreal, 1997.

[3] Michael Bett and Jade Goldstein. Automated query-relevant document summarization. In *Proceedings of Tipster Text Phase III 12-Month Workshop*, 1997.

[4] Michael Chrzanowski, Therese Firmin, Lynette Hirschman, David House, Inderjeet Mani, Leo Obrst, Sara Shelton, Beth Sundheim, and Sandra Wagner. (SUMMAC) call for participation. http://www.tipster.org/summcall.htm, January 1998.

[5] Michael John Collins. A New Statistical Parser Based on Bigram Lexical Dependencies. In *Proceedings of the 34th Annual Meeting of the ACL*, 1996.

[6] Therese Hand. Tipster summarization evaluation task:dry-run evaluation results. In *Proceedings of Tipster Text Phase III 12-Month Workshop*, 1997.

[7] Daniel Karp, Yves Schabes, Martin Zaidel, and Dania Egedi. A freely available wide coverage morphological analyzer for english. In *Proceedings of the 15th International Conference on Computational Linguistics*, 1994.

[8] Adwait Ratnaparkhi. A Maximum Entropy Part of Speech Tagger. In Eric Brill and Kenneth Church, editors, *Conference on Empirical Methods in Natural Language Processing*, University of Pennsylvania, May 17-18 1996.

[9] Jeffrey C. Reynar and Adwait Ratnaparkhi. A maximum entropy approach to identifying sentence boundaries. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pages 16–19, Washington, D.C., April 1997.

[10] Tomek Strzalkowski, Fang Lin, Jin Wang, Langdon White, and Bowden Wise. Natural language information retrieval and summarization. In *Proceedings of Tipster Text Phase III 12-Month Workshop*, 1997.

[11] Ellen M. Voorhees and Donna Harman. Overview of the fifth Text REtrieval Conference (TREC-5). In *Proceedings of the Fifth Text REtrieval Conference (TREC-5)*, pages 1–28. NIST 500-238, 1997.

[12] Nina Wacholder, Yael Ravin, and Misook Choi. Disambiguation of proper names in text. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, May 1997.

Multilingual robust anaphora resolution

Ruslan Mitkov
School of Languages and European Studies
University of Wolverhampton
Stafford Street
Wolverhampton WV1 1SB, United Kingdom
Email R.Mitkov@wlv.ac.uk

Lamia Belguith
LARIS - FSEG
University of Sfax
B.P. 1088
3018 Sfax, Tunisia
Email belguith.lamia@planet.tn

Malgorzata Stys
Computer Laboratory
University of Cambridge
New Museums Site, Pembroke Street
Cambridge CB2 3QG
United Kingdom
Email Malgorzata.Stys@cl.cam.ac.uk

Abstract

Most traditional approaches to anaphora resolution rely heavily on linguistic and domain knowledge. One of the disadvantages of developing a knowledge-based system, however, is that it is a very labour-intensive and time-consuming task. This paper presents a robust, knowledge-poor approach to resolving pronouns in technical manuals. This approach is a modification of the practical approach (Mitkov 1998a) and operates on texts pre-processed by a part-of-speech tagger. Input is checked against agreement and a number of antecedent indicators. Candidates are assigned scores by each indicator and the candidate with the highest aggregate score is returned as the antecedent. We propose this approach as a platform for multilingual pronoun resolution. The robust approach was initially developed and tested for English, but we have also adapted and tested it for Polish and Arabic. For both languages, we found that adaptation required minimum modification and that further, even if used unmodified, the approach delivers acceptable success rates. Preliminary evaluation reports high success rates in the range of and over 90%

1. Introduction: robust, knowledge poor anaphora resolution and multilingual NLP

For the most part, anaphora resolution has focused on traditional linguistic methods (Carbonell & Brown 1988; Carter 1987; Hobbs 1978; Ingria & Stallard 1989; Lappin & McCord 1990; Lappin & Leass 1994; Mitkov 1994; Rich & LuperFoy 1988; Sidner 1979; Webber 1979). However, to represent and manipulate the various types of linguistic and domain knowledge involved requires considerable human input and computational expense.

While various alternatives have been proposed, making use of e.g. neural networks, a situation semantics framework, or the principles of reasoning with uncertainty (e.g. Connoly et al. 1994; Mitkov 1995; Tin & Akman 1995), there is still a strong need for the development of robust and effective strategies to meet the demands of practical NLP systems, and to enhance further the automatic processing of growing language resources.

Several proposals have already addressed the anaphora resolution problem by deliberately limiting the extent to which they rely on domain and/or linguistic knowledge (Baldwin 1997; Dagan & Itai 1990; Kennedy & Boguraev 1996; Mitkov 1998; Nasukawa 1994; Williams et al. 1996). Our work is a continuation of these latest trends in the search for inexpensive, rapid and reliable procedures for anaphora resolution. It shows how pronouns in a specific genre can be resolved quite successfully without any sophisticated linguistic knowledge or even without parsing, benefiting instead from corpus-based NLP techniques such as sentence splitting and part-of-speech tagging.

On the other hand, none of the projects reported so far, has looked at the multilingual aspects of the approaches that have been developed, or, in particular, how a specific approach could be used or adapted for other languages. Furthermore, in addition to the monolingual orientation of all approaches so far developed, most of the work has concentrated on pronoun resolution in one language alone (English).

While anaphora resolution projects have been reported for French (Popescu-Belis & Robba 1997, Rolbert 1989), German (Dunker & Umbach 1993; Fischer et al. 1996; Leass & Schwall 1991; Stuckardt 1996; Stuckardt 1997), Japanese (Mori et al. 1997; Nakaiwa & Ikehara 1992; Nakaiwa & Ikehara 1995; Nakaiwa et al. 1995; Nakaiwa et al. 1996; Wakao 1994), Portuguese (Abraços & Lopes 1994), Swedish (Fraurud, 1988) and Turkish (Tin & Akman, 1994), the research on languages other than English constitutes only a small part of all the work in this field.

In contrast to previous work in the field, our project has a truly multilingual character. We have developed a knowledge-poor, robust approach which we propose as a platform for multilingual pronoun resolution in technical manuals. The approach was initially developed and tested for English, but we have also adapted and tested it for Polish and Arabic. We found that the approach could be adapted with minimum modifications for both languages and further, even if used without any modification, it delivers acceptable success rates. Evaluation shows a success rate of 89.7% for English, 93.3% for Polish and 95.2% for Arabic.[1]

2. The approach: general overview

With a view to avoiding complex syntactic, semantic and discourse analysis, we developed a robust, knowledge-poor approach to pronoun resolution which does not make use of parsing, syntactic and semantic constraints or any other form of linguistic or non-linguistic knowledge. Instead, we rely on the efficiency of sentence segmentation, part-of-speech tagging, noun phrase identification and the high performance of the antecedent indicators (knowledge is limited to a small noun phrase grammar, a list of terms, a list of (indicating) verbs, a list of genre-specific synonyms, and a set of antecedent indicators).

The core of the approach lies in activating a list of multilingual[2] "antecedent indicators" after filtering candidates (from the current and two preceding sentences) on the basis of gender and number agreement. Before that, the text is pre-processed by a sentence splitter which determines the sentence boundaries, a part-of-speech tagger which identifies the parts of the speech and a simple phrasal grammar which detects the noun phrases (In addition, in the case of complex sentences, heuristic "clause identification" rules track the clause boundaries). Non-anaphoric occurrences of "it" in constructions such as "It is important", "It is necessary" etc., are eliminated by a "referential filter".

After passing the "agreement filter", the genre-specific antecedent indicators are applied to the remaining candidates (see section 2.2). The noun phrase with the highest aggregate score is proposed as antecedent; in the rare event of a tie, priority is given to the candidate with the higher score for immediate reference. If immediate reference has not been identified, then priority is given to the candidate with the best collocation pattern score. If this does not help, the candidate with the higher score for indicating verbs is preferred. If still no choice is possible, the most recent from the remaining candidates is selected as the antecedent.

2.1 Agreement filter

The detected noun phrases (from the sentence where the anaphor is situated and the two preceding sentences, if available) are passed on to a gender and number agreement test. In English, however, there are certain collective nouns which do not agree in number with their antecedents (e.g. "government", "team", "parliament" etc. can be referred to by "they"; equally some plural nouns such as "data" can be referred to by "it") and are exempted from the agreement test. For this purpose we have drawn up a comprehensive list of all such cases; to our knowledge, no other computational treatment of pronominal anaphora resolution has addressed the problem of "agreement exceptions".

The gender and number agreement of an anaphor and its antecedent in Polish is compulsory. Polish gender distinctions are much more diverse than in English (e.g. feminine and masculine do not apply to a restricted number of nouns). Moreover, one pronominal form can potentially refer to nouns of different gender. For instance, the singular genitive form "jego" can equally well refer to either masculine or neuter nouns. In addition, certain pronouns such as the accusative form "je" can refer to either singular neuter or plural feminine nouns. Finally, unlike English, zero anaphors (in subject position) are typical in Polish in declarative sentences.

Agreement rules in Arabic are different. For instance, a set of non-human items (animals, plants, objects) is referred to by a singular feminine pronoun. Since Arabic is an agglutinative language, the pronouns may appear as suffixes of verbs, nouns (e.g. in the case of possessive pronouns) and prepositions. In particular, in the genre of technical manuals there are five "agglutinative" pronouns. The pronoun "ho" is used to refer to singular masculine persons and

[1] Given that the evaluation of the English version was more extensive, the figures for English are expected to be statistically more representative.

[2] We term the antecedent indicators "multilingual" because they work well not only for English, but also for other languages (in this case Arabic and Polish).

objects, while "ha" refers to singular feminine ones. There are three plural anaphoric pronouns: "homa" which refers to a dual number (a set of two elements) of both masculine and feminine nouns, "hom" which refers to a plural number (a set of more than two elements) of masculine nouns and "honna" which refers to a plural number of feminine

2.2 Antecedent indicators

Antecedent indicators (preferences) play a decisive role in tracking down the antecedent from a set of possible candidates. Candidates could be given preferential treatment, or not, from the point of view of each indicator and assigned a score (-1, 0, 1 or 2) accordingly; the candidate with the highest aggregate score is proposed as the antecedent. The antecedent indicators have been identified on the basis of empirical studies of numerous hand-annotated technical manuals (referential links had been marked by human experts). These indicators can be related to salience (definiteness, givenness, indicating verbs, indicating noun phrases, lexical reiteration, section heading preference, "non-prepositional" noun phrases, relative pronoun), to structural matches (collocation, immediate reference, sequential instructions), to referential distance or to preference of terms. Whilst some of the indicators are more genre-specific (term preference) and others are less genre-specific ("immediate reference", "sequential instructions" and to a much lesser extent "indicating noun phrases"), the majority of them appear to be genre-independent. In the following we shall outline the indicators used and shall illustrate some of them by examples (the indicators are used in the same way for English, Polish and Arabic unless otherwise specified).

Definiteness

Definite noun phrases in previous sentences are more likely antecedents of pronominal anaphors than indefinite ones (definite noun phrases score 0 and indefinite ones are penalised by -1). In English we regard a noun phrase as definite if the head noun is modified by a definite article, or by demonstrative or possessive pronouns. This rule is ignored if there are no definite articles, possessive or demonstrative pronouns in the paragraph (this exception is taken into account because some English user's guides tend to omit articles).

Since in Polish there are no definite articles, definiteness is signalled by word order, demonstrative pronouns or repetition.

In Arabic, definiteness occurs in a richer variety of forms (Galaini 1992). In addition to the definiteness triggered by the definite article "al" (the), demonstrative and possessive pronouns, a noun phrase in Arabic is also regarded as definite if it is followed by a definite noun/noun phrase[3]. For example, the noun phrase "kitabu al-rajuli" (lit. book the man) which means "the book of the man", is considered definite since the non-definite noun "kitabu" (book) is followed by the definite noun "al-rajoli" (the man). This form of definiteness is called in Arabic "Al-ta'rif bi-al-idhafa" (definiteness by addition).

Givenness

Noun phrases in previous sentences representing the "given information" (theme)[4] are deemed good candidates for antecedents and score 1 (candidates not representing the theme score 0). In a coherent text (Firbas 1992), the given or known information, or theme, usually appears first, and thus forms a co-referential link with the preceding text. The new information, or rheme, provides some information

Indicating verbs

If a verb is a member of the Verb_set = {discuss, present, illustrate, identify, summarise, examine, describe, define, show, check, develop, review, report, outline, consider, investigate, explore, assess, analyse, synthesise, study, survey, deal, cover}, we consider the first NP following it as the preferred antecedent (scores 1 and 0). Empirical evidence suggests that because of the salience of the noun phrases which follow them, the verbs listed above are particularly good indicators.

The Verb_set in Polish contains the Polish equivalents of the above verbs and their synonyms.

Indicating noun phrases

If the head of the NP preceding the verb is the noun "chapter", "section", "table" then consider the NP following the verb as the preferred antecedent (scores 1 and 0)

The last two preferences can be illustrated by the example:

> This table shows a minimal configuration$_i$; it$_i$ does not leave much room for additional applications or other software for which you may require additional swap space.

[3]There are other forms of definiteness in Arabic which we shall not discuss in this paper since they are not typical of technical manuals.

[4]We use the simple heuristics that the given information is the first noun phrase in a non-imperative sentence.

Lexical reiteration

Lexically reiterated items are likely candidates for antecedent (a NP scores 2 if is repeated within the same paragraph twice or more, 1 if repeated once and 0 if not). Lexically reiterated items include repeated synonymous noun phrases which may often be preceded by definite articles or demonstratives. Also, a sequence of noun phrases with the same head counts as lexical reiteration (e.g. "toner bottle", "bottle of toner", "the bottle").

Section heading preference

If a noun phrase occurs in the heading of the section, part of which is the current sentence, then we consider it as the preferred candidate (1, 0).

"Non-prepositional" noun phrases

A "pure", "non-prepositional" noun phrase is given a higher preference than a noun phrase which is part of a prepositional phrase (0, -1)

> Insert the cassette$_i$ into the VCR making sure it$_i$ is suitable for the length of recording.

Here "the VCR" is penalised (-1) for being part of the prepositional phrase "into the VCR".

This preference can be explained in terms of salience from the point of view of the centering theory. The latter proposes the ranking "subject, direct object, indirect object" (Brennan et al. 1987) and noun phrases which are parts of prepositional phrases are usually indirect objects.

This criterion was extended in Polish to frequently occurring genitive constructions (e.g. liczba komputerow = number of computers). Nouns which are part of such genitive constructions and which are not in genitive form are penalised by "-1".

In Arabic the antecedent and the anaphor can belong to the same prepositional phrase (see next section). Therefore, we have modified this indicator for the "Arabic version" accordingly: if an NP belongs to a prepositional phrase which doesn't contain the anaphor, we penalise it by -1; otherwise we do not assign any score to it (0).

Relative pronoun indicator

This indicator is used only in the Arabic version and is based on the fact that the first anaphor following a relative pronoun refers exclusively to the most recent NP preceding it which is considered as the most likely antecedent (2,0).

Example:

> Al-tahakkok min tahyiat al-moakkit
> Yomkino-ka a'rdh tahyiat moakkitoka li-at-tahakkok mina al-baramij$_i$ al-lati targhabo fi tasjili-ha$_i$.
> (Literal translation)
> Checking the Timer Settings
> You can display your timer settings to confirm the programmes$_i$ that you wish to recording it$_i$.
> Checking the Timer Settings
> You can display your timer settings to confirm the programmes you wish to record.

In this example the pronoun "ha" (it) is the first pronominal anaphor which follows the relative pronoun "al-lati" (that) and refers to the non-animate feminine plural "al-baramij" (the programmes; for agreement rules in Arabic see section 2.1) which is the most recent NP preceding "al-lati".

Collocation pattern preference

This preference is given to candidates which have an identical collocation pattern with a pronoun (2,0). The collocation preference here is restricted to the pattern "noun/pronoun, verb" or "verb, noun/pronoun" (owing to lack of syntactic information, this preference is somewhat weaker than the collocation preference described in (Dagan & Itai 1990).

> Press the key$_i$ down and turn the volume up... Press it$_i$ again.

The collocation pattern preference in Arabic has been extended to patterns "(un)V-NP/anaphor", i.e. verbs with a "undoing action" meaning are considered for the purpose of our approach to fall into collocation patterns along with their "doing action" counterparts. This extended new rule would help in cases such as "Loading a cassette or unloading it". This rule is soon to be integrated into the English and Polish versions.

Immediate reference

In technical manuals the "immediate reference" clue can often be useful in identifying the antecedent. The heuristics used is that in constructions of the form "...(You) V_1 NP ... con (you) V_2 it (con (you) V_3 it)", where con $\in$ {and/or/before/after...}, the noun phrase immediately after V_1 is a very likely candidate for antecedent of the pronoun "it" immediately following V_2 and is therefore given preference (scores 2 and 0).

This preference can be viewed as a modification of the collocation preference. It is also quite frequent with imperative constructions.

> To print the paper, you can stand the printer$_i$ up or lay it$_i$ flat.

To turn on the printer, press the Power button$_i$ and hold it$_i$ down for a moment.
Unwrap the paper$_j$, form it$_j$ and align it$_j$, then load it$_j$ into the drawer.

Sequential instructions

This new antecedent indicator has recently been incorporated for Arabic but it works equally well for English and is to be implemented in the English version soon as well. It states than in sequential instructions of the form "To V_1 NP_1, V_2 NP_2. (Sentence). To V_3 it, V_4 NP_4" the noun phrase NP_1 is the likely antecedent of the anaphor "it" (NP_1 is assigned a score of 2).

Example:

To turn on the video recorder, press the red button. To programme it, press the "Programme" key.
To turn the TV set ON, press the mains ON/OFF switch. The power indicator illuminates to show that the power is on. To turn the TV set off, press it again.

Referential distance

In English complex sentences, noun phrases in the previous clause[5] are the best candidate for the antecedent of an anaphor in the subsequent clause, followed by noun phrases in the previous sentence, then by nouns situated 2 sentences further back and finally nouns 3 sentences further back (2, 1, 0, -1). For anaphors in simple sentences, noun phrases in the previous sentence are the best candidate for antecedent, followed by noun phrases situated 2 sentences further back and finally nouns 3 sentences further back (1, 0, -1)

Since we found out that in Arabic the anaphor is more likely to refer to the most recent NP, the scoring system for Arabic gives a bonus to such candidates: the most recent NP is assigned a score of 2, the one that precedes it immediately 1 and the rest 0.

Term preference

NPs representing terms in the field are more likely to be the antecedent than NPs which are not terms (score 1 if the NP is a term and 0 if not).

As already mentioned, each of the antecedent indicators assigns a score with a value $\in \{-1, 0, 1, 2\}$. These scores have been determined experimentally on an empirical basis and are constantly being updated. Top symptoms like "lexical reiteration" assign score "2" whereas "non-prepositional" noun phrases are given a negative score of "-1". We should point out that the antecedent indicators are preferences and not absolute factors. There might be cases where one or more of the antecedent indicators do not "point" to the correct antecedent. For instance, in the sentence "Insert the cassette into the VCR$_i$ making sure it$_i$ is turned on", the indicator "non-prepositional noun phrases" would penalise the correct antecedent. When all preferences (antecedent indicators) are taken into account, however, the right antecedent is still very likely to be tracked down - in the above example, the "non-prepositional noun phrases" heuristics (penalty) would be overturned by the "collocational preference" heuristics.

The antecedent indicators have proved to be reasonably efficient in assigning the right antecedent and our results show that for the genre of technical manuals they may be no less accurate than syntax- and centering-based methods (see Mitkov 1998b). The approach described is not dependent on any theories or assumptions; in particular, it does not operate on the assumption that the subject of the previous utterance is the highest-ranking candidate for the backward-looking center - an approach which can sometimes lead to incorrect results. For instance, most centering-orientated methods would propose "the utility" incorrectly as the antecedent of "it" in the sentence "The utility (CDVU) shows you a LIST4250, LIST38PP, or LIST3820 file on your terminal for a format similar to that in which it will be printed" because of the preferential treatment of the subject as the most salient candidate (e.g. RAP, see Dagan et al. 1995). The "indicating verbs" preference of our approach, however, would give preference to the correct antecedent "LIST4250, LIST38PP, or LIST3820 file".

3. Evaluation

For practical reasons, the approach presented does not incorporate syntactic and semantic knowledge (other than a list of domain terms) and it is not realistic to expect its performance to be as good as an approach which makes use of syntactic and constraints and preferences. The lack of syntactic information, for instance, means giving up c-command constraints and subject preference (or on other occasions object preference, see Mitkov 1995) which could be used in center tracking. Syntactic parallelism, useful in discriminating between identical pronouns on the basis of their syntactic function, also has to be forgone. Lack of semantic knowledge rules out the use of verb semantics and semantic parallelism. Our evaluation, however, suggests that much less is lost than might be feared. In fact, our evaluation shows that the results are comparable to and

[5]Identification of clauses in complex sentences is done heuristically.

even better than syntax-based methods (Lappin & Leass 1994). The evaluation results also show superiority over other knowledge-poor methods (Baldwin 1997; see also below)[6]. We believe that the good success rate is due to the fact that a number of antecedent indicators are taken into account and no factor is given absolute preference. In particular, this strategy can often override incorrect decisions linked with strong centering preference (see 2.2) or syntactic and semantic parallelism preferences (Mitkov 1998b).

We have carried out evaluations on sample texts from technical user's guides both for English and Arabic and the results show comparable success rates. The success rate for Arabic is slightly higher and we should mention that in addition to tuning the approach for Arabic, the "Arabic improved" version uses 2 new indicators recently introduced which have not been included in the "Robust English" version yet.

3.1 English

The first evaluation exercise for English (Mitkov & Stys 1997) was based on a random sample text from a technical manual (Minolta 1994). There were 71 pronouns in the 140 page technical manual; 7 of the pronouns were non-anaphoric and 16 exophoric. The resolution of anaphors was carried out with a success rate of 95.8%. The approach being robust (an attempt is made to resolve each anaphor and a proposed antecedent is returned), this figure represents both "precision" and "recall" if we use the MUC terminology. To avoid any terminological confusion, we shall therefore use the more neutral term "success rate" while discussing the evaluation.

We conducted a second evaluation[7] of the robust approach on a different set of English sample texts from the genre of technical manuals (47-page Portable Style-Writer User's Guide (Stylewriter 1994). Out of 223 pronouns in the text, 167 were non-anaphoric (deictic and non-anaphoric "it"). The evaluation carried out was manual to ensure that no added error was generated (e.g. due to possible wrong sentence/clause detection or POS tagging). Another reason for doing it by hand is to ensure a fair comparison with other knowledge-poor methods (Baldwin 1997), which not being available to us, had to be hand-simulated.

The second evaluation indicated an 83.6% success rate for our robust approach. Baldwin's CogNIAC scored 75% on the same data, while J. Hobb's algorithm achieved 71% (Mitkov 1998b).

On the basis of both evaluation experiments a success rate of 89.7% could be regarded as a statistically more representative figure for the performance of "English version" of the robust approach[8]. In addition, our evaluation results indicate 82% "critical success rate", which we consider quite a satisfactory score (for definition of the concept "critical success rate" which is limited to the evaluation of the so-called "critical cases" - the resolution of "tough" anaphors which have already passed the agreement filter, see Mitkov 1998b). Finally, in order to evaluate the effectiveness of the approach and to explore whether or by how much it is superior to the baseline models for anaphora resolution, we also tested the sample texts on (i) a Baseline Model which checks agreement in number and gender and, where more than one candidate remains, picks as antecedent the most recent subject matching the gender and number of the anaphor and (ii) a Baseline Model which picks as antecedent the most recent noun phrase that matches the gender and number of the anaphor. The evaluation results suggest a success rate of 48.55% for the first baseline model and a success rate 65.95% for the second (Mitkov 1998b).

If we regard as "discriminative power" of each antecedent indicator the ratio "number of successful antecedent identifications when this indicator was applied"/"number of applications of this indicator" (for the non-prepositional noun phrase and definiteness being penalising indicators, this figure is calculated as the ratio "number of unsuccessful antecedent identifications"/"number of applications"), the immediate reference emerges as the most discriminative indicator (100%), followed by non-prepositional noun phrase (92.2%), collocation (90.9%), section heading (61.9%), lexical reiteration (58.5%), givenness (49.3%), term preference (35.7%) and referential distance (34.4%). The relatively low figures for the majority of indicators should not be regarded as a surprise: firstly, we should bear in mind that in most cases a candidate was picked (or rejected) as an antecedent on the basis of applying a number of different indicators and secondly, that most anaphors had a relatively high number of candidates for antecedent.

In terms of frequency of use ("number of non-zero applications"/"number of anaphors"), the most frequently used indicator proved to be referential distance used in 98.9% of the cases, followed by term preference (97.8%), givenness (83.3%), lexical reit-

[6] This applies to the genre of technical manuals; for other genres results may be different

[7] We are indebted to Lowenna Ansell for carrying out the second evaluation

[8] Please note that we have recently modified some of the rules/added some more rules but we have not evaluated the improved English version yet.

eration (64.4%), definiteness (40%), section heading (37.8%), immediate reference (31.1%) and collocation (11.1%). As expected, the most frequent indicators were not the most discriminative ones.

3.2 Arabic

We evaluated the robust approach for Arabic operating in two modes: the first mode consisted of using the robust approach directly, without any adaptation/modification for Arabic, whereas the second mode used an adapted/enhanced version which included modified rules (see section 2.2) designed to capture some of the specific aspects of Arabic plus a few new indicators.

The evaluation was based on 63 examples from a technical manual (Sony 1992). The first mode (i.e. using the robust approach without any adaptation for Arabic - this version is referred to as "Arabic direct" in the table below) reported a success rate of 90.5% (57 out of 63 anaphors were correctly resolved). Typical failures were examples in which the antecedent and the anaphor belonged to the same prepositional phrase:

> Tathhar al-surah fi awal kanat$_i$ ta-stakbilo-ha$_i$ fi mintakati-ka.
> Appears the-picture on first channel$_i$ you-receive-it$_i$ in area-your. (Literal translation)
> The picture appears when the first channel received in your area is detected.

Such failure cases were not detected in the improved version for Arabic in which the "non-prepositional phrase" rule was changed (see section 2.2).

Another typical problem which was rectified by changing the referential distance in Arabic was the case in which the anaphor appeared as part of a PP modifying the antecedent-NP:

> Kom bi-taghtiat thokb al-lisan bi-sharit plastic aw ista'mil kasit akhar$_i$ bi-hi$_i$ lisan al-aman.
> Cover slot the-tab with-tape plastic or use cassette another$_i$ in it$_i$ tab the- safety.
> Cover the safety tab slot with plastic tape, or use another cassette with a safety tab.

The candidates for antecedent in this example are the noun phrases "safety tab slot", "plastic tape" and "another cassette". If we use the robust approach without any modification, each candidate gets 2 for referential distance; the aggregate score for "safety tab slot" is 3, for "plastic tape" it is 2 and for "another cassette" is 2 as well (they all get an additional 1 score for "term preference"). Using the new referential distance scores, however, the correct candidate "another cassette" scores an aggregate of 2 as opposed to the other two candidates which are assigned an aggregate score of 1.

The second evaluation mode (evaluating the version adapted and improved for Arabic which is referred to as "Arabic improved" in the table below) reported a success rate of 95.2% (60 out of 63 anaphors were correctly resolved).

The evaluation for Arabic also showed a very high "critical success rate" as well. The robust approach used without any modification scored a "critical success rate" of 78.6%, whereas the improved Arabic version scored 89.3%.

The most discriminative indicators for Arabic proved to be immediate reference, collocation and sequential instructions with 100% discriminative power, followed by non-prepositional noun phrase (89.2%), term preference (82.1%), definiteness (78.6%), referential distance_score_2 (67.9%) and section heading (63.6%). The higher contribution of referential distance for Arabic is in tune with our empirical finding that referential distance is a more important indicator for Arabic than for English and that in particular, the most recent NPs in Arabic are more likely to be antecedents than in English (see section 2.2, indicator "referential distance").

The most frequently used indicators for Arabic were referential distance (100%, of which 34.6% with score 2 and 34.6% with score 1) and term preference (87.7%). Again, the most discriminative indicators could not be frequently used: collocation was applied in only 2.5% of the cases, whereas immediate reference and sequential instructions could be activated in 1.2% of the cases only.

3.3 Polish

The evaluation for Polish was based technical manuals available on the Internet (Internet Manual, 1994; Java Manual 1998). The sample texts contained 180 pronouns among which were 120 instances of exophoric reference (most being zero pronouns). The robust approach adapted for Polish demonstrated a high success rate of 93.3% in resolving anaphors.

Similarly to the evaluation for English, we compared the approach for Polish with (i) a Baseline Model which discounts candidates on the basis of agreement in number and gender and, if there were still competing candidates, selects as the antecedent the most recent subject matching the anaphor in gender and number (ii) a Baseline Model which checks agreement in number and gender and, if there were still more than one candidate left, picks up as the antecedent the most recent noun phrase that agrees with the anaphor.

The Polish version of our robust approach showed clear superiority over both Polish baseline models.

The first Baseline Model (Baseline Subject) was successful in only 23.7% of the cases, whereas the second (Baseline Most Recent) had a success rate of 68.4%. These results demonstrate the dramatic increase in precision, which is due to the use of antecedent tracking indicators.

The Polish version also showed a very high "critical success rate" of 86.2%. Used without any modification ("Polish direct"), the approach scored a 90% success rate.

The most discriminative antecedent indicators for Polish appear to be the sequential instructions, immediate reference and indicating verbs (100%), followed by referential distance (84.1%) and givenness (80 %).

The most frequently used indicators for Polish were definiteness (97.2% of the cases), referential distance (94.4%), givenness (61.1%) and non-prepositional noun phrase (52.8%). The least frequently used indicators proved to be indicating verbs (16.7%), lexical reiteration (13.9%) and immediate reference (2.8%).

The success rates obtained can be summarised as follows:

	Success rate
Robust English	89.7%
Polish direct	90%
Polish improved	93.3%
Arabic direct	90.5%
Arabic improved	95.2%

Table 1: Success rates of the robust approach

	Success rate
Baseline subject English	31.6% / 48.6%
Baseline most recent English	65.9%
Baseline subject Polish	23.7%
Baseline most recent Polish	68.4%

Table 2: Success rates of the baseline models

Since the approach is robust, the success rates equal both recall and precision except for "Baseline subject English": since there are cases in which "Baseline subject" may not be able to pick up an antecedent (e.g. paragraphs with zero subjects), this version can be measured in terms of both precision (the higher figure in table 2) and recall (the lower figure).

4. Future work

Future work includes adapting the approach for French, Spanish and Bulgarian as well as testing it on (and if necessary, modifying it to cover) a wider variety of genres. In addition, we plan to use the statistically-based multicriteria approach (Pomerol & Barbara-Romero, 1992) to fine-tune scoring.

5. Conclusion

We have described a genre-specific modification of the practical approach to pronoun resolution (Mitkov 1998a) and have shown its multilingual nature: we have adapted and tested the approach for Polish and Arabic. The evaluation reports success rates which are comparable to (and even better than) syntax-based methods and show superiority over other methods with limited knowledge.

References

Abraços, Jose & José G. Lopes. 1994. "Extending DRT with a focusing mechanism for pronominal anaphora and ellipsis resolution". *Proceedings of the 15th International Conference on Computational Linguistics (COLING'94)*, 1128-1132, Kyoto, Japan.

Baldwin, Breck. 1997. "CogNIAC: high precision coreference with limited knowledge and linguistic resources". *Proceedings of the ACL'97/EACL'97 workshop on Operational factors in practical, robust anaphora resolution*, 38-45, Madrid, Spain.

Brennan, S., M. Fridman and C. Pollard. 1987. A centering approach to pronouns. *Proceedings of the 25th Annual Meeting of the ACL (ACL'87)*, 155-162. Stanford, CA, USA.

Carbonell, James G. & Ralf D. Brown. 1988. "Anaphora resolution: a multi-strategy approach". *Proceedings of the 12. International Conference on Computational Linguistics (COLING'88)*, Vol.I, 96-101, Budapest, Hungary.

Carter, David M. 1987. *Interpreting anaphora in natural language texts*. Chichester: Ellis Horwood

Connoly, Dennis, John D. Burger & David S. Day. 1994. "A Machine learning approach to anaphoric reference". *Proceedings of the International Conference "New Methods in Language Processing"*, 255-261, Manchester, United Kingdom.

Dagan, Ido & Alon Itai. 1990. "Automatic processing of large corpora for the resolution of anaphora references". *Proceedings of the 13theInternational Conference on Computational Linguistics (COLING'90)*, Vol. III, 1-3, Helsinki, Finland.

Dagan, Ido, John Justeson, Shalom Lappin, Herbert Leass & Amnon Ribak. 1995. Syntax and lexical statistics in anaphora resolution. *Applied Artificial Intelligence*, 9.

Dunker, Guido & Carla Umbach. 1993. *Verfahren zur Anaphernresolution in KIT-FAST*. Internal Report KIT-28, Technical University of Berlin.

Firbas, Jan. 1992. *Functional sentence perspective in written and spoken communication*. Cambridge: Cambridge University Press.

Fischer, Ingrid, Bernd Geistert & Günter Görz 1996. "Incremental anaphora resolution in a chart-based semantics construction framework using l-DRT". *Proceedings of the International Colloquium on Discourse Anaphora and Anaphora Resolution. Lancaster (DAARC)*, 235-244, Lancaster, UK.

Fraurud, Kari. 1988. "Pronoun Resolution in unrestricted text". *Nordic Journal of Linguistics* 11, 47-68

Galaini, Chikh Mustafa. 1992. *Jami'u al-durus al-arabiah* (Arabic lessons collection). Beirut: Manshurat al-maktabah al-asriyah (Modern library).

Hasan, Abbes. 1975. *Al-nahw al-wafi ma'a rabtihi bi-al-asalib al-rafiah wa al-hayah al-loghawiah al-mutajadidah* (Complete grammar referring to good styles and the changing language). Egypt: Dar al-ma'arif (Knowledge bookstore).

Hobbs, Jerry R. 1978 "Resolving pronoun references". *Lingua*, 44, 339-352.

Ingria, Robert J.P. & David Stallard. 1989. "A computational mechanism for pronominal reference". *Proceedings of the 27th Annual Meeting of the ACL*, 262-271, Vancouver, British Columbia.

Internet Manual. 1994. *Translation of Internet Manual Internet i okolice: Przewodnik po swiatowych sieciach komputerowych*. Tracy LaQuey, Jeanne C. Ryer Translated by Monika Zielinska, BIZNET Poland.

Java Manual. 1998. *Jezyk Java*. Clico, Krakow.

Kennedy, Christopher & Branimir Boguraev, 1996. "Anaphora for everyone: pronominal anaphora resolution without a parser". *Proceedings of the 16th International Conference on Computational Linguistics (COLING'96)*, 113-118. Copenhagen, Denmark.

Lappin, Shalom & Michael McCord. 1990. "Anaphora resolution in slot grammar". *Computational Linguistics*, 16:4, 197-212.

Lappin, Shalom & Herbert Leass. 1994. "An algorithm for pronominal anaphora resolution". *Computational Linguistics*, 20(4), 535-561.

Leass Herbert & Ulrike Schwall. 1991. *An anaphora resolution procedure for machine translation*. IBM Germany Science Center. Institute for Knowledge Based Systems, Report 172.

Minolta. 1994. *Minolta Operator's Manual for Photocopier EP5325*. Technical Manual Minolta Camera Co., Ltd., Business Equipment Division 3-13, 2-Chome, Azuchi, -Machi, Chuo-Ku, Osaka 541, Japan.

Mitkov, Ruslan. 1994. "An integrated model for anaphora resolution". *Proceedings of the 15th International Conference on Computational Linguistics (COLING'94)*, 1170-1176, Kyoto, Japan.

Mitkov, Ruslan. 1995. "Un uncertainty reasoning approach for anaphora resolution". *Proceedings of the Natural Language Processing Pacific Rim Symposium (NLPRS'95)*, 149-154, Seoul, Korea.

Mitkov, Ruslan. 1998a. "Pronoun resolution: the practical alternative". In T. McEnery, S. Botley(Eds) *Discourse Anaphora and Anaphor Resolution*. John Benjamins.

Mitkov, Ruslan. 1998b. "Evaluating anaphora resolution approaches" (forthcoming).

Mitkov, Ruslan & Malgorzata Stys. 1997. "Robust reference resolution with limited knowledge: high precision genre-specific approach for English and Polish". *Proceedings of the International Conference "Recent Advances in Natural Language Proceeding" (RANLP'97)*, 74-81. Tzigov Chark, Bulgaria.

Mori, Tatsunori, Mamoru Matsuo, Hiroshi Nakagawa. 1997. Constraints and defaults of zero pronouns in Japanese instruction manuals. *Proceedings of the ACL'97/EACL'97 workshop on Operational factors in practical, robust anaphora resolution*, 7-13. Madrid, Spain.

Nakaiwa, Hiromi & Satoru, Ikehara. 1992. " Zero pronoun resolution in a Japanese-to-English Machine Translation system by using verbal semantic attributes". *Proceedings of 3rd Conference on Applied Natural Language Processing (ANLP'92)*, Trento, Italy.

Nakaiwa, Hiromi & Satoru, Ikehara. 1995. "Intrasentential resolution of Japanese zero pronouns in a Machine Translation system using semantic and pragmatic constraints". *Proceeedings of the International Conference on Theoretical and Methodological Issues in Machine Translation (TMI'95)*, 96-105, Leuven, Belgium.

Nakaiwa, Hiromi, S. Shirai, Satoru Ikehara & T. Kawaoka. 1995. "Extrasentential resolution of Japanese zero pronouns using semantic and pragmatic constraints". *Proceedings of the AAAI 1995 Spring Symposium Series: Empirical methods in discourse interpretation and generation.*

Nakaiwa, Hiromi & Francis Bond, Takahiro Uekado & Yayoi Nozawa. 1996. "Resolving zero pronouns in texts using textual structure". *Proceedings of the International Conference "New Methods in Language Processing" (NeMLaP-2)*, Ankara, Turkey.

Nasukawa, Tetsuya. 1994. "Robust method of pronoun resolution using full-text information". *Proceedings of the 15th International Conference on Computational Linguistics (COLING'94)*, 1157-1163, Kyoto, Japan.

Pomerol, Jean-Charles & Sergio Barbara-Romero. 1992. *Choix multicritère dans l'entreprise: principes et pratique*. Paris: HERMES.

Popescu-Belis, Andrei & Isabelle Robba. 1997. "Cooperation between pronoun and reference resolution for unrestricted texts". *Proceedings of the ACL'97/EACL'97 workshop on Operational factors in practical, robust anaphora resolution*, 30-37. Madrid, Spain.

Rich, Elaine & Susann LuperFoy. 1988. "An architecture for anaphora resolution". *Proceedings of the Second Conference on Applied Natural Language Processing (ANLP-2)*, 18-24, Texas, U.S.A.

Rolbert, Monique. 1989. *Résolution de formes pronominales dans l'interface d'interogation d'une base de données*. Thèse de doctorat. Faculté des sciences de Luminy.

Sidner, Candy L. 1979. *Towards a computational theory of definite anaphora comprehension in English discourse*. Technical Report No. 537. M.I.T., Artificial Intelligence Laboratory.

Sony. 1992. *Video cassette recorder*. Operating Instructions. Sony Corporation.

Stuckardt, Roland. 1996. "An interdependency-sensitive approach to anaphor resolution". *Proceedings of the International Colloquium on Discourse Anaphora and*

Anaphora Resolution. Lancaster (DAARC), 400-413. Lancaster, UK.

Stuckardt, Roland. 1997. "Resolving anaphoric references on deficient syntactic descriptions". *Proceedings of the ACL'97/EACL'97 workshop on Operational factors in practical, robust anaphora resolution*, 30-37. Madrid, Spain.

Stylewriter 1994. *Portable StyleWriter*. User's guide. Apple Computers.

Tin, Erkan & Varol, Akman. 1994. "Situated processing of pronominal anaphora". *Proceedings of the KONVENS"94 Conference*, 369-378, Vienna, Austria.

Wakao, Takahiro. 1994. "Reference resolution using semantic patterns in Japanese newspaper articles". *Proceedings of the 15th International Conference on Computational Linguistics (COLING'94)*, 1133-1137. Kyoto, Japan.

Webber, Bonnie L. 1979. *A formal approach to discourse anaphora*. London: Garland Publishing.

Williams, Sandra, Mark Harvey & Keith Preston. 1996. "Rule-based reference resolution for unrestricted text using part-of-speech tagging and noun phrase parsing". *Proceedings of the International Colloquium on Discourse Anaphora and Anaphora Resolution (DAARC)*, 441-456. Lancaster, UK.

Aligning Clauses in Parallel Texts

Sotiris Boutsis and **Stelios Piperidis**
Institute for Language and Speech Processing - ILSP
Artemidos & Epidavrou 151-25, Athens, Greece
tel:+301 6800959, fax:+301 6854270
email: {sboutsis, spip}@ilsp.gr

National Technical University of Athens - NTUA

Abstract

This paper describes a method for the automatic alignment of parallel texts at clause level. The method features statistical techniques coupled with shallow linguistic processing. It presupposes a parallel bilingual corpus and identifies alignments between the clauses of the source and target language sides of the corpus. Parallel texts are first statistically aligned at sentence level and then tagged with their part-of-speech categories. Regular grammars functioning on tags, recognize clauses on both sides of the parallel text. A probabilistic model is applied next, operating on the basis of word occurrence and co-occurrence probabilities and character lengths. Depending on sentence size, possible alignments are fed into a dynamic programming framework or a simulated annealing system in order to find or approximate the best alignment. The method has been tested on a small English-Greek corpus consisting of texts relevant to software systems and has produced promising results in terms of correctly identified clause alignments.

Introduction

The availability of large collections of texts in electronic form, has given rise to a wide range of applications aiming at the elicitation of linguistic resources such as translation dictionaries, transfer grammars and retrieval of translation examples (Dagan et al., 1991; Matsumoto et al., 1993), or even the building of fully-blown machine translation systems (Brown et al., 1990). The purpose of this paper is to describe a technique for extracting translation correspondences at bellow sentence level by employing statistical techniques coupled with shallow linguistic processing catering for the segmentation of sentences into clauses.

Statistical processing has proved powerful for the extraction of translation equivalences at sentence and intra-sentence level. Brown et al. (1991) described a method based on the number of words contained in sentences. The general idea is that the closer in length two sentences are, the most likely they are to align. Moreover, certain anchor points and paragraph markers are considered. Dynamic programming and HMMs are pipelined to produce alignments at sentence level. The method has been applied to the Hansard-Corpus, achieving an accuracy of 96%-97%. Gale and Church (1991) proposed a method that relies on a simple statistical model of character lengths. The model is based on the observation that longer sentences in one language tend to be translated into longer sentences in the other language while shorter ones tend to be translated into shorter ones. A probabilistic score is assigned to each pair of proposed sentence pairs, and a dynamic programming framework calculates the most probable alignment. Although the apparent efficacy of the Gale-Church algorithm is undeniable and validated on different pairs of languages, it faces problems when handling complex alignments(1-0, 1-2, 2-2).

Simard et al. (1992) argue that a small amount of linguistic information is necessary in order to overcome the inherited weaknesses of the purely statistical techniques. They proposed using cognates, which are pairs of tokens of different languages sharing "obvious" phonological or orthographic and semantic properties, since these are likely to be used as mutual translations. Papageorgiou et al. (1994) proposed a generic alignment scheme invoking surface linguistic information coupled with information about possible unit delimiters depending on the level at which alignment is sought. Each unit, sentence, clause or phrase, is represented by the sum of its content part of speech (POS) tags. The results are then fed into a dynamic programming framework that computes the optimum alignment of text units.

Brown (1988) uses a probabilistic measure to estimate word similarity of two languages in the context of statistically-based machine translation. Kay and Roescheisen (1993) present an algorithm for aligning bilingual texts on the basis of internal evidence. Processing is performed in many iterations and each new iteration uses the results of the previous one in order to calculate more accurate word and sentence correspondences. In

each iteration, processing consists of calculating correspondences between sentences on the basis of their relative positions, and then calculating word correspondences on the basis of word co-occurrences in related sentences. The Dice coefficient is used as the similarity measure between words of two languages in an attempt to secure the correctness of the alignment of parallel texts at sentence level. Kitamura and Matsumoto (1995) have used the same Dice coefficient to calculate the word similarity between Japanese-English parallel corpora. Single word correspondences have also been investigated by Gale and Church (1991b) using a statistical evaluation of contingency tables. Piperidis et al. (1997) and Boutsis and Piperidis (1996) describe methods for extracting single and multi-word equivalences based on a parallel corpus statistically aligned at sentence level and employing a similarity metric along the lines of the Dice coefficient with comparable performance.

Collocational correspondences have been studied by Smadja (1992) and Smadja et al. (1996), in an attempt to find translation patterns for continuous and discontinuous collocations in English and French. Meaningful collocations are first extracted in the source language while their corresponding French ones are found by calculating the mutual information between instances of the English collocation and various single word candidates in English-French aligned corpora. Recent work has broadened the scope identifying correspondences between word sequences. Kupiec (1993) proposes a method for extracting translation patterns of noun phrases from English-French parallel corpora. The corpus is tagged at part-of-speech (POS) level and then finite-state recognizers specified by regular expressions defined in terms of POS categories detect noun phrases on either side. Probabilities of correspondences are then calculated using an iterative EM-like algorithm. Kumano and Hirakawa (1994) presuppose an ordinary bilingual dictionary and non-parallel corpora, attempting to find bilingual correspondences in a Japanese-English setting at word, noun phrase and unknown word level. Extending previous work, Kitamura and Matsumoto (1996) apply the Dice coefficient on word sequence correspondence extraction.

This paper describes a method for the automatic alignment of parallel texts at clause level. Texts are first aligned at sentence level using statistical techniques. Part-of-speech tagging takes place next annotating each word form with the appropriate part of speech. Processing in this step and the next one is monolingual, so each language side of the text is treated independently of the other. Surface syntactic analysis is performed next on the basis of regular grammars. Shallow parsing results in the recognition of clauses. Statistical processing follows taking into account different sources of information, aiming at identifying intra-sentence alignments formed by the clauses of the parallel sentences of the bitext. The method caters for alignments of type 1-0, 1-1, 1-2, 2-1, and 2-2. A first pass through the text computes occurrence and co-occurrence probabilities for content words on both language sides. A probabilistic score, expressing the probability that a clause (or a pair of clauses) of the source language is translated into a clause (or a pair of clauses) of the target language, is computed on the basis of the previously calculated word probabilities, and a model of character lengths. Possible clause alignments are examined by a dynamic programming framework deciding on the best alignment. Avoiding combinatorial explosion requires that large sentences be channeled into a module that approximates the optimal alignment through simulated annealing, operating in polynomial time. EM iterative training caters for the estimation of the model's parameters, given the lack of hand-aligned training material. The overview of the processing is pictured in Figure 1.

Test Corpus

The corpus used to develop and test the proposed algorithms consists of text from the HP-VUE software platform documentation set. The Greek text contains 35726 wordforms and the English text 28872. The number of different words is 4512 for the Greek text and 3219 for the English text. The richer morphology of the Greek language accounts for the approximately 30% difference between these two figures.

Text Handling

Recognizing and labeling surface phenomena in the text is a necessary prerequisite for most Natural Language Processing (NLP) systems. In order to be able to make full use of the corpus, texts should be rendered in an appropriate form. To this end, parallel texts are normalized and handled. In the framework of the presented method, basic text handling is performed with the use of a Multext-like tokeniser, (Di Christo et al., 1995). Identification of word boundaries, sentence boundaries, abbreviations etc. takes place. Following common practice, the tokeniser makes use of a regular-expression based definition of words, coupled with downstream precompiled lists for the Greek and English language and simple heuristics. This proves to be quite successful in recognizing sentences and words effectively.

Sentence Alignment

Alignment consists in establishing correspondence links between units in a bilingual text. At this stage, the method aligns input text at sentence level. Processing caters for sentence substitution (one sentence translates into one), deletion (a sentence is not translated at all), insertion (a sentence with no equivalent in the source text

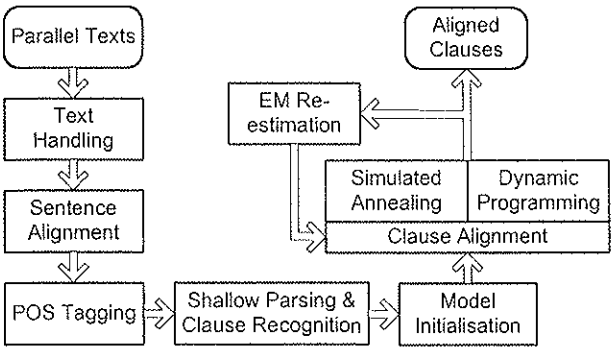


Figure 1: Processing Overview

is introduced by the translator), contraction (two consecutive sentences translate into one), expansion (one sentence translates into two) and merging (two sentences translate jointly into two).

The heart of the alignment scheme, employed at this stage, is a method for aligning sentences based on a simple statistical model of character lengths, (Gale and Church, 1991). The method relies on the assumption that longer sentences in the source language tend to be translated into longer sentences in the target and vice-versa. A probabilistic score is assigned to each pair of proposed sentence pairs, based on the ratio of lengths of the sentences and the variance of this ratio. This probabilistic score is used in a dynamic programming framework in order to find the maximum likelihood alignment of sentences. Additionally, following (Brown et al., 1991) certain points of the texts can be anchored thus dividing them into smaller sections that need to be aligned. Besides anchors, paragraph markers are also considered. Anchor points are specific to the text to be aligned and they usually appear in both texts. They are divided into major and minor anchors and alignment proceeds in two steps, first aligning major anchor points and then minor anchor points, followed by sentence alignment. The alignment algorithm has been tested in the setting of a multilingual text processing system and has been reported to yield accuracy between 96% and 100%, (Piperidis, 1995).

Part of speech tagging

Both English and Greek texts are analyzed morphosyntactically. The words in the parallel sentences are tagged with their corresponding POS categories. The corpus is thus represented as a bitext of tagged mutual sentence translations where every word is accompanied by its corresponding POS tag.

For Greek

Tagging with part-of-speech information for Greek takes place in two steps. First, each word is endowed with all possible tags through lexicon lookup, and then a disambiguation module decides on the most probable annotation.

Lexicon lookup operates on a morphological lexicon of modern Greek. It endows the words of the text with the characteristics found in the lexicon. The tagset used has been devised for the morphological annotation of Greek corpora and conforms to the guidelines set up by EAGLES and PAROLE, trying, at the same time, to capture the morphological peculiarities of the Greek language.

Text produced at the output of lexicon lookup is annotated with below POS information i.e. subcategorisation information for each POS category. Each wordform recognised as noun, for example, is annotated for case, number, gender etc. Ambiguous wordforms are endowed with all possible annotations. However, not all available morphological information is necessary for later processing. In addition, wordforms grammatically fully characterized with below POS information are highly ambiguous. Retaining all such information would impose a heavy burden on the disambiguation process. Experimentation has proved that performance of next stages is not seriously affected by reducing the tagset. To this end, a simplified tagset has been used helping reduce ambiguous wordforms notably. In addition, words not found in the lexicon are assigned possible tags on the basis of a probabilistic model operating on word suffixes. In case of multiple tagging, a disambiguator based on trigrams and contextual rules trained on Greek texts, suggests the tag that is most likely to be the correct, (Papageorgiou, 1996). This stage produces around 95% correct results.

For English

Tagging for English is based on mainstream statistical processing. A tagger implementing hidden markov model techniques is employed. The tagger has been trained on a large preannotated text collection and is then used to tag the HP-VUE test corpus. For training purposes, a set of technical texts annotated at POS level, drawn from the British National Corpus (BNC), has been used, (Burnard, 1995). Texts classified under the field codes: "Written: Domain: Informative: Natural and pure sciences" and "Written: Domain: Informative: Applied Science" have been selected. The size of the text collection is ca. 5,000,000 words. Text is annotated with POS tags according to the BNC tagset (Leech, 1995). This text collection is used to train the Acquilex HMM tagger (Elworthy, 1997) and estimate model parameters. After training, the HP-VUE corpus is tagged by application of the Viterbi algorithm.

Clause recognition

This stage, like the previous one, processes each language side of the text independently of the other. It aims at breaking sentences of both languages into clauses with well-defined boundaries.

In order to recognise clauses, this stage takes advantage of a shallow parser equipped with grammars for Greek and English. Syntactic analysis consists of parsing via finite state automata. Under this approach, a text can be analysed syntactically on the basis of grammars containing non-recursive rules written in the form of regular expressions. Rules are numbered in order to be applied in a certain order. The grammar is translated into finite-state automata with standard techniques (Aho et al., 1986) and automata are connected in a pipeline in order to form a cascade, which is used to annotate text in an incremental way. Each rule (regular expression) describes a specific phenomenon and higher-order rules can be expressed on the basis of the already described ones. Rules are designed to be reliable when they are applied using longest match, in order to avoid the need for disambiguation between different length instances of the same constituent type.

A basic characteristic of this method is that parsing is deterministic and no backtracking takes place. No ambiguity is produced since each automaton takes a definite decision about a constituent's existence or non-existence. This doesn't mean that ambiguities are resolved but that they are enclosed inside syntactic chunks, whose boundaries have been recognised, although their internal structure may have not been decided. Enclosure of ambiguity helps generate only one partial parse for each sentence, since ambiguity is kept local and does not cause the production of multiple parses for the whole sentence.

It should be noted that the method does not depend on the exact method adopted for clause recognision. Another system performing clause recognition could be used instead. This has also to do with the availability of the relevant linguistic processing modules. On the other hand, being aware of the complete partial parse can be very useful, if one is up to extend the method to cover other types of sub-sentence alignments (e.g. alignment of np's). It is also significant that the additional processing of shallow parsing does not impose serious speed overheads since the speed of analysis is measured in tens of hundreds of words/second. Clause boundaries for each analysed sentence are channelled into the next stages of processing. No distinction is made between different clause types. A sample output of this stage is shown in Figure 2.

[cl SEVERAL UTILITIES HELP YOU cl] [cl DIAGNOSE CONFIGURATION AND DATABASE ERRORS cl]

[cl ΠΟΛΛΑ ΒΟΗΘΗΤΙΚΑ ΠΡΟΓΡΑΜΜΑΤΑ ΒΟΗΘΟΥΝ cl] [cl ΝΑ ΔΙΑΓΝΩΣΕΤΕ ΣΦΑΛΜΑΤΑ ΔΙΑΜΟΡΦΩΣΗΣ ΚΑΙ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ cl]

[cl IF YOUR SYSTEM IS PROPERLY CONFIGURED cl] [cl TO AUTOMATICALLY RUN HP VUE cl] , [cl YOU WILL SEE THE HP VUE LOGIN SCREEN cl] [cl WHEN YOUR SYSTEM IS BOOTED cl]

[cl ΑΝ ΤΟ ΣΥΣΤΗΜΑ ΣΑΣ ΕΙΝΑΙ ΣΩΣΤΑ ΔΙΑΜΟΡΦΩΜΕΝΟ cl] [cl ΓΙΑ ΝΑ ΕΚΤΕΛΕΙ ΑΥΤΟΜΑΤΑ ΤΟ HP VUE cl] [cl ΘΑ ΔΕΙΤΕ ΤΗΝ ΟΘΟΝΗ ΣΥΝΔΕΣΗΣ ΤΟΥ HP VUE cl] [cl ΟΤΑΝ ΤΟ ΣΥΣΤΗΜΑ ΣΑΣ ΕΚΚΙΝΕΙ cl]

[cl IF YOU HAVE NO CONSOLE cl] , [cl YOU MUST LOG IN FROM A REMOTE SYSTEM cl]

[cl ΑΝ ΔΕΝ ΥΠΑΡΧΕΙ cl] [cl ΠΡΕΠΕΙ ΝΑ ΕΙΣΕΛΘΕΤΕ ΑΠΟ ΕΝΑ ΑΠΟΜΑΚΡΥΣΜΕΝΟ ΣΥΣΤΗΜΑ cl]

Figure 2: Parallel text with marked clause boundaries

Translation model

Part a

In this section we present the basic translation model, which is used for the purposes of clause alignment. Let's consider two corresponding sentences of the parallel text which are translations of each other, the source sentence $\underline{S_i} = \underline{sc_{i1}}\ \underline{sc_{i2}} \ldots \underline{sc_{il}}$ and its translation into the target language $\underline{T_i} = \underline{tc_{i1}}\ \underline{tc_{i2}} \ldots \underline{tc_{im}}$ where $\underline{sc_i}$ and $\underline{tc_i}$ are clauses identified during the previous stage. We approximate sentence translation with the assumption that clauses can be translated from the source into the target language in the following ways:

A. 1-0 and 0-1, when a clause of the source or the target sentence has no equivalent clause in the other language.
B. 1-1, when a clause of the source sentence is translated into one clause of the target sentence.
C. 1-2 and 2-1, when a clause of the source is translated into two clauses of the target or two clauses of the source translate into one of the target.
D. 2-2, when two clauses jointly translate into two clauses of the other language.

We view each group of aligned sentences of the parallel text as a sequence of clause-beads (after sentence-beads in (Brown et al., 1991)) where a bead accounts for a group of clauses that align with each other according to one of the above mentioned ways. A clause-alignment

$A_i = \{ a_{i1}\ a_{i2} \ldots a_{in} \}$ for a given pair i of sentences is a set of clause-beads a_{ij} covering all clauses of the source and target sentence under the condition that each clause participates to one and only one clause-bead. Figure 3 shows a schematic example of a clause-alignment between two sentences containing four and three clauses each. Making the assumption that translation of clauses in a bead is independent of clauses belonging to other beads we seek the alignment that maximises the joint distribution:

$$\Pr(\underline{S_i}, \underline{T_i}, A_i) = \Pr(n) \prod_{j=1}^{n} \Pr(a_{ij}) \qquad (1)$$

and assuming that $\Pr(n)$ (where n is the number of beads in the alignment) is independent of S_i, T_i and n we get:

$$\Pr(\underline{S_i}, \underline{T_i}, A_i) = \varepsilon \prod_{j=1}^{n} \Pr(a_{ij}) \qquad (2)$$

ε is ignored for the rest of the analysis, since it is a multiplicative constant factor having the same value for all clause-alignments.

Part b

Finding the correct alignment requires that we estimate clause-bead probabilities $\Pr(a_{ij})$ which express the probability for the source sentence clauses of the bead to be translated into the corresponding target sentence clauses. We consider a 1-1 bead covering the source and target clauses:

$\underline{sc_{is}} = sw_{is1}\ sw_{is2} \ldots sw_{isp}$ and

$\underline{tc_{it}} = tw_{it1}\ tw_{it2} \ldots tw_{itq}$

(where sw_{isp} is the pth word of the sth clause of the ith source sentence of the parallel text etc.) A first writing of $\Pr(a_{ij})$ can be as follows:

$$\Pr(a_{ij}) = P_{1-1}\ \Pr(\underline{sc_{is}}, \underline{tc_{it}}) \qquad (3)$$

where P_{1-1} is the probability of a '1-1' clause alignment. Referring to the second factor of (3), in order to approximate $\Pr(\underline{sc_{is}}, \underline{tc_{it}})$ we take into account two parameters: a) the length of the source and target clauses and b) the source language and target language words contained in $\underline{sc_{is}}$ and $\underline{tc_{it}}$. We model the probability that source text with character length $l(\underline{sc_{is}})$ is translated into target text with length $l(\underline{tc_{it}})$ with a distribution $\Pr\left(l(\underline{sc_{is}}), l(\underline{tc_{it}})\right)$. Under the assumption that the model used by the sentence aligner ("Sentence Alignment" section , (Gale and Church, 1991)) expressing sentence alignment probabilities on the basis of character lengths is valid when applied to clause-lengths, we estimate $\Pr\left(l(\underline{sc_{is}})\, l(\underline{tc_{it}})\right)$ with the same model.

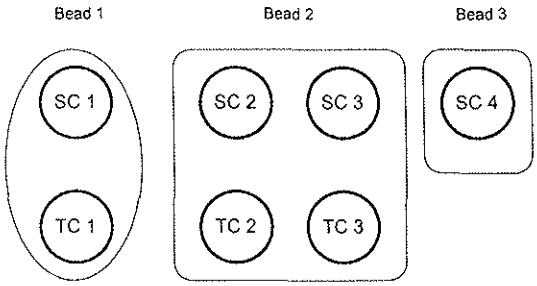


Figure 3: An alignment with three beads
(SC:Source sentence Clause
TC:Target sentence Clause)

Furthermore, we approximate clauses by unordered sets focusing on content carrying words i.e. content words, which are taken to be verbs, nouns, adjectives and adverbs. Thus, we assume that content words contribute the most to the examined probability. $\underline{tc_{it}}$ and $\underline{sc_{is}}$ are represented by the unordered sets of the content words they contain. Following that, equation (3) can be written as:

$$\Pr(a_{ij}) = P_{1-1} \cdot \Pr\left(l(\underline{sc_{is}}), l(\underline{tc_{it}})\right) \cdot \Pr\left(\{scw_{is1}, \ldots, scw_{isv}\}, \{tcw_{it1}, \ldots, tcw_{itw}\}\right) \qquad (4)$$

where scw stands for source clause content word and tcw stands for target clause content word. To approximate the third factor of Eq. (4) we assume that the content words of the source clause are independent events and the same is valid for the words of the target clause. That is:

$$\Pr(\{scw_{is1}, scw_{is2}, \ldots, scw_{isv}\}) = \Pr(scw_{is1})\ \Pr(scw_{is2}) \ldots \Pr(scw_{isv}) \qquad (5)$$

$$\Pr(\{tcw_{it1}, tcw_{it2}, \ldots, tcw_{itw}\}) = \Pr(tcw_{it1})\ \Pr(tcw_{it2}) \ldots \Pr(tcw_{itw}) \qquad (6)$$

Under this model each word of the target clause depends on zero or one word of the source clause. To il-

lustrate, let's consider the source clause $\underline{sc} = \{ scw_1, scw_2, scw_3 \}$ the target clause $\underline{tc} = \{ tcw_1, tcw_2, tcw_3 \}$ and a word alignment W_j so that tcw_1 depends on scw_1, tcw_2 depends on scw_2 while tcw_3 and scw_3 are independent events. In this case,

$$\Pr_{W_j}(\{ scw_1, scw_2, scw_3 \},\{ tcw_1, tcw_2, tcw_3 \}) =$$

$$\Pr(tcw_1, scw_1)\Pr(tcw_2, scw_2)\Pr(tcw_3)\Pr(scw_3) \quad (7)$$

given the computation of Figure 4.

Consequently, when estimating bead probability $\Pr(a_{ij})$, we need to sum probabilities over all possible word alignments W_j. This would require however to inspect an exponentially large set of possible word-alignments. Thus, we would like to approximate the sum with its biggest term. This is not feasible, either. So, a greedy-like technique is followed, which does not guarantee to find the best word alignment but usually comes up with a big enough value to distinguish between good and not so good clause alignments. The largest word-pair probabilities are selected first while probabilities of any unmatched words are taken into account next. In order to select a pair of words for Eq. (7) two heuristic conditions must be met: 1) the occurrence frequencies of the two words should not differ more than 50%, 2) their co-occurrence frequency in the bitext should not differ more than 50% from their occurrence frequencies in the texts.

In case of a non '1-1' alignment between clauses, the same model is used, where P_{1-1} is substituted by P_{1-2}, P_{2-1}, P_{2-2}, P_{1-0} and P_{0-1}. We take $P_{1-2} = P_{2-1}$ and $P_{1-0} = P_{0-1}$. The distribution on character lengths is also taken to be independent of the alignment type.

Model Training

In order to calculate clause-alignment probabilities, given the model presented in the previous section, estimations for several model parameters should be available. At this stage, parameters are estimated on the basis of simple corpus statistics. The probability of a single word of the source or target text is taken to be:

$$\Pr(w) = \frac{f(w)}{\sum_{w'} f(w')} \quad (8)$$

where the denominator of Eq. (8) is the sum of the frequencies of all words i.e. the length of the source or the target text in words. Correspondingly, the probability relating a word of the source text with a word of the target text is estimated by:

$$\Pr(sw, tw) = \frac{f(sw, tw)}{\sum_{(sw', tw')} f(sw', tw')} \quad (9)$$

For the presented application of the method, these probabilities are computed over the whole corpus. In very large texts it is adequate to estimate the probabilities in a representative large portion of the text. It would be also possible to use pre-computed probabilities from another text of the same domain, given that both texts share

$$\Pr_{W_j}(\{ scw_1, scw_2, scw_3 \},\{ tcw_1, tcw_2, tcw_3 \}) =$$

$$\Pr_{W_j}(\{ tcw_1, tcw_2, tcw_3 \} | \{ scw_1, scw_2, scw_3 \}) \Pr(\{ scw_1, scw_2, scw_3 \}) = \quad \textit{(Eq.(5), (6))}$$

$$\Pr_{W_j}(tcw_1 | \{ scw_1, scw_2, scw_3 \}) \Pr_{W_j}(tcw_2 | \{ scw_1, scw_2, scw_3 \}) \Pr_{W_j}(tcw_3 | \{ scw_1, scw_2, scw_3 \}) \cdot$$

$$\Pr(scw_1)\Pr(scw_2)\Pr(scw_3) =$$

$$\Pr(tcw_1 | scw_1)\Pr(tcw_2 | scw_2)\Pr(tcw_3)\Pr(scw_1)\Pr(scw_2)\Pr(scw_3) =$$

$$\frac{\Pr(tcw_1, scw_1)}{\Pr(scw_1)} \frac{\Pr(tcw_2, scw_2)}{\Pr(scw_2)} \Pr(tcw_3)\Pr(scw_1)\Pr(scw_2)\Pr(scw_3) =$$

$$\Pr(tcw_1, scw_1)\Pr(tcw_2, scw_2)\Pr(tcw_3)\Pr(scw_3)$$

Figure 4: Computation of $\Pr_{W_j}(\{ scw_1, scw_2, scw_3 \},\{ tcw_1, tcw_2, tcw_3 \})$

the same characteristics with respect to language use, coverage and translation.

Estimating $P_{1\text{-}1}, P_{1\text{-}2}, P_{2\text{-}2}$ and $P_{0\text{-}1}$ is less straightforward. Given the lack of training material, that is marked-up text aligned at clause level, no safe set of values can be computed for these parameters. To work around this, we first make an educated guess and then apply the EM (Expectation-Maximization) algorithm. The EM algorithm consists of two major steps: an expectation step followed by a maximization step. The expectation uses the current estimates of the parameters to process input data and the maximization provides next a new estimate of these parameters. These two steps iterate until convergence. EM is not guaranteed to converge to a global maximum; if many points of local convergence exist, the point where the method will convergence will depend on the initial parameter estimations. The initial parameter values we used and the estimated ones after the process converged are displayed in the Table 1.

If an alignment type does not occur in the output ('1-0' alignment in this case), the relevant probability takes a very small value (1E-4).

Best Clause-Alignment Selection

This stage aims at finding the best alignment between the clauses of two parallel sentences (or in the case of a non '1-1' sentence alignment e.g. '1-2', an alignment is sought between the clauses of the source sentence and the clauses of the two target sentences). Two schemes are considered, dynamic programming and simulated annealing.

Dynamic programming is a generalization of the greedy technique. It can be used to solve problems, whose solutions can be considered as a sequence of decisions. Usually dynamic programming is used to address an optimization problem, seeking the sequence of decisions giving the optimal solution. In many problems, decisions taken on the basis of local data always lead to optimal solutions; this is the case of problems solved by greedy techniques. On the other hand, there are problems, including alignment, for which this doesn't hold true. In this case one would have to generate all possible decision sequences and evaluate them. Dynamic programming can be used to exclude sub-optimal decision sequences so that they may not be considered. The principle of optimality governing dynamic programming is: "Any sub-sequence of the optimal decision sequence is optimal for the sub-problem corresponding to this sub-sequence of decisions".

Although dynamic programming is successfully applied to sentence alignment, it comes close to its limits when dealing with sub-sentence alignments given that the assumption of the left-to-right translation made for sentence alignment, is not valid at the bellow sentence level, or in other words, the order of the clauses in the source language is not the same in the target language. To handle cases of clause-alignments involving a number of clauses in the order of ten or more, we use a simulated annealing framework to approximate the optimal alignment. Simulated annealing (Metropolis et al., 1953), (Kirkpatrick et al. 1983), is a method for optimising functions depending on a large number of parameters. Annealing is a metallurgical term and the method is inspired by the controlled cooling of metals getting from the liquid to the solid state. The algorithm has been successfully applied for optimization purposes, including the approximate solution of TSP (Traveling Salesman Problem). This algorithm does not guarantee to find the best solution, but it may come up with a good approximation of it in non-exponential time. Processing starts with a random clause-alignment A. Initial temperature setting is $T=45$ and after each iteration it is reduced by 0.9. Each iteration is performed through 1000 steps. In each step, a random change in A is proposed and the cost function (negative logarithm of the clause-alignment probability) is computed. If the new alignment is better, the change is adopted, if not, it is adopted with probability $P = e^{-\frac{\Delta E}{T}}$, where ΔE is the change in the cost function. Once the loop is computed with no change in the configuration, or 10 iterations have been performed, the best alignment that has been found till that time is proposed.

Alignment Type	Initial Probability Estimation	Probability after Convergence
1-0	0.05	0.0001
1-1	0.8	0.6986
1-2	0.1	0.2465
2-2	0.05	0.0548

Table 1 : Initial and estimated probabilities

Results

The method has been applied to the corpus presented in section 2. A sample output of the method is displayed hereunder. Each table contains a source sentence, a target sentence and the set of proposed clause alignments (underlined alignments are wrong):

Alignment type:2-2, Dynamic Programming (DP)

[cl IF YOU HAVE NO CONSOLE cl] , [cl YOU MUST LOG IN FROM A REMOTE SYSTEM cl]
[cl ΑΝ ΔΕΝ ΥΠΑΡΧΕΙ cl] [cl ΠΡΕΠΕΙ ΝΑ ΕΙΣΕΛΘΕΤΕ ΑΠΟ ΕΝΑ ΑΠΟΜΑΚΡΥΣΜΕΝΟ ΣΥΣΤΗΜΑ cl]
IF YOU HAVE NO CONSOLE <-> ΑΝ ΔΕΝ ΥΠΑΡΧΕΙ *YOU MUST LOG IN FROM A REMOTE SYSTEM <-> ΠΡΕΠΕΙ ΝΑ ΕΙΣΕΛΘΕΤΕ ΑΠΟ ΕΝΑ ΑΠΟΜΑΚΡΥΣΜΕΝΟ ΣΥΣΤΗΜΑ*

Alignment type:3-3, DP

[cl THERE ARE SEVERAL REASONS cl] [cl THAT HP VUE MIGHT FAIL cl] [cl TO START cl]
[cl ΥΠΑΡΧΟΥΝ ΠΟΛΛΟΙ ΛΟΓΟΙ cl] [cl ΓΙΑ ΤΟΥΣ ΟΠΟΙΟΥΣ ΤΟ HP VUE ΜΠΟΡΕΙ ΝΑ ΑΠΟΤΥΧΕΙ cl] [cl ΝΑ ΞΕΚΙΝΗΣΕΙ cl]
THERE ARE SEVERAL REASONS <-> ΥΠΑΡΧΟΥΝ ΠΟΛΛΟΙ ΛΟΓΟΙ *THAT HP VUE MIGHT FAIL <-> ΓΙΑ ΤΟΥΣ ΟΠΟΙΟΥΣ ΤΟ HP VUE ΜΠΟΡΕΙ ΝΑ ΑΠΟΤΥΧΕΙ* *TO START <-> ΝΑ ΞΕΚΙΝΗΣΕΙ*

Alignment type:4-3, DP

[cl WHEN HP VUE FAILS cl] [cl TO BEHAVE cl] [cl AS EXPECTED cl] , [cl YOU SHOULD OPEN THE APPROPRIATE ERROR-MONITORING FILE cl]
[cl ΟΤΑΝ ΤΟ HP VUE ΑΠΟΤΥΓΧΑΝΕΙ cl] [cl ΝΑ ΣΥΜΠΕΡΙΦΕΡΘΕΙ ΚΑΤΑ ΤΟ ΑΝΑΜΕΝΟΜΕΝΟ cl] [cl ΘΑ ΠΡΕΠΕΙ ΝΑ ΑΝΟΙΞΕΤΕ ΤΟ ΚΑΤΑΛΛΗΛΟ ΑΡΧΕΙΟ ΠΑΡΑΚΟΛΟΥΘΗΣΗΣ ΣΦΑΛΜΑΤΩΝ cl]
WHEN HP VUE FAILS <-> ΟΤΑΝ ΤΟ HP VUE ΑΠΟΤΥΓΧΑΝΕΙ *TO BEHAVE AS EXPECTED <-> ΝΑ ΣΥΜΠΕΡΙΦΕΡΘΕΙ ΚΑΤΑ ΤΟ ΑΝΑΜΕΝΟΜΕΝΟ* *YOU SHOULD OPEN THE APPROPRIATE ERROR-MONITORING FILE <-> ΘΑ ΠΡΕΠΕΙ ΝΑ ΑΝΟΙΞΕΤΕ ΤΟ ΚΑΤΑΛΛΗΛΟ ΑΡΧΕΙΟ ΠΑΡΑΚΟΛΟΥΘΗΣΗΣ ΣΦΑΛΜΑΤΩΝ*

Alignment type:3-2, DP

[cl CREATING A SIMPLE ACTION cl] [cl COVERS cl] [cl HOW TO USE CREATEACTION cl]
[cl Η ΔΗΜΙΟΥΡΓΙΑ ΜΙΑΣ ΑΠΛΗΣ ΕΝΕΡΓΕΙΑΣ ΚΑΛΥΠΤΕΙ ΤΟ cl] [cl ΠΩΣ ΝΑ ΧΡΗΣΙΜΟΠΟΙΗΣΕΤΕ ΤΗ " CREATEACTION " cl]
CREATING A SIMPLE ACTION <-> Η ΔΗΜΙΟΥΡΓΙΑ ΜΙΑΣ ΑΠΛΗΣ ΕΝΕΡΓΕΙΑΣ ΚΑΛΥΠΤΕΙ ΤΟ *COVERS HOW TO USE CREATEACTION <-> ΠΩΣ ΝΑ ΧΡΗΣΙΜΟΠΟΙΗΣΕΤΕ ΤΗ " CREATEACTION "*

Alignment type:6-6, Simulated Annealing(SA)

[cl IF YOU PREVIOUSLY USED SOFTBENCH cl] [cl AND HAVE A PERSONAL <DIR>/HOMEDIRECTORY/.SOFTINIT </DIR> FILE cl] , [cl YOU MAY NEED cl] [cl TO REMOVE THE FILE cl] [cl OR EDIT IT cl] [cl TO INCLUDE THE HP VUE TOOLS cl]
[cl ΑΝ ΠΡΟΗΓΟΥΜΕΝΩΣ ΧΡΗΣΙΜΟΠΟΙΗΣΑΤΕ ΤΟ SOFTBENCH cl] [cl ΚΑΙ ΕΧΕΤΕ ΕΝΑ ΠΡΟΣΩΠΙΚΟ ΑΡΧΕΙΟ <DIR>/HOMEDIRECTORY/.SOFTINIT</DIR> cl] [cl ΜΠΟΡΕΙ ΝΑ ΧΡΕΙΑΣΤΕΙ cl] [cl ΝΑ ΑΦΑΙΡΕΣΕΤΕ ΤΟ ΑΡΧΕΙΟ cl] [cl Η ΝΑ ΤΟ ΤΡΟΠΟΠΟΙΗΣΕΤΕ cl] [cl ΩΣΤΕ ΝΑ ΠΕΡΙΛΑΜΒΑΝΕΙ ΤΑ ΕΡΓΑΛΕΙΑ HP VUE cl]
IF YOU PREVIOUSLY USED SOFTBENCH <-> ΑΝ ΠΡΟΗΓΟΥΜΕΝΩΣ ΧΡΗΣΙΜΟΠΟΙΗΣΑΤΕ ΤΟ SOFTBENCH *AND HAVE A PERSONAL <DIR>/HOMEDIRECTORY /.SOFTINIT</DIR> FILE <-> ΚΑΙ ΕΧΕΤΕ ΕΝΑ ΠΡΟΣΩΠΙΚΟ ΑΡΧΕΙΟ <DIR>/HOMEDIRECTORY/.SOFTINIT</DIR>* *YOU MAY NEED <-> ΜΠΟΡΕΙ ΝΑ ΧΡΕΙΑΣΤΕΙ* *TO REMOVE THE FILE <-> ΝΑ ΑΦΑΙΡΕΣΕΤΕ ΤΟ ΑΡΧΕΙΟ* *OR EDIT IT <-> Η ΝΑ ΤΟ ΤΡΟΠΟΠΟΙΗΣΕΤΕ* *TO INCLUDE THE HP VUE TOOLS <-> ΩΣΤΕ ΝΑ ΠΕΡΙΛΑΜΒΑΝΕΙ ΤΑ ΕΡΓΑΛΕΙΑ HP VUE*

The performance has been evaluated on a text portion containing ca. 250 sentences and overall precision of the output has been calculated to be 85.7%. If we exclude cases of misalignments due to errors in stages of processing preceding clause-alignment, we can calculate the precision of the last stage. In this case, precision is higher than 96%, so the error-rate introduced during clause-alignment is less than 4%. In addition to the low error-rate, clause-alignment corrects some of the errors caused by the previous stages, as it is mentioned in the next section.

Discussion

Given the incremental and engineering approach adopted, the results obtained so far are quite encouraging. The accuracy of the output lies around +85%, making the method quite reliable and suitable to be used in real world application systems.

Most of the errors were introduced by the first three primary processing stages, that is sentence-alignment, POS tagging and clause recognition. Major improvements in performance will certainly require further optimization of some or all of these stages along with any refinements to the statistical clause-alignment model used in the last stage. Regarding refinements to clause-alignment, there are several sources of information that could be readily taken into account. For example, pre-compiled bilingual dictionaries could be of help in order to establish reliable word associations in very short texts, which do not allow the safe estimation of the required word probabilities, while preference rules on clause types could be used to reduce search space, favoring alignments between certain clause types and penalising others. Future developments are believed to help improve accuracy and performance and broaden the coverage of the system in order to cover additional types of sub-sentence alignments. An interesting remark is that errors introduced by preceding stages are sometimes repaired by clause-alignment. For example, it may happen that a sentence is mistakenly chunked into clauses due to tagging or other errors. Then '1-2' and '2-2' clause-alignments may function in such a way that illegally separated sentence pieces are brought back together.

It is well understood that linguistic resources building is one of the important stumbling blocks in the localization/internationalization exercise. Methods approximating the automatic generation of such resources prove to be effective on a cost/time basis. Besides gains in speed and efficiency, the data driven approach improves consistency, which is an important requirement for systems

operating in a multilingual setting. By adopting a data driven approach and exploiting existing linguistic processing modules, the method produces textual parallel data of high resolution which can give a competitive advantage to multilingual processes and systems, such as semi-automatic lexicon builders, machine aided translation systems and retrieval of multilingual material.

References

Aho A., R. Sethi, and J. Ullman. 1986. Compilers, Principles, Techniques and Tools. Reading, Masschusets: Addison Wesley

Burnard, L. 1995. Users Reference Guide for the British National Corpus, British National Corpus Consortium Report, Oxford, England.

Boutsis, S., and S. Piperidis. 1996. Automatic Extraction of Bilingual Lexical Equivalences from Parallel Corpora. In Proc. Multilinguality in Software Industry /ECAI, 27-31.12 August, Budapest, Hungary.

Brown, P. 1988. A Statistical Approach to Language Translation. In Proc.12th International Conference on Computational Linguistics, vol. 1, 71-76. Budapest, Hungary.

Brown, P., J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, J. Lafferty, R. Mercer, and P. Roosin. 1990. A Statistical Approach to Machine Translation. Computational Linguistics. June: 79-85.

Brown P., J. Lai, and R. Mercer. 1991. Aligning Sentences in Parallel Corpora. In Proc. 29th Annual Meeting of the ACL, 169-176. 18-21 June, Berkley, Calif.

Dagan, I., A. Itai, and U. Schwall.1991. Two languages are more informative than one. In Proc. 29th Annual Meeting of the Association for Computational Linguistics, 130-137.18-21 June, Berkley, Calif.

Di Christo, P., S. Harie, C. De Loupy, N. Ide, and J. Veronis. 1995. Set of programs for segmentation and lexical look up, MULTEXT LRE 62-050 project Deliverable 2.2.1.

Elworthy, D. 1997. Tagger Suite User's Manual. Cambridge University Computer Laboratory Report, Cambridge, England.

Gale, W.A., and K.W. Church. 1991. A Program for Aligning Sentences in Parallel Corpora. In Proc. of the 29th Annual Meeting of the Association for Computational Linguistics, 177-184. 18-21 June, Berkley, Calif.

Gale, W.A., and K.W. Church. 1991b. Identifying word correspondences in parallel texts. Proceedings of the Fourth DARPA Speech and Natural Language Workshop, 152-157.

Kay, M., and M. Roescheisen. 1993. Text-translation Alignment. Computational Linguistics. March:121-142.

Kirkpatrick, S., C. Gelatt, and M.P. Vecchi. 1983. Optimisation by Simulated Annealing. Science Vol 220. pp. 671-680.

Kitamura, M., and Y. Matsumoto. 1995. A Machine Translation System based on Translation Rules Acquired from Parallel Texts. In Proc. Recent Advances in Natural Language Processing, 27-44. 14 - 16 September, Tzigov Chark, Bulgaria.

Kitamura, M., and Y. Matsumoto. 1996. Automatic Extraction of Word Sequence Correspondences in Parallel Corpora. In Proc. 4th Workshop on Very Large Corpora, 79-87. 4 August, Copenhagen, Denmark.

Kumano, A., and H. Hirakawa. 1994. Building an MT Dictionary from Parallel Texts Based on Linguistic and Statistical Information. In Proc. 15th International Conference on Computational Linguistics,76-81. 5-9 August, Kyoto, Japan.

Kupiec, J. 1993. An algorithm for Finding Noun Phrase Correspondences in Bilingual Corpora. In Proc. 31st Annual Meeting of the Association for Computational Linguistics, 17-22. 22-26 June, Columbus, Ohio.

Leech, G. 1995. A brief users' guide to the grammatical tagging of the British National Corpus. British National Corpus Consortium Report, Oxford, England.

Matsumoto, Y., H.Ishimoto, T. Utsuro. 1993. Structural Matching of Parallel Texts. In Proc. 31st Annual Meeting of the Association for Computational Linguistics, 23-30. 22-26 June, Columbus, Ohio.

Metropolis, N., A. Rosenbluth, M. Teller, A. Teller, and E. Teller. 1953. Journal Chem. Phys. Vol. 21. Pp 1087.

Papageorgiou, H., L. Cranias, and S. Piperidis. 1994. Automatic Allignment in Parallel Corpora. In Proc. 32nd Annual Meeting of the Association for Computational Linguistics, 334-336. 27-30 June, Las Cruses, New Mexico.

Papageorgiou H. 1996. Part of Speech Disambiguation. In Hybrid Techniques for Bilingual Corpus Processing 63-83. PhD thesis, National Technical University of Athens, Greece

Piperidis S. 1995. Interactive Corpus-based Translation Drafting Tool. Aslib Proceedings. March: 83-92.

Piperidis, S., S. Boutsis, and I. Demiros. 1997. Automatic Translation Lexicon Generation. In Proc. Multilinguality in Software Industry /IJCAI. 25 August, Nagoya, Japan.

Simard, M., G. Foster, and P. Isabelle. 1992. Using cognates to align sentences in bilingual corpora. Proc. TMI-92. Montréal, Québec.

Smadja, F. 1992. How to compile a bilingual collocational lexicon automatically.In Proc. AAAI Workshop on Statistically -based NLP Techniques, 67-71. San Jose, California.

Smadja, F., K.R. McKeown, and V. Hatzivassiloglou. 1996. Translating Collocations for Bilingual Lexicons: A Statistical Approach. Computational Linguistics. March: 1-38.

Automatic Insertion of Accents in French Text

Michel Simard
Laboratoire de recherche appliquée en linguistique informatique (RALI)
Université de Montréal
simardm@iro.UMontreal.CA

Abstract

Automatic accent insertion (*AAI*) is the problem of re-inserting accents (diacritics) into a text where they are missing. Unaccented French texts are still quite common in electronic media, as a result of a long history of character encoding problems and the lack of well-established conventions for typing accented characters on computer keyboards. We present an AAI method for French, based on a stochastic language model. This method was implemented into a program and *C* library of functions, which are now commercially available. Our experiments show that French text processed with this program contains less than one accent error per 130 words. We also show how our AAI method can be used to do on-the-fly accent insertions within a word-processing environment, which makes it possible to write in French without having to type accents. A prototype of such a system was integrated into the Emacs editor, and is now available to all students and employees of the Université de Montréal's computer science department.

1 Introduction

Even in this era of flashy, high-speed multimedia information, unaccented French texts (i.e texts without diacritics) are still routinely encountered in electronic media. Two factors account for this: first, the computer field has long suffered from a lack of sufficiently widespread standards for encoding accented characters, which has resulted in a plethora of problems in the electronic transfer and processing of French texts. Even now, it is not uncommon for one of the software links in an E-mail distribution chain to deliberately remove accents in order to avoid subsequent problems. Secondly, when using a computer keyboard that is not specifically designed for French, keying in French accented characters can turn out to be a laborious activity. This is a matter of both standards and ergonomics. As a result, a large number of French-speaking users systematically avoid using accented characters, at least in informal communication.

If this situation remains tolerable in practice, it is essentially because it is extremely rare that the absence of accents renders a French text incomprehensible to the human reader. Cases of ambiguity do nonetheless occur: for instance, "Ce chantier ferme a cause des emeutes" could be interpreted as "Ce chantier *ferme à cause* des émeutes" ("This work-site is closing because of the riots") or "Ce chantier *fermé a causé* des émeutes" ("This closed work-site [more naturally put, this work-site closure] has caused riots"). From a linguistic point of view, the lack of accents in French simply increases the relative degree of ambiguity inherent in the language. At worst, it slows down reading and proves awkward, much as a text written entirely in capital letters might do.

The fact remains, however, that while unaccented French text may be tolerated under certain circumstances, it is not acceptable in common usage, especially in the case of printed documents. Furthermore, unaccented texts pose serious problems for automatic processing: NLP-based applications such as information retrieval, information extraction, machine translation, human-machine conversation, speech synthesis, as well as many others, will usually require that French texts be properly accented to begin with.

Actually, for human readers, unaccented texts is probably the most benign of a more general class of ill treatments to which French texts are subjected. For example, it is not uncommon for older programs that are not "8-bit clean" to "strip" the eighth bit of each character, thus irreversibly mapping French characters onto the basic ASCII set. When this treatment is applied to an ISO-Latin text, '*é*' becomes '*i*', '*è*' becomes '*h*', etc. Other programs will simply delete accented characters, or replace them with a unique character, such as a question mark. The texts that result rapidly become unreadable.

All of the above factors prompted the initial interest in methods of *automatic accent insertion* (or *AAI*). Of course, as standards such as *Unicode* (multilingual character-coding standard) and *MIME* (multipurpose Internet mail extensions) gain ground, the accent legacy problem slowly disappears. The problem of typing accents, however, is likely to remain. For this reason, we have become interested in meth-

ods that would perform automatic accent insertion *on-the-fly*, in real time. It appears to us that such a tool would be a valuable addition to any word-processing environment, equally useful for native and non-native speakers of French.

In what follows, we first present a general automatic accent insertion method, based on a stochastic language model. This method was implemented into a program called Réacc, which is now commercially available through Alis Technologies[1]. We then examine how this method can be adapted to perform accent insertions on-the-fly within a word-processing environment. As we go along, we describe the various experiments we designed to evaluate the performance of the system in different contexts, and present the results obtained. Finally, we briefly describe how a prototype "on-the-fly accentuation" (*OTFA*) system was implemented within the Emacs text-editor.

Although our research focuses on unaccented French texts, we believe that our approach could be adapted to other languages that use diacritical marks, as well as to other types of text corruption, such as those mentioned above. The AAI problem and the solutions that we propose are also related to the more general problems of word-sense disambiguation and spelling and grammar checking.

2 Basic Automatic Accent Insertion

In its simplest form, the automatic accent insertion problem can be formulated this way: we are given as input an unaccented French text, in the form of a sequence of unaccented words $w_1 w_2 \ldots w_n$. To every one of these input words w_i may correspond any number of valid words (accented or not) $w_{i1} \ldots w_{im}$: our task is to disambiguate each word, i.e. to select the correct words w_{ik_i} at every position in the text, in order to produce a properly accented text.

An examination of the problem reveals that the vast majority (approximately 85%) of the words in French texts carry no accents at all, and that the correct form of more than half of the remaining words can be deduced deterministically on the basis of the unaccented form. Consequently, with the use of a good dictionary, accents can be restored to an unaccented text with a success rate of nearly 95% (i.e., an error in accentuation will occur in approximately every 20 words). The problems that remain at this point mostly revolve around *ambiguous* unaccented words, i.e. words to which more than one valid form may correspond, whether accented or not[2].

Obviously, for many such ambiguities in French, a simple solution is to systematically select the most frequent alternative. For instance, the most frequent word in most French texts is usually the preposition *de*, which turns out to be ambiguous, because there is also a French word *dé*, meaning either *dice* or *thimble*. If we simply ignore the latter form, we are likely to produce the correct form over 99% of the time, even in texts related to gambling and sewing! This general strategy can be implemented by determining *a priori* the most frequent alternative for each set of ambiguous words in a dictionary, by means of frequency statistics extracted from a corpus of properly accented French text. Using this simple method, we achieve a success rate of approximately 97%, i.e. roughly one error per 35 words.

Clearly, to attain better performances than these, an automatic accent insertion system will need to examine the context within which a given ambiguous word appears, and then resort to some form of linguistic knowledge. *Statistical language models* seem to be particularly well fit to this task, because they provide us with quantitative means of comparing alternatives.

We propose an automatic accent insertion (*AAI*) method that proceeds in two steps.

1. **Hypotheses generation**: identify for each input word the list of valid alternatives to which it may correspond;
2. **Candidate Selection**: select the best candidate in each list of hypotheses.

This is illustrated in Figure 1.

2.1 Hypotheses Generation

Hypotheses generation produces, for each word w_i of the input, a list of possible words $w_{i1} \ldots w_{im}$ to which it may correspond. For example, the form *pousse* may correspond to either *pousse* or *poussé*; *cote* to *cote*, *côte*, *coté* or *côté*; the only valid form for *francais* is *français* (with a cedilla), and *ordinateur* is its own unique correct form. In theory, nothing precludes generating *invalid* as well as valid hypotheses at this stage: for instance, for *cote*, also generate *cötè* and *çote*. But to limit the number of possibilities that the system must consider, hypotheses are produced using a list of known French word-forms, indexed on their unaccented version. On the other hand, when the hypotheses generator encounters word-forms that it does not know, it simply reproduces them verbatim.

2.2 Candidate Selection

Once lists of hypotheses have been identified for each input word, the best candidate of each list must be identified. For this, we rely on a stochastic language model, which can assign a score to any sequence of words, corresponding to the probability that the model generate this sequence. Given an input sequence of words $w_1 w_2 \ldots w_n$, and for each word w_i

[1] Alis Technologies: http://www.alis.com

[2] As we will see later on, other problems are caused by *unknown* words , i.e. words for which *no* valid forms are known.

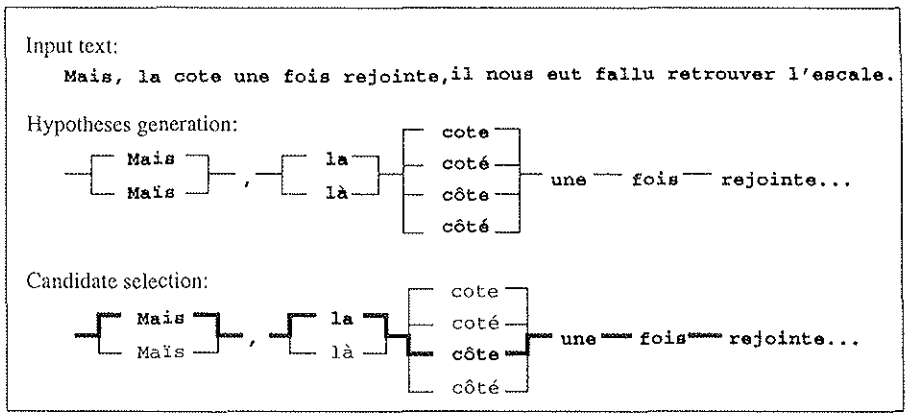


Figure 1: Automatic accent insertion method

in the sequence, a list of hypotheses $(w_{i1}, \ldots, w_{im})$, our goal can be reformulated as finding the sequence of hypotheses $w_{1k_1} w_{2k_2} \ldots w_{nk_n}$ that maximizes the overall likelihood of the output sequence.

The stochastic model we use is a *Hidden Markov Model* (HMM), within which a text is viewed as the result of two distinct stochastic processes. The first process generates a sequence of abstract symbols. In our case, these symbols correspond to *morpho-syntactic tags*, e.g. *"common noun, masculine-singular"*, *"verb, present indicative form, third person plural"*. In an N-tag HMM, the production of a tag depends on the $N-1$ preceding tags, so that the probability of observing a given tag t_i in a given context follows a conditional distribution $P(t_i|t_{i-N} \ldots t_{i-1})$.

Then, for each tag in this first sequence, a second stochastic process generates a second symbol: in our case, these symbols correspond to actual words in the language.

The parameters that define the model are:

- $P(t_i|h_{i-1})$: the probability of observing tag t_i, given the previous $N-1$ tags (h_{i-1} designates the series of $N-1$ tags ending at position $i-1$);
- $P(w_i|t_i)$: the probability of observing word w_i given the underlying tag t_i.

Given these parameters, the probability of generating some sequence of words $w = w_1 w_2 \ldots w_n$ can be evaluated. If T is the *tag alphabet*, and T^n denotes the set of all possible sequences of n tags of T, then:

$$P(w) = \sum_{t \in T^n} \prod_{i=1}^{n} P(t_i|h_{i-1}) P(w_i|t_i)$$

The direct calculation of this equation requires a number of calculation that is exponential in the length of the sequence. However, there exists an algorithm that computes the value of $P(w)$ in polynomial time (Rabiner and Juang, 1986).

To find the sequence of hypotheses that maximizes the probability of the text, each individual combination of hypotheses is examined. Because the number of possible combinations grows exponentially with the length of the text, we will want to segment the text into smaller pieces, whose probabilities can be maximized individually. Sentences are usually considered to be syntactically independent, and so we may assume that maximizing the probability of each sentence will yield the same result as maximizing the whole text. Even within sentences, it is sometimes possible to find subsegments that are "relatively" independent of one another. Typically, the inner punctuation of sentences (semicolons, commas, etc.) separates segments that are likely to be independent of one another. In the absence of inner punctuation, it is still possible to segment a sentence around regions of "low ambiguity".

Our AAI method relies on a heuristic segmentation method, which cuts up each sentence into a number of segments, such that the number of combinations of hypotheses to examine in each segment does not exceed a certain fixed threshold, while minimizing dependencies between segments. This segmentation strategy effectively guarantees that the accent-insertion can be done in polynomial time. But we sometimes end up segmenting the text at "sub-optimal" locations. This will have consequences on performance, as we will see in the next section.

Segments are processed in a left-to-right fashion. In practice, we have realized that one way of minimizing the negative impact of sub-optimal segmentations is to prepend to each segment the last few words of the previous segment, as output by the AAI system. This seems to have the effect of "priming" the model. The prepended words are then simply dropped when the

final result is pieced together.

2.3 Implementation

The method presented in the previous section was implemented in a program called `Réacc`. This program, given a hypotheses generator, the parameters of a HMM and an input, unaccented French text, produces an accented version of that text on the output.

The hypotheses generator we used was produced from a list of over 250 000 valid French words, extracted from our French morpho-syntactic electronic dictionary. Such a large dictionary is probably overkill, and in fact, it may even be the case that it uselessly slows down processing, by proposing extremely rare (although probably valid) words. (The only francophones we met that had heard of a *lé* were crossword puzzle addicts.)

The language model used is a 2-tag HMM, based on a set of approximately 350 morpho-syntactic tags. The parameters of the HMM were first estimated by direct frequency counts on a 60 000 words, hand-tagged extract of the Canadian Hansard. The parameters were then refined, using Baum-Welch reestimation (Baum, 1972), on a 3 million word (untagged) corpus consisting of equal parts of Hansards, Canadian National Defense documents and French press revues (*Radio-France International*).

2.4 Performance Evaluation

One of the interesting properties of the AAI problem is that the performance assessment of a given program is a very straightforward affair: all we need is a corpus of correctly accented French text, and a "de-accentuation" program. Performance can be measured by counting the number of words that differ in the original text and its re-accented counterpart.

For the purpose of our evaluation, we used a test corpus made up of various types of text. It contains Hansard, National Defense and RFI documents (distinct from those used in training), but also United Nations documents, court transcripts, computer manuals as well as some literary texts. The whole corpus contains 57 966 words (as counted by the standard `wc` UNIX program).

Apart from the hypotheses generator and the language model parameters, a number of parameters affect the performance of the program. The most important of these is the maximum number of combinations per subsegment, that it used in the segmentation heuristic. In what follows, we refer to this parameter as S. The results obtained for different values of S are presented in Table 1. All tests were done on a SparcSTATION 10 computer, with 32 MB of memory.

A cursory look at the results reveals that there is much to be gained by allowing the system to work on longer segments. However, beyond a certain limit, the quality of the results tends to level off, while the running time increases radically. Depending on the context of application of the program and the resources available, it would seem that acceptable results can be obtained with S set at around 16 or 32. In this setting, the system will process anywhere between 10 000 and 20 000 words per minute.

It is interesting to look at where `Réacc` goes wrong. Table 2 provides a rough classification of accent-restoration errors made by the program on our test corpus with S set at 16. The largest category of accentuation errors includes a rather liberal grouping of errors that have a common feature: they are the result of an incorrect choice pertaining to an acute accent on a final *e*. In most cases (although not all), this corresponds to an ambiguity between a finite and participle forms of a verb, e.g. *aime* as opposed to *aimé*. The next group of errors are those that stem from inadequacies in the hypotheses generator – i.e. cases in which the generator simply does not know the correct accented form. In most cases (nearly half), proper nouns are involved, but, especially in more technical texts, there are also many abbreviations, non-French words and neologisms (e.g. *réaménagement*, *séropositivité*). The next category concerns a unique word pair: the preposition *à*, and *a*, the third person singular present indicative form of the verb *avoir*.

2.5 Related Work

El-Bèze et al. (1994) present an AAI method that is very similar to ours. It also proceeds in two steps: hypotheses generation, which is based on a list of valid words, and candidate selection, which also relies on a Hidden Markov Model. The main difference between their method and ours is how the HMM is used to score competing hypotheses. While we segment the text into "independent segments" and maximize the probability of these segments, their program processes the text from left to right, using a fixed width "sliding window":

- For each word w_i, the hypotheses generator produces a list of possible *word/tag* alternatives: $(w_{i1}, t_{i1}), \ldots, (w_{ik}, t_{ik})$;
- Candidate Selection proceeds by selecting a specific pair (w_{ij}, t_{ij}) at each position; the goal is to find the sequence of *word/tag* pairs whose probability is maximum according to the model:

$$\prod_{i=1}^{n} P(w_{ij_i}|t_{ij_i}) P(t_{ij_i}|t_{i-1j_{i-1}}, t_{i-2j_{i-2}})$$

- To avoid combinatorial problems, instead of computing this product for all possible sequences, the system finds at each position i in the sequence the pair (w_{ij}, t_{ij}) that *locally* maximizes that part

Max. no. of combinations per segment (S)	Running time (seconds)	Total number of errors (words)	Average distance between errors (words)
2	68	821	70
4	85	560	103
8	132	466	124
16	169	441	130
32	277	429	134
64	429	425	136
128	731	420	137

Table 1: Results of AAI Experiments on 58K-word Test Corpus

Type of error	Number of occurrences	Percentage
-e VS. *-é* ending	171	38.8%
Unknown words	111	25.2%
a VS. *à*	69	15.7%
Other	90	20.4%
Total	441	100.0%

Table 2: Classification of Accent Restoration Errors ($S = 16$)

of the global computation within which it is involved:

$$P_i \times P_{i+1} \times P_{i+2}$$

where $P_i = P(w_{ij_i}|t_{ij_i})P(t_{ij_i}|t_{i-1j_{i-1}}, t_{i-2j_{i-2}})$.

- These computations proceed from left to right, so that the optimal tag found for position i will be used in the computation of the optimal *word/tag* pairs at positions $i + 1$ and $i + 2$.

The experimental results reported in El-Bèze et al. (1994) indicate success levels slightly superior to ours. This may be explained in part by the use of a better language model (their HMM is three-tag, ours is two-tag). It must be said, however, that their test-corpus was relatively small (in all, a little over 8000 words), and that the performances varied wildly from text to text, with average distances between errors varying between 100 and 600 words.

A method which exploits different sources of information in the candidate selection task is described in Yarowsky (1994b): this system relies on local context (e.g., words within a 2- or 4-word window around the current word), global context (e.g. a 40-word window), part-of-speech of surrounding words, etc. These are combined within a unifying framework known as *decision lists*. Within this framework, the system bases its decision for each individual candidate selection on the single most reliable piece of evidence.

Although the work described in Yarowsky (1994b) does address the problem of French automatic accentuation, it mostly focuses on the Spanish language. Furthermore, the evaluation focuses on specific ambiguities, from which it is impossible to get a global performance measure. As a result, it is unfortunately not currently possible to compare these findings with ours in a quantitative way.

In Yarowsky (1994a), the author compares his method with one based on the stochastic part-of-speech tagger of Church (1988), a method which obviously has a number of points in common with ours. In Mr Yarowsky's experiments, this method is clearly outperformed by the one based on decision lists. This is most apparent in situations where competing hypotheses are "syntactically interchangeable": pairs of words with identical morpho-syntactic features, or with differences that have no direct syntactic effects, e.g. present/preterite verb tenses. Such ambiguities are better resolved with non-local context, such as temporal indicators. As it happens, however, while such situations are very common in Spanish, they are rare in French. Furthermore, Mr Yarowsky's language model was admittedly quite weak: in the absence of a hand-tagged training corpus, he based his model on an *ad hoc* set of tags.

3 On-the-fly Automatic Accent Insertion

As mentioned earlier, the existence of unaccented French texts can in part be explained by the lack of a standard keying convention for French accents: conventions vary from computer to computer, from keyboard to keyboard, sometimes even from program to program. Many users type French texts without accents simply because they are unfamiliar with the conventions in a particular environment, or because these conventions are too complicated (e.g. hitting three keys in sequence to type a single accented character).

Clearly, in some situations, automatic accent insertion offers a simple solution to this problem: type the text without accents, run an AAI program on the text, and revise the output for accentuation mistakes. Of course, such a solution, if acceptable for one-time production of short texts, is not very practical in general. If a text is subjected to a number of editions and re-editions, or if it is produced cooperatively by several authors working in different environments, then it may need to go through a series of local re-accentuations. This process, if managed by hand, is error-prone and, in the end, probably more laborious than typing the accents by hand.

If, however, the accents are automatically inserted on-the-fly, as the user types the text, then accent revision and corrections can also be done as the text is typed. If such an *on-the-fly accentuation* (*OTFA*) system is capable of producing acceptable results in real-time, it may become a realistic alternative to the manual insertion of accents. In what follows, we examine how this may be done.

3.1 Method

How does OTFA differ from the basic AAI problem? In Section 2, the input was considered to be a static and (hopefully) complete text. In OTFA, the text is dynamic: it changes with every edit operation performed by the user. Therefore, the OTFA method that is conceptually the simplest is to re-compute the accentuation of the whole text after each edit, i.e. repeatedly apply to the entire text an AAI method such as that proposed earlier.

Of course, such a method is impractical, mainly because it will likely be computationally excessively expensive. It is also overkill, because changes in one region of the text are unlikely to affect the accentuation of the text in more or less distant regions. In fact, if we use the AAI method of Section 2, changes in one location will have no effects outside the sentence within which the edit occurs, because sentences are all treated independently. Because sentences are themselves sub-segmented, it is tempting to think that the effect of a given edit will be even further restricted, to the segment of the sentence within which it takes place. This, however, is not generally true, firstly because an edit is likely to affect the sub-segmentation process itself, and also because changes in one segment can have cascading effects on the subsequent segments, as the last words of each segment are prefixed to the following segment as additional context.

So a more practical solution is to process only the sentence within which the latest edit occurred. There are still problems with this approach, however. While the user is editing a sentence, chances are that at any given time, this sentence is "incomplete". Furthermore, although modern text-editors allow insertions and deletions to be performed in any order and at any position of the text, in a normal text-editing context, given the natural tendency of humans to write in a beginning-to-end fashion, the majority of the edits in a French text will be left-to-right insertions at the end of sentences. This means that at any given time, the text to the left of the latest edit is likely to constitute relevant context for the AAI task, while the text to the right is likely not to be relevant. In fact, taking this text into consideration could very well mislead the AAI process, as it may belong to a completely different sentence.

This suggests a further refinement: after each edit, process only that part of the current sentence that lies to the left of the location where the edit took place.

Also, it seems that there is no real need to take any action *while* the user is modifying a given word, and that it would be wiser to wait until all edits on that particular word are finished before processing it. By doing so, we will not only save computational time, we will also avoid annoying the user with irrelevant accentuations on "partial" words. Notice, however, that detecting the exact moment when the user has "finished" typing or modifying a word can be a tricky business. We will deal with this question in Section 3.4.

One of the potential benefits of performing accentuation on-the-fly, as opposed to *a posteriori* AAI, is that the user can correct accent errors as they happen. In turn, because accentuation errors sometimes cascade, such on-the-fly corrections may help the AAI "stay on the right track".

If we want to capitalize on user-corrections, we will need to:

1. *somehow distinguish "corrections" from other types of edits*: the reason is that we don't want to override the user's decisions when performing AAI. This question will also be dealt with when we discuss implementation details (Section 3.4).
2. *limit the scope of the AAIs to a small number of words around the location of the last edit*: the user can only correct the error that he *sees*; in theory, the effect of AAI after each edit is limited to the current sentence, but sentences come in all sizes. If a given "round" of AAI affects text too far away from the site of the last edit, which is usually also the focus of the user's attention, then he is likely not to notice that change. For this reason, it seems reasonable to restrict the actual scope of the AAI process to just a few words: intuitively, three or four words would be reasonable. Note that this doesn't imply restricting the amount of *context* that we provide the AAI with, but only limiting the size of the region that it is allowed to modify.

To summarize, the OTFA method that we propose essentially follows these lines:

- OTFA is performed by repeatedly applying an AAI method (such as that of Section 2) on the text.
- AAI rounds are triggered every time the user finishes editing a word.
- The scope of AAI (which we call the *AAI window*) is limited to a fixed number of words to the left of the last word edited.
- If this can be useful to the AAI process, more context can be given, in the form of additional words belonging to the same sentence to the left of the AAI window (what we call the *context window*).

3.2 Performance Evaluation

The ultimate goal of OTFA is to facilitate the editing of French texts. Therefore, it would be logical to evaluate the performance of an OTFA system in those terms. Unfortunately, the "ease of typing" is a notion that is hard to quantify. In theory, typing speed would seem to be the most objective criterion. But measuring performance using such a criterion would obviously require setting up a complex experimental protocol. On the other hand, the number and nature of parameters involved prohibits a "theoretical" evaluation in these terms.

What we can reliably evaluate, however, is the absolute performance of an OTFA system, in terms of the number of accentuation errors, for a given editing "session". Such a measure gives us an intuitive idea of the impact of the OTFA system on the "ease of typing".

We conducted a number of experiments along this line, to evaluate how an OTFA system based on the AAI system of Section 2 would perform. All experiments were done by simulation, using the same corpus that was used in Section 2.4. The editing "session" we simulated followed a very simple scenario: the user types the whole test corpus, from beginning to end, without typing accents, without making errors, and without correcting those made by the OTFA system.

As was the case with the Réacc program, several parameters affect the quality of the results and the computation time required. The only parameter that is specific to our OTFA method, however, is the size of the AAI window. This parameter, which we refer to as W, is measured in words. We conducted distinct experiments with various values for W, the results of which are summarized in Table 3. In all of these experiments, the segmentation factor S was set at 16.

The first conclusion that we can draw from Table 3 is that there is much to be gained in using an AAI window of more than one word: setting $W = 2$ allows to cut down the number of errors by almost 60%. Performance quickly levels off, however, so that near-optimal results are obtained with a three- or four-word window. This is encouraging, because it seems reasonable to assume that the user can effectively monitor a window of that size, and therefore detect accentuation errors when they occur.

Another point that is very encouraging, and perhaps surprising, is that with $W = 3$, the performance of our OTFA system rivals with that of the basic AAI experiments reported in Section 2.4. One possible explanation is that because the OTFA works with only a small number of words at each round (i.e. only the words in the AAI window), the system never has more than $S = 16$ combinations to examine, and therefore never needs to segment sentences into smaller pieces. In the end, both ways of proceeding are probably more or less equivalent, although more experimentation would be required to determine this for sure. The major difference, of course, is that since OTFA recomputes accentuation with every new word, its computational cost is accordingly higher. However, as seen in Section 2.4, our AAI system can process 20 000 words per minute. Since very few typists can enter more than 100 words per minute, even a straightforward OTFA implementation should be able to handle the required computations in real-time.

3.3 User-feedback

We mentioned earlier that one of the expected benefits of OTFA, as opposed to applying AAI on a text *a posteriori*, is that the user can spot accent errors as soon as they happen, and correct them right away. In fact, we believe that this form of *user-feedback* can even be further exploited, to improve the performance of the system itself. As pointed out in Section 2.4, about a quarter of AAI errors are caused by *unknown words*, i.e. words in the correctly accented version of the text which are unknown to the hypotheses generator. This suggests an easy way of exploiting user-feedback: systematically add to the hypotheses generator all user-corrected words whose form is unknown.

In principle, if we add such a mechanism to our OTFA system, and if the user corrects the AAI errors as soon as they happen, unknown words will be lexicalized right after their first appearance, and the system should only make one error per unknown word. In preliminary experiments with this idea, the average distance between errors passed from 138 to 156 words, a reduction of almost 12% on the total number of errors. Our test corpus being heterogeneous by design, unknown words do not repeat very often. We suspect that even better improvements would be observed on homogeneous texts of similar size.

This idea of exploiting user-feedback to modify the parameters of the OTFA dynamically can actually be pushed further. One of the current problems with

AAI window (W)	Total errors (words)	Average distance between errors (words)
1	1125	52
2	461	126
3	420	138
4	417	139
8	417	139
16	417	139

Table 3: OTFA Simulation Results

our OTFA system is its sometimes annoying tendency to systematically select the most frequent alternative when confronted with syntactically interchangeable words. For example, the two French words *cote* and *côte* have similar morpho-syntactic features (common noun, feminine singular) and so, from a grammatical point of view, are totally interchangeable. It so happens, however, that in the language model's training corpus, the second form, which is highly polysemous, is much more frequent. Therefore, the OTFA will systematically produce that form rather than the other. If the user of the system is writing about the stock market for example, he is likely to want to use the first form *cote*, and therefore to react negatively to the system's insistence on putting a circumflex accent where none should appear.

To solve this problem, some form of *dynamic language modeling* is required. We have begun experimenting with an approach initially proposed by Kuhn and Mori (1990) to solve a similar problem in speech recognition applications. Essentially, they suggest using local context to estimate the parameters of a unigram Markov model, and to use this model in conjunction with the static HMM to evaluate competing alternatives. Preliminary results with this approach are encouraging, although much work remains to be done.

3.4 Implementation

As mentioned earlier, the AAI method presented in Section 2 has been implemented as a program and *C* function library. Based on this implementation, a prototype OTFA system was developed and integrated to the *Emacs* text-editor. Although *Emacs* is not generally viewed as a true word-processing environment, it was a natural choice for prototyping because of its openness and extendibility.

In our implementation, the user of *Emacs* has access to a special editing mode called *Réacc-mode* (technically speaking, a *minor-mode*). When in this mode, the user has access to all the usual editing functions: he can move the cursor around, insert, delete, etc. The main difference with the normal "fundamental" mode is that now, accents are automatically inserted as words are typed, without the user having to explicitly type them.

The implementation follows the general lines of the OTFA method presented in Section 3.1: every time a new word is inserted, the system identifies the AAI window, submits the words that fall within this window to the AAI system, and replaces the content of the window with the newly accented words.

In practice, Emacs and the AAI program run as separate processes, and communicate asynchronously: when a new word is typed, Emacs sends the AAI window to the AAI process, along with other relevant information (context, position, etc.), and returns the control to the user. The AAI program processes the "accentuation request" in the background, and sends the results back to Emacs as soon as they are ready. When this happens, Emacs interrupts whatever it was doing, and replaces the original contents of the AAI window with the newly arrived words. This way, user-interaction is not significantly slowed down by the AAI process, because time-consuming computations typically take place during the editor's idle time, between keystrokes.

It is the editing process' responsibility to initiate AAI rounds, and therefore to determine when a new word has been typed. After experimenting with various strategies, we opted for a relatively simple method, based on the possibility to mark individual characters of the text with specific "properties" in Emacs. When words are processed by the AAI program and re-inserted into the text, they are systematically marked as *auto-accented*. By contrast, characters typed by the user do not carry this mark. Every time the user types a space or newline character, we examine the word immediately preceding the cursor: if all its characters are unmarked, then a new AAI round must be initiated.

We mentioned earlier that it was important for an OTFA system not to override the user's decisions. Two situations are particularly important to consider: when the user manually types an accent within a new word, and when the user corrects the accentuation of a word. In both cases, it is undesirable that the OTFA modify the words in question. The character mark-

ing capabilities of Emacs are also used to detect these situations. The first case (new word with accents) will be identified easily by the presence of accented characters within an unmarked word. The second situation (accent corrections) is more difficult to detect, but in general, a mix of marked and unmarked characters within a single word is a good indicator that corrections have taken place.

When these two situations occur, not only do we not initiate an AAI round, we also inhibit any further re-accentuations on these words, by marking their characters as *user-validated.* Words bearing this mark will never be touched by AAI. This type of marking is not limited to user-inserted accents and user-corrections: when the user turns *Réacc-mode* on, all existing text is initially marked that way. Later on, when AAI rounds are initiated and the system locates the AAI window, all text outside this window is also marked as *user-validated.* This way of proceeding, while allowing the OTFA system to do its work during simple text insertions, limits the possibility of "unpleasant surprises" when more complex interactions take place (deletions, corrections, cut-and-paste operations, etc.).

4 Conclusion

We have presented a method for automatically inserting accents into French text, based on a stochastic language model. This method was implemented into a program and *C* library of functions, which are commercially available from Alis Technologies. We have also shown how this method can be used to do on-the-fly accent insertions within a word-processing environment. A prototype OTFA system was also implemented and integrated into the Emacs editor.

Text processed with our system contains less than one accent error per 130 words on average, regardless of whether the system is used on its own or within an OTFA environment. On a Sun SparcSTATION 10 computer, with 32 MB, the system will process approximately 20 000 words per minute. Within the Emacs OTFA prototype, because AAI is performed asynchronously, the performance of the editor itself is not affected, and accents are inserted faster than this typist can type[3].

The program has been made available to students and employees of the Université de Montréal's computer science department, and initial feedback has been positive. We are currently examining the possibility of integrating our OTFA method to a "real" word-processor, such as Microsoft Word.

Acknowledgments

I am greatly indebted to Guy Lapalme, George Foster and Pierre Isabelle for their invaluable advice and constructive comments, as well as to Elliott Macklovitch, for helping me translate my thoughts into readable English. Many thanks also go to all the members of the RALI who contributed to the development of the Réacc system, as well as François Yergeau of Alis Technologies.

[3]These performance figures were obtained with a segmentation factor S set at 16.

References

L. E. Baum. 1972. An Inequality and Associated Maximization Technique in Statistical Estimations of Probabilistic Functions of Markov Processes. *Inequalities*, 3:1–8.

Kenneth W. Church. 1988. A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text. In *Proceedings of the Second Conference on Applied Natural Language Processing.* ACL.

Marc El-Bèze, Bernard Mérialdo, Bénédicte Rozeron, and Anne-Marie Derouault. 1994. Accentuation automatique de textes par des méthodes probabilistes. *Technique et sciences informatiques*, 13(6):797–815.

Roland Kuhn and Renato De Mori. 1990. A Cache-based Natural Language Model for Speech Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(6).

L. R. Rabiner and B. H. Juang. 1986. An Introduction to Hidden Markov Models. *IEEE ASSP Magazine*, pages 4–16, jan.

David Yarowsky. 1994a. A Comparison of Corpus-based Techniques for Restoring Accents in Spanish and French Texts. In *Proceedings of the Second Annual Workshop on Very Large Corpora*, Kyoto, Japan.

David Yarowsky. 1994b. Decision Lists for Lexical Ambiguity Resolution: Applications to Accent Restoration in Spanish and French. In *Proceedings of ACL-94*, Las Cruces, New Mexico.

Valence Induction with a Head-Lexicalized PCFG

Glenn Carroll and Mats Rooth
IMS, Universität Stuttgart
{glenn,mats}@ims.uni-stuttgart.de

Introduction

Either directly or indirectly, the lexicon for a natural language specifies *complementation frames* or *valences* for open-class words such as verbs and nouns. Constructing a lexicon of complementation frames for large vocabularies constitutes a challenge of scale, with the further complication that frame usage, like vocabulary, varies with genre and undergoes ongoing innovation in a living language. This paper addresses this problem by means of a learning technique based on probabilistic lexicalized context free grammars and the expectation-maximization (EM) algorithm. Given a hand-written grammar and a text corpus, frequencies of a head word accompanied by a frame are estimated using the inside-outside algorithm, and such frequencies are used to compute probability parameters characterizing subcategorization. The procedure can be iterated for improved models. We show that the scheme is practical for large vocabularies and accurate enough to capture differences in usage, such as those characteristic of different domains.

A grammar and formalism

The core of the grammar is an $\bar{X}$ grammar (Jackendoff [1977]) of phrases including noun phrases, prepositional phrases, and verbal clusters. A representative verbal structure is given on the left in Figure 1. The symbol VFC is read "finite verb chunk"; similarly we work with noun chunks (NC), prepositional chunks (PC), and so forth. Our use of the chunk concept follows Abney [1991], Abney [1995]. Categories are interpretable in terms of a feature decomposition, but are treated as atomic in the formalism. We depart from a standard context-free formalism in that heads are marked on the right hand sides of rules, using a prime (').

The grammar includes complementation rules for verbs, nouns, and adjectives. Complements are attached at a level above the chunk, which we call the phrasal level. For instance, the category VFP is expanded as a finite verb chunk VFC and a sequence of complements. This is illustrated on the right in Figure 1, where the VFC headed by *decided* takes a VTOP complement, the VTOC headed by *emphasize* takes an NP complement, and so forth.

Finally, the least standard part of the grammar is a large set of *state* or *n-gram* rules which form a parse without constructing a standard clause-level analysis. Instead, phrasal categories are strung together with context-free rules modelling a finite state machine, where the states are categories consisting of an ordered pair of phrasal categories. This results in right-branching structures, as illustrated Figure 2. Note that the entire tree on the right in Figure 1 could be substituted for the finite verb phrase VFP in the tree on the left in Figure 2. The state rules allow almost all the sentences (about 97%) in the corpus to be parsed, at the price of not assigning linguistically realistic higher-level structure.

We now define headed context-free grammars in the sense employed here.

Definition. A headed context free grammar is a tuple $\langle N, T, W, \mathcal{L}, \mathcal{R}, s\rangle$, where: (i) N and T are disjoint sets, interpreted as the non-terminal and terminal categories respectively. (ii) W is a set, interpreted as the set of words. (iii) $\mathcal{L}$ is a relation between W and T, indicating the possible terminal categories (parts of speech) for a given word. (iv) The set of headed productions $\mathcal{R}$ is a finite subset of $N \times N^* \times (N \cup T) \times N^*$, such that each non-terminal occurs as the left hand side of some rule and each terminal occurs on the right hand side of some rule. (v) $s \in N$, with the interpretation of a start symbol.

We typically use $\bar{n}$ as a variable for mother categories, n for head daughter categories, and α and β for the category sequences flanking the head on the right hand side, so that $\langle \bar{n}, \alpha, n, \beta\rangle$ represents a rule. x is used as a variable for non-head categories.

A category $\bar{n}$ in N is a *projection* of a category n in $N \cup T$ if there is some rule of the form $\langle \bar{n}, \alpha, n, \beta\rangle$. The set of *lexicalized nonterminals* $\mathcal{N} \subseteq W \times N$ is the composition of $\mathcal{L}$ with the transitive closure of the

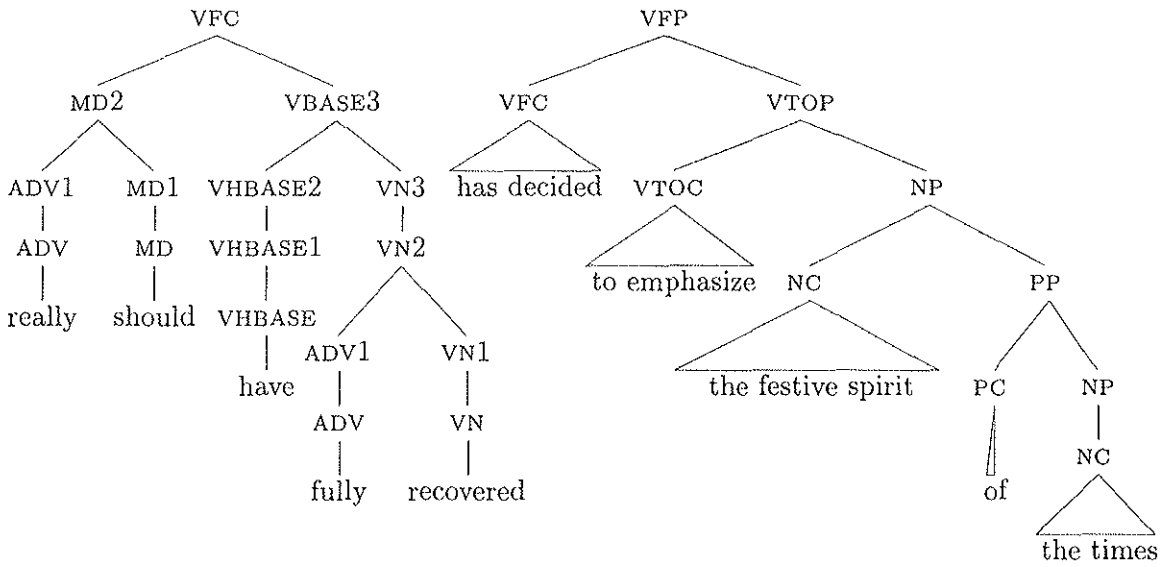


Figure 1: Illustrations of a finite verb chunk and complementation.

projection relation. We have $\langle w, n\rangle \epsilon \mathcal{N}$ if the word w can be the lexical head of the nonterminal category n (in a complete or incomplete tree).

Lexicalization and the probability model

This section defines a parameterized family of probability distributions over the trees licensed by a head-lexicalixed CFG. The main ideas on the parameterization of a lexicalized context free grammar which are employed here derive from Charniak [1995]; see also the remarks on lexicalization in Charniak [1993, section 8.4].

The head marking on rules is used to project lexical items up a chain of categories. In the transitive verb phrase on the right in Figure 2, *question* is projected to the NP level, and *asked* is projected to the VFP level. In this tree, the non-terminal nodes are lexicalized non-terminals, while the terminal nodes are members of $\mathcal{L}$. The point of projecting head words is to make information which probabilistically conditions rules and lexical choices available at the relevant level. At the top level in this example, the head *asked* is used to condition the choice of the phrase structure rule VFP $\rightarrow$ VFC′ NP as well as the choice of *question*, the head of the object.

We now define events which characterize choices of rules and of lexical heads.

Definition. Given a grammar $G = \langle N, T, W, \mathcal{L}, \mathcal{R}, s\rangle$ with lexicalized non-terminals $\mathcal{N}$, the set of rule events $ER(G)$ is the set of tuples $\langle w, \bar{n}, \alpha, n, \beta\rangle$ such that $\langle w, \bar{n}\rangle$ is an element of $\mathcal{N}$ and $\langle \bar{n}, \alpha, n, \beta\rangle$ is an element of $\mathcal{R}$. The set of lexical choice events $EL(G)$ is the set of tuples $\langle w, \bar{n}, x, v\rangle$ such that (i) $\langle w, \bar{n}\rangle$ and $\langle v, x\rangle$ are elements of $\mathcal{N}$;[1] (ii) in some rule of the form $\langle \bar{n}, \alpha, n, \beta\rangle$, x is an element of one or both of the category sequences α and β; and

By virtue of the length of tuples, $ER(G)$ and $EL(G)$ are disjoint, and the union $E(G)$ can be formed without confusing lexical with rule events.

A head-lexicalized PCFG is represented as a function mapping events to real numbers.

Definition. Let G be a headed context free grammar. A head-lexicalized probabilistic context free grammar with signature G is a function p with domain $E(G)$ and range $[0, 1]$ satisfying the conditions: (i) Fixing any lexicalized non-terminal $\langle \bar{w}, \bar{n}\rangle$, $\sum_{\alpha,n,\beta} p_{\bar{w},\bar{n},\alpha,n,\beta} = 1$; (ii) Fixing any lexicalized non-terminal $\langle \bar{w}, \bar{n}\rangle$ and possible non-head daughter x, $\sum_{x,w} p_{\bar{w},\bar{n},x,w} = 1$. Here the value of the function p on a rule event is written as $p_{\bar{w},\bar{n},\alpha,n,\beta}$, and on a lexical event as $p_{\bar{w},\bar{n},x,w}$.

To assign probability weights to trees, we use a tree-licensing and labelling interpretation of the grammar; a node in a tree analysis is labeled with event corresponding to the rule used to expand the node, and the list of lexical events for the non-head daughters of the node. Where τ is a labeled tree li-

[1] In the events, conditioning factors are ordered in the way they are dropped off in the smoothing procedure described below. In a lexical event $\langle w, \bar{n}, x, v\rangle$, the choice of the word v is conditioned on the parent lexical head w, the parent category $\bar{n}$, and the child category x. In the first smoothing distribution, the first conditioning factor, i.e. the parent head w, is dropped.

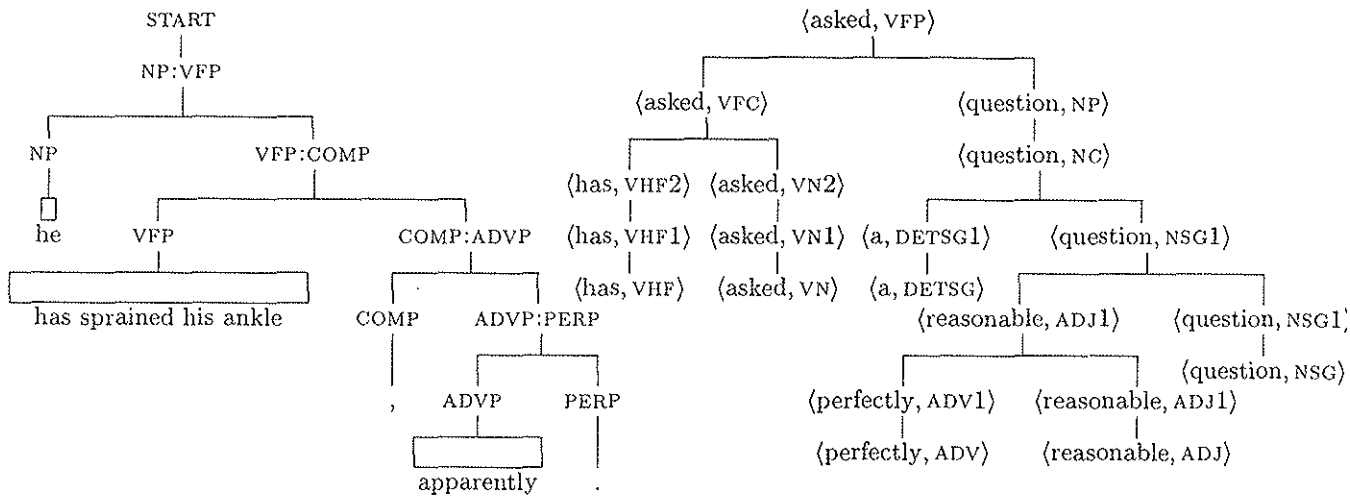


Figure 2: Left: finite-state structure; Right: Lexicalization.

censed by G, we define $e(\tau) : E(G) \rightarrow I\!N$ to be a function counting occurrences of events as labels in τ. Algebraically, we think of $e(\tau)$ as a monomial in the variables $E(G)$; the exponent of a given variable (or event) z is the number of occurrences of z in τ. We denote the evaluation of a polynomial or monomial ϕ in the variables $E(G)$ by subscripting: ϕ_p is the value of ϕ at the vector of reals p. Relative to a parameter setting p, $[e(\tau)]_p$ is interpreted as the probabilistic weight of the labeled tree τ.[2]

These notions are exemplified in Figure 3, which is a phrase structure tree for the N1 (read: N-bar) *big big problem* in a grammar where N1 is the sentence category. Each non-terminal is labeled with a phrase structure rule, and with lexical choice events for non-head daughters. In this case, the only non-head daughters are the two A1's headed with head *big*. ⟨problem,N1,A1,big⟩ is a lexical choice event where *big* is selected as the head of an A1 with parent category N1, and parent head *problem*. An event monomial corresponding to the event tree is obtained as the symbolic product of the events labeling the tree.

Parameter Estimation

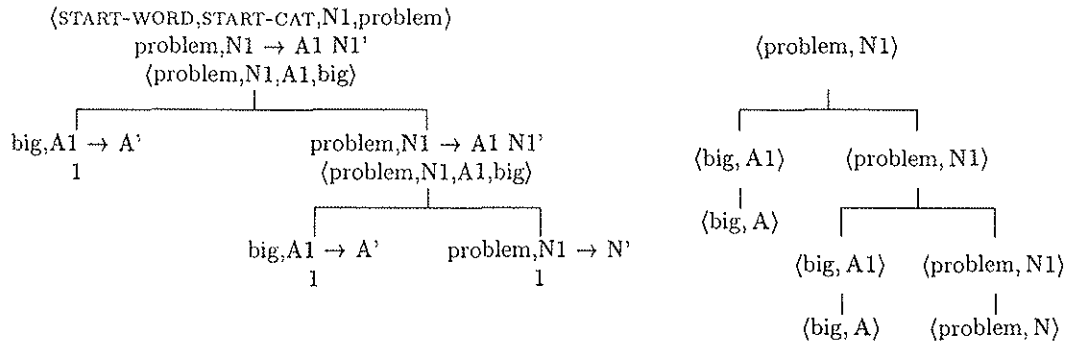
Given a grammar G, the inductive problem is to estimate a head-lexicalized PCFG with signature G. We work with the standard method for estimating PCFGs, based on the Expectation-Maximization framework (Baum & Sell [1968]; Dempster, Laird & Rubin [1977]).

Above, we defined the event polynomial $e(\tau)$ for an event tree τ licensed by G. The event polynomial for a sentence σ is the sum of the event polynomials for the event trees with yield σ. Where corpus $\mathcal{C}$ is a sequences of sentences, the corpus event polynomial $e(\mathcal{C})$ is the (polynomial) product of the event polynomials for the sentences in $\mathcal{C}$. In these terms, maximum likelihood estimation selects a parameter setting p such that the value $[e(\mathcal{C})]_p$ of the corpus polynomial is maximized; this corresponds to selecting a parameter setting which maximizes the probability of the corpus.

The E step of the EM algorithm computes an expected event count function which can be defined in terms of the corpus polynomial. In the estimation of PCFGs using the inside-outside algorithm, event counts are computed iteratively, sentence by sentence. The computation uses a packed parse forest, a compact and-or graph representing a set of trees and the sentence event polynomial, and which allows efficient computation of expected event counts. Somewhat more formally, we use the Inside-outside algorithm (Baker [1979]). to compute $E_p(z|\sigma) : E(G) \rightarrow I\!R$ where z ranges over events in the join rule and lexical event space $E(G)$, defined earlier. $c(\sigma, p)(z)$ has the probabilistic interpretation of the expected number of occurrences of the event z in the set of trees with yield σ.

Given a parameter setting p, event counts are computed and summed over the sentences in the corpus. In the algorithm of Baum and Sell, new parameter values would be defined as relative frequencies of event counts, i.e. maximum-likelihood estimation based on hidden data in the EM framework. We

[2] As with ordinary PCFGs, depending on the parameters, the construction may or may not define a probability measure on the set of finite trees licensed by G. For the general case, infinite trees can be included in the sample space. This requires an extension in the definition of the measure but does not affect the probabilities of finite trees.



$$(\text{problem,N1} \rightarrow \text{A1 N1'})^2(\text{START-WORD,START-CAT,N1,problem})^1(\text{big,A1} \rightarrow \text{A'})^2(\text{problem,N1,A1,big})^2(\text{problem,N1} \rightarrow \text{N'})^1$$

Figure 3: On the left, an event tree. On the right, the corresponding lexicalized tree. On the bottom, the event monomial obtained as a symbolic product of the event labels. The lexical choice event involving START-CAT chooses the head of the sentence, in this case *problem*.

use instead a modified M step involving a smoothing scheme in order to deal with the size of the parameter space and the resulting problems that (i) counts are zero for the majority of events, and (ii) the parameter space is too large to be represented directly in computer memory. Lexicalized rules are smoothed against non-lexicalized rules in a standard back-off scheme (Katz [1980]). The smoothed probability is defined as a weighted sum of the maximum-likelihood estimates for the lexicalized and unlexicalized rule probabilities. The smoothing weight is allowed to vary through five discrete values as a function of the frequency of the word-category pair. The parameters give greater weight to the lexicalized distribution when enough data is present to justify it. The smoothing parameters are set using the EM algorithm on reserved data.

For the lexical choice distributions, an absolute discounting scheme from Ney, Essen & Kneser [1994] is used, which is similar to Good-Turing, but somewhat simpler to work with.

The experiment

We estimated a head-lexicalized PCFG from parts of the British National Corpus (BNC Consortium [1995]), using the grammar described in the first section and the estimation method of the previous section. A bootstrapping method was used, in which first a non-lexicalized probabilistic model was used to collect lexicalized event counts. On the next iteration, counts were estimated based on a lexicalized weighting of parses, as described in the previous section.

Analyses were restricted to those consistent with the part of speech tags specified in the BNC, which are produced with a tagger. In each lexicalized iteration, event counts were collected over a contiguous five million word segment of the corpus. Parameters were re-computed in the way described above, and the procedure was iterated on the next contiguous five-million word segment. Results from all iterations were pooled to form a single model estimated from 50M words. Table 1 illustrates lexical distributions in this model.

This training scheme allows the frame distributions for high-frequency words a chance to converge on their true distributions, whereas a single 50M word iteration would not. The strategy derives from a variant generalized EM algorithm presented in Neal & Hinton [1998]. In a nutshell, re-estimating the parameters during the course of a single training iteration will still lead to convergence on a maximum-likelihood estimate, provided certain conditions are met. Foremost among these is the requirement that no parameter setting can be prematurely set to zero; this is met by our smoothing strategy. This is not to say that precisely the same strategy, pursued across multiple iterations, would produce a maximum-likelihood estimate; it would not. However, "classical" EM, requiring repeated iteration over the entire training set, is both relatively inefficient and infeasible given our present computational resources.

Dictionary Evaluation

The comparison to frames specified in a dictionary we use was introduced by Brent [1993] and subsequently used by Manning [1993], Ersan & Charniak [1995] and Briscoe & Carroll [1996]. The measure uses *precision* and *recall* to compare the set of induced frames to those in the standard. Precision is the percentage of frames that the system proposes that are correct (i.e. in the standard). Recall is the percentage of frames in the standard that the system

$p_{\text{NP},satisfactory,\text{ADJP},w}$		$p_{\text{VFP},address,\text{NP},w}$	
adverb	prob	noun	prob
entirely	0.17	question	0.086
highly	0.11	issue	0.086
most	0.09	themselves	0.059
very	0.075	issues	0.031
quite	0.055	structure	0.031
wholly	0.032	argument	0.014
uncommonly	0.0037	questions	0.0043
especially	0.0037	electorate	0.0043
...		...	

Table 1: On the left: the eight largest parameters in the lexical choice distribution describing modifying adjectives selected by *satisfactory*. On the right: parallel information for the distribution describing heads of objects of the verb *address*.

	cutoff	TP	FP	FN	prec	rec
Intrans	0.15	20	24	12	0.6471	0.78
NP	0.021	3	5	1	0.9479	0.98
ADJ	0.079	92	0	6	1	0.2
PP	0.045	27	15	6	0.7761	0.89
PART	0.027	60	5	14	0.8077	0.6
VTOP	0.079	83	1	7	0.9	0.56
NP PP	0.040	26	11	10	0.8281	0.84
NP PART	0.0099	68	6	12	0.7	0.53
NP NP	0.036	81	6	8	0.4545	0.38
NP VTOP	0.018	84	1	6	0.9	0.6
VING	0.019	86	3	6	0.625	0.45
NP VING	0.017	93	3	2	0.4	0.5
NP VINF	0.019	99	1	0	0	–
NP ADJ	0.016	85	1	12	0.6667	0.14
PP VTOP	0.014	97	1	1	0.5	0.5
		310	83	103	0.7888	0.75

Table 2: Precision/recall broken down by frame.

proposes. If the results are broken down into true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN), precision is defined as $TP/(TP+FP)$ and recall is $TP/(TP+FN)$. To produce measurements from our system, we must first reduce our distributions to set membership. Brent proposed a stochastic filter for this reduction, consisting of a set of per-frame probability cutoffs, which are applied independently of the lexical head. Although though the independence assumption is certainly dubious, we have adopted this method, without change, except for the introduction of a heuristic for finding the frame cutoffs.

The key property of cutoffs is that they control the tradeoff of precision versus recall. Raising the cutoff will generally produce a higher precision, but lower recall, and contrariwise. As we are neutral about this tradeoff, we set the cutoffs at the crossover point, where the difference in precision and recall changes sign. This is not entirely deterministic, as the measures may cross more than once; in that case, we optimize for the best precision.

For our dictionary, we used *The Oxford Advanced Learner's Dictionary* (Hornby [1985]), also used by Ersan/Charniak and Manning. We reduced our frame set and the dictionary's to a common set, mapping some frames and eliminating others. For evaluation, we selected 200 verbs at random from among those that occurred more than 500 times in the training data; half were used to set the optimal cutoff parameters, and precision and recall were measured with the remainder.

Table shows results broken down by frame. The largest source of error is the intransitive frame. It is not hard to understand why: our robust parsing architecture resolves unparsable constructs as intransitives. In addition to sentences where verbs are not linked up with their complements because of interjections, complex conjunctions or ellipses, this includes frames such as SBAR and WH-complements which are not included in the chunk/phrase grammar. While it would be possible in principle to extract these from the present word collocation statistics, we plan instead to pursue a solution involving extensions in the grammar.

A second major source of error is prepositional phrases. The complementation model embodied in the PCFG does not distinguish complements from adjuncts, and therefore adjunct prepositional phrases are a source of false positives. Thus the NP PP frame is scored as a false positive for the verb *meet*, because the OALD does not list the frame, although the combination appears often in the corpus data. While such frames lead to a loss of precision in the dictionary evaluation, we do not necessarily consider them a flaw in the information learned by the system, since the argument/adjunct distinction is often tenuous, and adjuncts are in many cases lexically conditioned.

Lastly, there are many false negatives for the particle frame and noun plus particle. This is mainly due to disagreements between BNC particle tagging and particle markup in the OALD.

Despite these difficulties, the summary shown in table shows results that are on the whole favorable. In comparison with other work with a comparable number of frames (Manning, Ersan/Charniak), the system is well ahead on recall and well behind on precision. If one takes the sum of precision and recall to be the final performance indicator, than we are slightly ahead: 1.54 vs. 1.44 for Ersan and 1.33

	precision%	recall %	no. of frames
lex PCFG	79	75	15
Briscoe	66	36	159
Charniak	92	52	16
Manning	90	43	19*

Table 3: Type precision/recall comparison. Some of Manning's frames are parameterized for a preposition.

for Manning. Briscoe and Carroll's work, with ten times as many target frames, is so different that the numbers may be regarded as incomparable.

Obviously, precision and recall measured against a standard relies on the completeness and accuracy of that standard. In checking false positives, Ersan and Charniak found that the OALD was incomplete enough to have a serious impact on precision. Symmetrically, false negatives conflate deficiencies in the corpus with poor learning efficiency. It is impossible to say based on table which of the systems is more efficient at learning. While our system shows the best recall, this could be attributed to our having the best training data. Charniak used 40M words of training data, comparable to our 50M, but his data was homogeneous, all taken from the Wall Street Journal. As we will show below, frame usage varies across genres, so the BNC, which includes texts from a wide variety of sources, shows more varied frame usage than the WSJ, and thus provides better data for frame acquisition.

Cross entropy evaluation

The information-theoretic notion of *cross entropy* provides a detailed measure of the similarity of the acquired probabilistic lexicon to the distribution of frames actually exhibited in the corpus (which we call the empirical distribution). The cross entropy of the estimated distribution q with the empirical distribution p obeys the identity

$$CE(p,q) = H(p) + D(p||q)$$

where H is the usual entropy function and D is the relative entropy, or Kullback-Leibler distance. The entropy of a distribution over frames can be conceptualized as the average number of bits required to designate a frame in an ideal code based on the given distribution. In this context, entropy measures the complexity of the observed frame distribution. The relative entropy is the penalty paid in bits when the frame is chosen according to the empirical distribution p, but the code is derived from the model's estimated distribution, q. Relative entropy is always non-negative, and reaches zero only when the two

obs freq			est freq	
imag	natsci	frame	imag	natsci
51	39	NP VTOP	40.4	34.2
21	43	NP	20.7	33.1
13	6	NP NP	8.8	3.9
6	1	NP PP	3.2	4.7
5	1	NP PART	1.7	1.0
2	11	PP	1.8	10.2
1	0	SBAR	0	0
1	0	Intrans	9.3	7.6
2.130	1.913	entropy	2.476	2.423

Table 4: True and estimated frame frequencies for *allow*.

distributions are identical. Our goal, then, is to minimize the relative entropy. For more in-depth discussion of entropy measures, see Cover & Thomas [1991], or any introductory information theory text.

For relative entropy to be finite, the estimated distribution q must be non-zero whenever p is. However, some observed frames are not present in the grammar, for one of two reasons. Some well-known frames such as SBAR require high-level constructs not available in the chunk/phrase grammar and unusual/unorthodox frames turn up in the data, e.g. PART PP PP. Since the model lacks these frames, smoothing against the unlexicalized rules is insufficient. Instead, for all the estimated distributions, we smooth against a Poisson distribution over categories, which assigns non-zero probability to all frames, observed or not. This allows us to spell out the unknown frame using a known finite alphabet, the grammar categories, while retaining a reasonable average length over frames.

For our entropy measurements, we selected three verbs, *allow*, *reach*, and *suffer* and extracted about 200 occurrences of each from portions of the BNC not used for training. Half of each sample was drawn from "imaginative" text and the other half from the natural or applied sciences, as indicated by BNC text mark-up. The true frame for each verb occurrence was marked by a human judge[3]. The empirical distribution was taken as the maximum-likelihood estimate from these frequencies. Tables 4 and 5 indicate the observed frequencies and the entropy of the resulting distributions.

Alongside the observed frequencies, we indicate a set of estimated frequencies. These were generated by taking the 50M word model described above, parsing the test sentences, and extracting the estimated frequencies. The sum of estimated frequencies is gen-

[3]For this judgment, the frame set was unrestricted, i.e. included frames not in the grammar.

obs freq			est freq	
imag	natsci	frame	imag	natsci
63	88	NP	50.1	74.5
13	15	NP PP	5.9	10.9
9	1	PART	5.9	0.8
6	0	PART PP	2.7	0
5	3	PP	6.7	3.4
4	1	Intrans	15.2	6.8
2	0	PART NP	0.5	0
1	0	NP PART	0	0.1
2.0	0.979	entropy	2.101	1.473

obs freq			est freq	
imag	natsci	frame	imag	natsci
41	6	Intrans	34.9	13.4
31	54	PP	27.4	50.5
21	36	NP	18.9	23.0
4	1	NP VTOP	2.1	0.7
3	4	NP PP	0.9	5.2
1.936	1.580	entropy	1.936	1.907

Table 5: True and estimated frame frequencies for *reach* (top) and *suffer* (bottom).

			$D(p\|q)$ for various q		
head, genre		$H(p)$	other genre	50M lex	50M unlex
allow	imag	2.06	0.50	0.40	3.13
	natsci	1.78	0.49	0.42	2.27
reach	imag	1.99	0.91	0.35	1.07
	natsci	0.90	0.37	0.37	1.36
suffer	imag	1.86	0.87	0.24	0.70
	natsci	1.51	0.59	0.37	1.19
	mean	1.68	0.62	0.36	1.62

Table 6: Frame relative entropy for three verbs in two genres. The first column names the lexical head and genre, and the second the entropy (H) of the empirical distribution over frames, p. By empirical distribution we mean the relative frequencies from examples scored by a human judge. Columns three through five give the relative entropy $D(p\|q)$ for various related distributions. In column three, q is the empirical frame distribution for the same head, but with the complementary genre. In column four q is the (genre-independent) distribution derived from the 50M word lexicalized model. Column five uses the unlexicalized frame distribution derived from the 50M model, i.e. a distribution insensitive to the head verb. Lower relative entropy is better.

erally less than the observed frequencies due to tagging errors, parse failures, and frequency assigned to frames not shown in the tables. However, an eyeball inspection of the tables shows that the parser does a good job of reproducing the target distribution.

One striking feature in the tables is the variation across genre. In particular, *suffer* used in the imaginative genre shows a very different distribution than *suffer* in the natural sciences. A chi-squared test applied to each pair indicates that the samples come from distinct distributions (confidence $> 95\%$).

The column labeled "50M lex" in Table 6 provides a quantitative measure of the agreement between the 50M word combined model and the empirical distributions for the three verbs in two genres in the form of relative entropy. The first column repeats the entropy of the data distributions. For purposes of comparison, the second column indicates the relative entropy of one data distribution with the other data distribution filling the role of the estimated distribution (i.e. q) in the discussion above. The relative entropy is lower when the estimated distribution is used for q than when the data distribution for the other genre is used for q in each case but one, where the figures are the same. This suggests the combined model contains fairly good overall distributions.

To numerically evaluate whether the system was able to learn the distribution exhibited in a given collection of sentences, we tuned the lexicon by parsing the test sentences for each genre separately with the 50M word model, extracting the frequencies, and estimating the distribution from these. The results are the column 4 labeled "50M lexicalized extraction" in 7. The following columns give the same figures for freqency extraction with other models. Extraction with the large lexicalized model gives the best results, and gives better relative entropy than the 50M lexicalilazed model itself (in column 2). Notice that only the distributions estimated with the two 50M models are better than the 50M lexicalized model, though the unlexicalized one is only marginally better. In this sense, only the 50M lexicalized parser proves to be a good enough parser for genre tuning. Notice that with this model, tuning in no case gives worse relative entropy, and in five out of six cases give an improvement.

Notice also that relative entropy for the distributions obtained by tuning with the 50M model are a good deal lower than the cross-genre figures from Table 6. This suggests that if we wanted to have a good probabilistic lexicon for, say, the imaginative genre, we would be better off using the automatic extraction procedure on data drawn from that genre than using a *perfect* parser (or a lexicographer) on data drawn from some other genre, such as the natural sciences. This provides a calibration of the accuracy of the lexicalized parser's estimates, and conversely demonstrates that words are not used in the same

		$D(p\|q)$				
head, genre		50M lex mod	50M lex extr	5M lex extr	50M unl. extr	5M unl. extr
allow	imag	0.40	0.32	1.32	0.47	1.32
	natsci	0.42	0.28	0.28	0.52	0.86
reach	imag	0.35	0.35	0.63	0.32	0.63
	natsci	0.37	0.19	0.34	0.28	0.34
suffer	imag	0.24	0.11	0.38	0.12	0.38
	natsci	0.37	0.20	0.88	0.34	0.88
	mean	0.36	0.24	0.64	0.34	0.74

Table 7: Relative entropy of distributions estimated by parsing the test sentences with various models, and using the Inside-outside algorithm to produce estimated distributions q. The first column names empirical distributions p. The second column repeats relative entropy for the 50M lexicalized model from the previous table. The third gives relative entropy where q is obtained by parsing and estimating frequencies in the test sentences with the 50M lexicalized model. The following columns give the corresponding figures for a q obtained by following the same procedure with a 5M word lexicalized model, a 50M word unlexicalized model, and a 5M word unlexicalized model.

way in different genres.

Optimal parses

Although identifying a unique parse does not play a role in our experiment, it is potentially useful for applications. A simple criterion is to pick a parse with maximal probability; this is identified in a parse forest by iterating from terminal nodes, multiplying child probabilities and the local node weight at *and*-nodes (chart edges), and choosing a child with maximal probability at *or*-nodes (chart constituents). Figures 1 and 4 give examples of maximal probability probability parses.

Other optimality criteria can be defined. The structure on noun chunks is often highly ambiguous, because of bracketing and part of speech ambiguities among modifiers. For many purposes, the internal structure of an noun chunk is irrelevant; one just wants to identify the chunk. From this point of view, a probability estimate which considers just one analysis might underestimate the probability of a noun chunk. In what we call a sum-max parse, probabilities are summed within chunks by the inside algorithm. Above the chunk level, a highest-probability tree is computed, as described above.

Notes on the implementation and parsing times

Software is implemented in C++. The parser used for the bootstrap phase is a vanilla CFG chart parser, operating bottom-up with top-down predictive filtering. Chart entries are assigned probabilities using the unlexicalized PCFG, and the lexicalized frequencies are found by carrying out a modified inside-outside algorithm which simulates lexicalization of the chart.

In the iterative training phase, an unlexicalized context-free skeleton is found with the same parser. We transform this into its lexicalized form—categories become $\langle w, n \rangle$ pairs and rules acquire lexical heads—and carry out the standard inside-outside using the more elaborate head-lexicalized PCFG model. Average speed of the parser during iterative training, including parsing, probability calculation, and recording observations, is 10.4 words per second on a Sun SPARC-20. The memory requirements for a model generated from a 5M word segment are about 90Mbyte. The upshot of all this is that we can train about 1M words per day on one machine, and a single 5M word iteration requires one machine work week.

Discussion

We believe the formalism and methodology described here have the following advantages:

- The grammar is under the control of the computational linguist and is of a familiar kind, making it possible to incorporate standard linguistic analyses, and making results interpretable in terms of linguistic theory. In contrast, approaches where context free rules are learned are likely to produce structures which are uninterpretable in terms of linguistic theory and practice.
- Because of the context free framework, efficient parsing algorthims (chart parsing) and probabilistic algorithms (the inside-outside algorithm) can be applied. With an efficient implementation, this makes it possible to construct representations of all the tree analyses for the sentences in corpora on the scale of ten to a hundred million words, and to map such a corpus to a probabilistic lexicon.
- With the robustness introduced by the state model, almost all sentences in the corpus can be parsed.
- The model assigns probabilities to sentences and trees, which is useful for applications independent of the lexicon-induction problem discussed here.
- The word-selection model, which threads a word bigram model through head relations in the syntactic tree, allows a large body of word-word collocations to be learned from the corpus, and put to use in weighting of competing analyses.

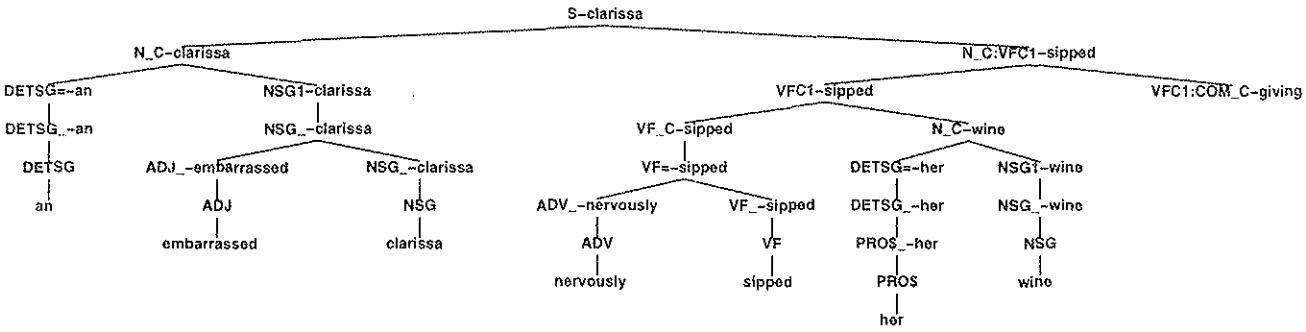


Figure 4: The first part of maximum probability parse.

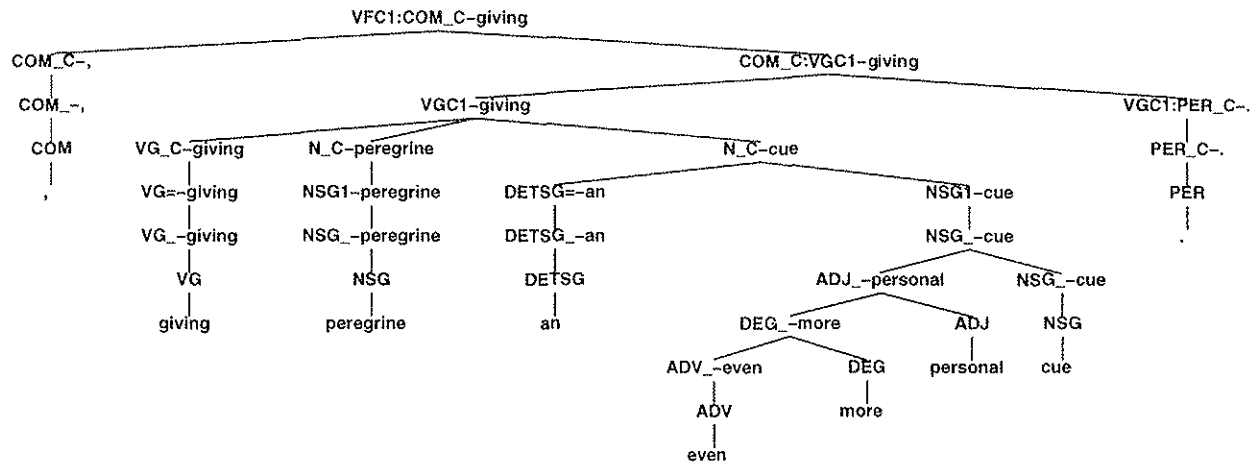


Figure 5: The second part.

- The valence information learned, rather than being simply a set of subcategorization frames, is a probability distribution which reflects the freqency of frames in a given training sample, and which can be plugged back into the parser and used to analyze further text.

Some of these benefits are purchased at the cost of a lack of sophistication in the grammar formalism, compared to constraint-based formalisms used in contemporary computational linguistics. This compromise is made in order to make large-scale experiments achievable; our interest is in conducting scientific experiments—observational and modeling experiments—with large bodies of language use. It is natural that this should require incorporating approximations in computational models. Notably, the compromises made in our approach are not so severe that the grammatical analyses identified and the probability parameters learned are out of touch with linguistic reality. This is in contrast to the situation with other approaches using similar mathematical methods, such as terminal-string n-gram modeling.

Conclusion

We have presented a statistically-based method for valence induction, based on the idea of automatic tuning of the probability parameters of a grammar. On the standard precision/recall measures, our system performs better on precision, worse on recall, and on the whole somewhat better than other published systems. We have provided a more precise evaluation via entropy measures, showing that the model learns efficiently and builds accurate models of frame distributions. The cross-domain entropy of the data frame distributions provides numerical evidence that frame usage varies across domains, similar to word usage. This, in turn, suggests that automatic acquisition and stochastic tuning are a must for large-scale NLP applications and computational linguistic models.

Bibliography

Abney, S. [1991], "Parsing by Chunks," in *Views on Phrase Structure*, D. Bouchard & K. Leffel, eds., Kluwer Academic Publishers.

———— [1995], "Chunks and dependencies: Bringing processing evidence to bear on syntax," in *Linguistics and Computation*, Jennifer S. Cole, Georgia M. Green & Jerry L. Morgan, eds., CSLI Publications.

BNC Consortium [1995], *The British National Corpus*, Oxford University, http://info.ox.ac.uk/bnc/.

Baker, J. K. [1979], "Trainable grammars for speech recognition," *Proceedings of the Spring Conference of the Acoustical Society of America*, Cambridge, MA.

Baum, L. E. & Sell, G. R. [1968], "Growth Transformations for Functions on Manifolds," *Pacific Journal of Mathematics* 27.

Brent, M. R. [1993], "From Grammar to Lexicon: Unsupervised Learning of Lexical Syntax," *Computational Linguistics* 19, 243–262.

Briscoe, T. & Carroll, J. [1996], "Automatic Extraction of Subcategorization from Corpora," *MS* , http://www.cl.cam.ac.uk/users/ejb/.

Charniak, E. [1993], *Statistical Language Learning*, MIT, Cambridge, MA.

———— [1995], "Parsing with Context-free Grammars and Word Statistics," Department of Computer Science, Brown University, Technical Report CS-95-28.

Cover, T. M. & Thomas, J. A. [1991], *Elements of Information Theory*, John Wiley and Sons, Inc., New York.

Dempster, A. P., Laird, N. M. & Rubin, D. B. [1977], "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistics Society* 39, 1–38, Series B.

Ersan, M. & Charniak, E. [1995], "A Statistical Syntactic Disambiguation Program and what it learns," Brown CS Tech Report CS-95-29.

Hornby, A. S. [1985], *Oxford Advanced Learner's Dictionary of Current English*, Oxford University Press, Oxford, 4th Ed..

Jackendoff, R. [1977], $\overline{X}$ *syntax: A study in phrase structure.*, MIT Press, Cambridge, MA.

Katz, S. M. [1980], "Estimation of probabilities from sparse data for the language model component of a speech recognizer," *IEEE Transactions on Acoustics, Speech and Signal Processing* 35, 400–401.

Manning, C. [1993], "Automatic acquisition of a large subcategorization dictionary from corpora," *Proceedings of the 31st Annual Meeting of the ACL.*

Neal, R. M. & Hinton, G. E. [1998], "A New View of the EM Algorithm that Justifies Incremental and Other Variants," in *Learning in Graphical Models*, Michael I. Jordan, ed., Kluwer Academic Press.

Ney, H., Essen, U. & Kneser, R. [1994], "On structuring probabilistic dependences in stochastic language modelling," *Computer Speech and Language* 8, 1–38.

Measures for corpus similarity and homogeneity

Adam Kilgarriff*
ITRI, University of Brighton

Tony Rose
Canon Research Centre Europe

Abstract

How similar are two corpora? A measure of corpus similarity would be very useful for NLP for many purposes, such as estimating the work involved in porting a system from one domain to another. First, we discuss difficulties in identifying what we mean by 'corpus similarity': human similarity judgements are not fine-grained enough, corpus similarity is inherently multi-dimensional, and similarity can only be interpreted in the light of corpus homogeneity. We then present an operational definition of corpus similarity which addresses or circumvents the problems, using purpose-built sets of "known-similarity corpora". These KSC sets can be used to evaluate the measures. We evaluate the measures described in the literature, including three variants of the information theoretic measure 'perplexity'. A χ^2-based measure, using word frequencies, is shown to be the best of those tested.

The Problem

How similar are two corpora? The question arises on many occasions. In NLP, many useful results can be generated from corpora, but when can the results developed using one corpus be applied to another? How much will it cost to port an NLP application from one domain, with one corpus, to another, with another? For linguistics, does it matter whether language researchers use this corpora or that, or are they similar enough for it to make no difference? There are also questions of more general interest. Looking at British national newspapers: is the Independent more like the Guardian or the Telegraph?[1]

What are the constraints on a measure for corpus similarity? The first is simply that its findings correspond to unequivocal human judgements. It must match our intuition that, eg, a corpus of syntax papers is more like one of semantics papers than one of shopping lists. The constraint is key but is weak. Direct human intuitions on corpus similarity are not easy to come by, firstly, because large corpora, unlike coherent texts, are not the sorts of things people read, so people are not generally in a position to have any intuitions about them. Secondly, a human response to the question, "how similar are two objects", where those objects are complex and multi-dimensional, will themselves be multi-dimensional: things will be similar in some ways and dissimilar in others. To ask a human to reduce a set of perceptions about the similarities and differences between two complex objects to a single figure is an exercise of dubious value.

This serves to emphasise an underlying truth: corpus similarity is complex, and there is no absolute answer to "is Corpus 1 more like Corpus 2 than Corpus 3?". All there are, are possible measures which serve particular purposes more or less well. Given the task of costing the customisation of an NLP system, produced for one domain, to another, a corpus similarity measure is of interest insofar as it predicts how long the porting will take. It could be that a measure which predicts well for one NLP system, predicts badly for another. It can only be established whether a measure correctly predicts actual costs, by investigating actual costs.[2]

Having struck a note of caution, we now proceed on the hypothesis that there is a single measure which corresponds to pre-theoretical intuitions about 'similarity' and which is a good indicator of many properties of interest – customisation costs, the likelihood that linguistic findings based on one corpus apply to another, etc. We would expect the limitations of the hypothesis to show through at some point, when different measures are shown to be suited to different purposes, but in the current situation, where there has been almost no work

*Kilgarriff's part of the work was undertaken under EPSRC grant GR/K/18931

[1]The work presented here develops and extends that presented in Kilgarriff (1997).

[2]Cf. Ueberla (1997), who looks in detail at the appropriateness of perplexity as a measure of task difficulty for speech recognition, and finds it wanting.

Corpus 1	Corpus 2	Distance	Interpretation
equal	equal	equal	same language variety/ies
equal	equal	high	different language varieties
high	low	high	corpus 2 is homogeneous and falls within the range of 'general' corpus 1
high	low	higher	corpus 2 is homogeneous and falls outside the range of 'general' corpus 1
high	high	low	impossible
low	low	a bit lower	overlapping; share some varieties
high	high	a bit lower	similar varieties

Table 1: **Interactions between homogeneity and similarity:** a similarity measure can only be interpreted with respect to homogeneity.
High means a large distance between corpora, or large within-corpus distances, so the corpus is heterogeneous/corpora are dissimilar; **low**, that the distances are low, so the corpus is homogeneous/corpora are similar. **High, low** and **equal** are relative to the other columns in the same row, so, in row 2, 'equal' in the first two columns reads that the within-corpus distance (homogeneity) of Corpus 1 is roughly equal to the within-corpus distance of Corpus 2, and 'high' in the Distance column reads that the distance between the corpora is substantially higher than these within-corpus distances.

on the question, it is a good starting point.

Similarity and homogeneity

How homogeneous is a corpus? The question is both of interest in its own right, and is a preliminary to any quantitative approach to corpus similarity. In its own right, because a sublanguage corpus, or one containing only a specific language variety, has very different characteristics to a general corpus (Biber, 1993) yet it is not obvious how a corpus's position on this scale can be assessed. As a preliminary to measuring corpus similarity, because it is not clear what a measure of similarity would mean if a homogeneous corpus (of, eg, software manuals) was being compared with a heterogeneous one (eg. Brown). Ideally, the same measure can be used for similarity and homogeneity, as then, Corpus 1/Corpus 2 distances will be directly comparable with heterogeneity (or "within-corpus distances") for Corpus1 and Corpus2. This is the approach adopted here.

Not all combinations of homogeneity and similarity scores are logically possible. A corpus cannot be much more similar to something else than it is to itself. Some of the permutations, and their interpretations, are shown in Table 1.

The last two lines in the table point to the differences between general corpora and specific corpora. High within-corpus distance scores will be for general corpora, which embrace a number of language varieties. Corpus similarity between general corpora will be a matter of whether all the same language varieties are represented in each corpus, and in what proportions. Low within-corpus distance scores will typically relate to corpora of a single language variety, so here, scores may be interpreted as a measure of the distance between the two varieties.

Related Work

There is very little work which explicitly aims to measure similarity between corpora. Johansson and Hofland (1989) aim to find which genres, within the LOB corpus, most resemble each other. They take the 89 most common words in the corpus, find their rank within each genre, and calculate the Spearman rank correlation statistic ('spearman').

Rose, Haddock, and Tucker (1997) explore how performance of a speech recognition system varies with the size and specificity of the training data used to build the language model. They have a small corpus of the target text type, and experiment with 'growing' their seed corpus by adding more same-text-type material. They use spearman and log-likelihood (Dunning, 1993) as measures to identify same-text-type corpora. Spearman is evaluated below.

There is a large body of work aiming to find words which are particularly characteristic of one text, or corpus, in contrast to another, in various fields including linguistic variation studies (Rayson, Leech, and Hodges, 1997), author identification (Mosteller and Wallace, 1964) and information retrieval (Salton, 1989; Dunning, 1993). Biber (1988, 1995) explores and quantifies the differences between corpora from a sociolinguistic perspective. While all of this work touches on corpus-similarity, none looks at is as a topic of itself.

Sekine (1997) explores the domain dependence of parsing. He parses corpora of various text genres and counts the number of occurrences of each subtree of

depth one. This gives him a subtree frequency list for each corpus, and he is then able to investigate which subtrees are markedly different in frequency between corpora. Such work is highly salient for customising parsers for particular domains. Subtree frequencies could readily replace word frequencies for the frequency-based measures below.

In information-theoretic approaches, perplexity is a widely-used measure. Given a language model and a corpus, perplexity "is, crudely speaking, a measure of the size of the set of words from which the next word is chosen given that we observe the history of ... words" (Roukos, 1996). Perplexity is most often used to assess how good a language modelling strategy is, so is used with the corpus held constant. Achieving low perplexity in the language model is critical for high-accuracy speech recognition, as it means there are fewer high-likelihood candidate words for the speech signal to be compared with.

Perplexity can be used to measure a property akin to homogeneity if the language modelling strategy is held constant and the corpora are varied. In this case, perplexity is taken to measure the intrinsic difficulty of the speech recognition task: the less constraint the domain corpus provides on what the next word might be, the harder the task. Thus Roukos (1996) presents a table in which different corpora are associated with different perplexities.

Perplexity measures are evaluated below.

"Known-Similarity Corpora"

A "Known-Similarity Corpora" (KSC) set is built as follows: two reasonably distinct text types, A and B, are taken. Corpus 1 comprises 100% A; Corpus 2, 90% A and 10% B; Corpus 3, 80% A and 20% B; and so on. We now have at our disposal a set of fine-grained statements of corpus similarity: Corpus 1 is more like Corpus 2 than Corpus 1 is like Corpus 3. Corpus 2 is more like Corpus 3 than Corpus 1 is like Corpus 4, etc. Alternative measures can now be evaluated, by determining how many of these 'gold standard judgements' they get right. For a set of n Known-Similarity Corpora there are

$$\sum_{i=1}^{n}(n-i)\left(\frac{i(i+1)}{2}-1\right)$$

gold standard judgements (see Appendix for proof) and the ideal measure would get all of them right. Measures can be compared by seeing what percentage of gold standard judgements they get right.

Two limitations on the validity of the method are, first, there are different ways in which corpora can be different. They can be different because each represents one language variety, and these varieties are different, or because they contain different mixes, with some of the same varieties. The method only directly addresses the latter model.

Second, if the corpora are small and the difference in proportions between the corpora is also small, it is not clear that all the 'gold standard' assertions are in fact true. There may be a finance supplement in one of the copies of the Guardian in the corpus, and one of the copies of Accountancy may be full of political stories: perhaps, then, Corpus 3 *is* more like Corpus 5 than Corpus 4. This was addressed by selecting the two text types with care so they were similar enough so the measures were not 100% correct yet dissimilar enough to make it likely that all gold-standard judgements were true, and by ensuring there was enough data and enough KSC-sets so that oddities of individual corpora did not obscure the picture of the best overall measure.

Measures

All the measures use spelt forms of words. None make use of linguistic theories. Comments on an earlier version of the paper included the suggestion that lemmas, or word senses, or syntactic constituents, were more appropriate objects to count and perform computations on than spelt forms. This would in many ways be desirable. However there are costs to be considered. To count, for example, syntactic constituents requires, firstly, a theory of what the syntactic constituents are; secondly, an account of how they can be recognised in running text; and thirdly, a program which performs the recognition. Shortcomings or bugs in any of the three will tend to degrade performance, and it will not be straightforward to allocate blame. Different theories and implementations are likely to have been developed with different varieties of text in focus, so the degradation may well effect different text types differentially. Moreover, practical users of a corpus-similarity measure cannot be expected to invest energy in particular linguistic modules and associated theory. To be of general utility, a measure should be as theory-neutral as possible.

While we are planning to explore counts of lemmas and part-of-speech categories, in these experiments we consider only raw word-counts.

Word Frequency measures

Two word frequency measures were considered. For each, the statistic did not dictate which words should be compared across the two corpora. In a preliminary investigation we had experimented with taking the most frequent 10, 20, 40 ... 640, 1280, 2560, 5120 words in the union of the two corpora as data points, and had

achieved the best results with 320 or 640. For the experiments below, we used the most frequent 500 words.

Both word-frequency measures can be directly applied to pairs of corpora, but only indirectly to measure homogeneity. To measure homogeneity:

1. divide the corpus into 'slices';
2. create two subcorpora by randomly allocating half the slices to each;
3. measure the similarity between the subcorpora;
4. iterate with different random allocations of slices;
5. calculate mean and standard deviation over all iterations.

Wherever similarity and homogeneity figures were to be compared, the same method was adopting for calculating corpus similarity, with one subcorpus comprising a random half of Corpus 1, the other, a random half of Corpus 2.

Spearman Rank Correlation Co-efficient

Ranked wordlists are produced for Corpus 1 and Corpus 2. For each of the n most common words, the difference in rank order between the two corpora is taken. The statistic is then the normalised sum of the squares of these differences,

$$1 - \frac{6\Sigma d^2}{n(n^2-1)}$$

Comment Spearman is easy to compute and is independent of corpus size: one can directly compare ranked lists for large and small corpora. However there was an *a priori* objection to the statistic. For very frequent words, a difference of rank order is highly significant: if *the* is the most common word in corpus 1 but only 3rd in corpus 2, this indicates a high degree of difference between the genres. At the other end of the scale, if *bread* is in 400th position in the one corpus and 500th in the other, this is of no significance, yet Spearman counts the latter as far more significant than the former.

χ^2

For each of the n most common words, we calculate the number of occurrences in each corpus that would be expected if both corpora were random samples from the same population. If the size of corpora 1 and 2 are N_1, N_2 and word w has observed frequencies $o_{w,1}, o_{w,2}$, then expected value $e_{w,1} = \frac{N_1 \times (o_{w,1}+o_{w,2})}{N_1+N_2}$ and likewise for $e_{w,2}$; then

$$\chi^2 = \Sigma \frac{(o-e)^2}{e}$$

Comment The inspiration for the statistic comes from the χ^2-test for statistical independence. As Kilgarriff (1996) shows, the statistic is not in general appropriate for hypothesis-testing in corpus linguistics: a corpus is never a random sample of words, so the null hypothesis is of no interest. But once divested of the hypothesis-testing link, χ^2 is suitable. The $(o-e)^2/e$ term gives a measure of the difference in a word's frequency between two corpora, and, while the measure tends to increase with word frequency, in contrast to the raw frequencies it does not increase by orders of magnitude.

The measure does not directly permit comparison between corpora of different sizes.

Perplexity and Cross-entropy

From an information-theoretic point of view, *prima facie*, entropy is a well-defined term capturing the informal notion of homogeneity, and the cross-entropy between two corpora captures their similarity. Entropy is not a quantity that can be directly measured. The standard problem for statistical language modelling is to aim to find the model for which the cross-entropy of the model for the corpus is as low as possible. For a perfect language model, the cross-entropy would be the entropy of the corpus (Church and Mercer, 1993; Charniak, 1993).

With language modelling strategy held constant, the cross-entropy of a language model (LM) trained on Corpus 1, as applied to Corpus 2, is a similarity measure. The cross-entropy of the LM based on nine tenths of Corpus 1, as applied to the other 'held-out' tenth, is a measure of homogeneity. We standardised on the 'tenfold cross-validation' method for measures of both similarity and homogeneity: that is, for each corpus, we divided the corpus into ten parts[3] and produced ten LMs, using nine tenths and leaving out a different tenth each time. (Perplexity is the log of the cross-entropy of a corpus with itself: measuring homogeneity as self-similarity is standard practice in information theoretic approaches.)

To measure homogeneity, we calculated the cross-entropy of each of these LMs as applied to the left-out tenth, and took the mean of the ten values. To measure similarity, we calculated the cross-entropy of each of the Corpus 1 LMs as applied to a tenth of Corpus 2 (using a different tenth each time). We then repeated the procedure with the roles of Corpus 1 and Corpus 2 reversed, and took the mean of the 20 values.

[3]For the KSC corpora, we ensured that each tenth had an appropriate mix of text types, so that, eg, each tenth of a corpus comprising 70% Guardian, 30% BMJ, also comprised 70% Guardian, 30% BMJ.

All LMs were trigram models. All LMs were produced and calculations performed using the CMU/Cambridge toolkit (Rosenfeld, 1995).

The treatment of words in the test material but not in the training material was critical to our procedure. It is typical in the language modelling community to represent such words with the symbol UNK, and to calculate the probability for the occurrence of UNK in the test corpus using one of three main strategies.

Closed vocabulary The vocabulary is defined to include all items in training and test data. Probabilities for those items that occur in training but not test data, the 'zerotons', are estimated by sharing out the probability mass initially assigned to the singletons and doubletons to include the zerotons.

Open, type 1 The vocabulary is chosen independently of the training and test data, so the probability of UNK may be estimated by counting the occurrence of unknown words in the training data and dividing by N (the total number of words).

Open, type 2 The vocabulary is defined to include all and only the training data, so the probability of UNK cannot be estimated directly from the training data. It is estimated instead using the discount mass created by the normalisation procedure.

All three strategies were evaluated.

Data

All KSC sets were subsets of the British National Corpus (BNC)[4]. A number of sets were prepared as follows.

For those newspapers or periodicals for which the BNC contained over 300,000 running words of text, word frequency lists were generated and similarity and homogeneity were calculated (using χ^2). We then selected pairs of text types which were moderately distinct, but not too distinct, to use to generate KSC sets. (In initial experiments, more highly distinct text types had been used, but then both Spearman and χ^2 had scored 100%, so 'harder' tests involving more similar text types were selected.)

For each pair a and b, all the text in the BNC for each of a and b was divided into 10,000-word tranches. These tranches were randomly shuffled and allocated as follows:

first 10 of a	into	b0a
next 9 of a, first 1 of b	into	b1a
next 8 of a, next 2 of b	into	b2a
next 7 of a, next 3 of b	into	b3a
...		

until either the tranches of a or b ran out, or a complete 11-corpus KSC-set was formed. A sample of KSC sets are available on the web.[5] There were 21 sets containing between 5 and 11 corpora. The method ensured that the same piece of text never occurred in more than one of the corpora in a KSC set.

The text types used were:
Accountancy (`acc`); The Art Newspaper (`art`); British Medical Journal (`bmj`); Environment Digest (`env`); The Guardian (`gua`); The Scotsman (`sco`); and Today ('low-brow' daily newspaper, `tod`).

To the extent that some text types differ in content, whereas others differ in style, both sources of variation are captured here. Accountancy and The Art Newspaper are both trade journals, though in very different domains, while The Guardian and Today are both general national newspapers, of different styles.

Results

For each KSC-set, for each gold-standard judgement, the 'correct answer' was known, eg., "the similarity 1,2 is greater than the similarity 0,3". A given measure either agreed with this gold-standard statement, or disagreed. The percentage of times it agreed is a measure of the quality of the measure. Results for the cases where all four measures were investigated are presented in Table 2.

	spear	χ^2	closed	type 1	type 2
KSC-set					
acc_gua	93.33	91.33	82.22	81.11	80.44
art_gua	95.60	93.03	84.00	83.77	84.00
bmj_gua	95.57	97.27	88.77	89.11	88.77
env_gua	99.65	99.31	87.07	84.35	86.73

Table 2: Comparison of four measures

The word frequency measures outperformed the perplexity ones. It is also salient that the perplexity measures required far more computation: ca. 12 hours on a Sun, as opposed to around a minute.

Spearman and χ^2 were tested on all 21 KSC-sets, and χ^2 performed better for 13 of them, as shown in Table 3

	spear	χ^2	tie	total
Highest score	5	13	3	21

Table 3: Spearman/χ^2 comparison on all KSCs

[4] http://info.ox.ac.uk/bnc

[5] http://www.itri.bton.ac.uk/~Adam.Kilgarriff/KSC/

The difference was significant (related t-test: t=4.47, 20DF, significant at 99.9% level). χ^2 was the best of the measures compared.

Conclusions and further work

We have argued that computational linguistics is in urgent need of measures for corpus similarity and homogeneity. Without one, it is very difficult to talk accurately about the relevance of findings based on one corpus, to another, or to predict the costs of porting an application to a new domain. We note that corpus similarity is complex and multifaceted, and that different measures might be required for different purposes. However, given the paucity of other work in the field, at this stage it is enough to seek a single measure which performs reasonably.

The Known-Similarity Corpora method for evaluating corpus-similarity measures was presented, and measures discussed in the literature were compared using it. For the corpus-size used and this approach to evaluation, χ^2 and Spearman both performed better than any of three cross-entropy measures. These measures have the advantage that they are cheap and straightforward to compute. χ^2 outperformed Spearman.

Further work is to include:

- developing a scale-independent χ^2-based statistic
- investigating a 2-dimensional measure for similarity, with one dimension for closed-class words and another for open-class words, to see whether differences in style and in domain can be distinguished
- evaluation of a log-likelihood-based measure, and of different vocabulary-sizes for open models. Then it will be possible to compare the 500-word measure for spearman and χ^2 more directly with the perplexity measures
- gathering data on the actual costs of porting systems, for correlation with results given by similarity measures
- comparing the method with Biber's feature-set and analysis.

References

Biber, Douglas. 1988. *Variation across speech and writing.* Cambridge University Press.

Biber, Douglas. 1993. Using register-diversified corpora for general language studies. *Computational Linguistics*, 19(2):219–242.

Biber, Douglas. 1995. *Dimensions in Register Variation.* Cambridge University Press.

Charniak, Eugene. 1993. *Statistical Language Learning.* MIT Press, Cambridge, Mass.

Church, Kenneth W. and Robert L. Mercer. 1993. Introduction to the special issue on computational linguistics using large corpora. *Computational Linguistics*, 19(1):1–24.

Dunning, Ted. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74.

Johansson, Stig and Knut Hofland, editors. 1989. *Frequency Analysis of English vocabulary and grammar, based on the LOB corpus.* Clarendon, Oxford.

Kilgarriff, Adam. 1996. Which words are particularly characteristic of a text? a survey of statistical approaches. In *Language Engineering for Document Analysis and Recognition*, pages 33–40, Brighton, England, April. AISB Workshop Series.

Kilgarriff, Adam. 1997. Using word frequency lists to measure corpus homogeneity and similarity between corpora. In *Proceedings, ACL SIGDAT workshop on very large corpora*, pages 231–245, Beijing and Hong Kong, August.

Mosteller, Frederick and David L. Wallace. 1964. *Applied Bayesian and Classical Inference - The Case of The Federalist Papers.* Springer Series in Satistics, Springer-Verlag.

Rayson, Paul, Geoffrey Leech, and Mary Hodges. 1997. Social differentiation in the use of English vocabulary: some analysis of the conversational component of the British National Corpus. *International Journal of Corpus Linguistics*, 2(1):133–152.

Rose, Tony, Nicholas Haddock, and Roger Tucker. 1997. The effects of corpus size and homogeneity on language model quality. In *Proceedings, ACL SIGDAT workshop on very large corpora*, pages 178–191, Beijing and Hong Kong, August.

Rosenfeld, Ronald. 1995. The CMU Statistical Language Modelling Toolkit and its use in the 1994 ARPA CSR Evaluation. In *Proc. Spoken Language Technology Workshop*, Austin, Texas.

Roukos, Salim, 1996. *Language Representation*, chapter 1.6. National Science Foundation and European Commission, `www.cse.ogi/CSLU/HLTsurvey.html`.

Salton, Gerard. 1989. *Automatic Text Processing.* Addison-Wesley.

Sekine, Satshi. 1997. The domain dependence of parsing. In *Proc. Fifth Conference on Applied Natural Language Processing*, pages 96–102, Washington DC, April. ACL.

Ueberla, Joerg. 1997. Towards an improved performance measure for language models. Technical Report DERA/CIS/CIS5/TR97426, DERA. `cmp-lg/9711009`.

Appendix

The proof is based on the fact that the number of similarity judgements is the triangle number of the number of corpora in the set (less one), and that each new similarity judgement introduces a triangle number of gold standard judgements (once an ordering which rules out duplicates is imposed on gold standard judgements).

- A KSC set is ordered according to the proportion of text of type 1. Call the corpora in the set 1...n.
- A similarity judgement ('sim') between a and b (a,b) compares two corpora. To avoid duplication, we stipulate that a<b. Each sim is associated with a number of steps of difference between the corpora: dif(a,b)=b-a.
- A gold standard judgement ('gold') compares two sims; there is only a gold between a,b and c,d if a<b and c<d (as stipulated above) and also if a<=c, b>=d, and not (a=c and b=d). Each four-way comparison can only give rise to zero or one gold, as enforced by the ordering constraints. Each gold has a difference of difs ('difdif') of (b-a)-(d-c) (so, if we compare 3,5 with 3,4, difdif=1, but where we compare 2,7 with 3,4, difdif = 4). difdif(X,Y) = dif(X)-dif(Y).
- Adding an nth corpus to a KSC set introduces n-1 sims. Their difs vary from 1 (for (n-1),n) to n-1 (for 1,n).
- The number of golds with a sim of dif m as first term is a triangle number less one, $\sum_{i=2}^{m} i$ or $\frac{m(m+1)}{2} - 1$ For example, for 2,6 (dif=4) there are 2 golds of difdif 1 (eg with 2,5 and 3,6), 3 of difdif 2 (with 2,4, 3,5, 4,6), and 4 of difdif 3 (with 2,3, 3,4, 4,5, 5,6).
- With the addition of the nth corpus, we introduce n-1 sims with difs from 1 to n-1, so we add $\sum_{i=1}^{n-1} \frac{i(i+1)}{2} - 1$ golds. For the whole set, there are $\sum_{i=1}^{n} \sum_{j=1}^{i-1} \frac{j(j+1)}{2} - 1$ and collecting up repeated terms gives $\sum_{i=1}^{n} (n-i)(\frac{i(i+1)}{2} - 1)$

Word-Sense Distinguishability and Inter-Coder Agreement *

Rebecca Bruce† and **Janyce Wiebe‡**
†Department of Computer Science
University of North Carolina at Asheville
Asheville, NC 28804-8511
‡Department of Computer Science
New Mexico State University, Las Cruces, NM 88003
bruce@cs.unca.edu, wiebe@cs.nmsu.edu

Abstract

It is common in NLP that the categories into which text is classified do not have fully objective definitions. Examples of such categories are lexical distinctions such as part-of-speech tags and word-sense distinctions, sentence level distinctions such as phrase attachment, and discourse level distinctions such as topic or speech-act categorization. This paper presents an approach to analyzing the agreement among human judges for the purpose of formulating a refined and more reliable set of category designations. We use these techniques to analyze the sense tags assigned by five judges to the noun *interest*. The initial tag set is taken from Longman's Dictionary of Contemporary English. Through this process of analysis, we automatically identify and assign a revised set of sense tags for the data. The revised tags exhibit high reliability as measured by Cohen's κ. Such techniques are important for formulating and evaluating both human and automated classification systems.

Introduction

It is common in Natural Language Processing (NLP) that the categories into which text is classified do not have fully objective definitions. Examples of such categories are lexical distinctions such as part-of-speech tags and word-sense distinctions, sentence level distinctions such as phrase attachment, and discourse level distinctions such as topic or speech-act categorization. This paper presents an approach to analyzing the agreement among human judges for the purpose of formulating a refined and more reliable set of category designations.

We performed a case study of the classification process, involving multiple judges performing a word-sense disambiguation task. Table 1 presents the data for two judges assigning one of six senses to each instance of *interest* used as a noun in the corpus. The data is represented as a *contingency table*, often referred to as a *confusion* matrix; it depicts the "confusion" among the judges' classifications. Evidence of confusion among the classifications in Table 1 can be found in the marginal totals, n_{i+} and n_{+j}, where i and j range from 1 to 6. We see that, on average, judge A has a higher preference for senses 1 and 3 than judge E does, while judge E has a higher preference for sense 2 than judge A does. These *biases* are one aspect of *agreement* (or the lack of it) among judges.

A second aspect of agreement is the extent to which judges agree on the tags of individual words (*category distinguishability*). We see from the diagonal frequencies in Table 1 that these judges agree on 2097 out of 2369 of them, which is 88.5% of the individual tags.

Cohen (1960) proposed the coefficient of agreement, κ, for measuring the agreement between two judges. κ compares the actual agreement to that which would be expected if the decisions made by each judge were statistically independent (i.e., "chance agreement"). A number of previous studies have used κ to evaluate inter-coder reliability (e.g., Carletta 1996, Litman & Passonneau 1995; Moser & Moore 1995; Hirschberg & Nakatani 1996; Wiebe et al. 1997). However, in looking at agreement among judges, we are often not as concerned with describing how well two particular judges

This research was supported by the Office of Naval Research under grant number N00014-95-1-0776.

sense 1	"readiness to give attention"
sense 2	"quality of causing attention to be given"
sense 3	"activity, subject, etc., which one gives time and attention to"
sense 4	"advantage, advancement, or favor"
sense 5	"a share (in a company, business, etc.)"
sense 6	"money paid for the use of money"

Figure 1: Noun Senses of *Interest* in LDOCE

agree as in measuring how well any observer can distinguish the categories from one another. In other words, the issue is the precision of the *classification process*.

In this paper, we present a study of a classification process. The section *Agreement Among Judges* presents an analysis of the patterns of agreement among the judges. Agreement is a function of the differences among the judges (i.e., their biases) and the distinguishability of the categories themselves. We study bias using the models for symmetry, marginal homogeneity, and quasi-independence (in the subsection *Observer Differences*). We study category distinguishability using Darroch & McCloud's (1986) *degree of distinguishability*, δ_{ij} (in the subsection *Category Distinguishability*). Guided by these analyses, in the section *Modification of the Classification Process* we investigate modifications to the classification process that improve reliability. We analyze the effects both of removing judges and collapsing categories. A technique is presented for formulating a tag set which can be automatically derived from the original tag set. The technique is successful in the study presented here: the derived tag set yields improved reliability, as measured by Cohen's κ.

The Data

The classification process performed in this study involved five human judges independently assigning sense tags to 2369 instances of the noun *interest* taken from the Wall Street Journal Treebank Corpus (Marcus et al. 1993). The senses given to the taggers, shown in Figure 1, are from the Longman's Dictionary of Contemporary English (LDOCE).

The annotation instructions were minimal. They were asked to use their judgment in assigning to each usage of *interest* the single tag that best characterizes its meaning. It is likely that more explicit tagging instructions including examples and default rules would improve agreement among judges. Indeed, an analysis of the classification process such as performed here could be used to formulate and interactively revise a set of tagging instructions, but this application is not considered here.

Five human judges, referred to as A through E, participated in the study. Two of the judges (judges C and D) were involved in the project and had participated in previous sense tagging experiments. The remaining three judges (judges A, B and E) were not members of the project and had no previous background in NLP or linguistics.

Agreement Among Judges

All of the techniques that we present for the analysis of agreement are appropriate for category classifications assigned to multiple objects (in this case, words) by two judges.[1] We analyze the agreement among all five judges by evaluating the agreement between all pair-wise combinations of these judges. We exclusively use maximum likelihood estimates of model parameters.

The Basics

Tables 1-5 present half of the data, in contingency table format. Each table is for one pair-wise combination of the five judges. The rest of the data, for the other five combinations, is available on the World Wide Web at *http://crl.nmsu.edu/Research/Projects/graphling*. In each table, the rows correspond to the senses assigned by the first judge while the columns correspond to those assigned by the second judge. Let n_{ij} denote the number of words that judge one classifies as i and judge two classifies as sense j. If we let p_{ii} be the probability that the judges will agree that a randomly selected usage is sense i, then $\sum_i p_{ii}$ is the total probability of agreement across all senses. p_{ii} can be estimated as $\frac{n_{ii}}{n_{++}}$ (a maximum likelihood estimate), and the total probability of agreement can be estimated as $\sum_i \hat{p}_{ii} = \sum_i \frac{n_{ii}}{n_{++}}$, where $n_{++} = \sum_{ij} n_{ij} = 2369$.

[1] Several of these techniques are also applicable to the analysis of multiple judges.

The simplest measure of agreement is the estimated probability of agreement, i.e., $\sum_i \hat{p}_{ii}$, where the possible values are affected by the marginal totals (i.e., the row and column totals). Cohen's κ compares the total probability of agreement to that expected if the ratings were statistically independent (i.e., "chance agreement"). That value is then normalized by the maximum possible level of agreement given the marginal distributions. The marginal distributions can be estimated from the marginal counts as: $\hat{p}_{i+} = \frac{n_{i+}}{n_{++}}$ and $\hat{p}_{+i} = \frac{n_{+i}}{n_{++}}$. The complete formulation of κ is:

$$\kappa = \frac{\sum_i \hat{p}_{ii} - \sum_i \hat{p}_{i+}\hat{p}_{+i}}{1 - \sum_i \hat{p}_{i+}\hat{p}_{+i}} \quad (1)$$

κ is 0 when the agreement is that expected by chance, and is 1.0 when there is perfect agreement.

An extension of κ for the case of multiple judges (three or more) is presented in Davies and Fleiss (1982) and used in this study.

Analyzing Patterns of Agreement

In a classification experiment, the two judges are assumed to classify any given usage independently of each other, but it is clear in the formulation of κ that we expect the data to exhibit dependence, i.e., $\hat{p}_{ij} \neq \hat{p}_{i+} \times \hat{p}_{+j}$. Where does this dependence come from? It arises from three factors and their possible interactions: (1) the heterogeneity of the objects being classified (i.e., the usages of *interest*), (2) the heterogeneity of the judges, and (3) the distinctions made in the category definitions.

We focus on the latter two factors and their interaction. Rather than simply measuring agreement we measure the contributions to agreement made by these two factors and propose changes to the classification process based on the analysis. Just as overall agreement can be assessed as a function of the counts in the pair-wise confusion matrices, so can the measures of *observer difference* (bias) and *category distinguishability.*

Observer Differences (Bias) The hypothesis of no difference between two judges is the hypothesis of complete **symmetry** (*Sym* in Table 6), that is, $\hat{p}_{ij} = \hat{p}_{ji}$ or $\frac{\hat{p}_{ij}}{\hat{p}_{ji}} = 1$ for all i, j. If this ratio equals one for all i, j then it follows that the observers' interpretations are indistinguishable.

Complete symmetry implies marginal symmetry, that is, $\hat{p}_{i+} = \hat{p}_{+i}$. Bias of one judge relative to another is evidenced as a discrepancy between these marginal distributions. Bias decreases as the marginal distributions become more nearly equivalent. The measure of bias is the test for **marginal homogeneity** (*M.H.* in Table 6), $\hat{p}_{i+} = \hat{p}_{+i}$ for all i.

It is possible to access the similarity of two judges even when there is evidence of bias. The model for **quasi-independence** (*Q.I.* in Table 6) (Bishop et al. 1975) tests whether two judges' decisions are independent if we consider only the off-diagonal counts—the counts corresponding to disagreement (i.e., $\hat{p}_{ij} = \hat{p}_{i+} \times \hat{p}_{+j}$ for $i \neq j$). Quasi-independence holds when, given that the judges disagree, there is no pattern of association in the categories they assign.

In the tests for symmetry, marginal homogeneity, and quasi-independence, a model is formulated that enforces the hypothesized constraint, e.g., $p_{ij} = p_{ji}$ in the case of symmetry. The degree to which the data is approximated by a model is called the *fit* of the model. In this work, the fit of each model is reported in terms of the likelihood ratio statistic, G^2, and its significance. The higher the G^2 value, the poorer the fit. The fit of a model is considered acceptable if its reference significance level is greater than 0.001 (i.e., if there is greater than a 0.001 probability that the data sample was randomly selected from a population described by the model).

Category Distinguishability The ratio $\tau_{ij} = \frac{\hat{p}_{ij} \times \hat{p}_{ji}}{\hat{p}_{ii} \times \hat{p}_{jj}}$, referred to as the *diagonal cross-product-ratio*, represents the odds for disagreement over agreement on categories i, j. Darroch and McCloud (1986) define the *degree of distinguishability*, δ_{ij}, for categories i, j as:

$$\delta_{ij} = 1 - \tau_{ij} = 1 - \frac{\hat{p}_{ij} \times \hat{p}_{ji}}{\hat{p}_{ii} \times \hat{p}_{jj}} \quad (2)$$

If $\delta_{ij} = 1$, we say that the categories are completely distinguishable, and, if $\delta_{ij} = 0$, they are completely indistinguishable.

Majority Consensus When multiple judges are involved in a study, it is possible to formulate a

majority tag for each object, that is, the tag that the majority of the judges assign to each object. It represents majority opinion and is useful in identifying outlyers, as shown in the next section.

Results

Table 6 presents the results of the tests for observer differences and Table 7 presents the measures of category distinguishability. All evaluations are performed on each pair-wise confusion matrix. The columns labeled $M|A$ through $M|E$ refer to similar tables comparing the majority tag to the assignments made by each judge (e.g., judge A, in the case of $M|A$). These tables are not included in the paper.

While the κ values in Table 6 are reasonably high, the judges display bias and cannot be considered interchangeable. The only exception is the strong similarity between the majority tag and the assignments made by judge C (i.e., the column labeled $M|C$ in Table 6); these tags are symmetric and unbiased. Among the five judges, the most similar are judges C and D, the two experienced judges. While their scores for symmetry and marginal homogeneity are not significant, indicating a relative bias, their score for quasi-independence is significance (i.e., $0.004 > 0.001$, the cutoff we use to judge significance). This indicates that, although judges C and D are not indistinguishable, there is no **systematic** difference of opinion between them. Judge D also shows some similarity to the majority tag.

The judge that is least similar to the others is judge E; this is particularly evident when judge E is compared to the majority tag.

The distinguishability, δ_{ij}, of all pair-wise combinations of tags are evaluated in Table 7. All scores are at or near the maximum of 1.0, with the exception of those measuring the distinguishability of tags 1 and 2. It is particularly low in Table $A|B$ (i.e., Table 2).

Modification of the Classification Process

Based on the results presented above, we modified the classification process in two ways: (1) judge E is removed, and (2) sense tags 1 and 2 are conflated to form a single sense distinction. The poor marks for distinguishability between these senses seem to be reflected in a closeness in meaning (see in Figure 1), supporting the decision to conflate them.

Removing judge E from the study removes the tables with the lowest κ scores. As a result, the agreement among all judges increases from 0.874 to 0.898, as measured by Davies and Fleiss' extension of κ.

The process of conflating two tags is accomplished using the *latent class model* (Goodman 1974)[2]. This procedure has historically been used to identify a set of *latent* categories that explain the interdependencies among the observable categories. In this case, the observable categories are the sense tags assigned by the remaining four judges, while the latent categories correspond to the unobservable *true* meanings of the noun *interest*. Once the desired number of latent categories has been specified, these categories are assigned via the EM algorithm as described in Goodman (1974) and applied in Pedersen & Bruce (1997)[3].

Using the EM algorithm as described above, all usages of *interest* are assigned to one of five latent sense groupings. The mapping between the derived (i.e., latent) categories and the observed senses is established to maximize the correlation between latent categories and observed senses. This correlation for each judge, is estimated as part of the process of assigning latent categories. As an example, Table 10 presents the correlation for judge C. The values recorded in the table are the probabilities of judge C assigning sense tag i and the EM algorithm assigning latent tag j. As can be seen, correlation is maximized when the mapping of observed tags to latent tags is as follows: $1 \Rightarrow 1$, $2 \Rightarrow 1$, $3 \Rightarrow 2$, $4 \Rightarrow 3$, $5 \Rightarrow 4$, and $6 \Rightarrow 5$. This mapping conflates senses 1 and 2 while leaving all other senses intact. This corresponds to our expectations based on the study of agreement presented in the previous section. Using this mapping, the observer difference measures among the

[2] Also referred to as the *Naive Bayes model* (Langley et al. 1992).

[3] This is a well known unsupervised learning alobserved tagsgorithm; other notable references to this procedure are Lazarfeld (1966), Pearl (1988), and AutoClass (Cheeseman 1990).

		Latent Tag				
		1	2	3	4	5
Judge C	1	0.142	0.010	0.001	0.001	0.002
	2	0.003	0.001	0.001	0.000	0.000
	3	0.000	0.024	0.005	0.000	0.000
	4	0.001	0.000	0.074	0.001	0.000
	5	0.001	0.003	0.000	0.206	0.000
	6	0.000	0.000	0.000	0.000	0.526

Table 10: Tag Correlation for Judge C

four judges for the latent tag set are presented in Table 8, and the distinguishability of latent tags is presented in Table 9. As compared to the original classification process, the agreement among all judges increases from 0.874 to 0.916 for the revised tag set with four judges.

Recent work has proposed various methods for pruning senses for word instances and tuning tag sets to a particular domain using corpus information and existing linguistic knowledge sources (e.g., Yarowsky 1992, Jing et al. 1997, Basili et al. 1997). We have presented an automatic method for refining a tag set using an important additional source of information: the manual annotations assigned by human judges.

Conclusion

There is increasing awareness of the need to manage the uncertainty inherent in many classification systems. We have presented procedures that can be used to analyze and refine any classification system that makes use of nominal categories. These techniques can be used to study and improve the reliability of human judges as well as refine categorizations that can be applied automatically and, in the process, establish an upper bound on the accuracy of automatic classification, i.e., the agreement among the human judges. In future work, we will apply these techniques to the analysis and evaluation of automated classification systems.

References

[1] Bishop, Y. M., Fienberg, S., & Holland, P. (1975). *Discrete Multivariate Analysis: Theory and Practice.* Cambridge: The MIT Press.

[2] Basili, R., Della Rocca, M., Pazienza, M. T. (1997). Toward a bootstrapping framework for corpus semantic tagging. In *Proc. SIGLEX Workshop on Tagging Text with Lexical Semantics*, pp. 58-65.

[3] Carletta, J. (1996). Assessing agreement on classification taks: the kappa statistic. *Computational Linguistics 22* (2).

[4] Cheeseman, P. & Stutz, J. (1996). Bayesian Classification (AutoClass): Theory and Results. In Fayyad, Piatetsky–Shapiro, Smyth, & Uthurusamy editors, *Advances in Knowledge Discovery and Data Mining*, AAAI Press/MIT Press.

[5] Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psych. Meas. 20*: 37–46.

[6] Davies, M. & Fleiss, J. (1982). Measuring Agreement for Multinomial Data *Biometrics*,38:1047–1051.

[7] Darroch & McCloud. (1986). Category Distinguishability and Observer Agreement. *Austral. Journal of Statistics*, 28(3):371–388.

[8] Goodman, L. (1974). Exploratory latent structure analysis using both identifiable and unidentifiable models. *Biometrika*, 61(2):215–231.

[9] Hirschberg, J. & Nakatani, C. (1996). A prosodic analysis of discourse segments in direction-giving monologues. In *Proc. ACL-96*, pp. 286-293.

[10] Jing, H., Hatzivassiloglou, V., Passonneau, R., and McKeown, Kathleen (1997). Investigating complementary methods for verb sense pruning. In *Proc. SIGLEX Workshop on Tagging Text with Lexical Semantics*, pp. 58-65.

[11] Langley, P., Iba, W. & Thompson, K. (1992). An analysis of bayesian classifiers. In *Proceedings of the 10th National Conference on Artificial Intelligence*, pp. 223–228.

[12] Lazarfeld, P. (1966). Latent structure analysis. In S. A. Stouffer, L. Guttman, E. Suchman, P.Lazarfeld, S. Star, and J. Claussen (Ed.), *Measurement and Prediction*, New York: Wiley.

[13] Litman, D. & Passonneau, R. (1995). Combining multiple knowledge sources for discourse segmentation. In *Proc. 33rd Annual Meeting of the Assoc. for Computational Linguistics*, MIT, pp. 130-143.

[14] Marcus, M., Santorini, B., & Marcinkiewicz, M. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics 19* (2): 313–330.

[15] Moser, M. & Moore, J. (1995). Investigating cue selection and placement in tutorial discourses. In *Proc. 33rd Annual Meeting of the Assoc. for Computational Linguistics*, MIT, pp. 130-143.

[16] Pearl, J. (1988). *Probabilistic Reasoning In Intelligent Systems: Networks of Plausible Inference*. San Mateo, Ca.: Morgan Kaufmann.

[17] Pedersen, T. & Bruce, R. (1997). Distinguishing Word Senses in Untagged Text. *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing (EMNLP-97)*, August 1997.

[18] Wiebe, J., O'Hara, T., McKeever, K., and Öhrström-Sandgren, T. (1997). An empirical approach to temporal reference resolution. *Proc. 2nd Conference on Empirical Methods in Natural Language Processing (EMNLP-97)*, Association for Computational Linguistics, Brown University, August 1997, pp. 174–186.

[19] Yarowsky, D. (1992). Word-sense disambiguation using statistical models of Roget's categories trained on large corpora. In *Proc. COLING-92.*

		Judge 2 = E						
		sense1	*sense2*	*sense3*	*sense4*	*sense5*	*sense6*	
Judge 1 = A	*sense1*	$n_{11} = 174$	$n_{12} = 115$	$n_{13} = 11$	$n_{14} = 8$	$n_{15} = 6$	$n_{16} = 2$	$n_{1+} = 316$
	sense2	$n_{21} = 7$	$n_{22} = 8$	$n_{23} = 1$	$n_{24} = 2$	$n_{25} = 1$	$n_{26} = 0$	$n_{2+} = 19$
	sense3	$n_{31} = 25$	$n_{32} = 24$	$n_{33} = 40$	$n_{34} = 12$	$n_{35} = 4$	$n_{36} = 3$	$n_{3+} = 108$
	sense4	$n_{41} = 3$	$n_{42} = 1$	$n_{43} = 3$	$n_{44} = 156$	$n_{45} = 8$	$n_{46} = 1$	$n_{4+} = 172$
	sense5	$n_{51} = 1$	$n_{52} = 1$	$n_{53} = 6$	$n_{54} = 12$	$n_{55} = 474$	$n_{56} = 6$	$n_{5+} = 500$
	sense6	$n_{61} = 0$	$n_{62} = 0$	$n_{63} = 1$	$n_{64} = 2$	$n_{65} = 6$	$n_{66} = 1245$	$n_{6+} = 1254$
		$n_{+1} = 210$	$n_{+2} = 149$	$n_{+3} = 62$	$n_{+4} = 192$	$n_{+5} = 499$	$n_{+6} = 1257$	$n_{++} = 2369$

Table 1: Confusion Matrix for Judges A and E

		Judge 2 = B						
		1	2	3	4	5	6	
Judge 1 = A	1	242	37	21	7	8	1	316
	2	13	2	1	1	1	1	19
	3	32	5	53	15	1	2	108
	4	2	0	1	161	6	2	172
	5	3	0	20	16	458	3	500
	6	0	0	1	1	6	1246	1254
		292	44	97	201	480	1255	2369

Table 2: Confusion Matrix for Judges A and B

		Judge 2 = C						
		1	2	3	4	5	6	
Judge 1 = A	1	303	2	0	6	3	2	316
	2	10	6	1	1	1	0	19
	3	42	3	56	5	1	1	108
	4	4	0	8	154	6	0	172
	5	4	0	1	13	480	2	500
	6	5	1	1	1	5	1241	1254
		368	12	67	180	496	1246	2369

Table 3: Confusion Matrix for Judges A and C

		Judge 2 = D						
		1	2	3	4	5	6	
Judge 1 = C	1	342	1	2	2	12	9	368
	2	1	10	0	0	0	1	12
	3	2	1	48	12	3	1	67
	4	8	1	3	160	7	1	180
	5	4	0	0	0	489	3	496
	6	1	0	0	0	0	1245	1246
		358	13	53	174	511	1260	2369

Table 4: Confusion Matrix for Judges C and D

		Judge 2 = E						
		1	2	3	4	5	6	
Judge 1 = C	1	206	131	11	6	7	7	368
	2	0	11	0	0	0	1	12
	3	1	6	42	13	2	3	67
	4	1	1	5	164	8	1	180
	5	1	0	4	7	481	3	496
	6	1	0	0	2	1	1242	1246
		210	149	62	192	499	1257	2369

Table 5: Confusion Matrix for Judges C and E

Test	*A|E*	*A|B*	*A|C*	*A|D*	*B|C*	*B|D*	*B|E*	*C|D*	*C|E*	*D|E*	*M|A*	*M|B*	*M|C*	*M|D*	*M|E*
Sym.:															
G^2	165	70	77	75	105	101	109	46	226	214	81	84	22	39	212
Sig.	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.102	0.001	0.000
M. H.:															
G^2	150	30	47	58	69	79	90	37	213	210	64	42	15	39	206
Sig.	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.010	0.000	0.000
Q. I.:															
G^2	154	143	79	61	94	81	186	42	135	120	67	82	34	25	120
Sig.	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.004	0.000	0.000	0.000	0.000	0.016	0.051	0.000
Kappa	0.825	0.866	0.916	0.903	0.882	0.873	0.821	0.951	0.856	0.849	0.929	0.901	0.977	0.964	0.874

Table 6: Tests of Observer Differences (Bias) for Five Judges and Six Senses

Senses	*A|E*	*A|B*	*A|C*	*A|D*	*B|C*	*B|D*	*B|E*	*C|D*	*C|E*	*D|E*	*M|A*	*M|B*	*M|C*	*M|D*	*M|E*
1 – 2	0.422	0.006	0.989	0.986	0.765	0.662	0.183	1.000	1.000	1.000	0.990	0.675	1.000	1.000	1.000
1 – 3	0.960	0.948	1.000	0.997	0.959	0.964	0.950	1.000	0.999	0.997	1.000	0.968	1.000	1.000	0.999
1 – 4	0.999	0.999	1.000	0.999	0.999	0.999	0.999	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
1 – 5	1.000	1.000	1.000	1.000	1.000	0.999	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
1 – 6	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2 – 3	0.925	0.953	0.991	0.978	0.964	0.979	1.000	1.000	1.000	1.000	0.988	0.966	1.000	1.000	1.000
2 – 4	0.998	1.000	1.000	1.000	1.000	1.000	0.999	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2 – 5	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2 – 6	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
3 – 4	0.994	0.998	0.995	0.994	0.997	0.994	0.986	0.995	0.991	0.993	0.999	0.998	0.999	0.999	0.996
3 – 5	0.999	0.999	1.000	1.000	1.000	1.000	0.994	1.000	1.000	0.999	1.000	1.000	1.000	1.000	1.000
3 – 6	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
4 – 5	0.999	0.999	0.999	0.999	1.000	1.000	0.999	1.000	0.999	1.000	1.000	1.000	1.000	1.000	1.000
4 – 6	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
5 – 6	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

Table 7: Measure of Category Distinguishability for Five Judges and Six Senses

Test	*A|B*	*A|C*	*A|D*	*B|C*	*B|D*	*C|D*	*M|A*	*M|B*	*M|C*	*M|D*
Sym.:										
G^2	56	63	63	72	70	44	72	72	17	36
Sig.	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.068	0.000
M. H.:										
G^2	19	39	52	38	53	37	57	43	7	29
Sig.	0.001	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.136	0.000
Q. I.:										
G^2	72	68	50	46	23	37	60	37	26	19
Sig.	0.000	0.000	0.000	0.000	0.016	0.000	0.000	0.000	0.006	0.017
Kappa	0.898	0.924	0.910	0.909	0.902	0.952	0.943	0.926	0.978	0.964

Table 8: Tests of Observer Differences (Bias) for Four Judges and Five Senses

Senses	*A|B*	*A|C*	*A|D*	*B|C*	*B|D*	*C|D*	*M|A*	*M|B*	*M|C*	*M|D*
1 – 2	0.948	0.997	0.994	0.957	0.964	1.000	1.000	0.968	1.000	1.000
1 – 3	1.000	0.999	0.999	0.999	0.999	1.000	1.000	1.000	1.000	1.000
1 – 4	1.000	1.000	1.000	1.000	0.999	1.000	1.000	1.000	1.000	1.000
1 – 5	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2 – 3	0.998	0.995	0.993	0.997	0.994	0.995	0.999	0.998	1.000	0.997
2 – 4	0.999	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2 – 5	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
3 – 4	0.999	0.999	0.998	0.999	1.000	1.000	1.000	1.000	1.000	1.000
3 – 5	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
4 – 5	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

Table 9: Measure of Category Distinguishability for Four Judges and Five Senses

Category Levels in Hierarchical Text Categorization

Stephen D'Alessio Keitha Murray
Robert Schiaffino
Department of Computer and Information Science
Iona College
New Rochelle, N.Y. 10801, USA
sdalessio@iona.edu, kmurray@iona.edu, rschiaffino@iona.edu

Aaron Kershenbaum
Department of Computer Science
Polytechnic University
Hawthorne, N.Y. 10532, USA
akershen@duke.poly.edu

Abstract

We consider the problem of assigning level numbers (weights) to hierarchically organized categories during text categorization. These levels control the ability of the categories to attract documents during the categorization process. The levels are adjusted to obtain a balance between recall and precision for each category. If a category's recall exceeds its precision, the category is too strong and its level is reduced. Conversely, a category's level is increased to strengthen it if its precision exceeds its recall.

The categorization algorithm used is a supervised learning procedure that uses a linear classifier based on the category levels. We are given a set of categories, organized hierarchically. We are also given a training corpus of documents already placed in one or more categories. From these, we extract vocabulary, words that appear with high frequency within a given category, characterizing each subject area. Each node's vocabulary is filtered and its words assigned weights with respect to the specific category. Then, test documents are scanned and categories ranked based on the presence of vocabulary terms. Documents are assigned to categories based on these rankings. We demonstrate that precision and recall can be significantly improved by solving the categorization problem taking hierarchy into account. Specifically, we show that by adjusting the category levels in a principled way, that precision can be significantly improved, from 84% to 91%, on the much-studied Reuters-21578 corpus organized in a three-level hierarchy of categories.

1 Introduction and Background

The volume of online information has drastically increased with the explosive use of the Internet and online databases. Text retrieval systems employed by search engines for accessing this information have difficulty keeping pace with the growth in the amount of data that needs indexing and searching. Categorization of the original text is a method of organizing and making more efficient the retrieval task by sorting information into pre-specified "category bins" that can then be queried against using natural language processing systems.

The document categorization problem is one of assigning newly arriving documents to categories within a given hierarchy of categories. In general, lower level categories may be part of more than one higher level category. Moreover, a document may belong to more than one low-level category. While the techniques described here can be applied to this more general problem, the experiments we have conducted, to date, have been carried out on a corpus where each document is a member of a single category and the categories form a tree rather than a more general directed acyclic graph. We limited the investigation to this more specific problem in order to focus the investigation on the effect of adjusting the category level numbers.

Most computational experience discussed in the literature deals with hierarchies that are trees. Indeed, until recently, most problems discussed dealt with categorization within a simple (non-hierarchical) set of categories (Frakes and Baeza-Yates, 1992). The Reuters-21578 corpus (available at David Lewis's home page:

http://www.research.att.com/ lewis) has been studied extensively. Yang (Yang, 1997) compares 14 categorization algorithms applied to this Reuters corpus as a flat categorization problem on 135 categories. This same corpus has been more recently studied by others treating the categories as a hierarchy (Chakrabarti et al., 1997)(Koller and Sahami, 1997)(Ng et al., 1997)(Yang, 1996). Yang examines a portion of the OHSUMED (Hersh et al., 1994) corpus of medical abstracts, a part of the National Library of Medicine corpus that has over 9 million abstracts organized into over 10,000 categories in a taxonomy (called MeSH) which is seven levels deep in some places.

We describe an algorithm for hierarchical document categorization where the vocabulary and term weights are associated with categories at each level in the taxonomy and where the categorization process itself is iterated over levels in the hierarchy. Thus a given term may be a discriminator at one level in the taxonomy receiving a large weight and then become a stopword at another level in the hierarchy.

There are two strong motivations for taking the hierarchy into account. First, experience to date has demonstrated that both precision and recall decrease as the number of categories increases (Apte et al., 1994) (Yang, 1996). One of the reasons for this is that as the scope of the corpus increases, terms become increasingly polysemous. This is particularly evident for acronyms, which are limited by the number of 3- and 4-letter combinations, and which are reused from one domain to another.

The second motivation for doing categorization within a hierarchical setting is it affords the ability to deal with very large problems. As the number of categories grows, the need for domain-specific vocabulary grows as well. Thus, we quickly reach the point where the index no longer fits in memory and we are trading accuracy against speed and software complexity. On the other hand, by treating the problem hierarchically, we can decompose it into several problems each involving a smaller number of categories and smaller domain-specific vocabularies and perhaps yield savings of several orders of magnitude.

Feature selection, deciding which terms to actually include in the indexing and categorization process, is another aspect affected by size of the corpus. Some methods remove words with low frequencies both in order to reduce the number of features and because such words are often unreliable. Depending on the size of the corpus, this may still leave over 10,000 features, which renders even the simplest categorization methods too slow to be of use on very large corpora and renders the more complex ones entirely infeasible.

Methods that incorporate additional feature selection have been studied (Apte et al., 1994) (Chakrabarti et al., 1997) (Deerwester et al. 1990) (Koller and Sahami, 1996) (Lewis, 1992) (Ng et al., 1997) (Yang and Pederson 1997). The effectiveness of these feature selection methods varies. Most reduce the size of the feature set by one to two orders of magnitude without significantly reducing precision and recall from what is obtained with larger feature sets. Some approaches assign weights to the features and then assign category ranks based on a sum of the weights of features present. Some weight the features further by their frequency in the test documents. These methods are all known as linear classifiers and are computationally simplest and most efficient, but they sometimes lose accuracy because of the assumption they make that the features appear independently in documents. More sophisticated categorization methods base the category ranks on groups of terms (Chakrabarti et al., 1997) (Heckerman, 1996) (Koller and Sahami, 1997) (Sahami, 1996) (Yang, 1997). The methods that approach the problem hierarchically compute probabilities and make the categorization decision one level in the taxonomy at a time.

Precision and recall are used by most authors as a measure of the effectiveness of the algorithms. Most of the simpler methods achieved values for these near 80% for the Reuters corpus (Apte et al., 1994) (Cohen and Singer, 1996). More computationally expensive methods using the same corpus, achieved results near 90% (Koller and Sahami, 1997) while methods that used hierarchy obtained small increases in precision and large increases in speed (Ng et al., 1997). As the number of categories increased in a corpus (OHSUMED), precision and recall decline to 60% (Yang 1996).

In a previous paper (D'Alessio et al., 1998) we show that it is possible to obtain more significant improvements in precision and recall by making use of the hierarchy. We describe an earlier version of the algorithm discussed here and show that treating the categorization problem within the context of a hierarchy is effective in realizing these improvements. The principal focus there was on the effect of the hierarchy itself and in refining the hierarchy. In some cases, moving categories from one place within the hierarchy to another within it can further improve the accuracy of the categorization. Here we extend that investigation and focus on the effect of adjusting the category levels to further improve accuracy. We are particularly interested in exploring the situations where one approach (hierarchy modification or level modification) works best.

2 Problem Definition

2.1 General Definition of Categories

We are given a set of categories where sets of categories may be further organized into supercategories. We are given a training corpus and, for each document, the category to which it belongs. Documents can, in general, be members of more than one category. In that case, it is possible to consider a binary categorization problem where a decision is made whether each document is or is not in each category. Here, we examine the M-ary categorization problem where we choose a single category for each document.

gories organized as a flat taxonomy. Although the collection does not have a pre-defined hierarchical classification structure, additional information on the category sets available at Lewis's site describes an organization that has 5 additional categories that become supercategories of all but 3 of the original topics categories. Adding a root forms a 3-level hierarchy (see Figure 1). The number of categories per supercategory varies widely from a minimum of 2 to a maximum of 78. All of the documents in the Reuters collection are assigned to 0 or more of the original 135 topics categories. In this case, documents are assigned only to leaf categories of the hierarchy while, in general, this is not necessarily the case.

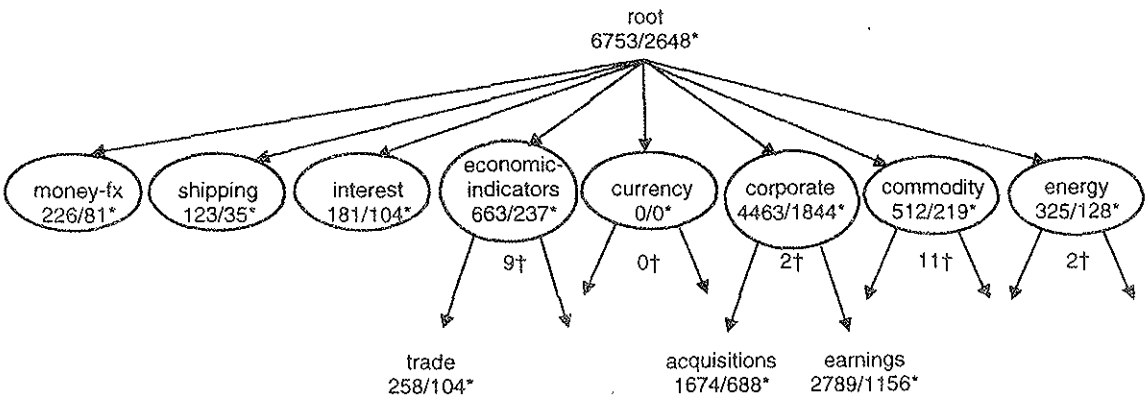


Figure 1 Reuters basic hierarchy

2.2 Document Corpus and Taxonomy

We use the Reuters-21578 corpus, Distribution 1.0, which is comprised of 21578 documents, representing what remains of the original Reuters-22173 corpus after the elimination of 595 duplicates by Steve Lynch and David Lewis in 1996. The size of the corpus is 28,329,337 bytes, yielding an average document size of 1,313 bytes per document. The documents are "categorized" along five axes - topics, people, places, organizations, and exchanges. We consider only the categorization along the topics axis. Close to half of the documents (10,211) have no topic and as Yang (Yang, 1996) and others suggest, we do not include these documents in either our training or test sets. Note, that unlike Lewis (acting for consistency with earlier studies), the documents that we consider no-category are those that have no categories listed between the topic tags in the Reuters-21578 corpus' documents. This leaves 11,367 documents with one or more topics. Most of these documents (9,495) have only a single topic. The average number of topics per document is 1.26.

The Reuters-21578 collection uses 135 topics cate-

The number of training documents per category also varies widely, from a minimum of 0 (for 71 such categories) to a maximum of 2,789 (earnings). On the other hand, document size does not vary greatly across categories. In the experiments described in this paper, we only considered categorizing test documents into categories having 20 or more training documents. This was done in order to focus on a problem where there was enough statistical significance in the features we extracted to make comparisons among different category levels meaningful. This limited the investigation to 27 categories and actually removed only 94 documents (less than 3.5%) from the test corpus. This increased the overall precision and recall by about 1.5%. However, since we are principally interested here in studying the effect of varying the category level numbers, this is not a problem as all the experiments described were carried out on the same corpus.

2.3 Performance Metrics

We measure the effectiveness of our algorithm by using the standard measures of microaveraged precision

and recall; i.e., the ratio of correct decisions to the total number of decisions and the ratio of correct decisions to the total number of documents, respectively. We do, however, sometimes leave documents in non-leaf categories and then, in measuring precision and recall, count these as "no-category", reducing recall but not precision.

3 Algorithm Description

3.1 Overview

We begin by creating training and test files using the 9,495 single-category documents from the Reuters-21578 corpus. While this led to somewhat higher precision and recall than would have been obtained by including multicategory documents, our 91% precision and 90% recall is also higher than the roughly 80% typically reported for categorization methods of comparable speed and complexity. Thus, our approach is comparable to those methods and serves as a reasonable baseline against which to study the effects of the hierarchy.

The corpus is divided randomly, using a 70%/30% split, into a training corpus of 6,753 training documents and 2,742 test documents. Documents in both the training and test corpora are then divided into words using the same procedure. Non-alphabetic characters (with the exception of "-") are removed and all characters are lowercased. Stopwords are removed. The document is then parsed into "words"; i.e., character strings delimited by whitespace, and these words are then used as features.

Next, we count the number of times each feature appears in each document and, from that, we compute the total number of times each feature appears in training documents in each category. We retain only features appearing 2 or more times in a single training document or 10 or more times across the training corpus. All other features are discarded as being insufficiently reliable.

Next we use a variant of the ACTION Algorithm (Wong et al. 1996), described in detail in Section 3.2 below, to associate features with nodes in the taxonomy. This is one of the two aspects that make our approach novel. By eliminating most features from most categories, we gain several advantages. First, by limiting the appearance of a feature to a small number of categories (usually, just one) where it is an unambiguous discriminator, we improve the precision of the categorization process. Second, by working with a small number of features, we avoid optimization over a large number of features, and have a procedure with low computational complexity that can be applied to large problems with many categories. (Currently the number of features is set to 50). Our feature selection procedure most closely resembles rule induction (Apte et al., 1994) but it differs from that approach in that it considers the interactions among a larger number of features for a given amount of computational effort.

Weights are now assigned to the surviving features in each category. We associate a weight, W_{fc} , with each surviving feature, f , in category c. We define W_{fc} by:

$$W_{fc} = (\lambda + (1 - \lambda)\frac{N_{fc}}{M_c}) \quad (1)$$

where N_{fc} is the number of times f appears in c, M_c is the maximum frequency of any feature in c, and is a parameter (currently set to 0.4).

We also assign a negative weight to features associated with siblings (successors of the same parent node) of each category. A feature appearing in one or more siblings of c but not in c itself, is assigned a negative weight

$$W_{fc} = -(\lambda + (1 - \lambda)\frac{N_{fp}}{M_p}) \quad (2)$$

where p is the parent of c in the hierarchy. Thus N_{fp} is the number of times f appears in the parent of c, which is in turn the number of times f appears in all siblings of c since it does not appear in c itself at all. M_p is the maximum frequency of any feature in c's parent.

Finally, we filter the set of positive and negative words associated with each category, both leaf and interior, retaining the most significant words. This process is described in the next section.

We now have an index suitable for use in the category ranking process. The index contains features and a weight, W_{fc} , associated with each feature in each category. Note that W_{fc} is implicitly 0 for any feature not associated with a particular category.

Given a document, d, a rank can now be associated with each category with respect to d. Let F be the set of features, f, in D. The ranking of category c with respect to document d, R_{cd}, is then defined to be:

$$R_{cd} = \sum_f N_{fd} W_{fc} \quad (3)$$

where the sum is over all positive and negative features associated with c and N_{fd} is the number of times f appears in d. Note that, in practice, the sum is taken only over features that are in the intersection of the sets of features actually appearing in d and actually associated with c. Note that R_{cd} may be positive, negative or zero.

Test document d is now placed in a category. Starting at r, the root of the hierarchy, we compute R_{cd} for all c which are successors of r. If all R_{cd} are zero or negative, d is left at r. If any R_{cd} is positive, let c' be the category with the highest rank. If c' is a leaf node, d is placed in c'. If c' is an interior node, the contest is repeated

at node c'. Thus, d is eventually placed either in a leaf category which wins a contest among its siblings or in an interior node none of whose children have a positive rank with respect to d. In this latter case, we may say that d is actually placed in the interior category, partially categorized or not categorized at all. Which of these we choose is dependent upon the application and on how much we value precision versus recall.

3.2 The ACTION Algorithm

The ACTION Algorithm was first described in (Wong et al., 1996) as a method of associating documents with categories within a hierarchy. Here, we use it to associate vocabulary with nodes in a hierarchy and associate documents with the nodes using the procedure described in Section 3.1 above. The original algorithm applied to problems with documents at interior and leaf nodes. Although our adaptations apply to the more general case also, we describe the algorithm with respect to that simpler case since the corpus we are using has documents only at leaf nodes.

The algorithm begins by counting N_{fc}, the number of times feature f appears in documents associated with category c in the training set, for all f and c. There is a level, , associated with each category, c, in the hierarchy. By convention, the root is at level 1; its immediate successors are at level 2, etc.

We then define EF_{fc}, the effective frequency of a subtree rooted at node c with respect to feature f as

$$EF_{fc} = \sum_{j \in S_C} N_{fj} \quad (4)$$

Thus, EF_{fc} is the total number of occurrences of f in c and all subcategories, S_c of node c.

Finally, we define V_{fc}, the significance value of c with respect to f, as

$$V_{fc} = L_c \times EF_{fc} \quad (5)$$

Thus, a node gets credit, in proportion to its level, for occurrences of f in itself and in its successors. The farther down the tree a node is, the more credit it is given for its level, but the higher up the tree a node is, the larger the subtree rooted at c and the larger the credit it gets for effective frequency. A competition thus takes place between each node and its parent (immediate predecessor). For each feature, f, EF_{fc} is compared with, EF_{fp} , where p is the parent of c and if EF_{fc} is smaller then f is removed from node c. Thus a parent can remove a feature from a child but not vice versa. In the case of a tie, the child loses the feature. All this competition proceeds from the leaves upward towards the root.

The net effect of this is that if a feature occurs in only a single child of a given parent, then the child retains the feature (as does the parent), but if the feature occurs significantly in more than one child of the same parent, then only the parent retains the feature.

Several advantages accrue from all this. First, common features, including stopwords, will naturally rise to the root, where they will not participate in any rankings. Thus, this algorithm is a generalized version of removing stopwords. If a feature is prominent in several children of the same node, the parent will remove it from all of them. Ideally, words that are important for making fine distinctions among categories farther down in the category hierarchy, but are ambiguous at higher levels, will participate only in places where they can help.

Note that we never directly remove a feature from the parent even when the child retains it. The reason for this is that we may need the feature to get the document to the parent; if it doesn't reach the parent it can never reach the child. In the case where a feature strongly represents only one category, there is no harm in the parent retaining it. In the cases where it is ambiguous at the level of the parent, the grandparent removes it from the parent (its child).

Thus, at the end of the algorithm when we filter the feature set for each category (leaf and non-leaf) retaining only the 50 most highly ranked positive and negative words, at non-leaf categories we also retain any words retained by their children.

3.3 Assignment of Category Level Values

The focus of the experiments described in Section 4 is to investigate the effect of modifying the category levels in the ACTION Algorithm and in the ranking process which actually selects document categories. We begin with the root at level 1 and with all other categories at a level one higher than that of their parents. We run a categorization and measure the resultant precision and recall for each category and for the corpus as a whole.

Next, we consider the effect of varying the level of the root, observing the effect on accuracy, and setting the level of the root (and all other categories, since their levels are set relative to that of the root) to the best value found. A simple, linear search is carried out at a fairly coarse scale (increments of .25). Experiments we carried out using a finer scale did not yield significantly better results and we thus limited all the experiments here to this stepsize of .25. Even with such a simple search, we obtained significant improvements in accuracy, over 7% overall. It is our intention in the future, after examining the effects of the interaction between hierarchy modifications and level modifications in more

detail, to return to the issue of searching over a narrower grid. At this stage in the investigation, however, we felt that doing so would only obscure the main results.

Actually, the category level numbers serve two purposes: word selection and document ranking. First, during the ACTION Algorithm (see Equation 5), they affect the competition (between parents and children) for words. A parent at level L will compete successfully with a child at level L+D, removing a word from the child's wordlist, if the F_p/F_c, the frequencies of the word in the subtrees rooted at the parent and child, respectively, exceeds (L+D)/L. Thus, the difference in the level number of the parent and child directly affects how high the relative frequency of the word must be, in the child relative to the parent, in order for the child to retain the word. Making D smaller strengthens the parent with respect to the child. Similarly, making L smaller while leaving D the same, weakens the parent with respect to the child. But altering L is fundamentally different from altering D as altering L also affects the parent's strength with respect to its own parent.

Thus, in modifying level numbers we must consider this interaction. We do so simply by looking for categories where the precision and recall are very different and where the interaction with other categories is marked. At each step, we consider the performance of a node relative to its parent, strengthening or weakening it as appropriate to balancing the node's precision and recall, specifically, its ability to attract the correct documents to its subtree.

Changing a node's level number also affects the ranking process. Again, the higher the level number, the stronger the node. Now, however, the change in level number also affects a node's strength with respect to its siblings as siblings compete directly for documents reaching their parent. We deal simply with this problem too. By examining the dispersion matrix, we observe which categories in the group under a common parent are too strong, aggressively stealing documents from their siblings, and which are victims. We begin by adjusting the node most out of kilter, or several nodes that are all out of kilter in the same direction and are not directly competing with one another. In practice this was found to be effective; experiments with more complex modification procedures did not produce significantly better results.

Actually, it is possible to consider two different level numbers, one for word selection and another for document ranking. In fact, the motivations for modifying a node's level number for word selection and for document ranking coincide thus making it reasonable to consider making similar adjustments. We plan to return to this issue as part of a broader investigation of refinements to the overall algorithm, preferring to concentrate here on the simpler case. Even using this simple approach, however, we obtained significant gains.

4 Computational Experience

There are a number of ways that the performance of a hierarchical categorization system can be tuned. Here we describe experiments performed in order to understand the effects of adjusting the level numbers (weights) of the categories within the hierarchy.

The purpose of this research is to investigate the role of a hierarchical organization of categories on the text categorization task. In particular we are considering a tree of categories with each node in the tree assigned a level number. As described above, this level number is used in evaluating the significance of features during the feature selection process, and in weighting of document features during the categorization process. The experiments reported here were conducted to determine the impact of this level number on feature selection and categorization of documents.

We begin with a base line case. We use the topics hierarchy supplied with the Reuters-21578 corpus, and consider only the leaf categories. We add a root category to make a simple tree structure. We assign the root a level number of 0 and the leaves a level of 1. With this organization of categories and level numbers the root is unable to remove any features from a node during the feature selection process. Therefore, it effectively becomes a set of nodes rather than a tree. When we apply our categorization algorithm to the test documents we achieve a precision of 83.6% and a recall of 83.5%. We refer to this case as Flat-0. Note that if no category gives a document a ranking above our threshold, currently set to zero, then the document remains unclassified. In the Flat-0 case there are 2 unclassified documents.

We modified the base case by giving the root a level of 1, and all leaves a level of 2. The root is now capable of extracting features from the leaves during the feature selection process. When we apply our categorization procedure to the same test data as above we achieve a precision of 90.6% and a recall of 87.2%. We refer to this case as Flat-1. In Flat-1 there are 99 unclassified documents, but the precision and recall are significantly improved.

With a level number of 1, the root aggressively removes features from the leaves. The result is that 97 more documents receive rankings below the threshold and remain unclassified in Flat-1 than in Flat-0. We hypothesize that if the root were less aggressive in re-

moving features from the leaves, the leaves would retain better features, resulting in better recall and precision. On the other hand, if the root has too low a level number the root does not remove any features from the leaves, and as a result the leaves retain features that are noisy. We tested this hypothesis by assigning the root a level of .75 and the leaves levels of 1.75. We refer to this case as Flat-75. Applying our feature selection and categorization algorithms as above resulted in a precision of 91.2% and a recall of 89.2%. In this case 60 documents were unclassified but both precision and recall were improved when compared to Flat-1. The results of the experiments on the test data for the three Flat hierarchy cases are in the summary Table 3.

These results support our hypothesis that the value of the level numbers affects the ability of the root to remove features. We conducted a further experiment to confirm this conclusion. Normally our program removes stopwords from the training and testing documents. Since we restrict the number of features at each node to 50, this insures that the retained features are useful. We modified our programs so that stopwords were not removed, then ran the feature selection and categorization processes. If our conclusions regarding the level numbers were correct, then using a level number of .75 should result in precision and recall approximately equal to the results described above for Flat-75. However if we run the program with a root level of 0 the precision and recall should deteriorate since the stopwords will impede performance. When we performed these experiments we achieved a precision of 90.7% and a recall of 88.9% with a root level of .75 and a precision and recall of 78.3% with a root level of 0. These results confirm that our feature selection algorithm together with appropriate level values significantly reduces noise and improves performance.

Our next objective was to determine if the level numbers could be tuned to improve performance in the case of a more elaborate hierarchy. For this set of experiments we also used the topics hierarchy provided with the Reuters-21578 corpus (Figure 1). This time we included the intermediate categories, corporate, commodities, economic indicators, energy and currency. We first established a base line for performance by assigning the root a level of .75 and increasing the level numbers by 1 at each lower level of the tree. We refer to this organization as Base-Hier. We ran our feature selection program using the training data, and our categorization program using the same test data as above. The result was a precision of 87.1% and a recall of 85.2%. This result is reported in the summary Table 3. In order to tune the level numbers we repeated a process of first using the training data to select a set of features for each node, then categorizing the training data and using the results to select new level numbers, then repeating the feature selection/categorization process on the training data until we arrived at an appropriate set of level numbers. At that point we could judge the effectiveness of the training by comparing our results against the base line case.

We began the process by using the training data for feature selection and categorization with Base-Hier. Based on our analysis of the results from the previous experiments we hypothesize that we could improve the categorization performance in two ways. First, if a category is achieving high precision and low recall, we could raise its level number; and second, if a category is achieving high recall and low precision we could lower its level number. For our first experiments we selected simple cases of nodes that were experiencing poor performance, and as we learned more about the tuning process moved onto more involved cases.

When we apply our feature selection and categorization programs to our training data using Base-Hier we get a precision of 89.2% and a recall of 87.5%. When we examine the results more closely we see that the categories of interest and money-fx are candidates for tuning. Interest has a precision of 95% but a recall of only 23% while money-fx has a precision of 89% and a recall of 60%. Both of these categories are direct descendents of the root and have no descendents. In both cases raising the level numbers should allow us to improve recall. We changed both level numbers from 1.75 to 2.75 and ran our feature selection and categorization procedures with the new hierarchy. Overall the precision and recall improved to 90.8% and 89.3% respectively. Interest has a precision of 98% and recall of 62% and money-fx has a precision of 85% and recall of 84%. Of course these results are from categorizing the training data, however they do indicate significant improvement.

If we look at the results of the previous experiment we see that with a precision of approximately 90% and a training set of 6493 documents, we are making approximately 650 errors. The largest single source of these errors occurs in the corporate subtree. Corporate has two subcategories, earnings and acquisitions. Earnings has a precision of 91% and a recall of 99% while acquisitions has a precision of 94% and a recall of 84%. The corporate category has a precision of 97% and a recall of 98%. These categories account for approximately 2/3 of the training data. From these results we can see that almost all of the earnings and acquisitions documents are correctly placed in the corporate category. Our categorizer must then decide if the documents are earnings or acquisitions documents. Our program is placing 22 earnings documents in the acqui-

sitions category and 211 acquisitions documents in the earnings category. In addition, 42 earnings and acquisitions documents are left in the corporate category since there was no positive rank. In all, this is a total of 275 mistakes, which accounts for a substantial portion of the total 650 mistakes. Clearly this set of categories is a good candidate for tuning. This case is more complex than the interest/money-fx case since earnings and acquisitions are not descendents of the root. As we tune their level values we want to improve the performance of earnings and acquisitions without having a negative impact on the performance of the corporate category.

Since many more acquisitions documents are being classified as earnings documents than the reverse and acquisitions' recall is significantly lower that its precision (see Table 1), we should lower the level number of earnings relative to acquisitions in order to make acquisitions stronger. At this point corporate is at level 1.75, and both earnings and acquisitions are at level 2.75. There are a number of ways that we might tune these levels, we explored three possibilities. The first alternative leaves corporate at 1.75 and acquisitions at level 2.75 but lowers earnings to 2.5. Call this A1. The second alternative leaves corporate at 1.75 and lowers both earnings and acquisitions, earnings to 2.25 and acquisitions to 2.5. Call this A2. The final possibility lowers corporate to 1.5 and earnings to 2.5 and leaves acquisitions at its 2.75 level. Call this A3.

We would expect that using A1 would result in acquisitions getting better features and consequently also getting more of its own documents, with the possible side effect of having more earnings documents classified as acquisitions. A2 makes corporate stronger relative to both earnings and acquisitions. As we saw in the flat cases this would mean that corporate would remove features more aggressively. We would expect therefore that acquisitions would have fewer of its documents classified as earnings, but there is the possibility that many more documents from both earnings and acquisitions will be unclassified. A3 makes both earnings and acquisitions stronger relative to corporate and we would expect to have fewer unclassified documents. We would also expect that fewer acquisitions documents would be classified as earnings. Since both A1 and A2 leave corporate at level 1.75 we would also expect that corporate would continue to achieve both high recall and precision. In fact, other branches of the hierarchy should be unaffected by the changes inside the corporate tree (one of the strengths of a hierarchical approach). A3 changes the level of corporate so there is the possibility that the performance of corporate relative to the rest of the hierarchy will deteriorate in this case.

We ran our feature selection and classification procedure for all three cases. The results are shown in the Table 1 below.

As we can see A1 allows acquisitions to retain better features and its recall improves significantly. By making earnings weaker it classifies fewer acquisitions documents as earnings and earnings achieves a higher precision with a slight decrease in recall. The weaker value for earnings also results in more unclassified corporate documents. A2 produces similar improvements in the precision of earnings and the recall of acquisitions but results in many more unclassified corporate documents resulting in a slightly lower overall recall. A3 gives the fewest unclassified corporate documents. Since corporate has a lower level number in this case it does not remove features as aggressively as in the other two cases. One side effect however is that many more documents are incorrectly classified as acquisitions, and the overall performance deteriorates. Of course there are other adjustments that could be made, but our objective was not to find the optimum combination, but rather to understand the effects of changing the levels. We selected A1 as the best alternative.

The next case we consider is the economic indicators subtree. This is a more complex case than those described above. Nine of the categories in this subtree have more than twenty training documents and are used in these experiments. Together there are 663 training documents for the categories in the subtree. Using the A1 hierarchy above the subtree achieves a recall of 88% and a precision of 84% on the training data. Within the subtree the performance is quite varied. Five of the nine subcategories have a precision of over 90% while four of the categories have recall below 70%. In some cases the difference between precision and recall is very large. The category cpi for example has a precision of 100% but a recall of only 40%. Balance of payments has a recall of 88% and a recall of only 29%. On the other hand, trade has a precision of only 73% and a recall of 88%.

All the categories within economic indicators have level 2.75. We tested our hypotheses regarding the effects of level numbers by adjusting the levels within the subtree. We increased the level of the two nodes with very low recall and high precision from 2.75 to 4.75. We increased the level of one node to 3.75 and we decreased the level of trade from 2.75 to 2.5. Nodes with recall and precision approximately equal were left unchanged. With these adjustments, our overall performance on the training data was a 93.0% precision and a 91.7% recall. Using our guidelines we performed a final round of tuning throughout the hierarchy (called Final-Hier) using the training data with a precision of 93.2% and a recall of 92.0%. The results of these experiments on the

	Level Numbers								
	Corp	Acq	Earn	Overall Prec/Rec	Earn Prec/Rec	Acq Prec/Rec	Unclass Corp Docs	Earn as Acq	Acq as Earn
Before	1.75	2.75	2.75	91/89	91/99	94/84	42	22	211
A1	1.75	2.75	2.50	93/91	96/98	92/93	55	47	55
A2	1.75	2.50	2.25	92/90	97/97	91/91	96	65	39
A3	1.50	2.75	2.50	91/89	97/94	88/93	15	160	50

Table 1: Precision, Recall, Unclassified Corporate Documents, Earnings Documents Classified as Acquisitions and Acquisitions Documents Classified as Earnings for Different Levels Number for Corporate, Acquisitions and Earnings using Training Data.

	Prec(%)	Rec(%)
Base-Hier	89.2	87.5
Interest/Money-fx at 2.75	90.8	89.3
Earnings at 2.5 (A1)	92.7	91.0
Adjusting Econ Inds	93.0	91.7
Final-Hier	93.2	92.0

Table 2: Results Using Training Data

	Prec(%)	Rec(%)
Flat-0	83.6	83.5
Flat-75	91.2	89.2
Flat-1	90.6	87.2
Base-Hier	87.1	85.2
Final-Hier	91.5	89.9
Final-Hier (Alt split)	91.1	89.9

Table 3: Results Using Test Data

training data are reported in Table 2.

We then used the resulting hierarchy, Final-Hier, to categorize the test data. The result was an overall precision of 91.5% and a recall of 89.9%. This compares favorably with our results on the test data using Base-Hier where we achieved a precision of 87.1% and a recall of 85.2%. See Table 3 for a summary of selected corresponding results using the test data.

We performed additional experiments to test the robustness of our final hierarchy. In all of the experiments above we restricted ourselves to categories that had at least 20 training documents. In the first test of robustness we relaxed this condition and only required 10 training documents. When we applied our categorizer to the test data we achieved a precision of 91.0% and a recall of 89.4%. In our second test we relaxed the condition further and considered all the categories regardless of the number of training documents. When we applied the categorizer in this case we achieved a precision of 90.0% and a recall of 88.4%. In our next test we kept the level values of the categories the same but retrained the graph using only 30% of the data as training data. We then tested the categorizer on the remaining 70%. In this experiment we again required 20 training documents for a category. The result was a precision of 89.8% and a recall of 89.4%. Finally, we tested the categorizer on an alternate 70/30 random split of the corpus and obtained similar results. This final result is also reported in Table 3.

5 Summary and Conclusions

In this paper, we have explored the effect of modifying the category level numbers in an algorithm for hierarchical text categorization and have shown that it is possible to obtain substantial improvements in precision and recall by doing so. Specifically, we improved precision and recall from an 84% level to over a 91% level, by adjusting the category level numbers. The procedure we used was a simply, greedy search heuristic guided by the principle that categories whose precisions significantly exceeded their recall were too weak and those whose recall exceeded their precision were too strong.

In a previous paper (D'Alessio et al., 1998) we explored the effect of modifying the hierarchy itself, moving categories from one part of the hierarchy to another, in order to achieve similar objectives. We found that approach effective also and have now shed additional light on the role of hierarchy in the categorization process and in the interaction between hierarchy modification and level modification. Close examination of the dispersion matrix has been very useful in this regard. We found that level modification was most useful in cases where a category was generally too weak or generally too strong. The row or column in the dispersion matrix containing many off-diagonal elements characterized these cases. On the other hand, when the problem was a single large off-diagonal element, moving a category from one part of the hierarchy to another was more effective. In some cases, both approaches were effective.

We have seen examples of all these cases. We illustrated we could achieve improvements by modifying the level numbers for earnings and acquisitions or, alternatively (in our previous work (D'Alessio et al., 1998)) by altering the hierarchy by removing the intermediate corporate category. The former approach, however, worked somewhat better. We found that we could gain by altering the level numbers of interest and money-fx or, alternatively making them children of economic indicators. Both approaches worked, but in this case, the latter worked better.

Based on our computational experience to date, our conclusion is that both types of adjustment are useful and that much of the obtainable gain can be achieved by making adjustments individually, focussing on simple adjustments and on those with large potential gains. Our next goal is to explore this interaction more closely and to automate the process of category level number modification. We also plan to explore the use of these techniques in problems with multi-category documents.

References

C. Apte, F. Damerau, S. M. Weiss. 1994. Automated Learning of Decision Rules for Text Categorization. *ACM Transactions on Information Systems*, 233-251.

S. Chakrabarti, B. Dom, R. Agarawal, P. Raghavan. 1997. Using Taxonomy, Discriminants and Signatures for Navigating in Text Databases. *Proceedings of the 23rd VLDB Conference*; Athens, Greece.

W.W. Cohen and Y. Singer. 1996. Context-Sensitive Learning Methods for Text Categorization. *Proceedings of the 19th Annual ACM/SIGIR Conference.*

S. D'Alessio, A. Kershenbaum, K. Murray, R. Schiaffino. 1998. Hierarchical Text Categorization. *Technical Report CS-98-BASK* (URL http://pride.poly.edu/textmining).

S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6), pp. 391-407.

W. B. Frakes and R. Baeza-Yates. 1992. Information Retrieval: *Data Structures and Algorithms.* Prentice-Hall.

D. Heckerman. 1996. Bayesian Networks for Knowledge Discovery. *Advances in Knowledge Discovery and Data Mining.* Fayyad, Piatetsky-Shapiro, Smyth and Uthurusamy eds., MIT Press.

W. Hersh, C. Buckley, T. Leone and D. Hickman. 1994. OHSUMED: An Interactive Retrieval Evaluation and a New Large Text Collection for Research. *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Philadelphia.

D. Koller and M. Sahami. 1996. Towards Optimal Feature Selection. *International Conference on Machine Learning*, Volume 13, Morgan-Kauffman.

D. Koller and M. Sahami. 1997. Hierarchically Classifying Documents using Very Few Words. *International Conference on Machine Learning*, Volume 14, Morgan-Kauffman, 1997.

L. Larkey and W.B. Croft. 1996. Combining Classifiers in Text Categorization. *Proceedings of the 19th Annual ACM/SIGIR Conference.*

D. Lewis. 1992. Text Representation for Intelligent Text Retrieval: A Classification-Oriented View. *Text-Based Intelligent Systems*, P.S. Jacobs, Lawrence-Erlbaum.

D. Lewis and M. Ringuette. 1994. A Comparison of Two Learning Algorithms for text Categorization. *Third Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, pp. 81-93.

H.-T. Ng, W.-B. Goh and K.-L.Low. 1997. Feature Selection, Perception Learning and a Usability Case Study. *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Philadelphia, July 27-31, pp. 67-73

M. Sahami. 1996. Learning Limited Dependence Bayesian Classifiers. *Proc. KDD-96*, pp.335-338.

G. Salton. 1989. *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*, Addison-Wesley.

C.J. van Rijsbergen. 1979. *Information Retrieval.* Buttersworth, London, second edition.

I.H. Witten, A. Moffat and T. Bell. 1994. *Managing Gigabytes.* Van Nostrand Reinhold.

J.W.T. Wong, W.K. Wan and G. Young. 1996. ACTION: Automatic Classification for Full-Text Documents. *SIGIR Forum 30(1)*, pp. 11-25.

Y. Yang. 1997. An Evaluation of Statistical Approaches to Text Categorization. *Technical Report CMU-CS-97-127, Computer Science Department, Carnegie Mellon University.*

Y. Yang. 1996. An Evaluation of Statistical Approaches to MEDLINE Indexing. *Proceedings of the AMIA* , pp. 358-362.

Y. Yang and C.G. Chute. 1992. A Linear Least Squares Fit Mapping Method for Information Retrieval from Natural Language Texts. *Proceedings of COLING '92*, pp. 447-453.

Y. Yang and J.P. Pederson. 1997. Feature Selection in Statistical Learning of Text Categorization, *International Conference on Machine Learning*, Volume 14, Morgan-Kauffman.

UMLS Knowledge Sources 8th Edition; National Library of Medicine, January, 1997.

An Empirical Approach to Text Categorization based on Term Weight Learning

Fumiyo Fukumoto and Yoshimi Suzuki†
Department of Computer Science and Media Engineering,
Yamanashi University
4-3-11 Takeda, Kofu 400-8511 Japan
{fukumoto@skye,ysuzuki@suwa†}.esi.yamanashi.ac.jp

Abstract

In this paper, we propose a method for text categorization task using term weight learning. In our approach, learning is to learn true keywords from the error of clustering results. Parameters of term weighting are then estimated so as to maximize the true keywords and minimize the other words in the text. The characteristic of our approach is that the degree of context dependency is used in order to judge whether a word in a text is a true keyword or not. The experiments using *Wall Street Journal* corpus demonstrate the effectiveness of the method.

Introduction

With increasing numbers of machine readable documents becoming available, an automatic text categorization which is the classification of text with respect to a set of pre-categorized texts, has become a trend in IR and NLP studies.

One of the important issues in text categorization task is how to characterize texts which are pre-categorized. There are at least two statistical approaches to cope with the issue, i.e. statistical approach that relies mainly on (1) *surface information* of words in texts, and (2) *semantic information* of words in texts.

Statistical approach based on surface information of words has been widely studied in IR. One representative is a vector model. In this model, each text is represented by a *vector*, i.e. every text which should be classified and texts which are pre-categorized in a training phase are characterized by a vector, each dimension of which is associated with a specific word in texts, and every coordinate of the text is represented by term weighting. Then, some similarity measure is used and the text is assigned to the most semantically similar set of texts which are pre-categorized. Term weighting method is widely studied [Luhn1958], [Salton and Yang1973], [Salton1988], [Jones1973]. Guthrie and Yuasa used word frequencies for weighting [Guthrie and Walker1994], [Yuasa et al.1995], and Tokunaga used weighted inverse document frequency (WIDF) which is a word frequency within the document divided by its frequency throughout the entire document collection [Tokunaga and Iwayama1994].

The other approach is based on a probabilistic model. This approach is widely used, since it has solid formal grounding in probability theory. Iwayama et. al. proposed a probabilistic model called *Single random Variable with Multiple Values* (*SVMV*) [Iwayama and Tokunaga1994]. They reported that the result of their experiment using *SVMV* was better than other probabilistic models; *Component Theory*(CT) [Kwok1989], *Probabilistic Relevance Weighting*(PRW) [Robertson and Jones1976] and *Retrieval with Probabilistic Indexing*(RPI) [Fuhr1989] in the task of categorizing news articles from the *Wall Street Journal*(*WSJ*). Most previous approaches seem to show the effect in entirely different texts, such as 'weather forecasts', 'medical reports' and 'computer manuals'. Because each different text is characterized by a large number of words which appear frequently in one text, but appear seldom in other texts. However, in some texts from the same domain such as 'weather forecasts', one encounters quite a large number of words which appear frequently over texts. Therefore, how to characterize every text is a serious problem in such the restricted subject domain.

The other statistical approach is based on semantic information of words. The technique developed by Walker copes with the discrimination of polysemy [Walker and Amsler1986]. The basic idea of his approach is that to disambiguate word-senses in articles might affect the accuracy of context dependent classification, since the meaning of a word characterizes the domain in which it is used. He used the semantic codes of the *Longman Dictionary of Contemporary English* to determine the subject domain for a set of texts. For a given text, each word is checked against the dictionary to determine the semantic codes associated with it. By accumulating the frequencies for these senses and then ordering the list of categories in terms of frequency, the subject matter of the text can be identified. However, Fukumoto reported that when using disambiguated word-senses within texts (49 different texts, each of which consists of 3,500 sentences) were up to only 7.5% as those when using word frequencies for

weighting, since in a restricted subject domain such as *Wall Street Journal*, lots of nouns in articles were used with the same sense. As a result, the results of word-sense disambiguation did not strongly contribute to an accurate classification [Fukumoto and Suzuki1996].

Blosseville et. al. proposed an automated method of classifying research project descriptions using textual and non-textual information associated with the projects. Textual information is processed by two methods of analysis: a NL analysis followed by a statistical analysis. Non-textual information is processed by a symbolic learning technique. The results using two classification sets showed that 90.6% for 7 classes and 79.9% for 28 classes could be classified correctly. Their method, however, requires a great effort, since the input data are not raw textual data, but rather the result of deep syntactic and semantic analysis of textual data.

In this paper, we propose an alternative method for an automatic classification, i.e. a method for term weight learning which is used to characterize texts. In our approach, learning is to learn true keywords from the error of clustering results. Parameters of term weighting are then estimated so as to maximize the true keywords and minimize the other words in the text. The characteristic of our approach is that the degree of context dependency is used in order to judge whether a word in a text is a true keyword or not. We applied our technique to the task of categorizing news articles from 1989 *WSJ* in order to see how our method can be used effectively to classify each text into a suitable category.

In the following sections, we first present a basic idea of context dependency, and describe how to recognize keywords. Next, we describe methods for term weight learning and for classifying texts using term weight learning. Then, we present a method for categorization task. Finally, we report on some experiments in order to show the effect of the method.

Training the Data

Recognition of Keywords

In our approach, learning is to learn true keywords from the error of clustering results. The basic idea of our term weight learning is to use the fact that whether a word is a key in a text or not depends on the domain to which the text belongs.

We will focus on the *WSJ* corpus. Let 'stake' be a keyword and 'today' not be a keyword in the text (article). If the text belongs to a restricted subject domain, such as 'Economic news', there are other texts which are related to the text. Therefore, the frequency of 'stake' and 'today' in other texts are similar to each other. Let us further consider a broad coverage domain such as all texts of the *WSJ*; i.e. the text containing the words 'stake' and 'today' belongs to the *WSJ* which consists of different subject domains such as 'Economic news' or 'International news'. 'Today' should appear frequently with every text even in such a domain, while 'stake' should not. Our technique for recognition of true keywords explicitly exploits this feature of context dependency of word: how strongly a word is related to a given context?

Like Luhn's assumption of keywords, our method is based on the fact that a writer normally repeats certain words (keywords) as he advances or varies his arguments and as he elaborates on an aspect of a subject [Luhn1958]. Figure 1 shows the structure of the *WSJ* corpus.

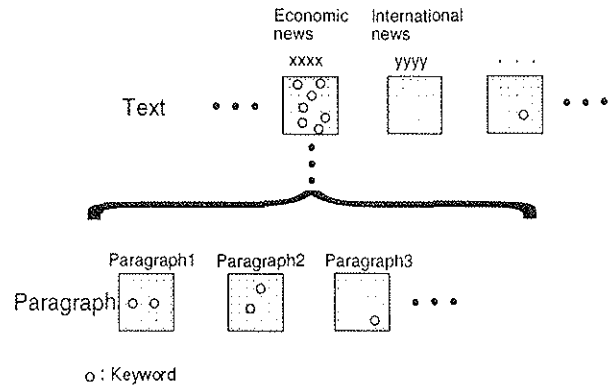


Figure 1: The structure of the *WSJ* corpus

In Figure 1, 'xxxx' and 'yyyy' shows a title name of a text which belongs to the category, 'Economic news' and 'International news', respectively.

We introduce a degree of context dependency into the structure of the *WSJ* corpus shown in Figure 1 in order to recognize keywords. A degree of context dependency is a measure showing how strongly each word is related to a particular paragraph or text. In Figure 1, let '◯' be a keyword in the text 'xxxx'. According to Luhn's assumption, '◯' frequently appears throughout paragraphs. Therefore, the deviation value of '◯' in the paragraph is small. On the other hand, the deviation value of '◯' in the text is larger than that of the paragraph, since in texts, '◯' appears in the particular text, 'xxxx'. We extracted keywords using this feature of the degree of context dependency. In Figure 1, if a word is a keyword in a given text, it satisfies that the deviation value of a word in the paragraph is smaller than that of the text, and is shown in formula (1) [Fukumoto et al.1997].

$$\frac{\chi P_w^2}{\chi T_w^2} < 1 \tag{1}$$

where,

$$\chi P_w^2 = \sqrt{\frac{\sum_{j=1}^{n}(\chi P_{wj}^2 - \bar{v}_w)}{w}} \tag{2}$$

$$\chi P_{wj}^2 = \frac{(x_{wj} - \bar{v}_{wj})^2}{\bar{v}_{wj}} \tag{3}$$

$$\bar{v}_{wj} = \frac{\sum_{j=1}^{n} x_{wj}}{\sum_{w=1}^{m}\sum_{j=1}^{n} x_{wj}} \times \sum_{w=1}^{m} x_{wj} \qquad (4)$$

In formula (1), w of χP_w^2 and χT_w^2 is a word in paragraph and text, respectively. χP_w^2 and χT_w^2 is the deviation value of a set of paragraph and text, respectively. In formula (2), n is the number of paragraphs, and $\bar{v}_w$ is the mean value of the total frequency of word w in paragraphs which consist of n. In formula (3), x_{wj} is the frequency of word w in the j-th paragraph. $\bar{v}_{wj}$ in formula (3) is shown in (4) where m is the number of different words and n is the number of paragraphs [1].

Term Weight Learning

In our method, non-overlapping group average clustering algorithm based on frequency-based term weighting is applied to every text which is pre-categorized. If a text which could not be clustered correctly in the process of clustering, then, recognition of keywords is performed.

Let T_x and $T_{x'}$ be the same category and T_y not be the same one with T_x. Let also T_x and T_y be judged to be the same category incorrectly. Recognition of keywords is shown in Figure 2.

In Figure 2, (a-1) and (b-1) are the procedures to extract keywords, and (a-2) and (b-2) are the procedures to extract other words. In (a), for example, when w is judged to be a keyword, term weighting of w is $\alpha \times f(w)$, where $f(w)$ is a frequency of w. On the other hand, when w is judged not to be a keyword, term weighting of w is $\beta \times f(w)$. Here, α and β is a variable which is concerned with a true keyword and the other words, respectively[2]. In $\frac{\chi P_w^2}{\chi T_w^2} < 1$ shown in Figure 2, the texts are T_x and T_y.

Clustering Texts based on Term Weight Learning

The clustering algorithm for pre-categorization of texts is shown in Figure 3.

As shown in Figure 3, the algorithm is composed of three procedures: **Make-Initial-Cluster-Set**, **Apply-Clustering** and **Term-Weight-Learning**[3].

1. **Make-Initial-Cluster-Set**

   The procedure **Make-Initial-Cluster-Set** produces all possible pairs of texts in the input with their similarity values. Firstly, every text which is the pre-categorization of texts is represented by a vector. Using a term weighting method, every text would be

[1] In formulae (2), (3) and (4), we can replace χP_w^2 with χT_w^2.

[2] In the experiment, two procedures are performed alternatively; (1) increment value of α is set to 0.001 and β is a constant value, (2) decrease value of β is set to 0.001 and α is a constant value.

[3] The largest value of α is empirically determined.

```
begin
   do Make-Initial-Cluster-Set
   for i := 1 to m(m-1)/2 do
      do Apply-Clustering
      if T_x such that T_x does not belong to
         the correct cluster
         then do Term-Weight-Learning
                 do Make-Initial-Cluster-Set
                 i := 1
      end_if
   end_for
end
```

Figure 3: Flow of the algorithm

represented by a vector of the form

$$T_i = (X_{i1}, X_{i2}, \cdots, X_{ix}) \qquad (5)$$

where x is the number of nouns in a text and X_{ij} is a frequency with which the noun X_j appears in text T_i.

Given a vector representation of texts T_1, $\cdots$, T_m (where m is the number of texts) as in formula (5), a similarity between two texts T_i and T_j would be obtained by using formula (6). The similarity between T_i and T_j is measured by the inner product of their normalized vectors and is defined as follows:

$$Sim(T_i, T_j) = \frac{T_i \cdot T_j}{|\, T_i \,||\, T_j \,|} \qquad (6)$$

The greater the value of $Sim(T_i, T_j)$ is, the more similar T_i and T_j. For texts T_1, $\cdots$, T_{m-1} and T_m, we calculate the similarity value of all possible pairs of texts. The result is a list of pairs which are sorted in the descending order of their similarity values. The list is called ICS (Initial Cluster Set). In the FOR-loop in the algorithm, a pair of texts is retrieved from ICS, one at each iteration, and passed to the next two procedures.

2. **Apply-Clustering**

   In this procedure, the clustering algorithm is applied to the sets and produces a set of clusters, which are ordered in the descending order of their semantic similarity values. We adopted non-overlapping group average method in our clustering technique [Jardine and Sibson1968]. Let T_x and $T_{x'}$ be the same category and T_y not be the same one with T_x. Let also T_x and T_y be judged to be the same category incorrectly. The next procedure, **Term-Weight-Learning** is applied to T_x, $T_{x'}$ and T_y.

3. **Term-Weight-Learning**

   For T_x, $T_{x'}$ and T_y ($T_{y'}$), recognition of keywords shown in Figure 2 is applied, and every text would be represented by a vector of the form

$$T_i = (X'_{i1}, X'_{i2}, \cdots, X'_{ix}) \qquad (7)$$

```
begin
(a) if T_y' such that T_y' and T_y be the same category exists
      for all w such that T_x ∩ T_y
        if w satisfies χP_w^2 / χT_w^2 < 1 and w is the element of T_x ∩ T_x' or T_y ∩ T_y'
(a-1)     then w is judged to be a keyword and parameter of term weighting of w is set to α (1 < α < 10)
        else if w does not satisfy χP_w^2 / χT_w^2 < 1 and w is the element of T_x ∩ T_x' or T_y ∩ T_y'
(a-2)     then w is judged not to be a keyword and parameter of term weighting of w is set to β (0 < β < 1)
        end_if
      end_for
(b) else
      for all w such that T_x ∩ T_y
        if w satisfies χP_w^2 / χT_w^2 < 1 and w is the element of T_x ∩ T_x'
(b-1)     then w is judged to be a keyword and parameter of term weighting of w is set to α (1 < α < 10)
        else if w does not satisfy χP_w^2 / χT_w^2 < 1 and w is the element of T_x ∩ T_x'
(b-2)     then w is judged not to be a keyword and parameter of term weighting of w is set to β (0 < β < 1)
        end_if
      end_for
    end_if
end
```

Figure 2: Recognition of keywords

where x is the number of nouns in a text and X'_{ij} is as follows;

$$X'_{ij} = \begin{cases} 0 & X'_j \text{ does not appear in } T_i \\ \alpha \times f(X'_j) & X'_j \text{ is a keyword and appears in } T_i \\ \beta \times f(X'_j) & X'_j \text{ is not a keyword and appears in } T_i \end{cases}$$

where $f(X'_j)$ is a frequency with which the noun X'_j appears in text T_i.

α and β are estimated so as to maximize $Sim(T_x, T_{x'})$ and $Sim(T_y, T_{y'})$ among all possible pairs of texts, T_x, $T_{x'}$, T_y and $T_{y'}$.

Make-Initial-Cluster-Set where every text except T_x, $T_{x'}$, T_y and $T_{y'}$ would be represented by a vector of the form shown in formula (5) and T_x, $T_{x'}$, T_y and $T_{y'}$ would be represented by a vector shown in formula (7), is applied to an arbitrary pair in texts, and the procedures are repeated.

If the newly obtained cluster contains all the texts in input, the whole process terminates.

Category Assignment

For the training data, $T_1, \cdots, T_m$ (where m is the number of texts), clustering algorithm which is shown in Figure 3 is applied, and all texts are classified into a suitable category. Given a new text T which should be classified, T would be represented by a term vector of the form shown in formula (5). The similarities between T and each text of the training data are calculated by using formula (6). Then, $T_1, \cdots, T_m$ are sorted in the descending order of their similarity values. T is assigned to the categories which are assigned to $T_1, \cdots, T_m$ with the descending order of their similarity values.

Lewis proposed the *proportional assignment strategy* based on the probabilistic ranking principle [Lewis1992]. Each category is assigned to its top scoring texts in proportion to the number of times the category was assigned in the training data. For example, a category assigned to 2% of the training texts would be assigned to the top scoring 0.2% of the test texts if the proportionality constant was 0.1, or to 10% of the test texts if the proportionality constant was 5.0. We used this strategy for evaluation.

Experiments

We have conducted two experiments to examine the effect of our method. The first experiment, **Text Categorization Experiment** shows how the results of term weight learning can be used effectively to categorize new texts. The second experiment, **Comparison to Other Methods**, we applied *chi-square* method as a *vector model* and Iwayama's *SVMV* as a *probabilistic model* to classify texts [Iwayama and Tokunaga1994], and compared them with our method.

Data

The training data we have used is 1989 *Wall Street Journal* (*WSJ*) in ACL/DCI CD-ROM which consists of 12,380 texts [Liberman1991]. The *WSJ* are indexed with 78 categories. Texts having no category were excluded. 8,907 texts remained. Each having 1.94 categories on the average. The largest category is "Tender Offers, Mergers, Acquisitions (TNM)" which encompassed 2,475 texts; the smallest one is "Rubber (RUB)",

assigned to only 2 texts. On the average, one category is assigned to 443 texts. All 8,907 texts were tagged by the tagger [Brill1992]. We used nouns in the texts. Inflected forms of the same words are treated as single units. For example, 'share' and 'shares' are treated as the same unit. We divided 8,907 texts into two sets; one for training(4,454 texts), and the other for testing(4,453 texts).

Text Categorization Experiment

Term weight learning is applied to 4,454 texts, and each word in the texts was weighted. For the result, we applied category assignment to the 4,453 test data. The best known measures for evaluating text categorization models are *recall* and *precision*, calculated by the following equations [Lewis1992].

$$Recall = \frac{\text{the number of categories that are correctly assigned to texts}}{\text{the number of categories that should be assigned to texts}}$$

$$Precision = \frac{\text{the number of categories that are correctly assigned to texts}}{\text{the number of categories that are assigned to texts}}$$

Note that recall and precision have somewhat mutually exclusive characteristics. To raise the recall value, one can simply assign many categories to each text. However, this leads to a degradation in precision, i.e. almost all the assigned categories are false. A *breakeven* point might be used to summarize the balance between recall and precision, the point at which they are equal. We calculated breakeven points in the experiment. The result of **Text Categorization Experiment** is shown in Table 1.

Table 1: The result of the experiment

Category	Training data	Test data	Breakeven
10	2,399	1,457	0.80
20	3,893	2,452	0.77
30	5,178	3,508	0.77
40	5,828	3,994	0.76
50	7,344	4,998	0.77
60	8,475	5,976	0.76
70	11,489	6,148	0.75
78	11,649	7,305	0.75

In Table 1, 'Category' shows the number of categories which are extracted at random. 'Training data' shows the number of training texts which are included in each category shown in the 'Category'. Most of the texts in *WSJ* are classified into more than one category. Each having 1.94 categories on the average. 'Test data' in Table 1 shows the total number of the texts which is classified into 'Category'.

Comparison to Other Methods

We reported on the results of our method comparing with other two methods, i.e. chi-square value for term weighting and *Single random Variable with Multiple Values(SVMV)* which is proposed by Iwayama et.al. [Iwayama and Tokunaga1994].

The reason why we compared our method with chi-square method is the following two points:

- Chi-square value is one of the conventional text classification [Iwadera and Kikui1997].
- In our method, chi-square value is used in order to introduce a degree of context dependency.

Iwayama et. al. proposed a new probabilistic model for text categorization called *SVMV*. The probability that the document d is classified into the category c is shown in formula (8).

$$P(c \mid d) = P(c) \sum_{t_i} \frac{P(T = t_i \mid c) P(T = t_i \mid d)}{P(T = t_i)} \quad (8)$$

where,

- $P(T = t_i \mid c) = \frac{NC_i}{NC}$: NC_i is the frequency of the term t_i in the category c, and NC is the total frequency of terms in c.
- $P(T = t_i \mid d) = \frac{ND_i}{ND}$: ND_i is the frequency of the term t_i in the document d, and ND is the total frequency of terms in d.
- $P(T = t_i) = \frac{N_i}{N}$: N_i is the frequency of the term t_i in the given training documents, and N is the total frequency of terms in the training documents.
- $P(c) = \frac{D_c}{D}$: D_c is the frequency of documents that is categorized to c in the given training documents, and D is the frequency of documents in the training documents.

They reported that in their experiment using *WSJ*, the result of the breakeven points of TF*IDF which was proposed by Salton et. al. was 0.48, while the result of *SVMV* was 0.63. Furthermore, their method is similar to our technique when the following two points are considered:

- Text categorization is defined as the classification of texts with respect to a set of pre-categorized texts.
- Category assignment is based on surface information of words in texts.

Therefore, we implemented Iwayama et. al.'s method and compared it with our method. The results are shown in Figure 4.

Figure 4 shows the recall/precision trade off for each method with proportional assignment strategy. 'learning', '*SVMV*' and 'χ^2' shows the result of our method, Iwayama's method and χ^2 value, respectively. Table 2 lists the breakeven points for each method. All the breakeven points were obtained when proportionality constant was about 1.0.

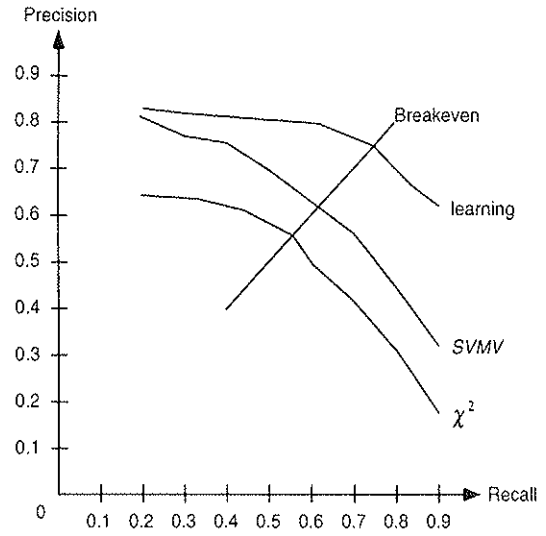


Figure 4: The result of comparative experiment

Table 2: Breakeven Points

Method	Breakeven Points
learning	0.75
SVMV	0.64
χ^2	0.56

Discussion

Text Categorization Experiment

Effectiveness of the Method According to Table 1, there are 7,305 test data in all which are classified into 78 categories, and the value of the breakeven points was 0.75. Comparing the ratios of correct judgments when the number of categories is large with when the number of it is small, the correctness of the former was higher in some cases. For example, when the number of categories was 40, the correct ratio was 0.76, while the number of categories was 50, the correct ratio was 0.77. This shows that our method can be used effectively to characterize each text without depending on the number of categories.

Table 3 shows the first top five of the highest weighted value of 12 categories which were selected from 78 categories at random.

In Table 3, 'Word' shows the extracted words, and 'Wt' shows its weighted value. 12 categories which are used in Table 3 are shown in Table 4.

According to Table 3, our technique for term weight learning is effective, though there are some nouns judged highly weighted but our intuition cannot explain why. For example, 'general' in 'FOD' is not a true keyword in our intuition.

Table 4: The category name

AIR:	Airlines	ARO:	Aerospace
BBK:	Buybacks	BNK:	Banks
FOD:	Food products	STK:	Stock market
ENV:	Environment	MED:	Media
ECO:	Economic news	PIP:	Pipeline
DIV:	Dividends	CPR:	Computers

Problem of the Method The test data which was the worst result, was the data which should be classified into 'STK'. There were 499 test data which should be classified into 'STK'. Of these, 159 data (32% in all) be judged to classify into 'BBK', incorrectly. According to Table 3, the first top three words in 'BBK' and those of 'STK' are the same, and the weighted values of these words of 'BBK' are higher than those of 'STK'. 'BBK' and 'STK are semantically similar with each other and it is difficult to distinct even for a human. Therefore, in this case, there are limitations to our method using term weight learning.

Comparison to Other Methods

(1) χ^2 method and our method Table 2 shows that the breakeven points using our method was 0.75, while χ^2 was 0.56. Table 5 shows the first top five of the highest weighted value of 12 categories using χ^2 method.

According to Table 5, every noun except 'devon' and 'hadson' in 'BBK' and 'transcanada' and 'westcoast' in 'PIP' are correctly weighted as keywords in every categories. On the other hand, the test data which was the worst result, was the same data as the result using our method, i.e. the data which should be classified into 'STK'. According to Table 5, three words in 'BBK' and those of 'STK' are the same, and the weighted values of these words of 'STK' are higher than those of 'STK'. As a result, it is difficult to distinct these two categories in χ^2 method.

One possible reason why the result of our method was better than χ^2 method is that the difference between weighting values of two words in χ^2 was smaller than those of our method. The deviation value between an arbitrary two keywords in both methods is shown in Table 6.

Table 6: Deviation value of χ^2 and our methods

Cat.	learning	χ^2	Cat.	learning	χ^2
AIR	4.63	3.64	ARO	4.20	4.12
BBK	3.80	2.57	BNK	2.23	2.25
FOD	2.25	2.72	STK	4.45	2.57
ENV	2.99	2.30	MED	3.89	6.10
ECO	4.44	2.55	PIP	3.94	3.11
DIV	4.93	3.41	CPR	4.50	3.86

Table 3: The first top 5 of the highest weighted words in our method

	AIR		ARO		BBK		PIP	
No	Word	Wt	Word	Wt	Word	Wt	Word	Wt
1	airline	522.1	aerospace	148.2	share	149.0	gas	58.0
2	mile	136.5	aircraft	143.0	stock	71.9	pipeline	37.0
3	passenger	120.5	air	730.	company	57.2	industry	29.0
4	revenue	85.0	army	51.0	bank	51.0	foothill	24.0
5	air	67.2	jetliner	43.3	security	43.5	oil	7.0
	BNK		FOD		STK		DIV	
No	Word	Wt	Word	Wt	Word	Wt	Word	Wt
1	bank	84.0	food	140.0	company	50.0	cent	85.0
2	branch	32.0	fda	27.0	share	37.7	share	70.0
3	credit	30.0	general	24.0	stock	31.7	company	60.9
4	tax	24.0	cereal	19.0	trade	10.1	dividend	54.6
5	letter	16.0	health	16.0	investment	9.4	split	46.7
	ENV		MED		ECO		CPR	
No	Word	Wt	Word	Wt	Word	Wt	Word	Wt
1	environment	78.0	news	281.0	gain	120.5	analytics	106.5
2	maquilas	19.0	d&b	108.0	tax	111.0	IBM	89.8
3	water	12.0	network	69.1	capital	83.4	machine	69.0
4	plant	10.1	report	69.0	rate	79.5	computer	62.0
5	health	9.4	broadcaster	44.8	economy	30.5	system	48.6

Table 5: The first top 5 of the highest weighted words in χ^2 method

	AIR		ARO		BBK		PIP	
No	Word	Wt	Word	Wt	Word	Wt	Word	Wt
1	airline	12109.1	boeing	4880.0	share	2348.7	pipeline	8521.7
2	ual	5268.5	force	4022.3	redemption	1902.4	foothill	5933.7
3	passenger	5142.3	aircraft	3886.7	devon	1779.4	gas	5744.4
4	pilot	4672.1	defense	2328.6	hadson	1641.1	transcanada	4948.0
5	flight	4050.8	missile	2060.7	buy-back	1616.4	weastcoast	4494.9
	BNK		FOD		STK		DIV	
No	Word	Wt	Word	Wt	Word	Wt	Word	Wt
1	bank	6196.4	spam	3148.4	stock	7265.4	dividend	10067.7
2	bnl	1517.3	food	2848.5	share	3563.2	share	4999.4
3	bond	1211.4	cereal	2627.7	buy-back	2302.0	company	3666.8
4	loan	1023.3	cholesterol	2518.2	redemption	1448.5	buy-back	2499.4
5	rate	890.1	cooke	2355.1	big	1018.6	henley	2166.6
	ENV		MED		ECO		CPR	
No	Word	Wt	Word	Wt	Word	Wt	Word	Wt
1	ozone	2650.7	magazine	4222.3	gain	2160.5	computer	13948.8
2	epa	2414.0	d&b	3313.7	democrat	1492.0	IBM	8470.1
3	asbestosis	2259.0	cable	2890.1	tax	1410.6	software	4709.2
4	anthrax	1483.5	network	2496.9	budget	1294.5	cray	3538.7
5	pollution	1165.3	broadcaster	1999.9	spending	1157.3	digital	3291.7

In Table 6, the deviation value using χ^2 method was smaller than our method except 'BNK', 'FOD' and 'MED'. This shows that χ^2 method can not represent the characteristic of the text more precisely than our method.

(2) *SVMV* method and our method According to Table 4, the breakeven points using our method was 0.75, while *SVMV* was 0.64, respectively.

A possible reason why the result of our method was better than *SVMV* is that term weight learning is effective to classify texts. Let A and B be a category name and the total number of words which were included in each category be the same. Let also w_1 is included in A, B and the test data with the same frequency, and the test data consists of only w_1. In *SVMV*, the probabilities of the test data which is classified into A and B are the same. Therefore, it could not be judged whether the test data is classified into A or B, correctly. However, our method introduces the degree of context dependency in order to judge whether a word in a text is a true keyword or not. Therefore, our method can classify the test data into A or B, when the keyword of the category A is judged to be the word w_1. As a result, our method can represent the characteristic of the texts more precisely than *SVMV*.

Conclusion

We have reported on an empirical study for term weight learning for an automatic text categorization. The characteristic of our approach is that the degree of context dependency is introduced in order to judge whether a word in a text is a true keyword or not. In the experiment using *WSJ*, we could obtain 0.75 breakeven points for 4,453 texts which are classified into 78 categories.

In our current method, category assignment is based on a word in texts, i.e. every text which should be classified and texts which are pre-categorized are characterized by a vector, each dimension of which is associated with a word in texts. As a result, two words are treated quite different even if these words are semantically similar. In order to get more accuracy, linking words with their semantically similar words might be necessary to be introduced into our framework.

Acknowledgments

The authors would like to thank the reviewers for their valuable comments. This work was partially supported by the Grant-in-aid for Scientific Research of the Ministry of Education, Science and Culture of Japan (No. 09780322).

References

[Brill1992] E. Brill. 1992. A simple rule-based part of speech tagger. In *Proc. of the 3rd Conference on Applied Natural Language Processing*, pages 152–155.

[Fuhr1989] N. Fuhr. 1989. Models for retrieval with probabilistic indexing. *Information Processing & Retrieval*, 25(1):55–72.

[Fukumoto and Suzuki1996] F. Fukumoto and Y. Suzuki. 1996. An automatic clustering of articles using dictionary definitions. In *Proc. of the 16th International Conference on Computational Linguistics*, pages 406–411.

[Fukumoto et al.1997] F. Fukumoto, Y. Suzuki, and J. Fukumoto. 1997. An automatic extraction of key paragraphs based on context dependency. In *Proc. of the 5th Conference on Applied Natural Language Processing*, pages 291–298.

[Guthrie and Walker1994] L. Guthrie and E. Walker. 1994. Document classification by machine: Theory and practice. In *Proc. of the 15th International Conference on Computational Linguistics*, pages 1059–1063.

[Iwadera and Kikui1997] T. Iwadera and G. Kikui. 1997. Automatic text categorization using trend-tracking technique. In *Proc. of the Natural Language Processing Pacific Rim Symposium*, pages 645–648.

[Iwayama and Tokunaga1994] M. Iwayama and T. Tokunaga. 1994. A probabilistic model for text categorization: Based on a single random variable with multiple values. In *Proc. of the 4th Conference on Applied Natural Language Processing*, pages 162–167.

[Jardine and Sibson1968] N. Jardine and R. Sibson. 1968. The construction of hierarchic and non-hierarchic classifications. pages 177–184.

[Jones1973] K. S. Jones. 1973. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11–21.

[Kwok1989] K. L. Kwok. 1989. Experiments with a component theory of probabilistic information retrieval based on single terms as document components. *ACM Transactions on Information Systems*, 8(4):363–386.

[Lewis1992] D. Lewis. 1992. An evaluation of phrasal and clustered representations on a text categorization task. In *SIGIR92*, pages 37–50.

[Liberman1991] M. Liberman, 1991. *CD-ROM I.* Association for Computational Linguistics Data Collection Initiative University of Pennsylvania.

[Luhn1958] H. P. Luhn. 1958. The automatic creation of literature abstracts. *IBM journal*, 2(1):159–165.

[Robertson and Jones1976] S. E. Robertson and K. Sparck Jones. 1976. Relevance weighting of search terms. Number 27, pages 129–146.

[Salton and Yang1973] G. Salton and C. S. Yang. 1973. On the specification of term values in automatic indexing. *Journal of documentation*, 29(4):351–372.

[Salton1988] G. Salton. 1988. In *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer.* Addison-Wesley.

[Tokunaga and Iwayama1994] T. Tokunaga and M. Iwayama. 1994. Text categorization based on weighted inverse document frequency. *SIG-IPS Japan*, 100(5):33–40.

[Walker and Amsler1986] D. Walker and R. Amsler. 1986. In *The Use of Machine-Readable Dictionaries in Sublanguage Analysis*, pages 69–84. Lawrence Erlbaum, Hillsdale, NJ.

[Yuasa et al.1995] N. Yuasa, T. Ueda, and F. Togawa. 1995. Classifying articles using lexical co-occurrence in large document databases. *Trans. of Information Processing Society Japan (In Japanese)*, 36(8):1819–1827.

An Empirical Evaluation on Statistical Parsing of Japanese Sentences using Lexical Association Statistics

SHIRAI Kiyoaki INUI Kentaro TOKUNAGA Takenobu TANAKA Hozumi

Department of Computer Science, Graduate School of Information Science and Engineering,
Tokyo Institute of Technology

Abstract

We are proposing a new framework of statistical language modeling which integrates lexical association statistics with syntactic preference, while maintaining the modularity of those different statistics types, facilitating both training of the model and analysis of its behavior. In this paper, we report the result of an empirical evaluation of our model, where the model is applied to disambiguation of dependency structures of Japanese sentences. We also discussed the room remained for further improvement based on our error analysis.

1 Introduction

In the statistical parsing literature, it has already been established that statistics of lexical association have real potential for improvement of disambiguation performance. The question is how lexical association statistics should be incorporated into the overall statistical parsing framework. In exploring this issue, we consider the following four basic requirements:

- *Integration of different types of statistics*:
  Lexical association statistics should be integrated with other types of statistics that are also expected to be effective in statistical parsing, such as short-term POS n-gram statistics and long-term structural preferences over parse trees.
- *Modularity of statistics types*:
  The total score of a parse derivation should be decomposable into factors derived from different types of statistics, which would facilitate analysis of a model's behavior in terms of each statistics type.
- *Probabilistically well-founded semantics*:
  The language model used in a statistical parser should have probabilistically well-founded semantics, which would also facilitate the analysis of the model's behavior.
- *Trainability*:
  Since incorporation of lexical association statistics would make the model prohibitively complex, the model's complexity should be flexibly controllable depending on the amount of available training data.

However, it seems to be the case that no existing framework of language modeling [2, 4, 12, 13, 14, 17, 18] satisfies these basic requirements simultaneously[1]. In this context, we newly designed a framework of statistical language modeling taking all of the above four requirements into account [8, 9]. This paper reports on the results of our preliminary experiment where our framework was applied to structural disambiguation of Japanese sentences.

In what follows, we first briefly review our framework (Section 2). We next describe the setting of our experiment, including a brief introduction of Japanese dependency structures, the data sets, the baseline of the performance, etc. (Section 3). We then describe the results of the experiment, which was designed to assess the impact of the the incorporation of lexical association statistics (Section 4). We finally discuss the current problems revealed through our error analysis, suggesting some possible solutions (Section 5).

2 Overview of our framework

As with the most statistical parsing frameworks, given an input string A, we rank its parse derivations according to the joint distribution $P(R, W)$, where W is a word sequence candidate for A, and R is a parse derivation candidate for W whose terminal symbols constitute a POS tag sequence L (see Figure 1[2]). We first decompose $P(R, W)$

[1] For further discussion, see [8]. This is also the case with recent works such as [3] and [5] due to the lack of modularity of statistical types.

[2] Although syntactic structure R is represented as a dependency structure in this figure, our framework

into two submodels, the syntactic model $P(R)$ and the lexical model $P(W|R)$:

$$P(R,W) = P(R) \cdot P(W|R) \quad (1)$$

The syntactic model, which is lexically insensitive, reflects both POS n-gram statistics and structural preference, whereas the lexical model reflects lexical association statistics. This division of labor allows for distinct modularity between the syntactic-based statistics and lexically sensitive statistics, while maintaining the probabilistically well-foundedness of the overall model.

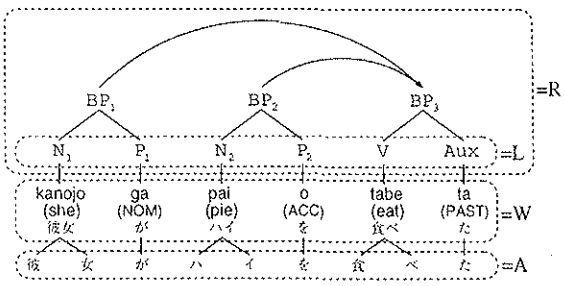


Figure 1: A parse derivation for an input string "彼女がパイを食べた (She ate a pie)"

2.1 The syntactic model

The syntactic model $P(R)$ can be estimated using a wide range of existing syntactic-based language modeling frameworks, from simple PCFG models to more context-sensitive models including those proposed in [2, 13, 19]. Among these, we, at present, use probabilistic GLR (PGLR) language modeling, which is given by incorporating probabilistic distributions into the GLR parsing framework [10, 21]. The advantages of PGLR modeling are (a) PGLR models are mildly context-sensitive, compared with PCFG models, and (b) PGLR models inherently capture both structural preferences and POS bigram statistics, which meets our integration requirement. For further discussion, see [10].

2.2 The lexical model

The lexical model $P(W|R)$ is the product of the probability of each lexical derivation $l_i \rightarrow w_i$, where $l_i \in L$ ($L \subset R$) is the POS tag of $w_i \in W$:

$$P(W|R) = \prod_i P(w_i|R, w_1, \ldots, w_{i-1}) \quad (2)$$

The key idea for estimating each factor $P(w_i|R, w_1, \ldots, w_{i-1})$ (a lexical derivation probability) is in assuming that each lexical derivation depends only on a certain small part of its whole context. We first assume that syntactic structure R in $P(w_i|R, w_1, \ldots, w_{i-1})$ can always be reduced to l_i ($\in R$), which allows us to deal with the lexical model separately from the syntactic model. The question then is which subset C of $\{w_1, \ldots, w_{i-1}\}$ has the strongest influence on the derivation $l_i \rightarrow w_i$. We refer to a member of such a subset C as a *lexical context* of the derivation $l_i \rightarrow w_i$.

Let us illustrate this through the previous example shown in Figure 1. Suppose that the derivation order for W is head-driven, as given below, to guarantee that, for each of the words subordinated by a head word, the context of the derivation of that subordinated word always includes that head word.

ta (PAST) $\rightarrow$ *tabe* (eat) $\rightarrow$ *ga* (NOM) $\rightarrow$ *o* (ACC) $\rightarrow$ *kanojo* (she) $\rightarrow$ *pai* (pie)

First, for each lexical item that we don't consider any lexical association, we estimate the probability of its derivation as follows.

$$P(ta|R) \approx P(ta|Aux) \quad (3)$$
$$P(tabe|R, ta) \approx P(tabe|V) \quad (4)$$

Second, we estimate the probability of deriving each slot-marker, e.g. "*ga* (NOM)" and "*o* (ACC)", by considering not only the dependency between the head word and each of its slot-markers, but also the dependency between slot-markers subordinated by the same head:

$$P(ga|R, tabe, ta) \approx P(ga|P_1[\mathrm{h}(tabe, [P_1, P_2])]) \quad (5)$$
$$P(o|R, ga, tabe, ta) \approx P(o|P_2[\mathrm{h}(tabe, [P_1 : ga, P_2])]) \quad (6)$$

where $\mathrm{h}(h, [s_1, \ldots, s_n])$ is a lexical context denoting a head word h that subordinates the set of slots $s_1, \ldots, s_n$, and $P(w_i|l_i[\mathrm{h}(h, [s_1, \ldots, s_n])])$ is the probability of a lexical derivation $l_i \rightarrow w_i$, given that w_i functions as a slot-marker of lexical head $\mathrm{h}(h, [s_1, \ldots, s_n])$.

Finally, we estimate the probability of deriving each slot-filler, e.g. "*kanojo* (she)" and "*pai* (pie)", in assuming that the derivation of a slot-filler depends only on its head word and slot:

$$P(kanojo|R, ga, o, tabe, ta) \approx P(kanojo|N[\mathrm{s}(tabe, ga)]) \quad (7)$$
$$P(pai|R, kanojo, ga, o, tabe, ta) \approx P(pai|N[\mathrm{s}(tabe, o)]) \quad (8)$$

where $\mathrm{s}(h, s)$ is a lexical context denoting a slot s of a head word h, and $P(w_i|l_i[\mathrm{s}(h, s)])$ is the

does not impose any restriction on the representation of syntactic structures.

probability of a lexical derivation $l_i \rightarrow w_i$ given that w_i functions as a filler of a slot $s(h, s)$.

Combining equations (3), (4), (5), (6), (7) and (8), we produce (9):

$$\begin{aligned} P(W|R) \approx\ & P(ta|Aux) \cdot P(tabe|V) \cdot \\ & P(ga|P[\mathrm{h}(tabe, [P, P])]) \cdot \\ & P(o|P[\mathrm{h}(tabe, [P : ga, P])]) \cdot \\ & P(kanojo|N[\mathrm{s}(tabe, ga)]) \cdot \\ & P(pai|N[\mathrm{s}(tabe, o)]) \end{aligned} \quad (9)$$

2.3 Handling multiple lexical contexts

Note that a lexical derivation may be associated with more than one lexical context (multiple lexical contexts). Multiple lexical contexts appear typically in coordinate structures. For example, in the sentence shown in Figure 2, "*kanojo-wa* (she-TOP)" functions as the case of both of the verbs "*tabe* (eat)" and "*dekake* (leave)".

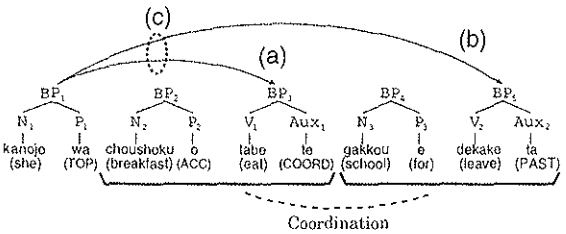


Figure 2: An example sentence containing a coordinate structure: "She ate breakfast and left for school"

Let us first consider the lexical derivation probability for the slot-filler "*kanojo* (she)". According to the assumption mentioned in Section 2.2, the lexical contexts of this slot-filler should be $s(tabe, wa)$ and $s(dekake, wa)$. Thus, the probability of deriving it is $P(kanojo|N_1[s(tabe, wa), s(dekake, wa)])$. More generally, if a slot-filler w_i is associated with two lexical contexts c_1 and c_2, then the probability of deriving w_i can be estimated as follows:

$$\begin{aligned} & P(w_i|l_i[c_1, c_2]) \\ &= \frac{P(l_i[c_1, c_2]|w_i) \cdot P(w_i)}{P(l_i[c_1, c_2])} && (10) \\ &\approx \frac{P(l_i[c_1]|w_i) \cdot P(l_i[c_2]|l_i, w_i) \cdot P(w_i)}{P(l_i[c_1]) \cdot P(l_i[c_2]|l_i])} && (11) \\ &= P(w_i|l_i) \cdot \frac{P(w_i|l_i[c_1])}{P(w_i|l_i)} \cdot \frac{P(w_i|l_i[c_2])}{P(w_i|l_i)} && (12) \\ &= P(w_i|l_i) \cdot D(w_i|l_i[c_1]) \cdot D(w_i|l_i[c_2]) && (13) \end{aligned}$$

In (13), we assume that the two lexical contexts c_1 and c_2 are mutually independent given l_i (and w_i):

$$P(l_i[c_2]|l_i[c_1]) \approx P(l_i[c_2]|l_i) \quad (14)$$

$$P(l_i[c_2]|l_i[c_1], w_i) \approx P(l_i[c_2]|l_i, w_i) \quad (15)$$

$D(w_i|l_i[c])$ is what we call a lexical dependency parameter, which is given by:

$$D(w_i|l_i[c]) = \frac{P(w_i|l_i[c])}{P(w_i|l_i)} \quad (16)$$

$D(w_i|l_i[c])$ measures the degree of the dependency between the lexical derivation $l_i \rightarrow w_i$ and its lexical context c. It is close to one if w_i and c are highly independent. It becomes greater than one if w_i and c are positively correlated, whereas it becomes less than one and close to zero if w_i and c are negatively correlated. Thus, if we set a lexical dependency parameter to one, that means we create a model that neglects the dependency associated with that parameter. For example, the probability of deriving "*kanojo* (she)" in Figure 2 is calculated as follows.

$$\begin{aligned} & P(kanojo|N_1[\mathrm{s}(tabe, wa), \mathrm{s}(dekake, wa)]) \\ &\approx P(kanojo|N_1) \cdot D(kanojo|N_1[\mathrm{s}(tabe, wa)]) \\ &\quad \cdot D(kanojo|N_1[\mathrm{s}(dekake, wa)]) \end{aligned} \quad (17)$$

Let us then move to the estimation of the probability of deriving the slot-markers "*wa* (TOP)", "*o* (ACC)", and "*e* (for)", where "*wa*" is associated with both "*tabe* (eat)" and "*dekake* (leave)", while "*o*" is associated only with "*tabe*", and "*ni*" is associated only with "*dekake*". To be mode general, let slot-marker w_0 is associated with two lexical contexts c_1 and c_2, and slot-markers w_1 and w_2 are, respectively, associated with c_1 and c_2. Assuming that w_1 and w_2 are mutually dependent, being both dependent on w_0, and c_1 and c_2 are mutually independent, the joint probability of the derivations of w_0, w_1 and w_2 can be estimated as (20) in Figure 3, similar to (13). For example, the probability of deriving "*wa* (TOP)", "*o* (ACC)", and "*e* (for)" in Figure 2 is calculated as (21) in Figure 3.

Summarizing equations (2), (13) and (16), the lexical model $P(W|R)$ can be estimated by the product of the context-free distribution of the lexical derivations $P_{cf}(W|L)$ and the degree of the dependency between the lexical derivations $D(W|R)$:

$$P(W|R) \approx P_{cf}(W|L) \cdot D(W|R) \quad (22)$$

$$P_{cf}(W|L) = \prod_{i=1}^{m} P(w_i|l_i) \quad (23)$$

$$D(W|R) = \prod_{i=1}^{m} \prod_{c \in C_{w_i}} D(w_i|l_i[c]) \quad (24)$$

where C_{w_i} is the set of the lexical contexts of w_i.

$$P(w_0, w_1, w_2 | l_0[\mathrm{h}(h_1, [l_0, l_1]), \mathrm{h}(h_2, [l_0, l_2])], l_1[\mathrm{h}(h_1, [l_0, l_1])], l_2[\mathrm{h}(h_2, [l_0, l_2])])$$

$$\approx P(w_0|l_0[\mathrm{h}(h_1, [l_0, l_1]), \mathrm{h}(h_2, [l_0, l_2])]) \cdot P(w_1|l_1[\mathrm{h}(h_1, [l_0 : w_0, l_1])]) \cdot P(w_2|l_2[\mathrm{h}(h_2, [l_0 : w_0, l_2])]) \quad (18)$$

$$\approx P(w_0|l_0) \cdot \frac{P(w_0|l_0[\mathrm{h}(h_1, [l_0, l_1])])}{P(w_0|l_0)} \cdot \frac{P(w_0|l_0[\mathrm{h}(h_2, [l_0, l_2])])}{P(w_0|l_0)} \cdot P(w_1|l_1[\mathrm{h}(h_1, [l_0 : w_0, l_1])]) \cdot P(w_2|l_2[\mathrm{h}(h_2, [l_0 : w_0, l_2])]) \quad (19)$$

$$= P(w_0|l_0) \cdot D(w_0|l_0[\mathrm{h}(h_1, [l_0, l_1])]) \cdot D(w_0|l_0[\mathrm{h}(h_2, [l_0, l_2])]) \cdot P(w_1|l_1) \cdot D(w_1|l_1[\mathrm{h}(h_1, [l_0 : w_0, l_1])]) \cdot P(w_2|l_2) \cdot D(w_2|l_2[\mathrm{h}(h_2, [l_0 : w_0, l_2])]) \quad (20)$$

$$P(wa, o, e | P_1[\mathrm{h}(tabe, [P_1, P_2]), \mathrm{h}(dekake, [P_1, P_3])], P_2[\mathrm{h}(tabe, [P_1, P_2])], P_3[\mathrm{h}(dekake, [P_1, P_3])])$$

$$\approx P(wa|P_1) \cdot D(wa|P_1[\mathrm{h}(tabe, [P_1, P_2])]) \cdot D(wa|P_1[\mathrm{h}(dekake, [P_1, P_3])]) \cdot P(o|P_2) \cdot D(o|P_2[\mathrm{h}(tabe, [P_1 : wa, P_2])]) \cdot P(e|P_3) \cdot D(e|P_3[\mathrm{h}(dekake, [P_1 : wa, P_3])]) \quad (21)$$

Figure 3: The joint probability of the derivations of slot-markers

2.4 Summary of our model

From equations (1) and (22), the overall distribution $P(R, W)$ can be decomposed as follows:

$$P(R, W) \approx P(R) \cdot P_{cf}(W|L) \cdot D(W|R) \quad (25)$$

where the first term $P(R)$ reflects part-of-speech bigram statistics and structural preference, the second term $P_{cf}(W|L)$ reflects the occurrence of each word, and the third term $D(W|R)$ reflects lexical association. Thus, equation (25) suggests that our model integrates these types of statistics, while maintaining modularity of lexical association.

Figure 4 shows the factors of the P(R,W) for the sentence in Figure 1. In this figure:

1. $P(R)$ reflects the syntactic preference.
2. $P_{cf}(W|L)$, which consists of $P(kanojo|N)$, $P(ga|P)$ etc., reflects the occurrence of each word.
3. $D(W|R)$, which consists of $D(o|N[\mathrm{h}(tabe, [])])$, $D(pai|N[\mathrm{s}(tabe, ACC)])$ etc., reflects the lexical association statistics.

In this way, our modeling maintains the modularity of different statistics types.

The modularity of the lexical model facilitates parameter estimation. Although the syntactic model ideally requires *fully* bracketed training corpora, training it is expected to be manageable since the model's parameter space tends to be only a small part of the overall parameter space. The lexical association statistics, on the other hand, may have a much larger parameter space, and thus may require much larger amounts of training data, as compared to the syntactic model. However, since our lexical model can be trained independently of syntactic preference, one can train it using *partially* parsed tagged corpora, which can be produced at a lower cost (i.e. automatically), as well as fully bracketed corpora. In fact, we used both a full-bracketed corpus and a partially parsed corpus in our experiment.

3 A preliminary experiment

Let us first briefly describe some fundamental features of Japanese syntax. A Japanese sentence can be analyzed as a sequence of so-called *bunsetu* phrases (BPs, hereafter) as illustrated in Figure 1. A BP is a chunk of words consisting of a content word (noun, verb, adjective, etc.) accompanied by some function word(s) (postposition, auxiliary, etc.). For example, the BP "*kanojo-ga*" (BP_1) in Figure 1 consists of the noun "*kanojo* (she)" followed by the postposition "*ga* (NOM)", which functions as a slot-marker. The BP "*tabeta*" (BP_3), on the other hand, consists of the verb "*tabe* (eat)" followed by the auxiliary "*ta* (PAST)".

Given a sequence of BPs, one can recognize dependency relations between them as illustrated in Figure 1. In Japanese, if BP_i precedes BP_j, and BP_i and BP_j are in a dependency relation, then BP_i is always the modifier of BP_j, and we say "BP_i modifies BP_j." For example, in Figure 1, both BP_1 and BP_2 modify BP_3.

For the preliminary evaluation of our model, we restricted our focus only on the model's performance for structural disambiguation excluding morphological disambiguation. Thus, the task of the parser was restricted to determination of the dependency structure of an input sentence, which is given together with the specification of word

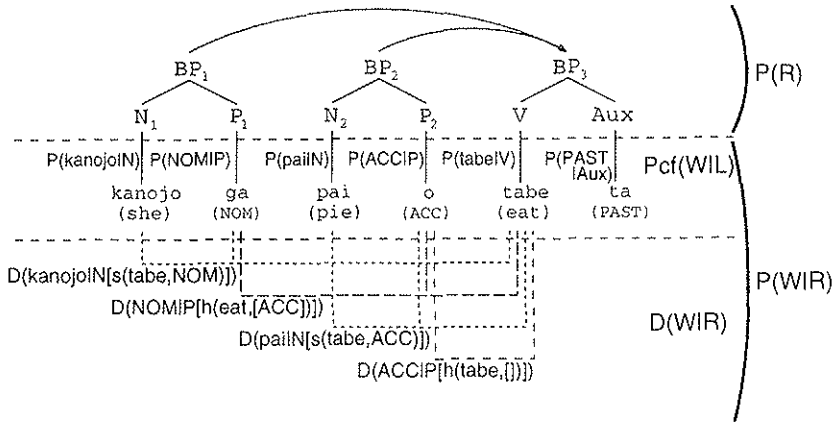


Figure 4: The summary of our model

segments, their POS tags, and the boundaries between BPs.

In developing the grammar used by our PGLR parser, we first established a categorization of BPs based on the POS of their constituents: postpositional BPs, verbal BPs, nominal predicative BPs, etc. We then developed a modification constraint matrix that describes which BP category can modify which BP category, based on examples collected from the Kyoto University text corpus [11]. We finally transformed this matrix into a CFG; for instance, the constraint that a BP of category C_i can modify a BP of category C_j can be transformed into context-free rules such as $\langle \bar{C}_j \rightarrow C_i \ \ C_j \rangle$, $\langle \bar{C}_j \rightarrow \bar{C}_i \ \ C_j \rangle$, etc., where $\bar{X}$ denotes a nonterminal symbol.

For the text data, we used roughly 10,000 sentences from the Kyoto University text corpus for training the syntactic model, and the whole EDR corpus [6] and the RWC POS-tagged corpus [16] for training the lexical model. For testing, we used 500 sentences collected from the Kyoto University text corpus with the average sentence length being 8.7 BPs. The data sets used for training and testing are mutually exclusive. The grammar used by our probabilistic GLR parser was a CFG automatically acquired from the training sentences, consisting of 967 context-free rules containing 50 nonterminal symbols and 43 terminal symbols (i.e. BP categories).

The baseline of the disambiguation performance was assessed by way of a naive strategy which selects the nearest possible modifiee (similarly to the right association principle in English) under the non-crossing constraint. The performance of this naive strategy was 62.4% in BP-based accuracy, where BP-based accuracy is the ratio of the number of the BPs whose modifiee is correctly identified to the total number of BPs (excluding the two rightmost BPs for each sentence). On the other hand, the syntactic model $P(R)$ achieved 72.1% in BP-based accuracy, 9.7 points above the baseline.

4 The contribution of the lexical model

In our experiment, we considered the following three lexical dependency parameters in the lexical model.

First, we considered the dependencies between slot-markers and their lexical head by using the lexical dependency parameter (26).

$$D(p|P[\mathrm{h}(h,[s_1,\ldots,s_n])]) \qquad (26)$$

(26) can be computed from $P(p^n|P^n[\mathrm{h}(h,[])])$, the distribution of n postpositions (slot-markers) given that all of them are subordinated by a single lexical head h. We trained this distribution using 150,000 instances of p^n-$\{verb,adjective,nominal\_predicate\}$ collocation collected from the EDR full-bracketed corpus. For parameter estimation, we used the maximum entropy estimation technique [1, 15]. For further details of this estimation process, see [20].

Next, we considered dependencies between slot-fillers and their head verb coupled with the corresponding slot-markers by using the lexical dependency parameter (27).

$$D(n|N[s(v,p)]) \qquad (27)$$

(27) was trained using 6.7 million instances of *noun-postposition-verb* collocation collected from both the EDR and RWC corpora. For parameter estimation, we used 115 non-hierarchical semantic noun classes derived from the NTT semantic

dictionary [7] to reduce the parameter space:

$$D(n|N[\mathrm{s}(v,p)]) \approx \frac{\sum_{c_n} P(c_n|N[\mathrm{s}(v,p)]) \cdot P(n|c_n)}{P(n|N)} \quad (28)$$

$P(c_n|N[\mathrm{s}(v,p)])$ was estimated using a simple back-off smoothing technique: for any given lexical verb v and postposition p, if the frequency of $\mathrm{s}(v,p)$ is less than a certain threshold λ (in our experiment, $\lambda = 100$), then $P(c_n|N[\mathrm{s}(v,p)])$ was approximated to be $P(c_n|N[\mathrm{s}(c_v,p)])$ where c_v is a class of v whose frequency is more than λ.

Finally, we considered the occurrence of postpositions by using the lexical dependency parameter (29).

$$D(p|P[head\_type]) \quad (29)$$

In Japanese, the distribution of the lexical derivation of postpositions, $P(p|P)$, is quite different depending on whether they function as slot-markers of verbs, adjectives and nominal predicates such as "*ga* (NOM)" and "*o* (ACC)" in Figure 1, or they function as slot-markers of nouns such as "*no* (of)" in the following sentence.

hana	*no*	*syashin*[3]
(flower)	(of)	(picture)

For such a reason, we introduced the lexical dependency parameter (29), where *head_type* denotes whether the postposition P functions as a slot-marker of a predicate or a noun. We estimated this dependency parameter using about 950,000 postpositions collected from the EDR corpus.

Table 1 summarizes the results of the experiment. The lexical model achieved 76.5% in BP-based accuracy, and the model using both the syntactic and lexical model achieved 82.8% in BP-based accuracy. According to these results, the contribution of lexical statistics for disambiguation is as great as that of syntactic statistics in our framework.

The bottom three lines in Table 1 denotes the setting where the only lexical dependency parameter (26), (27) and (29) are considered in the lexical model. Among these, the contribution of (29) was greatest.

Table 1: The contribution of the lexical model

	accuracy
base line	62.4 %
syntactic model only	72.1 %
lexical model only	76.5 %
syntactic + lexical model	82.8 %
syntactic model + (26)	73.4 %
syntactic model + (27)	78.3 %
syntactic model + (29)	81.3 %

5 Error analysis

In the test set, there were 574 BPs whose modifiee was not correctly identified by the system. Among these errors, we particularly explored 290 errors that were associated with postpositional BPs functioning as a case of either a verb, adjective, or nominal predicate, since, for lexical association statistics in the lexical model, we took the dependencies between slots (i.e. slot-markers and slot-fillers) and their heads into account. In this exploration, we identified three major error types: (a) errors associated with a coordinate clause, (b) errors associated with relative clauses, (c) errors associated with the lack of the consideration of dependency between slot-fillers.

5.1 Coordinate structures

One of the typical error types is associated with coordinate structures. The sentence in Figure 2 has at least three alternative interpretations in terms of which BP is modified by the left-most BP "*kanojo-wa* (she-TOP)": (a) "*tabe-ta* (eat-PAST)", (b) "*dekake-ta* (leave-PAST)", (c) both "*tabe-ta* (eat-PAST)" and "*dekake-ta* (leave-PAST)". Among these alternatives, the most reasonable interpretation is obviously (c), where the two predicative BPs constitute a coordinate structure.

In our experiment, however, neither the training data nor the test data indicates such coordinate structures. Thus, in the above sentence, for example, the system was required to choose one of two alternatives (a) and (b), where (b) is the preferred candidate according to the structural policy underlying our corpora. However, this choice is not really meaningful. Furthermore, the system systematically prefers (a), the wrong choice, since (i) the syntactic model tends to prefer shorter-distance modification relations (similarly to the right association principle in English), and (ii) the lexical model is expected to support both candidates because both $D(kanojo|N[\mathrm{s}(tabe,wa)])$ in (a) and $D(kanojo|N[\mathrm{s}(dekake,wa)])$ in (b) should be high. This problem makes the performance of our model lower than what it should be.

Obviously, the first step to resolving this problem is to enhance our corpora and grammar to enable the parser to generate the third interpretation, i.e. to explicitly generate a coordinate structure such as (c) if needed. Once such a setting is established, we then need to consider the

[3]This sentence means "a picture of a flower."

lexical contexts of each of the constituents modifying a coordinate structure, such as "*kanojo-wa* (she-TOP)" in the above sentence. In interpretation (c), since "*kanojo-wa* (she-TOP)" modifies both predicative BPs, it is reasonable to associate it with two lexical contexts, s(*tabe*, *wa*) and s(*dekake*, *wa*). As mentioned in Section 2, our framework allows us to deal with such multiple lexical contexts, namely:

$$D(kanojo|N[\mathrm{s}(tabe, wa), \mathrm{s}(dekake, wa)]) \approx D(kanojo|N[\mathrm{s}(tabe, wa)]) \cdot D(kanojo|N[\mathrm{s}(dekake, wa)]) \quad (30)$$

The correct interpretation (c) would assigned higher probability than (a) or (b), since the two lexical dependency parameters in (30), $D(kanojo|N[\mathrm{s}(tabe, wa)])$ and $D(kanojo|N[\mathrm{s}(dekake, wa)])$ are both expected to be sufficiently large.

5.2 Treatment of correference

One may have already noticed that the issue discussed above can be generalized as an issue associated with the treatment of correference in dependency structures. Namely, if a prepositional BP is correferred to by more than one clause as a participant, a naive treatment of this correference relation could require the parser to make a meaningless choice: which clause subordinates that BP. This problem in the treatment of correference is considered to cause a significant proportion of errors associated with relative/adverbial clauses or compound predicates. Such errors are expected to be resolvable through an extension of the model, as discussed in Section 5.1.

Let us briefly look at another example in Figure 5, where the matrix clause and relative clause correfer to the leftmost BP "*kanojo-wa* (she-TOP)", i.e. interpretation (c). Without any refined treatment of this correference relation, the parser would be required to make a meaningless choice between (a) and (b).

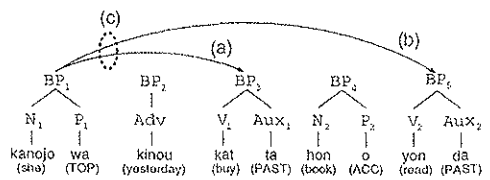


Figure 5: An example sentence containing a relative clause: "She read the book which she bought yesterday"

5.3 Dependency between slot fillers

According to the results summarized in Table 1, the contribution of the dependency between slot-fillers and their heads seems to be negligibly small. We can enumerate several possible reasons including that the estimation of these types of dependency parameters was not sufficiently sophisticated.

In addition to these reasons, we also found that the lack of the consideration of dependency between slot-fillers was also problematic in some cases; there are particular patterns where dependency between slot-fillers seems to be highly significant. For example, in the clause "*kanojo-wa* (she-TOP) *isha-ni* (doctor-DAT) *nat-ta* (become-PAST)" (she became a doctor), the distribution of the filler of the "*wa* (TOP)" slot is considered to be highly dependent on the filler of the "*ni* (DAT)" slot, "*isha* (doctor)", since its distribution would be markedly different if "*isha* (doctor)" was replaced with "*mizu* (water)". Similar patterns include, for example, "*A-wo* (ACC) *B-ni* (DAT) *suru* (make)", where *A* and *B* are highly dependent, and "*A-ga* (NOM) *B-wo* (ACC) *suru* (do)", where noun *B* indicating an action strongly influences the distribution of *A*.

In our framework, this type of problem can be treated by means of controlling the choice of lexical contexts. We are now conducting another experiment in which the dependencies between slot-fillers are additionally considered in particular patterns. Note that the refinement of our model in this manner illustrates that the modularity of lexical association statistics facilitates rule-based control in choosing the locations where lexical association is considered. This rule-based control allows us to incorporate qualitative knowledge such as linguistic insights and heuristics newly obtained from experiments based on the model.

6 Conclusion

In this paper, we first presented a new framework of language modeling for statistical parsing, which incorporates lexical association statistics while maintaining modularity. We then reported on the results of our preliminary evaluation of the model's performance, showing that both the syntactic and lexical models made a considerable contribution to structural disambiguation, and that the division of labor between those two models thus seemed to be working well to date.

Many issues remain unclear. First, we need to conduct experiments on the combination of the morphological and syntactic disambiguation tasks, which our framework intrinsically is designed for. Second, empirical comparison with other lexically sensitive models is also strongly

required. One interesting issue is whether the division of labor between the syntactic and lexical models presented in this paper works well language-independently, or conversely, whether the existing models designed for English are equally applicable to languages like Japanese.

Acknowledgements

The authors would like to thank the staff of NTT for making available their considerable electronic resources.

References

[1] A. L. Berger, S. A. Della Pietra, and V. J. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.

[2] E. Black, F. Jelinek, J. Lafferty, D. M. Magerman, R. Mercer, and S. Roukos. Towards history-based grammars: Using richer models for probabilistic parsing. In *Proceedings of the ACL*, pages 31–37, 1993.

[3] E. Charniak. Statistical parsing with a context-free grammar and word statistics. In *Proceedings of the AAAI*, 1997.

[4] M. Collins. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the ACL*, 1996.

[5] M. Collins. Three generative, lexicalised models for statistical parsing. In *Proceedings of the ACL*, 1997.

[6] EDR. The EDR electronic dictionary technical guide (second edition). Technical Report TR-045, Japan Electronic Dictionary Research Institute, 1995.

[7] S. Ikehara, M. Miyazaki, S. Shirai, A. Yokoo, H. Nakaiwa, K. Ogura, Y. Ooyama, and Y. Hayashi. *A Japanese Lexicon*. Iwanami Shoten, 1997. (In Japanese).

[8] K. Inui, K. Shirai, H. Tanaka, and T. Tokunaga. Integrated probabilistic language modeling for statistical parsing. Technical Report TR97-0005, Dept. of Computer Science, Tokyo Institute of Technology, 1997. `ftp://ftp.cs.titech.ac.jp/lab/tanaka/papers/97/inui97b.ps.gz`.

[9] K. Inui, K. Shirai, T. Tokunaga, and H. Tanaka. Integration of statistical techniques for parsing. In *summary collection of the IJCAI'97 poster session*, 1997.

[10] K. Inui, V. Sornlertlamvanich, H. Tanaka, and T. Tokunaga. A new formalization of probabilistic GLR parsing. In *Proceedings of the IWPT*, 1997.

[11] S. Kurohashi and M. Nagao. Kyoto university text corpus project. In *Proceedings of the 11th Annual Conference of JSAI*, pages 58–61, 1997. (In Japanese).

[12] H. Li. A probabilistic disambiguation method based on psycholinguistic principles. In *Proceedings of WVLC-4*, 1996.

[13] D. M. Magerman and M. Marcus. Pearl: A probabilistic chart parser. In *Proceedings of the EACL*, pages 15–20, 1991.

[14] D. M. Magerman. Statistical decision-tree models for parsing. In *Proceedings of the ACL*, pages 276–283, 1995.

[15] A. Ratnaparkhi, J. Reyner, and S. Roukos. A maximum entropy model for prepositional phrase attachment. In *Proceedings of the Human Language Technology Workshop*, pages 250–255, 1994.

[16] Real World Computing Partnership. RWC text database. `http://www.rwcp.or.jp/wswg.html`, 1995.

[17] P. Resnik. Probabilistic tree-adjoining grammar as a framework for statistical natural language processing. In *Proceedings of the COLING*, pages 418–424, 1992.

[18] Y. Schabes. Stochastic lexicalized tree-adjoining grammars. In *Proceedings of the COLING*, pages 425–432, 1992.

[19] S. Sekine and R. Grishman. A corpus-based probabilistic grammar with only two nonterminals. In *Proceedings of the IWPT*, 1995.

[20] K. Shirai, K. Inui, T. Tokunaga, and H. Tanaka. Learning dependencies between case frames using maximum entropy method. In *Proceedings of Annual Meeting of the Japan Association for Natural Language Processing*, 1997. (In Japanese).

[21] V. Sornlertlamvanich, K. Inui, K. Shirai, H. Tanaka, T. Tokunaga, and T. Takezawa. Empirical evaluation of probabilistic glr parsing. In *Proceedings of the NLPRS*, 1997.

Japanese Dependency Structure Analysis based on Lexicalized Statistics

Masakazu Fujio
NAIST
8916-5 Takayama, Ikoma
Nara, 630-0101 JAPAN
masaka-h@is.aist-nara.ac.jp

Yuji Matsumoto
NAIST
8916-5 Takayama, Ikoma
Nara, 630-0101 JAPAN
matsu@is.aist-nara.ac.jp

Abstract

We present statistical models of Japanese dependency analysis and report results of some experiments to investigate the performance of the models for the use fo a partical parsing system. The statistical models are rather simple compared with the recent complex models and intesively use lexical level information, such as morphemes, and part-of-speech tags..

We conducted several experiments to show the following properties of the models:

- performance of the models according to feature selection
- performance of the models as a partial parsing system.

The EDR[6] corpus was used for both training and evaluation of the system.

1. Introduction

A number of statistical parsing methods have been proposed. most of the systems focus on full parsing of sentences, and do not discuss the performance of partial parses, which is crucial for some applications, such as information retrieval or pre processing of corpus annotation.

Early approaches of statistical parsing [15, 10, 13] conditioned probabilities on syntactic rules. To take more contextual information into account, word collocation is applied to syntactic formalization, such as lexicalized PCFG, lexicalized tree adjoining grammar, and lexicalized link grammar.

The length of phrases or the distance between head-words were also considered in the several models [16, 8]

There are parsing methods that do not require a grammar. Collins [3] proposes a statistical parser based on probabilities of dependencies between head-words in parse trees. Yasuhara [18], constructs a system based on collocation counts as the only source of grammatical information. He uses co-occurrence patterns of the POS tags of head-words. The method, however, is not statistical, in that it only accumulates correct patterns for direct use.

Magerman [4] proposes a statistical parser based on a decision tree model, in which the probabilities are conditioned on the derivation history of the parse trees [4, 10]. He compares the decision tree model with the n-gram model, and claims that the amount of parameters in the resulting model remains relatively constant, depending mostly on the number of training examples.

Charniak [5] proposes a new model and compared it with Collins', and Magerman's models and shows what aspects of these systems affect their relative performance.

In general, statistical models suffer from the problem of data sparseness.

Instead of using a complex statistical model combined with various smoothing techniques [1, 2, 7, 9], We stick to a statistical model of simple setting aiming at an easy implementation, and pursue a way to select useful information for achieving higher parse accuracy.

The basic model is close to Collins' model[3] Japanese dependency structure are usually based on phrasal units (called "*bunsetsu*"). A *bunsetsu* basically consists of one (or a sequence of) content word(s) and its succeeding function words (that forms the smallest phrase, such as a simple noun phrase.).

We consider the dependency structure such that every *bunsetsu* in a sentence except the right most one modifies one of its following *bunsetsu*'s in the sentence and no two modifications may cross each other.

The difference of our model to Collins' model principally comes from the property of Japanese sentence structure. First, the type of modification relation (dependency relations) is uniqly determined by the function words or the ending form of the modifier. Second, the modification always direct from left to right since Japanese is a head-final language.

There are various features that may affect the parsing precision. We test a number of possible setting and try to find out the best combination of features. We also test the performance of partial parsing in several settings. 200,00 parsed Japanese sentences in EDR corpus is used for evaluation.

In the next section, the statistical model is described. Section 3 outlines the parsing algorithm is outlined. section 4 presents the evaluation method. Final section is for conclusion and future work.

2. The Statistical Model

We propose a statistical model based on the features of *bunsetsu*'s. Those features usually defined by the result of morphological analysis, such as part-of-speech (POS) tags, inflection types, punctuations, and other grammatical or surface information. Some features are determined not directly from the modifier and modifiee *bunsetsu's* For instance, the number of *bunsetsu* between a modifier and a modifiee can be a feature.

We first introduce notational conventions. $S = w_1, \ldots, w_n$ is a sentence, where w_i is the i-th word. T is a sequence of words and tag pairs, that is, $T = < w_1, t_1 >, \ldots, < w_n, t_n >$. F is a sequence of *bunsetsu* and feature pairs, that is, $F =< b_1, \mathbf{f}_1 >, \ldots, < b_m, \mathbf{f}_\mathbf{m} >$. We use the notation $Dep(i) = j$ to indicate that the i-th *bunsetsu* in the sequence is a modifier to the j-th *bunsetsu*. Here, the symbol $w_i, t_i, and b_i$ stand for word, tag, and *bunsetsu* respectively, and $\mathbf{f}_\mathbf{i}$ represents the set of features assigned to *bunsetsu* b_i. The subscripts m, *and* n stand for the number of *bunsetsu's* and words, respectively. L is the sequence of dependencies: $L = (Dep(1), Dep(2), \ldots, Dep(m-1))$.

In general, a statistical parsing model estimates the conditional probability, $P(P_t \mid S)$, for each candidate parse tree P_t for a sentence S. In Japanese dependency structure analysis, the final goal is to identify L rather than P_t, and we try to maximize the probability $P(L, F, T \mid S)$.

The most likely dependency structure analysis under the model is then:

$$\begin{aligned} L_{best} &= \operatorname*{argmax}_{L,F,T} P(L,F,T|S) \\ &= \operatorname*{argmax}_{L,F,T} P(L|F,T,S)\ P(F|T,S)\ P(T|S) \end{aligned}$$

We assume that *bunsetsu* construction only depend on word/tag pairs, hence $P(F \mid T, S) = P(F \mid T)$, and assume that a dependency structure can be determined only by *bunsetsu* features, thus $P(L \mid F, T, S) = P(L \mid F)$. The equation (1) is now written:

$$L_{best} = \operatorname*{argmax}_{L,F,T} P(L|F)\ P(F|T)\ P(T|S)$$

For simplicity, we assume that the morphological analysis and the *bunsetsu* construction are both deterministic. For the morphological analysis, we use the most likely output of the Japanese morphological analyzer ChaSen [11].

For the *bunsetsu* construction, we use a finite state transducer constructed from regular expressions of word/tag pairs.

What we need to do therefor is to estimate $P(L \mid F)$, and find L for each S that maximizes the conditional probability $P(L \mid F)$.

We assume that dependencies are mutually independent, that is,

$$P(L \mid F) = \prod_{i=1}^{m-1} P(Dep(i)=j \mid \mathbf{f}_1, \ldots, \mathbf{f}_\mathbf{m}) \qquad (1)$$

and no two modifications may cross each other.

$\mathbf{f}_1, .., \mathbf{f}_\mathbf{m}$ stands for the sequence of *bunsetsu* features assigned to the *bunsetsu*. Thus, $P(L \mid F)$ can be defined as the product of the probability of dependency pairs.

One point that differs from the Collins' model is that our model does not estimate the type of dependency relations. It only estimate the existence of the dependency relations. This is because the type of dependency is determined uniquely by the modifier in Japanese sentences.

The model estimate the probability of each dependency pair directly by maximum likelihood estimation based on *bunsetsu* features. Head-words, POS tags, word classes, function words, punctuations, and distance measure such as the number of *bunsetsu*'s are used available for the probability estimation.

We can expand each item of the equation (1) by using those features, and assuming independence of the co-occurrence of some features. In the following, we discriminate the *bunsetsu* features that directory relate to the modifier and modifiee and the distance features that relate to relative positions of the modifier and the modifiee.

$$\begin{aligned} & P\ (Dep(i)=j \mid \mathbf{f}_1, \ldots, \mathbf{f}_\mathbf{m}) \\ \approx\ & P_h(Dep(i)=j \mid \mathbf{f}_1, \ldots, \mathbf{f}_\mathbf{m}) \qquad (2) \\ & \times P_d(Dep(i)=j \mid \mathbf{f}_1, \ldots, \mathbf{f}_\mathbf{m}) \qquad (3) \end{aligned}$$

In the second equation, we assume independence of two kinds of probabilities. The first is the collocation probability between *bunsetsu features*, and the second one is the distance feature between two *bunsetsu*'s. The independency of these two probabilities reduce the size of the model.

We refer to the probability (2) as the collocation probability, and the probability (3) as the distance probability.

The remainder of this section explains these probabilities in detail.

Head Collocation Probability

Japanese language has dependency relations expressed by the function words or the ending form, and they play a crucial role in determining the dependency structure. The relation name (type) is usually determined by the function words.

If a *bunsetsu* has no function words, we use POS tag (and inflection type) of the right most content word of the *bunsetsu*.

Head word is basically defined by the right most content word in the each *bunsetsu*.

By using these features, we define two models of head-collocation probabilities. The first is the generation probability of features and the second is the collocation probability of features.

In the first model, we assume Japanese dependency structure is the result of selectional process of which each modifier selects a modifiee. The selectional probability is written as $F_g(h_j, r_j, p_j \mid h_i, r_i, p_i)$. In this expression, the modifiee's features are h_j, r_j, p_j given that modifier's features are h_i, r_i, p_i. The symbols $h_i, r_i, andp_i$ stand for head feature, relation type, and punctuation, respectively. With this setting, we make the following approximation:

$$P_h\ (Dep(i)=j \mid \mathbf{f_1}, \ldots, \mathbf{f_m}) \stackrel{\text{def}}{=} F_g(h_j, r_j, p_j \mid h_i, r_i, p_i)$$

The maximum-likelihood estimate of F_g is given as follows:

$$F_g\ (h_j, r_j, p_j \mid h_i, r_i, p_i) = \frac{C(Dep(i)=j, h_i, r_i, p_i, h_j, r_j, p_j)}{C(Dep(i)=j', h_i, r_i, p_i)}$$

$C(Dep(i)=j,\ h_i, r_i, p_i, h_j, r_j, p_j)$ is the number of times that feature pairs of h_i, r_i, p_i and h_j, r_j, p_j are in a dependency relation in the training data.

In the second model, we define the the selectional probability as $F_c(Dep(i)=j \mid h_i, r_i, p_i, h_j, r_j, p_j)$. This is the probability that *bunsetsu* b_i modifies *bunsetsu* b_j when those *bunsetsu*'s appear in the same sentence.

$$P_h\ (Dep(i)=j \mid \mathbf{f_1}, \ldots, \mathbf{f_m}) \stackrel{\text{def}}{=} F_c(Dep(i)=j \mid h_i, r_i, p_i, h_j, r_j, p_j)$$

The maximum-likelihood estimate of F_c is given as follows:

$$F_c\ (Dep(i)=j \mid h_i, r_i, p_i, h_j, r_j, p_j) = \frac{C(Dep(i)=j,\ \ h_i, r_i, p_i, h_j, r_j, p_j)}{C(h_i, r_i, p_i, h_j, r_j, p_j)}$$

$C_s(h_i, r_i, p_i, h_j, r_j, p_j)$ is the number of times h_i, r_i, p_i and h_j, r_j, p_j appear in the same sentence in the training data. $C_s(Dep(i)=j,\ h_i, r_i, p_i, h_j, r_j, p_j)$ is the number of times h_i, r_i, p_i and h_j, r_j, p_j are seen in the same sentence in the training data and b_i modifies b_j with the relation r_i.

For the head feature h_i, we can use the head word, as well as the POS tag or the word class of a head word. We use the Japanese thesaurus ' Bunrui Goi Hyou'(BGH)[12] to define word classes. BGH has a six-layered abstraction hierarchy, in which more than 80,000 words are assigned at the leaves.

For each of those probabilities explained above, we tested the following models for feature selection.

POS model	uses POS tags for the head feature.
LEX model	uses POS tags and lexical forms for the head feature.
BGH model	uses POS tags, lexical forms, and word classes for the head feature.

To acquire the statistics, we have to resolve the following ambiguities:

- Which level of thesaurus hierarchy is appropriate as the class for head-word
- How much information from the function words should be considered to define the dependency relation names.

For the limitation of computer resources, we could not use all the combination of word classes (the combination of modifier and modifiee). The collocation of word classes in the same layer in BGH was learned (from the 2nd to 6th layer) and used separately.

In the current implementation, we count the statistics for various length of dependency relation names. Consider the examples in Table 1.

Relation feature of modifier in 3 → 4 may be "まで-に" or "に". Relation feature of modifiee in 3 → 4 may be "せる" or empty.

Then, head collocation feature combinations defined fo 3 → 4 are as follows (in the case of LEX model):

[私 は]$_1$ [それ を]$_2$ [春 まで − に]$_3$ [完成 させる]$_4$ (I complete it untill this spring)

	modifier's features		modifiee's features	
	relation name	head	head	relation name
1 →4	私	は (particle)	完成	させる
2 →4	それ (demonstrative pronoun)	を (case particle)	完成	させる
3 →4	春	まで (particle)–に (case particle)	完成	させる

Table 1: Example of dependency relations. Each square bracket represents a *bunsetsu*

modifier's feature		modifiee's feature	
relation name	head	head	relation name
まで-に	春	完成	させる
まで-に	春	完成	-
に	春	完成	させる
に	春	完成	-
まで-に	Noun	完成	させる
まで-に	Noun	完成	-
に	Noun	完成	させる
に	Noun	完成	-
まで-に	春	Noun	させる
まで-に	春	Noun	-
に	春	Noun	させる
に	春	Noun	-
まで-に	Noun	Noun	させる
まで-に	Noun	Noun	-
に	Noun	Noun	させる
に	Noun	Noun	-

The Distance Probability

Distance measure of dependency relations is an important factor to disambiguate dependency structure. For instance, relation type "ha/particle" has a tendency to modify a distant phrasal unit.

For the distance measure of a pair of bunsetsu's, we use the numbers of the *bunsetsu's* and punctuations between them.

Two types of probabilities are considered for the probabilities of head-collocation described above.

Generation probability model of the distance features is as follows:

$$P_d(Dep(i)=j \mid \mathrm{f_1}, \ldots, \mathrm{f_m}) \approx F_g^d(r_i, d_{ij}, p_{ij} \mid r_i) = \frac{C(Dep(i)=j, r_i, d_{ij}, p_{ij})}{C(Dep(i)=j', r_i)}$$

Collocation probability version of the distance features is as follows:

$$P_d(Dep(i)=j) \approx F_c^d(Dep(i)=j \mid r_i, d_{if}, p_{ij}) = \frac{C(Dep(i)=j, r_i, d_{ij}, p_{ij})}{C(r_i, d_{ij}, p_{ij})}$$

d_{ij}, and p_{ij} indicate the number of *bunsetsu's* and the number of punctuations, respectively.

Same as the case of estimation of head collocation probabilities, modification relations of various length was extracted from each modification pair.

3. The Algorithm

Full Parse

1. Tokenization and POS-tagging is applied to the input
2. Construct *bunsetsu'* and define its features,
3. Calculate the probabilities of every *bunsetsu* pair, by using statistics derived from the EDR corpus.
4. Compose the most likely (or n-best) dependency structure based on the statistical model described in section 2.

For the first step, we use the morphological analyzer, ChaSen[11].

For the second step, tokens are analized into *bunsetsu'* based on pre-defined regular expressions, and then *bunsetsu* features are extracted. The basic rules for assigning features are as follows:

- The right most content word in the *bunsetsu* becomes the head feature.
- Morphological information (such as word, tag, and inflection form) of function words in the *bunsetsu* defines the dependency relation.

There is a room to customize the rules by a user to cope with exceptional cases which do not fall into a general pattern, and to cope with conceptual differences between system designs.

For the fourth step, we consider the dependency structure such that:

- Every *bunsetsu* in S except the right most one modifies one of its succeeding *bunsetsu*'s in the sentence
- No two modifications may cross each other (crossing constraint)

Under those constraints, we use CYK algorithm to effectively select the most likely (n-best) combination of dependency relations.

Partial Parse

We propose three types of partial parsing, which focuses on the probabilities of each dependency pairs (p0), the probabilities of whole dependency structure (p1), and some specific dependency relations (p2).

(p0) Output dependency relations of which probability is higher than a particular threshold. The result is the set of dependencies.

(p1) N-best parses are firstly obtained. Then, the dependencies that are included in all of the N-best parses are selected as the result.

(p2) Only the dependencies of the specified relations are produced.

In the *p0* algorithm, we do not use CYK algorithm. If there are more than two modifiees whose dependency probabilities are higher than the threshold, the highest one is chosen (in other words, do not care about "crossing constraint"). Although this method is very simple, it is useful, for example, to help interactive correction procedure of tree-bank construction.

To use the *p2* algorithm, we must evaluate the precision for each relation typef. Some experiments are given in the following section.

4. System Evaluation

For the training and test corpora, we used EDR Japanese bracketed corpus [6], which contains about 208,000 sentences collected from articles of newspapers and magazines.

We splitted the sentences into twenty files. One of these files is held out for evaluation and others are used for training.

Full parse accuracy is evaluated by the precision of correct dependency pairs. Partial parse accuracy is evaluated by the precision and recall of correct dependency pairs.

Precision and recall are defined as follows:

$$Precision = \frac{\text{Number of correct dependencies generated by the system}}{\text{Number of system's output of dependencies}}$$

$$Recall = \frac{\text{Number of correct dependencies generated by the system}}{\text{Total number of dependencies}}$$

Evaluation of Full Parse

The precision of the number of dependency pairs was calculated under the following models.

(a) Base-line

(b) POS model

(c) LEX model

(d) BGH model

The model (a) is used as the base-line, in which all modifiers modify its immediate right *bunsetsu*. "POS model" means that POS tags of head-words are used as the head feature. "LEX model" means that POS tags of head-words and lexical items are used. "BGH model" means that POS tags, lexical items, and word classes are used as the head feature. The level of the layers in the thesaurus is altered from 2 to 6 (leaf layer).

For each of (b), (c), (d) models, we applied two probability models described in section 2 (generation probability and collocation probability) to each of head-collocation probability and distance probability. Then each (a), (b), (c), and (d) models has four different models. But we only shows the result of the following two models, for the each POS, LEX, and BGH model.

- head-collocation (collocation model) + distance (generation model) → model-1
- head-collocation (collocation model) + distance (collocation model) → model-2

Since the other two models give the performance (precision) as low as 70 %, we will not go into more detail of those models. The amount of training data was changed and evaluated in terms of the precision of correct dependency relations.

Figure 1 shows the result of the precision for the inside and outside data under "model-1"[1] . Figure 2 shows the result of the precision for the inside and the outside data under "model-2".

"BGH:6" in the figure means that the sixth-layer of the thesaurus is used for the word class. It slightly outperforms other models that use higher layers in the thesaurus.

When evaluating with outside data, we imposed certain frequency threshold on the statistical data, that is, the collocation data whose occurrence frequency is less than i-times was discarded, where i is a predetermined threshold.

Figure 3 show the resulting change of precisions under the POS, LEX, BGH models. The value of "i" was changed from 2 to 10.

[1] By "inside data", we mean that the training data is used also for the test data, whereas "outside data" means that the held-out data is used for the test data.

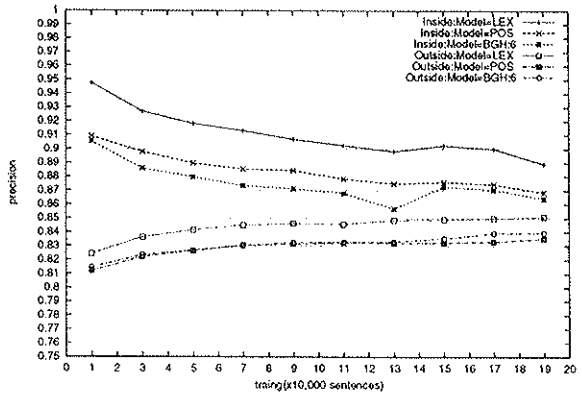


Figure 1: Precision under model-1.

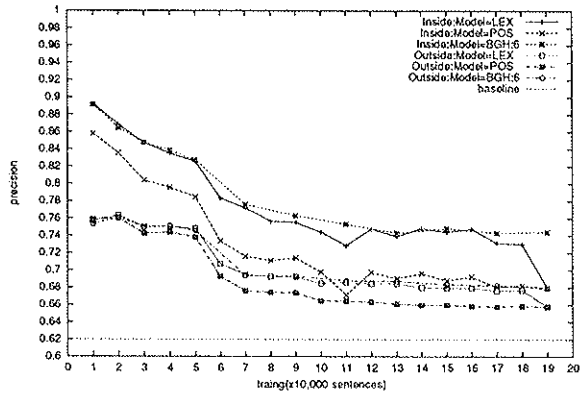


Figure 2: Precision under model-2.

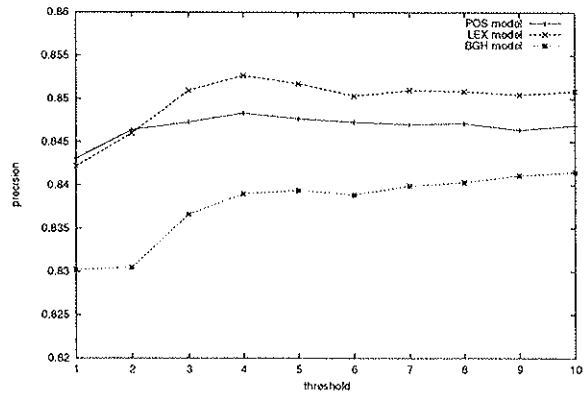


Figure 3: Precision of full parses. Trained from 190,000 sentences. Evaluated by 1,000 sentences

From this experiment, we decided to set the value of i to 4.

The LEX model shows the highest performance in both cases, and the result of model-1 outperforms that of model-2 constantly.

Surprisingly, the BGH model shows poor performance than the POS model. A part of the reason may comes from the fact that we only used one layer of word classes for each experiments. Other reason may be that the hierarchy of "Bunrui Goi Hyou" is not adequate for the syntactic analysis.

The graph shows that the performance of the inside data decreases when the size of training data increases.

The precision of the outside data in "model-1" constantly close up to the precision of the inside data.

We use "model-1" for further analysis.

Contribution of Head-Collocation Probability and Distance Probability

To test which features of head-collocation and distance feature contribute to the accuracy of parsing, the following models are tested.

(e) Distance probability

(f) POS model without the distance probability

(g) LEX without the distance probability

(h) BGH without the distance probability

Each model is trained by 190,000 sentences, and evaluated by 1,000 sentences held out from the training data.

model	precision %	correct/total
(e)	66.07	5087/7610
(f)	79.09	6019/7610
(g)	80.09	6095/7610
(h)	77.58	5819/7610

Table 2: Precision for 1,000 sentences.

The distance probability makes little contribution to the parsing accuracy compared to the head collocation probability. This is because the features used for the distance probability is too simple.

Sentence Level Evaluation

We evaluate sentence level accuracy in this section. A sentence is regarded as correct if the correct structure is found in the n-best parse of the parser, where n is a predetermined value.

Figure 4 shows the rate of correct parses appearing in the n-best parses, where n is changed from 1 to 10. The average number of *bunsetsu*'s in a sentence is 7.

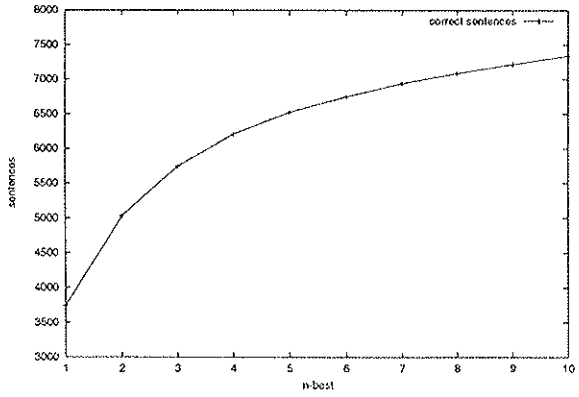


Figure 4: Distribution of correct parses (out of 10,000 sentences). Trained under LEX model by 190,000 sentences.

When n is 5 the precision is 65.21 %, and when n is 10, it becomes 73.40 %.

Evaluation of Each Relation Types

We also check the precision of relation types. The results are shown in Table 5. The first column specifies the type of dependency, which consists of a word, a tag or an inflection form. The second column in Table 5 indicates the ratio of correct dependencies over the total system output.

It is seen that the frequencies of relation type, noun base-form-verb, and ha-particle are high, and influence system's performance, since the precisions for these relations are bad. The particle "ha', "verb/renyou", and "verb/tekei" can construct subordinate clauses in Japanese, and in some cases, it is difficult even for human to consistently determine its modifiee.

A noun + punctuation pattern is also a problematic case, because it can be a part of conjunction phrases. They behave like adverbs (temperal noun and adverbial noun) or form subordinate clauses.

In these cases, it is reasonable to leave these modifiees unspecified. This doesn't conflict the purpose of using the system for practical fields or preprocessor of higher NLP, because it is favorable to output reliable partial parses rather than output unreliable full parses.

Evaluation of Partial Parsing

The results of full parsing accuracy show that model-1 under the LEX model outperforms other models.

For the model, we further examined partial parsing methods explained in section 3, and evaluated its precision and recall.

Table 3 shows the result of *p0* algorithm. The first column in Table 3 indicates the threshold on the probability of each dependency relation. The degree of the reliability (hence the degree of the precision) can be controlled by the value of the threshold on the probabilities.

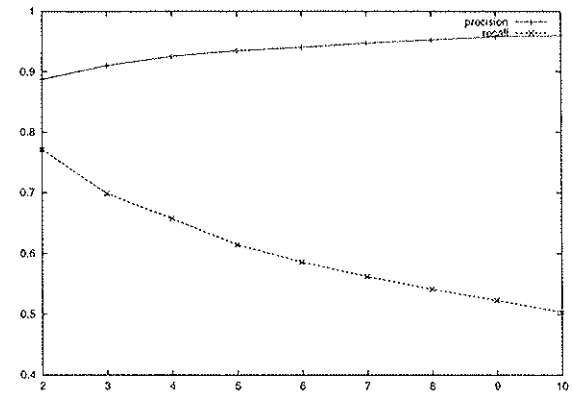


Figure 6: Evaluation of *p1* algorithm. LEX model learned from 190,000 sentences was used.

threshold	precision %(correct/total)	recall % (correct/total)
0.5	86.16 (6356/7377)	83.52 (6356/7610)
0.6	88.23 (6193/7019)	81.38 (6193/7610)
0.7	90.24 (5999/6648)	78.83 (5999/7610)
0.8	92.33 (5705/6179)	74.97 (5705/7610)
0.9	95.19 (5149/5409)	67.66 (5149/7610)

Table 3: Evaluation of *p0* algorithm. LEX model learned from 190,000 sentences was used.

Figure 5 shows the result of *p1* algorithm. The value of "n" in the *p1* algorithm is varied from 2 to 10.

The degree of the precision can be controlled by the value of "n". Figure 6 depicts the results in graphs.

threshold	precision %(correct/total)	recall % (correct/total)
2	88.77 (5149/5409)	77.14 (5149/7610)
3	91.03 (5705/6179)	69.91 (5705/7610)
4	92.53 (5999/6648)	65.80 (5999/7610)
5	93.47 (6193/7019)	61.46 (6193/7610)
6	93.99 (6356/7377)	58.59 (6356/7610)
7	94.71 (6356/7377)	56.23 (6356/7610)
8	95.26 (6356/7377)	54.14 (6356/7610)
9	95.78 (6356/7377)	52.30 (6356/7610)
10	95.99 (6356/7377)	50.38 (6356/7610)

Table 4: Evaluation of *p1* algorithm. LEX model learned from 190,000 sentences was used.

Table 5 shows the result of *p2* algorithm. *p2* algorithm achieves slightly better precision than full parse, but is not as good as *p0* and *p1* algorithms.

When comparing three methods, *p0* algorithm shows highest performance, in terms of the precision and re-

relation name (lexicon/POS/inflection form)	precision (%)	correct	total
/adjective/rentai	95.41	1019	1068
/demonstrative/	93.72	1329	1418
wo/cp/	93.32	7000	7501
no/p/	92.15	11040	11980
ni/cp/	91.51	5769	6304
/adjective/renyou	88.14	959	1088
ga/cp/	87.94	5025	5714
/verb/base	87.32	1344	1539
to/cp/	85.49	1585	1854
mo/p/	83.54	1680	2011
de/cp/	81.83	991	1211
/verb/tekei	79.55	926	1164
/temporal noun/	78.20	1155	1477
da/declarative/tekei	77.96	902	1157
ha/p/	75.32	5790	7687
/noun/	75.29	1182	1570
/verb/renyou	72.43	796	1099

Figure 5: System's outputs were classified according to the right most constituent of relation type, and sorted with their precisions. The symbol cp, and p in the first column mean case-particle and particle. Renyou, rentai tekei and base are the names of inflection forms.

relation types	precision%
without "ha"	86.21 (5904/6808)
without "verb/renyou,tekei"	85.56 (6333/7402)
without "verb/renyou,tekei, ha"	86.57 (5748/6640)

Table 5: Dependency relations without some types of relations. Trained by 190,000 sentences. Evaluated by other 1,000 sentences.

call. When *p0* and *p1* algorithm shows same precision, *p0* algorithm shows higher recall.

p0 and *p1* algorithms can be controlled by a single parameter.

5. Conclusion and Future Works

We showed that the statistical method incorporating lexical level information without any grammar rule is effective in Japanese dependency structure analysis.

Instead of lexical items, we also tested word classes of the thesaurus as head features of phrasal units (BGH model). But that model showed poor performance than the POS model (which uses part-of-speech tags, as head features). This may be because that the hierarchy of applied thesaurus is not appropriate for the syntactic analysis.

85 % of precision (the number of correct dependency relations) is achieved by using LEX model.

In those experiments, the combinations of features are determined manually by human. There is a room to select the combinations of features automatically.

One reason of this comes from the fact that we applied various kinds of distance features, such as the number of noun phrases, the number of case particles, the number of verbs and other kinds of grammatical features between two *bunsetsu*'s, but finally it turned out that simple features, such as the number of *bunsetsu*'s and punctuations between two *bunsetsu*'s shows good performance. This may imply the limitaion of manual selection of combinations of features. Automatical selection of appropriate features is one of our future works.

We also proposed several partial parse methods. Among them, *p0* algorithm is exhibited highest performance in terms of precision and recall, in spite of its simplicity of algorithm.

In *p0* algorithm, the degree of reliability (in other word, degree of precision) is controllable by a single parameter.

Partial parse method can be used for other NLP applications, such as information retrieval or preprocessing of corpus annotation.

References

[1] S.F. Chen and. An empirical study of smoothing techniques for language modeling. *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pp. 310–318, Jun 1996.

[2] M. John Collins and James Brooks. Prepositional phrase attachment through a backed-off model. *Proceedings of the Third Workshop on Very Large Corpora*, pp. 27–38, Jun 1995.

[3] Michael John Collins. A new statistical parser based on bigram lexical dependencies. *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pp. 184–191, Jun 1996.

[4] D.Magerman. Statistical decision-tree model for parsing. *Proceedings of the 33th Annual Meeting of the Association for Computational Linguistics*, pp. 276–283, Jun 1995.

[5] E.Charniak. Statistical parsing with a context-free grammar and word statistics. AAAI, pp. pages 598–603., 1997.

[6] Japan Electronic Dictionary Research Institute, Ltd. *EDR Electronic Dictionary Technical Guide.* 1996.

[7] F.Jelinek and R.L.Mercer. Interpolated estimation of markov source parameters from sparse data. *Proceedings of the Workshop on Pattern Recognition in Practice*, pp. 400–401, March 1987.

[8] H.Li. A probabilistic disambiguation method based on psycholinguistic. *Proceedings of the Forth Workshop on Very Large Corpora*, Aug 1996.

[9] S. M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-35, No. 3*, pp. 400–401, March 1987.

[10] D. Magerman. and M. Marcus. Pearl: A probabilistic chart parser. Proceedings of the 1991 European ACL Conference, 1991. Berlin, Germany.

[11] Y. Matsumoto, O. Imaichi, T. Yamashita, A Kitauchi, and Tomoaki Imamura. *Morphological analysis system ChaSen version 1.0b5 user manual.* Matsumoto lab. Nara Institute of Science and Technology. (in Japanese), 1996.

[12] *Word List by Semantic Principles*, syuei syuppan. (in japanese), 1964,1993.

[13] F. Pereira. and Y. Schabes. Inside-outside reestimation from partially bracketed corpora. *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pp. pages 128–135., 1992.

[14] R Rivest. Learning decision lists. *Machine Learning*, pp. 229–246, 1987.

[15] T.Briscoe and John Carroll. Generalized probabilistic lr parsing of natural language (corpora) with unification-based grammars. *Computational Linguistics*, Vol. Vol.19, No.1, pp. 25–29, Mar 1993.

[16] W.R.Hogenhout and Y.Matsumoto. Training stochastic grammars on semantical categories. *Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing*, 1996.

[17] D. Yarowsky. Decision lists for lexical ambiguity resolution: Application to accent restoration in spanish and french. *Proceedings of the 32th Annual Meeting of the Association for Computational Linguistics*, pp. 88–95, Jun 1994.

[18] H Yasuhara. Kakari-uke dependency analysis with learning function based on reduced type cooccurrence relation. *Journal of Japanese Association for Language Processing*, pp. 87–101, Oct 1996.

A Comparison of Criteria for Maximum Entropy / Minimum Divergence Feature Selection

Adam Berger
Carnegie Mellon University
School of Computer Science
Pittsburgh, PA 15232
aberger@cs.cmu.edu

Harry Printz
IBM
Watson Research Center
Yorktown Heights, NY 10598
printz@watson.ibm.com

Abstract

In this paper we study the *gain*, a naturally-arising statistic from the theory of MEMD modeling [2], as a figure of merit for selecting features for an MEMD language model. We compare the gain with two popular alternatives—empirical activation and mutual information—and argue that the gain is the preferred statistic, on the grounds that it directly measures a feature's contribution to improving upon the base model.

Introduction

Maximum entropy / minimum divergence (MEMD) modeling is a powerful technique for building statistical models of linguistic phenomena. It has been applied to problems as diverse as machine translation [2], parsing [10], word morphology [5] and language modeling [6, 11, 3, 9]. The heart of the method is to choose a collection of informative *features*, each encoding some linguistically significant event, and then to incorporate these features into a family of conditional models.

A fundamental issue in applying this technique is the criterion used to select features. The work described in [3], for instance, incorporates every feature which either appears with above-threshold count in a training corpus, or which exhibits high mutual information. In [11] and [1], the authors select features based on a mutual information statistic. As we argue below, both these methods have drawbacks.

In this paper, we examine a statistic for selecting MEMD model features, called the *gain*. The gain was introduced in [4], and studied in greater detail in [5] and [2]. We present intuition, theory and experimental results for this statistic, as a criterion for selecting features for an MEMD language model. We believe our work marks the first time it has been used in MEMD language modeling, and the first side-by-side comparison with other selection criteria. Though our experimental results concern language models exclusively, we note that the gain can be used to select features for any MEMD model on a discrete space.

The language model we present is based on dependency grammars. It is similar to, but extends upon, the work reported in [3]. Two important differences between that work and ours are that ours is a true minimum-divergence model, and ours incorporates both link and trigger features.

The paper is organized as follows. In Section *Structure of the Model* we give a brief review of MEMD models in general, and of our dependency grammar model in particular. In Section *Linguistic Features* we describe and motivate the types of features we chose to investigate. In Section *Experimental Setup* we describe our experimental procedure. In Section *Selection of Features* we discuss feature selection; it is here that we develop the notion of gain. In Section *Additivity of the Gain* we discuss the additivity of gain, which measures the extent to which features contribute independently to a model. In Section *Tests and Results* we report our test results. Section *Summary* concludes the paper.

Structure of the Model

Use of a Linkage

Let $S = w^0 \ldots w^N$ be the sentence in question, and let $K(S)$ or just K stand for its linkage. A linkage is a planar graph, in which the nodes are the words of S, and the edges connect linguistically related word pairs. A typical sentence S, with its linkage K, appears in Figure 1. The relationship between the linkage of a sentence, and the familiar notion of a parse tree, is described in Section *Experimental Setup* below.

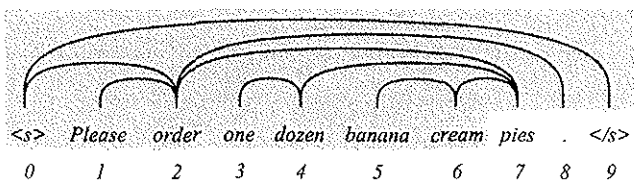


Figure 1: A Sentence S and its Linkage K. The shaded area represents the history h^7, which is the conditioning information available to the model at position 7. h^7 consists of the complete linkage K, and words w^0 through w^6 inclusive.

Our model, written $P(S \mid K)$, is not a language model proper, since it is conditioned upon the linkage. In principle we can recover $P(S)$ as $\sum_K P(S \mid K)P(K)$; in practice we simply take $P(S) \approx P(S \mid K)$. Moreover

since K itself depends upon S, the model cannot be applied incrementally, for instance in a real-time speech recognition system. However, such a model can be used to select from a list of complete sentences.

The value $P(S \mid K)$ computed by our model is formed in the usual way as the product of individual word probabilities; that is

$$P(S \mid K) = \prod_{i=0}^{N} p(w^i \mid w_0^{i-1} K) = \prod_{i=0}^{N} p(w^i \mid h^i). \quad (1)$$

Here we have written $h^i = \langle w_0^{i-1}, K \rangle$ for the *history* at position i; this is the information the model may use when predicting the next word. Here and below the notation w_i^j, with $i \leq j$, stands for the word sequence $w^i \ldots w^j$. Thus for the models in this paper, the history consists of the words $w^0 \ldots w^{i-1}$, plus the complete linkage K.

Fundamentals of MEMD Models

The individual word probabilities $p(w^i \mid h^i)$ appearing in equation (1) above are determined by a *minimum divergence model*. Here we review of the fundamentals of such models; a thorough description appears in reference [2].

As above, let w stand for the word or *future* to be predicted, and let h stand for the history upon which this prediction is based. Suppose that $f(w\ h)$ is a binary-valued indicator function of some linguistic event. For instance, f may take the value 1 when the most recent word of h is the definite article *the* and the word w is any noun; otherwise f is 0. Or f might be 1 when h contains the word *dog* in any location and w is the word *barked*. Any such function $f(w\ h)$ is called a *binary feature function*; clearly we can invent a large number of such functions.

Now suppose C is a large corpus. C can be regarded as a very long sequence of word-history pairs $w^i\ h^i$, where w^i is the word at position i and h^i is the history at that position. We can use C to define the empirical expectation $E_{\tilde{p}}[f]$ of any feature function f; it is given by

$$E_{\tilde{p}}[f] = \sum_i f(w^i\ h^i)/N \quad (2)$$

where i runs over all the positions of the corpus, and N is the number of positions. The sum $A_f = \sum_i f(w^i\ h^i)$ is called the *empirical activation* of the feature f; it is the number of corpus positions where the feature is active (attains the value 1).

Finally, let $q(w \mid h)$ be some selected statistical language model, for instance a trigram model. We call q the *base model*. When q is a trigram, it predicts w based exclusively upon the two most recent words appearing in h. Note however that an arbitrary feature function f can inspect any word of h, or the linkage itself if it comprises part of h. It is the enlarged scope of information available to f that we hope to exploit.

We can now enunciate the principle of minimum divergence modeling. Let $\bar{f} = \langle f_1 \ldots f_M \rangle$ be a vector of binary feature functions, with a known vector of empirical expectations $\langle E_{\tilde{p}}[f_1] \ldots E_{\tilde{p}}[f_M] \rangle$. We seek the model $p(w \mid h)$ of minimal Kullback-Liebler divergence from the base model $q(w \mid h)$, subject to the constraint that

$$\langle E_p[f_1] \ldots E_p[f_M] \rangle = \langle E_{\tilde{p}}[f_1] \ldots E_{\tilde{p}}[f_M] \rangle. \quad (3)$$

That is, the expectation of each f_i, according to the model p, must equal its empirically observed expectation on the corpus C.

By familiar manipulations with Lagrange multipliers, as detailed in [2], the solution to this problem can be shown to be

$$p(w \mid h) = \frac{1}{Z(\bar{\alpha}\ h)}\, q(w \mid h)\, e^{\bar{\alpha} \cdot \bar{f}(w\ h)} \quad (4)$$

where

$$Z(\bar{\alpha}\ h) = \sum_{w \in V} q(w \mid h)\, e^{\bar{\alpha} \cdot \bar{f}(w\ h)}. \quad (5)$$

Here $\bar{f}(w\ h)$ is a vector of 0s and 1s, depending upon the value of each feature function at the point $w\ h$. Likewise $\bar{\alpha}$ is a vector of real-valued exponents, which are adjusted during the training of the model so that equation (3) holds. V is a fixed vocabulary of words, and $Z(\bar{\alpha}\ h)$ is a normalizing value, computed according to equation (5). Finally $q(w \mid h)$ is the base model, which represents our nominal prediction of w from h. When q is the constant function $1/|V|$, the resulting model p is called a *maximum entropy model*; when q is non-constant, p is called a *minimum divergence model*. However the defining equations (4, 5) are the same, regardless of the nomenclature.

Use of a Base Model

In the work reported here, the base model q is decidedly *not* a constant: it is a linearly-interpolated trigram model, trained on a corpus of 44,761,334 words. This approach, while not novel [1], is one of the key departures of our work from [3].

This departure is significant for three reasons. First, it gives us a computationally efficient way to incorporate a large amount of valuable information into our model. To put this another way, we already know that the 14,617,943 trigrams, 3,931,078 bigrams and 56,687 unigrams that together determine q are useful linguistic predictors. But if we should try to incorporate each of these word-grams into a pure maximum entropy framework, via its corresponding feature function, we would be faced with an intractable computational problem.

Second, the use of raw word-gram feature functions, without some discounting of expectations, is believed to be problematic for maximum entropy models, since it can force solutions with unbounded exponents. By incorporating word-gram information via a linearly interpolated trigram model, we are less likely to encounter this problem.

Third, using a trigram base model raises a new and challenging version of the feature selection problem. How can we determine which features, when incorporated into the model, will actually yield an advance upon the trigram model? This is the central problem of this paper, which we proceed to address by using the gain statistic.

Linguistic Features

We now take up the question of how to exploit the information in the history h^i to more accurately estimate the probability of word w^i. We remind the reader that the base model already provides such an estimate, $q(w^i \mid h^i)$. But because in this case q is a trigram model, it discards all of h^i except the two most recent words, $w^{i-2}w^{i-1}$. Our aim is to find informative binary feature functions $f(w^i\ h^i)$ that are clues to especially likely or unlikely values of w^i. We chose to use two different kinds of features: triggers and links.

Trigger Features

As every speaker of English is aware, the appearance of one given word in a sentence is often strong evidence that another particular word will follow. For instance, knowing that *computer* appeared among the words of h^i, one might expect that *nerds* is more likely than normal to appear among the remaining words of the sentence. Some words are in fact good predictors of themselves: seeing *Japanese* once in a sentence raises the likelihood it will appear again later. Word pairs such as these, where the appearance of the first is strongly correlated with the subsequent appearance of the second, are called *trigger pairs* [1, 11]. Note that the trigger property is not necessarily symmetric: we would expect a left parenthesis *(* to trigger a right parenthesis *)*, but not the other way around.

Our model incorporates these relationships through *trigger features*. Let u, v be some trigger pair. A trigger feature f_{uv} is defined as

$$f_{uv}(w\ h) = \begin{cases} 1 & \text{if } w = v \text{ and } h \ni u \text{ with } |uv| \geq d_{\min} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Here $h \ni u$, read "h contains u," means that u appears somewhere in the word sequence of h. The notation $|uv| \geq d_{\min}$ means that the *span* of this pair, defined as the number of words from u to v, including u and v themselves, is not less than a predetermined threshold $d_{\min}$. Throughout this work we have used $d_{\min} = 3$.

Link Features

One shortcoming of trigger features is their profligacy. In a model built with the feature $f_{computer\ nerds}$, an appearance of *computer* will boost the probability of *nerds* at every position at distance $d_{\min}$ or more to its right. This will be so whether or not a position is a linguistically appropriate site for *nerds*. Moreover, if a model contains a large number of trigger features, there will be many triggered words at each position, and their heightened probabilities will tend to wash each other out.

For instance consider the sentence of Figure 2. The plausible trigger feature $f_{stocks\ rose}$ will boost the probability of *rose* at every word from position 4 onward, in particular at position 6. But here the acoustically confusable word *woes* appears, and so increasing the probability of *rose* at this position could yield an error. Thus the boost that $f_{stocks\ rose}$ gives to *rose*, which we desire in position 8, is just as clearly not desired in position 6. Unfortunately the trigger is blind to the distinction between these two sites, and it boosts *rose* in both places.

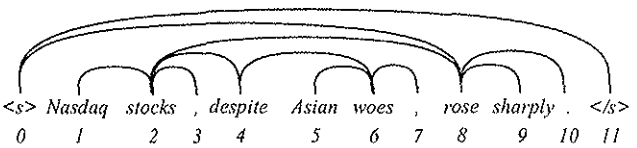


Figure 2: Links versus Triggers. The trigger feature for *stocks* and *rose* boosts the probability of *rose* at each position from 4 to 11, inclusive. The link feature also boosts *rose*, but only at positions 4 and 8. The linkage shown here is the actual one computed by our parser.

These considerations have led us and others to consider features that use the linkage. The aim is to focus the effect of words in the history upon the particular positions that are appropriate for them to influence. Figure 2 shows how the linkage of this sentence connects *stocks*, the headword of the subject noun phrase, with *rose*, the main verb of the sentence; note there is no such link from *stocks* to *woes*. These are precisely the linguistic facts that we wish to exploit, using an appropriate feature function. To do so, we will construct a feature function that (like a trigger) turns on only for a given word pair, and in addition only when the named words are connected by an arc of the linkage.

Because such features depend upon the the linkage of the sentence, we refer to them as *link features*. Such a feature $f_{u\widehat{\ }v}$, for words u and v, is defined as

$$f_{u\widehat{\ }v}(w\ h) = \begin{cases} 1 & \text{if } w = v \text{ and } h \ni u\widehat{\ }v \text{ with } |uv| \geq d_{\min} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

The notation $h \ni u\widehat{\ }v$, read "h contains u, linking v," means that word u appears in the history's word sequence, that an arc of K connects u with the current position, and that word v appears in the current position. In the example given above, the link feature $f_{stocks\widehat{\ }rose}$ attains the value 1 at position 8 only.

Experimental Setup

Here we describe the computation that underlies the work in this paper. Figure 3 is a schematic of the

complete computation, which divides into three phases: (1) prepare the corpus and train a parser and base model, (2) identify and rank features, and (3) select features and train an MEMD model. Our experiments, which we report later, concerned phases (2) and (3) only. We include a discussion of phase (1) for completeness, and to place our experiments in context.

In the first phase we trained a parser and base model, and parsed the corpus text. By parsed we mean that for each sentence S of the corpus text T, we have its linkage $K(S)$ at our disposal. The parser we trained and then used was a modified version of the decision-tree parser described in [7]. Our parser training corpus consisted of 990,145 words of Treebank Release II data, and our base model corpus consisted of 44,761,334 words of Wall Street Journal data, both prepared by the Linguistic Data Consortium.

This parser constructs a conventional parse tree. Since we needed linkages, we used the method of headword propagation to create them from the parser output; we now explain this method. To each parse tree we apply a small collection of headword propagation rules, which operate leaves-to-root. The result is a tree labeled with a headword at each node, where each headword is selected from the headwords of a node's children. (At the leaves, each word is its own headword.) The desired linkage is then obtained by drawing an arc from the headword of each child node to the headword of its parent, excluding self-loops. A conventional parse tree for the sentence of Figure 2 above, labeled with propagated headwords, appears in Figure 4.

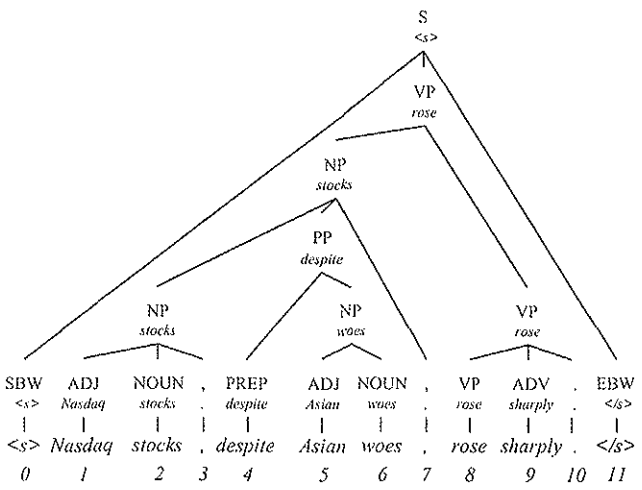


Figure 4: Conventional Parse Tree, with Propagated Headwords. The text explains how this headword-labelled tree can be transformed into the linkage of Figure 2.

For the base model q, we chose to use a linearly interpolated trigram language model, built from the same regularized WSJ corpus as the dependency grammar model itself.

In the second processing phase we identified and ranked features. The details of this phase, and in particular the figure of merit used for ranking, are the subject of Section *Selection of Features*. Here we explain its place in the overall scheme. By inspecting the parsed corpus C, we identify a set F of trigger and link candidate features. These are then *ranked* according to the chosen statistic. In this paper we advocate the use of the gain as the rank statistic. The gain depends upon both the corpus and the base model, and for this reason these are shown as inputs to the box rank features in Figure 3. The output is the same set of candidate features, ranked according to the figure of merit. It happens that the gain computation also yields initial estimates of the MEMD exponents; abbreviated exps in the figure.

In the final phase of processing, we inspected the ranked list of features and selected those to incorporate into the model. We then used the selected features, their initial exponent estimates, the corpus, and the base model to train the MEMD model. Different choices of features yield different models; Section *Tests and Results* below gives details and performance of the various models we built.

Selection of Features

Once the model's prior and feature types have been chosen—choices generally dictated by computational practicality, and the information available in the training corpus—the key open issue is which features to incorporate in the model. In general we cannot and will not want to use every possible feature. For one thing, we usually have too many features to train a model that includes all of them: the processing and memory requirements are just too great. Moreover, rescoring with a model that has a very large number of features is itself time-consuming. Finally, many features may be of little predictive value, for they may seldom activate, or may just repeat information that is already present in the prior.

In this section we describe a method for selecting precisely those features of greatest predictive power, over and above the base model q. The key idea of our method is to seek features that improve upon q's predictions of the training corpus itself. The measure of improvement is a statistic called the *gain*, which we define and motivate below. As we will demonstrate, computing the gain not only yields a principled way of selecting features; it can also be of great help in constructing the MEMD model that contains the selected features.

Our method proceeds in three steps: candidate identification, ranking, and selection. We now describe each step in greater detail.

Candidate Identification

By *candidate identification* we mean a pass over the training corpus (or some other corpus) to collect po-

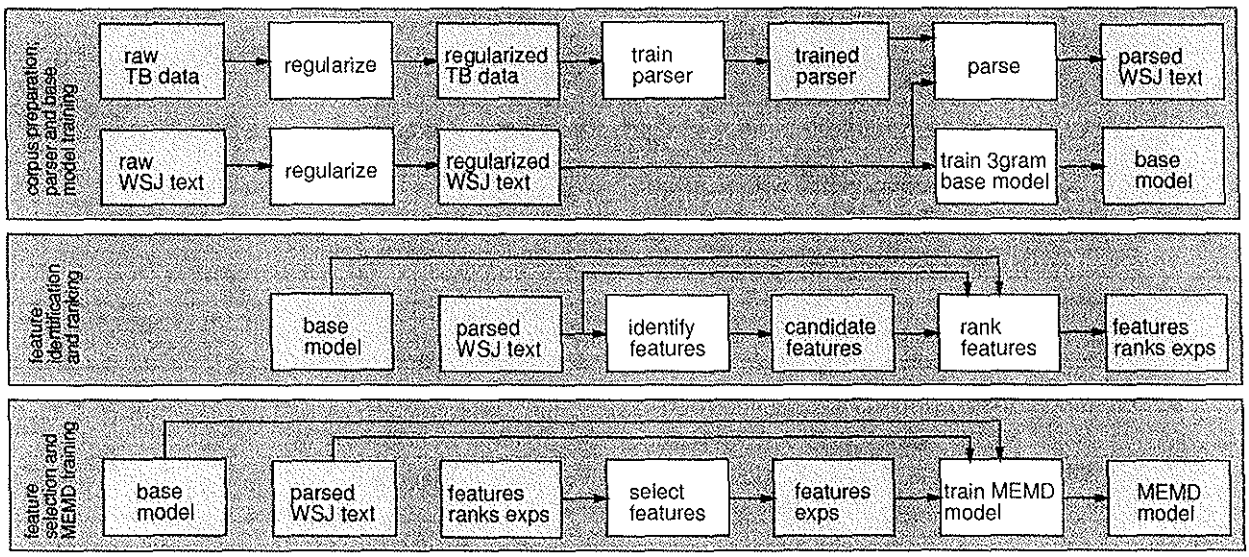


Figure 3: Corpus Preparation, Feature Ranking, and Model Training

tential features for the model. The result of this pass is a *candidate feature set*, denoted F. The candidate features are those that we will rank by gain in the next step.

Note that one or more criteria may be applied to decide which features, out of the many exhibited in the corpus, are placed into F in the first place. In the work reported here, we scanned the parsed corpus to collect potential features, both triggers and links. Since we were building a model using a trigram prior, we had good reason to believe that adjacent words were well-modeled by this prior, and so we ignored links or triggers of span 2. To keep from being swamped with features of no semantic importance, and which arise purely because the words involved are common ones, we likewise ignored triggers where either word was among the 20 most frequent in the corpus. Moreover we did not include any trigger pair with an empirical activation below 6, nor any link pair with a count below 4.

In this way we collected a total of 538,998 candidate link features (which were all those passing the criteria above) and 1,000,000 candidate trigger features (which were those passing the criteria above, and then the top 1,000,000 when sorted by mutual information). We supplied the resulting candidate set F, containing 1,538,998 features, to the next stage of the feature selection process.

Ranking

In this section we will motivate and develop the central feature of this paper, which is the notion of *gain*. First introduced in [2], and further developed in [5], the gain is a statistic computed for a given feature f, with respect to a base model, over some fixed corpus. We will argue that the gain is the appropriate figure of merit for ranking features.

Motivation At the heart of the issue lie the following two questions. First, how much does a feature f aid us in modeling the corpus? Second, to what extent does this feature help us to improve upon the base model? By giving quantitative answers to these questions, we will be led to the gain.

We begin by establishing some notation. Let $P(\mathcal{C})$ stand for the probability of the corpus, according to the base model q; that is $P(\mathcal{C}) = \prod_{i=0}^{N} q(w^i \mid h^i)$. For the model developed here, this should more properly be written $P(\mathcal{T} \mid \mathcal{K})$, where $\mathcal{T}$ represents the collected text of the corpus, and $\mathcal{K}$ consists of the linkage of each sentence of $\mathcal{T}$. However since our meaning is clear, for typographic simplicity we will use the shorter notation.

Now we remind the reader of the connection between MEMD training and maximum-likelihood estimation. Suppose we construct an exponential model, from base model q, that contains one single feature $f(w\ h)$. The form of this model will be

$$p_\alpha(w \mid h) = \frac{1}{Z(\alpha\ h)}\, q(w \mid h)\, e^{\alpha f(w\ h)} \tag{8}$$

where $Z(\alpha\ h)$ is the usual normalizer, and α is a free parameter. For any given value of α, the probability $P_{f\alpha}(\mathcal{C})$ of the entire corpus $\mathcal{C}$, as predicted by this model, is

$$P_{f\alpha}(\mathcal{C}) = \prod_{i=0}^{N-1} p_{\alpha^*}(w^i \mid h^i). \tag{9}$$

The MEMD trained value of α, denoted α^*, is determined as

$$\alpha^* = \operatorname*{argmax}_{\alpha} P_{f\alpha}(\mathcal{C}). \tag{10}$$

That is, the particular α that makes expression (8) the MEMD model is precisely the value α^* given by (10).

This fact is demonstrated in [5], along with a proof that the maximizing α^* is unique.

Thus the probability of the complete corpus, according to the MEMD model p_{α^*}, is just $P_{f\alpha^*}(\mathcal{C})$. When the identity of the feature is clear, we will abbreviate this by $P_{\alpha^*}(\mathcal{C})$.

We proceed to motivate and define the gain. At many positions of the corpus, the models q and p_{α^*} will yield the same value. But in those positions where they disagree, we would hope that p_{α^*} does a better job, in the sense that $p_{\alpha^*}(w^i \mid h^i) > q(w^i \mid h^i)$. That is, we wish that p_{α^*} distributes more probability mass than q on the word that actually appears in corpus position i. The extent to which this occurs is a measure of the predictive value of f, the feature that underlies p_{α^*}.

Of course, we do not want to gauge the value of f by a comparison of models on this or that particular corpus position. But we can judge the overall value of f by comparing $P_{\alpha^*}(\mathcal{C})$, the probability of the entire corpus according to a model that incorporates both q and f, with $P(\mathcal{C})$, the probability of the entire corpus according to q alone.

We can quantify the degree of improvement by writing

$$G_f(\alpha^*) = \frac{1}{N} \log \frac{P_{\alpha^*}(\mathcal{C})}{P(\mathcal{C})} = \frac{1}{N} \log P_{\alpha^*}(\mathcal{C}) - \frac{1}{N} \log P(\mathcal{C}). \tag{11}$$

We refer to $G_f(\alpha^*)$ as the *gain* of feature f. By the rightmost equality above, the gain measures the improvement in cross-entropy afforded by f, or more simply, the information content of f. When it is clear which feature we mean, we will write just $G(\alpha^*)$ for its gain. Likewise we will write G_f when we don't need to display the exponent. The seemingly ancillary quantity α^* is in fact of value, since it is an initial estimate of the feature's associated exponent, and may be used as a starting point in an MEMD training computation that includes this and other features.

Clearly, computing a feature's gain is intimately related to training an MEMD model containing this single feature. But because the model p_{α^*} involves only one feature, substantial computational speedup is possible. A fast algorithm for computing the gain appears in [8].

The notion of gain extends naturally to a set of features M. If $P_M(\mathcal{C})$ is the corpus probability according to a trained MEMD model built with feature set M, then we define $G_M = (1/N) \log (P_M(\mathcal{C})/P(\mathcal{C}))$.

Comparison with Other Criteria A key advantage of the gain as a figure of merit is that it overcomes shortcomings of two competing criteria: the feature's empirical activation, and the mutual information of its history with its future. There are clear rationales for both alternatives, but also clear drawbacks.

Selecting by empirical activation ensures that we are choosing features that could significantly reduce the corpus perplexity, for they are active at many corpus positions, and hence can often alter the base model probability. But there is no guarantee that they change the MEMD model much from the base model, since the selected features might simply express regularities of language that the base model already captures. Of course there is no harm in this, but it does not yield a better model.

Likewise, the mutual information criterion could choose features that coincide with, rather than depart from, the base model. Moreover this criterion can suffer from inaccurate estimates of its constituent probabilities, when the feature is rare.

The gain remedies these problems. It finds features that cause the MEMD model to depart, in a favorable way, from the base model. And if a feature is rare, it is ignored, unless it is very valuable in those cases where it appears.

To test this claim, we computed the gain, empirical activation, and mutual information of the 538,998 candidate link features that we collected earlier from our corpus. We then plotted the gain against empirical activation, and against mutual information; these plots appear in Figure 5. It is clear that gain is only weakly correlated with these competing statistics. In Section *Models Trained* below, we compare the perplexities of models built by selecting features with these three criteria.

Final Selection

Ranking places the features of F in order, from most to least gainful. However, though it is clear that we wish to choose features from F in rank order, say retaining the top 10,000 or 100,000 features, the ranking algorithm does not indicate how many features to select. Thus this last step—choosing where in the ranked list to draw the line—must be decided by hand by the modeler.

Since part of our aim was to compare the relative value of link and trigger features, we elected to build models containing the top T triggers and the top L links, for various values of T and L. We also built a model in which we simply retained the top 10,000 features by rank, without regard to their type.

For illustration, we provide in Table 1 a list of 25 selected trigger and link features, of the 1,538,998 in F, ranked by gain. The table also gives the value of α^* for each feature f; this number is reported as e^{α^*}, since this roughly corresponds the probability boost the future of each feature receives, when the feature is active.

Comparison with Feature Induction

In selection by ranking, we form a set F of candidate features, rank them by gain with respect to the base model q, and retain some number of top-ranked features to build the MEMD model p. We regard this approach as eminently reasonable. But there is this danger of inefficiency: we may incorporate two or more features that capture essentially the same linguistic information.

As a prophyllaxis against this, some authors [2] have advocated *feature induction*. Feature induction is an

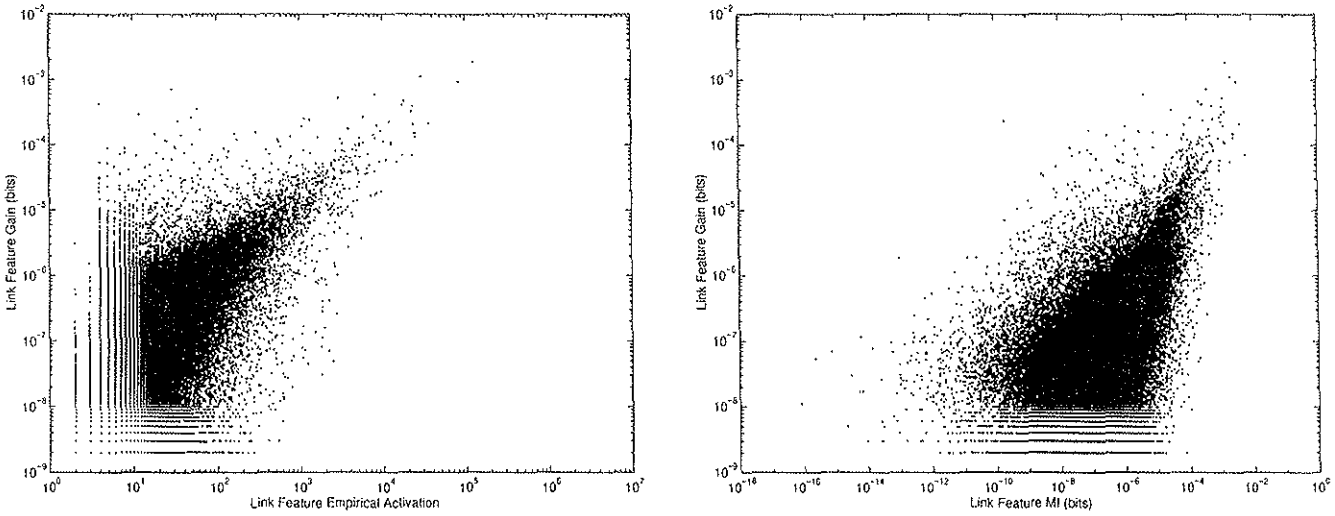


Figure 5: Comparison of 2Link Feature Gain with Empirical Activation and Mutual Information. Left: Scatterplot of feature gain against empirical activation. Right: Scatterplot of feature gain against mutual information.

iterative algorithm for choosing features; it selects one new feature on each iteration. One iteration consists of (1) complete training of an MEMD model using a current set of selected features, initially empty, (2) ranking all remaining candidates against this just-trained model, and (3) removing the single top-ranking feature from the candidate set, and adding it to the set of selected features. Feature induction terminates after incorporating some fixed number of features, or when the gain of the highest-ranked feature, with respect to the current model, drops below some threshold. In this way, if two features f and f' encode essentially the same information, only one is likely to be incorporated into the final model. This is so because after (let us say) feature f is selected, f' will probably have low gain with respect to the model that includes f.

We will show that at least for syntactic features, the feature induction computation is of little benefit. We begin our treatment of this issue by developing the notion of *gain additivity* in the next section. In Section *Empirical Study of Gain Additivity* we present results to support this claim.

Additivity of the Gain

A natural question is whether a selected collection of features $M \subset F$ will be as informative as the sum of its parts. For instance, suppose the words *stocks* and *bonds* are both informative as triggers of the word *rose*. We might reasonably doubt that these are really independent predictors of *rose*, since *stocks* and *bonds* themselves tend to occur together. Put another way, since the gain is a numerical measure of the value of a feature, we are asking if the value of these (or any) two features, when both are used in a model, equals the sum of the individual value of each. In this section we give a theoretical treatment of this issue, introducing the notion of *additivity*.

To begin we consider why it might be plausible that the gains would add. Consider a set $M = \{f_1, f_2\}$ of just two features. By equations (8, 9, 11) above and the associated discussion we have

$$G_{f_1} = \frac{1}{N} \log \frac{P_{f_1 \alpha_1^*}(C)}{P(C)}, \quad G_{f_2} = \frac{1}{N} \log \frac{P_{f_2 \alpha_2^*}(C)}{P(C)}. \tag{12}$$

Let us write $P_{\bar{f}\bar{\alpha}^*}$ for the MEMD model defined by features $\bar{f} = \{f_1, f_2\}$ and exponents $\bar{\alpha}^* = \{\bar{\alpha}_1^*, \bar{\alpha}_2^*\}$, yielding a gain

$$G_{\bar{f}} = \frac{1}{N} \log \frac{P_{\bar{f}\bar{\alpha}^*}(C)}{P(C)}. \tag{13}$$

Note that $\bar{\alpha}_1^*$, $\bar{\alpha}_2^*$ are decidedly *not* necessarily equal to α_1^* and α_2^*, as determined by equation pair (12) above.

Now let us write

$$\frac{P_{\bar{f}\bar{\alpha}^*}(C)}{P(C)} = \frac{P_{f_1 \alpha_1^*}(C)}{P(C)} \frac{P_{\bar{f}\bar{\alpha}^*}(C)}{P_{f_1 \alpha_1^*}(C)} \tag{14}$$

which yields

$$G_{\bar{f}} = G_{f_1} + \frac{1}{N} \log \frac{P_{f_1 f_2 \bar{\alpha}_1^* \bar{\alpha}_2^*}(C)}{P_{f_1 \alpha_1^*}(C)}. \tag{15}$$

Here we have written $P_{\bar{f}\bar{\alpha}^*}(C)$ out in full as $P_{f_1 f_2 \bar{\alpha}_1^* \bar{\alpha}_2^*}(C)$, and simplified using the definition of G_{f_1}. Thus the heart of the matter is how well the second term on the right hand side is approximated by G_{f_2}. We proceed to give a sufficient condition to ensure that the equation $G_{\bar{f}} = G_{f_1} + G_{f_2}$ is exact.

The key idea we will need for our argument is the *potential activation vector* of a feature f with respect to a corpus C, written $\bar{\phi}^C(f)$. In what follows we will relate

word pair		*gain* (mbits)	e^{α^*}	*active* ($\times 10^6$ words)
(	)	0.708	3.6	931
Mr.	*Mr.*	0.678	1.8	3351
Japanese	*Japanese*	0.472	8.1	276
his	*Mr.*	0.431	1.7	2501
Reserve	*Fed*	0.371	18.0	137
Motors	*G.*	0.264	9.8	140
Gorbachev	*Soviet*	0.261	15.6	104
Pennzoil	*Texaco*	0.257	47.7	69
Tokyo	*Japanese*	0.211	7.0	136
Exporting	*OPEC*	0.207	46.3	56
Lambert	*Drexel*	0.198	19.4	73
currency	*dollar*	0.191	3.9	233
prices	*million*	0.160	0.5	484
auto	*Ford*	0.153	10.6	75
Eastman	*Kodak*	0.148	163.2	31

trigger features

word pair		*gain* (mbits)	e^{α^*}	*active* ($\times 10^2$ words)
⟨*s*⟩	⟨/*s*⟩	9.639	10.7	16937
said	.	4.919	10.4	1561
⟨*s*⟩	*said*	2.920	3.8	1969
would	.	1.112	17.5	290
dollars	*cents*	0.934	70.6	230
yesterday	*closed*	0.261	67.1	39
rose	*to*	0.226	4.4	121
rose	*from*	0.197	5.3	84
its	*unit*	0.176	14.2	37
allow	*to*	0.164	38.1	36
A	*spokesman*	0.145	29.3	36
increased	*percent*	0.123	29.6	30
yield	*percent*	0.091	78.8	17
prevent	*from*	0.067	89.3	9
pence	*cents*	0.062	221.8	7

link features

Table 1: Selected Trigger and Link Features. These features are ranked according to gain, reported here in thousandths of a bit (mbits). The third column, e^{α^*}, represents the approximate boost (or deflation) of probability given to the second word of each pair, when the feature is active. The rightmost column lists the feature's empirical activation. Note that trigger features are active far more often than link features. The units used for column *active* differ by 10^4 words.

$\bar{\phi}^C(f)$ and the gain G. Note that both quantities are defined relative to a corpus. For typographic clarity, we elide the superscript from $\bar{\phi}^C$, with the understanding that our claims hold only when $\bar{\phi}$ and G share the same underlying corpus C.

As above, suppose the corpus C contains N positions, numbered 0 through $N-1$, with h^i the history at position i. Then we define $\phi_i(f)$, the ith component of $\bar{\phi}(f)$, by

$$\phi_i(f) = \begin{cases} 1 & \text{if } \exists w \in V \text{ such that } f(w\ h^i) = 1 \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

Thus, $\phi_i(f)$ is non-zero if and only if feature f does or could attain the value 1 at corpus position i. More succinctly, $\phi_i(f) = \max_{w \in V} f(w\ h^i)$; note that ϕ_i does not depend upon the word w^i that actually appears at position i. The potential activation vector $\bar{\phi}(f)$ is then defined componentwise as an N-element vector, the ith component of which is $\phi_i(f)$.

Lemma 1 *Let f_1 and f_2 be binary-valued features. If $\bar{\phi}(f_1) \cdot \bar{\phi}(f_2) = 0$, then*

$$G_{f_1 f_2} = G_{f_1} + G_{f_2}. \tag{17}$$

Proof: The set of corpus positions $I = \{0 \ldots N-1\}$ can be split into three sets

$$\begin{aligned} I_{f_1} &= \{i \mid \phi_i(f_1) = 1\} \\ I_{f_2} &= \{i \mid \phi_i(f_2) = 1\} \\ I_0 &= \{i \mid \phi_i(f_1) = 0 \text{ and } \phi_i(f_2) = 0\}. \end{aligned}$$

Since $\bar{\phi}(f_1) \cdot \bar{\phi}(f_2) = 0$, these three sets are mutually disjoint; by definition they cover I.

Observe that G_{f_1} depends only upon the positions that appear in I_{f_1}; likewise G_{f_2} depends only upon I_{f_2}. Moreover the maximization of $\bar{\alpha}_1$ in $\text{argmax}_{\bar{\alpha}} \log P_{f_1 f_2 \bar{\alpha}_1 \bar{\alpha}_2}(C)$ depends only upon positions appearing in I_{f_1}, since the log sum splits into independent terms just as I splits into I_{f_1}, I_{f_2} and I_0. Indeed, the term that corresponds to I_{f_1} is precisely the non-constant term in the maximization that yields α_1^*; thus $\bar{\alpha}_1^* = \alpha_1^*$. A similar argument holds for $\bar{\alpha}_2^*$. A simple calculation then yields the desired result. ∎

When $\bar{\phi}(f_1) \cdot \bar{\phi}(f_2) = 0$, we write $f_1 \perp\!\!\!\perp f_2$. If $M = \{f_j\}$ is a collection of features, and g is a feature such that $g \perp\!\!\!\perp f_j$ for each $f_j \in M$, we write $g \perp\!\!\!\perp M$. Finally, if for every $f_j \in M$, we have $f_j \perp\!\!\!\perp (M \backslash \{f_j\})$, where the right hand side stands for M with f_j removed, then we say the collection M is *ϕ-orthogonal.*

Theorem 1 *Let M be a ϕ-orthogonal collection of features. Then*

$$G_M = \sum_{f_j \in M} G_{f_j}. \tag{18}$$

Proof: By induction on the size of M. ∎

Of course, we do not mean to suggest that many practical feature collections are ϕ-orthogonal. And it should be clear that since ϕ is defined relative to a particular corpus C, it is entirely possible that a collection M that is ϕ-orthogonal for one corpus may not be for another.

Tests and Results

Our experiments were designed to address three issues. First, given a training corpus over 20 times larger than

the Switchboard transcripts used in [3], we were curious to see how large a model we could feasibly train. Second, we wanted to conduct an experimental study of the gain as a criterion for feature selection, compared to empirical activation and mutual information. Third, we wished to investigate the addivity of the gain. To answer these questions, we trained a number of models, varying the number of features, and the selection criterion, and measuring the resources the training consumed, and the perplexities of the resulting models.

Models Trained

We trained a total of fifteen models; in all cases we trained on the complete corpus. We performed MEMD training using the *improved iterative scaling* algorithm of [5], using the relative change in conditional perplexity, R_t, as a stopping criterion. This quantity is defined as $R_t = (\pi_{t-1} - \pi_t)/\pi_{t-1}$, where π_t is the conditional perplexity (that is, $\pi_t = P_t(T \mid \mathcal{K})^{-1/N}$, where $P_t(T \mid \mathcal{K})$ is the corpus probability according to our model at training iteration t). We required $R_t < .01$ before stopping. We write π_M for the perplexity of the final model M.

Table 2 summarizes our models, the characteristics of the training computation, and the model perplexities. Column t_{seg} is the time to complete one improved iterative scaling iteration on one segment (1/40th) of the complete training corpus on an IBM RS/6000 POWERstation, model 590H. Column *mem* is the total data memory required to process one segment of the corpus. The columns for G_M, $\hat{G}_M$ and δ_M are discussed below.

We draw three conclusions from the perplexity results in this table. First, models constructed only with 2link features have lower perplexity than those constructed only with 2trig features, when we compare models of the same size. This is evident in the comparison between 10k.2trig and 10k.2link, and also between 50k.2trig and 50k.2link. We believe this reflects the higher additivity of 2link gains, a point we discuss further in the next section. However, another possible explanation is that the training converges faster for 2link features than for 2trig features.

Second, the best performance is obtained by including both feature types. This can be seen by comparing among models 10k, 10k.2trig and 10k.2link, and likewise among 50k, 50k.2trig and 50k.2link.

Finally, models selected by gain do better than those selected by mutual information or empirical activation. This is evident from the perplexities of models 10k, 10k.mi and 10k.eact, and likewise 50k.2link, 50k.2link.mi and 50k.2link.eact.

Empirical Study of Gain Additivity

To investigate the additivity of the gain, we first computed the actual gain of each model M, defined as

$$G_M = \frac{1}{N} \log \frac{P_M(C)}{P(C)}. \tag{19}$$

Here $P_M(C)$ is the probability of the corpus, as given by model M. Note that the gain and the perplexity are related by $G_M = \log(\pi_q/\pi_M)$, where π_q is the perplexity of the base model. We then compared G_M with the gain as predicted by summing the individual feature gains, written

$$\hat{G}_M = \sum_{f \in M} G_f. \tag{20}$$

Table 2 reports both these values, and also their *defect* δ_M, which is defined as $\delta_M = \hat{G}_M - G_M$. The defect measures the extent to which the model fails to realize its potential gain. The smaller the defect, the more nearly the gains of the underlying features are additive.

We have argued that the additivity of the gain is related to the ϕ-orthogonality of the feature set, and we believe this is borne out by the figures in the table. Trigger features are clearly highly non-additive. This is to be expected, since in any collection of gainful trigger features, we would expect a large fraction of them to be potentially active at any one position.

By contrast, the link features appear to be very nearly additive. Moreover, the defect δ_M does not grow monotonically with the number of link features in the model. It would seem that the stanza of 300,000 lower-ranked link features are more nearly ϕ-orthogonal than the 200,000 higher-ranked ones. This is reasonable, since on balance the lower-ranked features are probably less often active, hence more likely to act independently of one another.

Summary

In this paper we have investigated the use of gain as a criterion for selecting features for MEMD language models. We showed how the gain of a feature arises naturally from consideration of the feature's predictive value in an MEMD model, compared to the predictions made by the base model. We argued that the gain is the prefered figure of merit for feature selection, since it identifies features that improve upon the base model.

We then applied this statistic to the problem of selecting features for a dependency grammar language model. We showed that when comparing models constructed from the same number of features, using gain as the figure of merit yields models of lower perplexity than either empirical activation of mutual information. Moreover, among models built exclusively from either trigger or link features, but having the same number of features, those built exclusively from links had lower perplexity. However, we achieved the lowest perplexity when we picked the most gainful features without regard to their type.

Finally, we showed that sets of link features have very low gain defect; this is defined as the gap between the set's true and predicted perplexity gains, where the prediction is the sum of individual feature gains. Thus the computationally expensive feature induction procedure appears dispensable, at least for link features.

model name M	t_{seg} (hrs)	mem (MB)	perplexity π_M	actual, predicted gain G_M (bits)	$\hat{G}_M$ (bits)	defect δ_M
baseline (q)			26.764			
10k	.5	20	22.769	.233196	.558733	.325537
10k.mi	.3	19	24.195	.145558	.159312	.013754
10k.eact	1.6	23	25.860	.049545	.143026	.093481
10k.2trig	.8	20	24.483	.128487	.454672	.326185
10k.2link	.4	18	23.835	.167206	.202876	.035670
50k	2.4	37	21.647	.306100	1.140826	.834726
50k.2trig	2.6	38	23.706	.175015	1.007069	.832054
50k.2link	.9	21	23.114	.211472	.256284	.044812
50k.2link.mi	.8	21	23.379	.195054	.213165	.018111
50k.2link.eact	.9	21	23.324	.198452	.208937	.010485
100k	4.2	64	21.212	.335386	1.524190	1.188804
100k.2link	1.2	25	22.805	.230900	.278472	.047572
150k.2link	1.4	28	22.607	.243499	.291138	.047639
200k.2link	1.6	32	22.507	.249903	.299675	.049772
500k.2link	3.8	53	22.232	.267657	.316176	.048519

Table 2: Model Features, Training Characteristics, Perplexities, Gains. Models are named by the following convention. The first part of the name gives the number of features; the letter k denotes a factor of 1,000. Thus *10k* is a model built of the 10,000 highest-ranking features of the candidate set F. The notation *2trig* or *2link* means that we used only trigger or link features respectively. Thus *10k.2link* is built of the 10,000 highest-ranking 2link features of F. Additional letters identify the figure of merit used for the ranking: *eact* stands for empirical activation, *mi* stands for mutual information. If neither appears, the figure of merit was the gain.

We hasten to point out that our results concern perplexity only. It remains to be seen if these conclusions carry over to word error rate, in a suitable speech recognition experiment.

Acknowledgements

We gratefully acknowledge support in part by the National Science Foundation, grant IRI-9314969.

References

[1] D. Beeferman, A. Berger, J. Lafferty, "A Model of Lexical Attraction and Repulsion," *Proc. of the ACL-EACL'97 Joint Conference*, Madrid, Spain.

[2] A. Berger, S. Della Pietra, V. Della Pietra, "A Maximum Entropy Approach to Natural Language Processing," *Computational Linguistics*, 22(1): 39–71, March 1996.

[3] C. Chelba, et. al., "Structure and Performance of a Dependency Language Model," *Proc. of Eurospeech '97*, Rhodes, Greece, September 1997.

[4] S. Della Pietra, V. Della Pietra, J. Gillett, J. Lafferty, H. Printz, L. Ureš, "Inference and Estimation of a Long-Range Trigram Model," *Second International Colloquium on Grammatical Inference*, Alicante, Spain, September 1994.

[5] S. Della Pietra, V. Della Pietra, and J. Lafferty, *Inducing Features of Random Fields*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(4): 380–393, April 1997.

[6] R. Lau, R. Rosenfeld, S. Roukos, "Trigger-Based Language Models: a Maximum Entropy Approach," *Proc. of the International Conference on Acoustics, Speech and Signal Processing*, pp II: 45–48, Minneapolis, MN, April 1993.

[7] D. Magerman, *Natural Language Parsing as Statistical Pattern Recognition*, Ph.D. Thesis, Department of Computer Science, Stanford University, Palo Alto, CA, February 1994.

[8] H. Printz, "Fast Computation of Maximum Entropy / Minimum Divergence Feature Gain," submitted to *International Conference on Speech and Language Processing*, Sydney, Australia, September 1998.

[9] P. S. Rao, S. Dharanipragada, S. Roukos, "MDI Adaptation of Language Models Across Corpora," *Proc. of Eurospeech '97*, pages 1979–1982, Rhodes, Greece, September 1997.

[10] A. Ratnaparkhi, J. Reynar, S. Roukos, "A Maximum Entropy Model for Prepositional Phrase Attachment," *Proc. of the Human Language Technology Workshop*, Plainsboro, NJ, March 1994.

[11] R. Rosenfeld, *Adaptive Statistical Language Modeling: A Maximum Entropy Approach*, Ph.D. Thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, April 1994.

AUTHOR INDEX