2006

COLING • ACL

COLING • ACL 2006

21st International Conference on
Computational Linguistics
and
44th Annual Meeting of the
Association for Computational Linguistics

Proceedings of the Main Conference Poster Sessions

17-18 July 2006
Sydney, Australia

Order copies of this and other ACL proceedings from:

Association for Computational Linguistics (ACL)
209 N. Eighth Street
Stroudsburg, PA 18360
USA
Tel: +1-570-476-8006
Fax: +1-570-476-0860
acl@aclweb.org

Table of Contents

Using Machine Learning Techniques to Build a Comma Checker for Basque

Iñaki Alegria Bertol Arrieta Arantza Diaz de Ilarraza Eli Izagirre Montse Maritxalar
Computer Engineering Faculty. University of the Basque Country.
Manuel de Lardizabal Pasealekua, 1
20018 Donostia, Basque Country, Spain.
{acpalloi,bertol,jipdisaa,jibizole,jipmaanm}@ehu.es

Abstract

In this paper, we describe the research using machine learning techniques to build a comma checker to be integrated in a grammar checker for Basque. After several experiments, and trained with a little corpus of 100,000 words, the system guesses correctly not placing commas with a precision of 96% and a recall of 98%. It also gets a precision of 70% and a recall of 49% in the task of placing commas. Finally, we have shown that these results can be improved using a bigger and a more homogeneous corpus to train, that is, a bigger corpus written by one unique author.

1 Introduction

In the last years, there have been many studies aimed at building a grammar checker for the Basque language (Ansa et al., 2004; Diaz De Ilarraza et al., 2005). These works have been focused, mainly, on building rule sets —taking into account syntactic information extracted from the corpus automatically— that detect some erroneous grammar forms. The research here presented wants to complement the earlier work by focusing on the style and the punctuation of the texts. To be precise, we have experimented using machine learning techniques for the special case of the comma, to evaluate their performance and to analyse the possibility of applying it in other tasks of the grammar checker.

However, developing a punctuation checker encounters one problem in particular: the fact that the punctuation rules are not totally established. In general, there is no problem when using the full stop, the question mark or the exclamation mark. Santos (1998) highlights these marks are reliable punctuation marks, while all the rest are unreliable. Errors related to the reliable ones (putting or not the initial question or exclamation mark depending on the language, for instance) are not so hard to treat. A rule set to correct some of these has already been defined for the Basque language (Ansa et al., 2004). In contrast, the comma is the most polyvalent and, thus, the least defined punctuation mark (Bayraktar et al., 1998; Hill and Murray, 1998). The ambiguity of the comma, in fact, has been shown often (Bayraktar et al., 1998; Beeferman et al., 1998; Van Delden S. and Gomez F., 2002). These works have shown the lack of fixed rules about the comma. There are only some intuitive and generally accepted rules, but they are not used in a standard way. In Basque, this problem gets even more evident, since the standardisation and normalisation of the language began only about twenty-five years ago and it has not finished yet. Morphology is mostly defined, but, on the contrary, as far as syntax is concerned, there is quite work to do. In punctuation and style, some basic rules have been defined and accepted by the Basque Language Academy (Zubimendi, 2004). However, there are not final decisions about the case of the comma.

Nevertheless, since Nunberg's monograph (Nunberg, 1990), the importance of the comma has been undeniable, mainly in these two aspects: i) as a due to the syntax of the sentence (Nunberg, 1990; Bayraktar et al., 1998; Garzia, 1997), and ii) as a basis to improve some natural language processing tools (syntactic analysers, error detection tools…), as well as to develop some new ones (Briscoe and Carroll, 1995; Jones, 1996). The relevance of the comma for the syntax of the sentence may be easily proved with some clarifying examples where the sentence is understood in one or other way, depending on whether a comma is placed or not (Nunberg, 1990):

a. Those students who can, contribute to the United Fund.

b. Those students who can contribute to the United Fund.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 1–8,
Sydney, July 2006.

In the same sense, it is obvious that a well punctuated text, or more concretely, a correct placement of the commas, would help considerably in the automatic syntactic analysis of the sentence, and, therefore, in the development of more and better tools in the NLP field. Say and Akman (1997) summarise the research efforts in this direction.

As an important background for our work, we note where the linguistic information on the comma for the Basque language was formalised. This information was extracted after analysing the theories of some experts in Basque syntax and punctuation (Aldezabal et al., 2003). In fact, although no final decisions have been taken by the Basque Language Academy yet, the theory formalised in the above mentioned work has succeeded in unifying the main points of view about the punctuation in Basque. Obviously, this has been the basis for our work.

2 Learning commas

We have designed two different but combinable ways to get the comma checker:

- based on clause boundaries
- based directly on corpus

Bearing in mind the formalised theory of Aldezabal et al. (2003)[1], we realised that if we got to split the sentence into clauses, it would be quite easy to develop rules for detecting the exact places where commas would have to go. Thus, the best way to build a comma checker would be to get, first, a clause identification tool.

Recent papers in this area report quite good results using machine learning techniques. Carreras and Màrquez (2003) get one of the best performances in this task (84.36% in test). Therefore, we decided to adopt this as a basis in order to get an automatic clause splitting tool for Basque. But as it is known, machine learning techniques cannot be applied if no training corpus is available, and one year ago, when we started this process, Basque texts with this tagged clause splits were not available.

Therefore, we decided to use the second alternative. We had available some corpora of Basque, and we decided to try learning commas from raw text, since a previous tagging was not needed. The problem with the raw text is that its commas are not the result of applying consistent rules.

Related work

Machine learning techniques have been applied in many fields and for many purposes, but we have found only one reference in the literature related to the use of machine learning techniques to assign commas automatically.

Hardt (2001) describes research in using the Brill tagger (Brill 1994; Brill, 1995) to learn to identify incorrect commas in Danish. The system was developed by randomly inserting commas in a text, which were tagged as incorrect, while the original commas were tagged as correct. This system identifies incorrect commas with a precision of 91% and a recall of 77%, but Hardt (2001) does not mention anything about identifying correct commas.

In our proposal, we have tried to carry out both aspects, taking as a basis other works that also use machine learning techniques in similar problems such as clause splitting (Tjong Kim Sang E.F. and Déjean H., 2001) or detection of chunks (Tjong Kim Sang E.F. and Buchholz S., 2000).

3 Experimental setup

Corpora

As we have mentioned before, some corpora in Basque are available. Therefore, our first task was to select the training corpora, taking into account that well punctuated corpora were needed to train the machine correctly. For that purpose, we looked for corpora that satisfied as much as possible our "accepted theory of Basque punctuation". The corpora of the unique newspaper written in Basque, called *Egunkaria* (nowadays *Berria*), were chosen, since they are supposed to use the "accepted theory of Basque punctuation". Nevertheless, after some brief verifications, we realised that the texts of the corpora do not fully match with our theory. This can be understood considering that a lot of people work in a newspaper. That is, every journalist can use his own interpretation of the "accepted theory", even if all of them were instructed to use it in the same way. Therefore, doing this research, we had in mind that the results we would get were not going to be perfect.

To counteract this problem, we also collected more homogeneous corpora from prestigious writers: a translation of a book of philosophy and a novel. Details about these corpora are shown in Table 1.

[1] From now on, we will speak about this as "the accepted theory of Basque punctuation".

	Size of the corpora
Corpora from the newspaper *Egunkaria*	420,000 words
Philosophy texts written by one unique author	25,000 words
Literature texts written by one unique author	25,000 words

Table 1. Dimensions of the used corpora

A short version of the first corpus was used in different experiments in order to tune the system (see section 4). The differences between the results depending on the type of the corpora are shown in section 5.

Evaluation

Results are shown using the standard measures in this area: precision, recall and f-measure[2], which are calculated based on the test corpus. The results are shown in two colums ("0" and "1") that correspond to the result categories used. The results for the column "0" are the ones for the instances that are not followed by a comma. On the contrary, the results for the column "1" are the results for the instances that should be followed by a comma.

Since our final goal is to build a comma checker, the precision in the column "1" is the most important data for us, although the recall for the same column is also relevant. In this kind of tools, the most important thing is to first obtain all the comma proposals right (precision in columns "1"), and then to obtain all the possible commas (recall in columns "1").

Baselines

In the beginning, we calculated two possible baselines based on a big part of the newspaper corpora in order to choose the best one.

The first one was based on the number of commas that appeared in these texts. In other words, we calculated how many commas appeared in the corpora (8% out of all words), and then we put commas randomly in this proportion in the test corpus. The results obtained were not very good (see Table 2, baseline1), especially for the instances "followed by a comma" (column "1").

The second baseline was developed using the list of words appearing before a comma in the training corpora. In the test corpus, a word was tagged as "followed by a comma" if it was one of the words of the mentioned list. The results (see baseline 2, in Table 2) were better, in this case, for the instances followed by a comma (column named "1"). But, on the contrary, baseline 1 provided us with better results for the instances not followed by a comma (column named "0"). That is why we decided to take, as our baseline, the best data offered by each baseline (the ones in bold in table 2).

	0			1		
	Prec.	Rec.	Meas.	Prec.	Rec.	Meas.
baseline 1	**0.927**	**0.924**	**0.926**	0.076	0.079	0.078
baseline 2	0.946	0.556	0.700	**0,096**	**0.596**	**0.165**

Table 2: The baselines

Methods and attributes

We use the WEKA[3] implementation of these classifiers: the Naive Bayes based classifier (NaiveBayes), the support vector machine based classifier (SMO) and the decision-tree (C4.5) based one (j48).

It has to be pointed out that commas were taken away from the original corpora. At the same time, for each token, we stored whether it was followed by a comma or not. That is, for each word (token), it was stored whether a comma was placed next to it or not. Therefore, each token in the corpus is equivalent to an example (an instance). The attributes of each token are based on the token itself and some surrounding ones. The application window describes the number of tokens considered as information for each token.

Our initial application window was [-5, +5]; that means we took into account the previous and following 5 words (with their corresponding attributes) as valid information for each word. However, we tuned the system with different application windows (see section 4).

Nevertheless, the attributes managed for each word can be as complex as we want. We could only use words, but we thought some morphosyntactic information would be beneficial for the machine to learn. Hence, we decided to include as much information as we could extract using the shallow syntactic parser of Basque (Aduriz et al., 2004). This parser uses the tokeniser, the lemmatiser, the chunker and the morphosyntactic disambiguator developed by the IXA[4] research group.

The attributes we chose to use for each token were the following:

- word-form
- lemma
- category
- subcategory
- declension case
- subordinate-clause type

[2] f-measure = 2*precision*recall / (precision+recall)

[3] WEKA is a collection of machine learning algorithms for data mining tasks (http://www.cs.waikato.ac.nz/ml/weka/).

[4] http://ixa.si.ehu.es

- beginning of chunk (verb, nominal, entity, postposition)
- end of chunk (verb, nominal, entity, postposition)
- part of an apposition
- other binary features: multiple word token, full stop, suspension points, colon, semicolon, exclamation mark and question mark

We also included some additional attributes which were automatically calculated:

- number of verb chunks to the beginning and to the end of the sentence
- number of nominal chunks to the beginning and to the end of the sentence
- number of subordinate-clause marks to the beginning and to the end of the sentence
- distance (in tokens) to the beginning and to the end of the sentence

We also did other experiments using binary attributes that correspond to most used collocations (see section 4).

Besides, we used the result attribute "comma" to store whether a comma was placed after each token.

4 Experiments

Dimension of the corpus

In this test, we employed the attributes described in section 3 and an initial window of [-5, +5], which means we took into account the previous 5 tokens and the following 5. We also used the C4.5 algorithm initially, since this algorithm gets very good results in other similar machine learning tasks related to the surface syntax (Alegria et al., 2004).

	0			1		
	Prec.	Rec.	Meas.	Prec.	Rec.	Meas.
100,000 train / 30,000 test	0,955	0,981	0,968	0,635	0,417	0,503
160,000 train / 45,000 test	0,947	0,981	0,964	0,687	0,43	0,529
330,000 train / 90,000 test	**0,96**	**0,982**	**0,971**	**0,701**	**0,504**	**0,587**

Table 3. Results depending on the size of corpora (C4.5 algorithm; [-5,+5] window).

As it can be seen in table 3, the bigger the corpus, the better the results, but logically, the time expended to obtain the results also increases considerably. That is why we chose the smallest corpus for doing the remaining tests (100,000 words to train and 30,000 words to test). We thought that the size of this corpus was enough to get good comparative results. This test, anyway, suggested that the best results we could obtain would be always improvable using more and more corpora.

Selecting the window

Using the corpus and the attributes described before, we did some tests to decide the best application window. As we have already mentioned, in some problems of this type, the information of the surrounding words may contain important data to decide the result of the current word.

In this test, we wanted to decide the best application window for our problem.

	0			1		
	Prec.	Rec.	Meas.	Prec.	Rec.	Meas.
-5+5	0,955	0,981	0,968	0,635	0,417	0,503
-2+5	0,956	0,982	0,969	0,648	0,431	0,518
-3+5	0,957	0,979	0,968	0,627	0,441	0,518
-4+5	0,957	0,98	0,968	0,634	0,446	**0,52**
-5+2	0,956	0,982	0,969	**0,65**	0,424	0,514
-5+3	0,956	0,981	0,969	0,643	0,432	0,517
-5+4	0,955	0,982	0,968	0,64	0,417	0,505
-6+2	0,956	0,982	0,969	0,645	0,421	0,509
-6+3	0,956	0,982	0,969	0,646	0,426	0,514
-8+2	0,956	0,982	0,969	0,645	0,425	0,513
-8+3	0,956	0,979	0,967	0,615	0,431	0,507
-8+8	0,956	0,978	0,967	0,604	0,422	0,497

Table 4. Results depending on the application window (C4.5 algorithm; 100,000 train / 30,000 test)

As it can be seen, the best f-measure for the instances followed by a comma was obtained using the application window [-4,+5]. However, as we have said before, we are more interested in the precision. Thus, the application window [-5 ,+2] gets the best precision, and, besides, its f-measure is almost the same as the best one. This is the reason why we decided to choose the [-5 ,+2] application window.

Selecting the classifier

With the selected attributes, the corpus of 130,000 words and the application window of [-5 , +2], the next step was to select the best classifier for our problem. We tried the WEKA implementation of these classifiers: the Naive Bayes based classifier (NaiveBayes), the support vector machine based classifier (SMO) and the decision tree based one (j48). Table 5 shows the results obtained:

	0			1		
	Prec.	Rec.	Meas.	Prec.	Rec.	Meas.
NB	0,948	0,956	0,952	0,376	0,335	0,355
SMO	0,936	0,994	0,965	**0,672**	0,143	0,236
J48	0,956	0,982	0,969	0,652	0,424	**0,514**

Table 5. Results depending on the classifier (100,000 train / 30,000 test; [-5, +2] window).

As we can see, the f-measure for the instances not followed by a comma (column "0") is almost the same for the three classifiers, but, on the contrary, there is a considerable difference when we refer to the instances followed by a comma (column "1"). The best f-measure gives the C4.5 based classifier (J48) due to the better recall, although the best precision is for the support vector machine based classifier (SMO). Definitively, the Naïve Bayes (NB) based classifier was discarded, but we had to think about the final goal of our research to choose between the other two classifiers. Since our final goal was to build a comma checker, we would have to have chosen the classifier that gave us the best precision, that is, the support vector machine based one. But the recall of the support vector machine based classifier was not as good as expected to be selected. Consequently, we decided to choose the C4.5 based classifier.

Selecting examples

At this moment, the results we get seem to be quite good for the instances not followed by a comma, but not so good for the instances that should follow a comma. This could be explained by the fact that we have no balanced training corpus. In other words, in a normal text, there are a lot of instances not followed by a comma, but there are not so many followed by it. Thus, our training corpus, logically, has very different amounts of instances followed by a comma and not followed by a comma. That is the reason why the system will learn more easily to avoid the unnecessary commas than placing the necessary ones.

Therefore, we resolved to train the system with a corpus where the number of instances followed by a comma and not followed by a comma was the same. For that purpose, we prepared a *perl* program that changed the initial corpus, and saved only x words for each word followed by a comma.

In table 6, we can see the obtained results. One to one means that in that case, the training corpus had one instance not followed by a comma, for each instance followed by a comma. On the other hand, one to two means that the training corpus had two instances not followed by a comma for each word followed by a comma, and so on.

	0			1		
	Prec.	Rec.	Meas.	Prec.	Rec.	Meas.
normal	0,955	0,981	0,968	**0,635**	0,417	0,503
one to one	0,989	0,633	0,772	0,164	**0,912**	0,277
one to two	0,977	0,902	0,938	0,367	0,725	0,487
one to three	0,969	0,934	0,951	0,427	0,621	0,506
one to four	0,966	0,952	0,959	0,484	0,575	0,526
one to five	0,966	0,961	0,963	0,534	0,568	**0,55**
one to six	0,963	0,966	0,964	0,55	0,524	0,537

Table 6. Results depending on the number of words kept for each comma (C4.5 algorithm; 100,000 train / 30,000 test; [-5, +2] window).

As observed in the previous table, the best precision in the case of the instances followed by a comma is the original one: the training corpus where no instances were removed. Note that these results are referred as *normal* in table 6.

The corpus where a unique instance not followed by a comma is kept for each instance followed by a comma gets the best recall results, but the precision decreases notably.

The best f-measure for the instances that should be followed by a comma is obtained by the one to five scheme, but as mentioned before, a comma checker must take care of offering correct comma proposals. In other words, as the precision of the original corpus is quite better (ten points better), we decided to continue our work with the first choice: the corpus where no instances were removed.

Adding new attributes

Keeping the best results obtained in the tests described above (C4.5 with the [-5, +2] window, and not removing any "not comma" instances), we thought that giving importance to the words that appear normally before the comma would increase our results. Therefore, we did the following tests:

1) To search a big corpus in order to extract the most frequent one hundred words that precede a comma, the most frequent one hundred pairs of words (bigrams) that precede a comma, and the most frequent one hundred sets of three words (trigrams) that precede a comma, and use them as attributes in the learning process.

2) To use only three attributes instead of the mentioned three hundred to encode the information about preceding words. The first attribute would indicate whether a word is or not one of

the most frequent one hundred words. The second attribute would mean whether a word is or not the last part of one of the most frequent one hundred pairs of words. And the third attribute would mean whether a word is or not the last part of one of the most frequent one hundred sets of three words.

3) The case (1), but with a little difference: removing the attributes "word" and "lemma" of each instance.

	0			1		
	Prec.	Rec.	Meas.	Prec.	Rec.	Meas.
(0): normal	0,956	0,982	0,969	0,652	0,424	0,514
(1): 300 attributes	**0,96**	**0,983**	**0,972**	**0,696**	**0,486**	**0,572**
(2): 3 attributes +	0,96	0,981	0,97	0,665	0,481	0,558
(3): 300 attributes +, no lemma, no word	0,955	0,987	0,971	**0,71**	0,406	0,517

Table 7. Results depending on the new attributes used (C4.5 algorithm; 100,000 train / 30,000 test; [-5, +2] window; not removed instances).

Table 7 shows that case number 1 (putting the 300 data as attributes) improves the precision of putting commas (column "1") in more than 4 points. Besides, it also improves the recall, and, thus, we improve almost 6 points its f-measure.

The third test gives the best precision, but the recall decreases considerably. Hence, we decided to choose the case number 1, in table 7.

5 Effect of the corpus type

As we have said before (see section 3), depending on the quality of the texts, the results could be different.

In table 8, we can see the results using the different types of corpus described in table 1. Obviously, to give a correct comparison, we have used the same size for all the corpora (20,000 instances to train and 5,000 instances to test, which is the maximum size we have been able to acquire for the three mentioned corpora).

	0			1		
	Prec.	Rec.	Meas.	Prec.	Rec.	Meas.
Newspaper	0.923	0.977	0.949	0.445	0.188	0.264
Philosophy	0.932	0.961	0.946	**0.583**	**0.44**	**0.501**
Literature	0.925	0.976	0.95	0.53	0.259	0.348

Table 8. Results depending on the type of corpora (20,000 train / 5,000 test).

The first line shows the results obtained using the short version of the newspaper. The second line describes the results obtained using the translation of a book of philosophy, written completely by one author. And the third one presents the results obtained using a novel written in Basque.

In any case, the results prove that our hypothesis was correct. Using texts written by a unique author improves the results. The book of philosophy has the best precision and the best recall. It could be because it has very long sentences and because philosophical texts use a stricter syntax comparing with the free style of a literature writer.

As it was impossible for us to collect the necessary amount of unique author corpora, we could not go further in our tests.

6 Conclusions and future work

We have used machine learning techniques for the task of placing commas automatically in texts. As far as we know, it is quite a novel application field. Hardt (2001) described a system which identified incorrect commas with a precision of 91% and a recall of 77% (using 600,000 words to train). These results are comparable with the ones we obtain for the task of guessing correctly when not to place commas (see column "0" in the tables). Using 100,000 words to train, we obtain 96% of precision and 98.3% of recall. The main reason could be that we use more information to learn.

However, we have not obtained as good results as we hoped in the task of placing commas (we get a precision of 69.6% and a recall of 48.6%). Nevertheless, in this particular task, we have improved considerably with the designed tests, and more improvements could be obtained using more corpora and more specific corpora as texts written by a unique author or by using scientific texts.

Moreover, we have detected some possible problems that could have brought these regular results in the mentioned task:

- No fixed rules for commas in the Basque language
- Negative influence when training using corpora from different writers

In this sense, we have carried out a little experiment with some English corpora. Our hypothesis was that a completely settled language like English, where comma rules are more or less fixed, would obtain better results. Taking a comparative English corpus[5] and similar learning attributes[6] to Basque's one, we got, for the instances followed by a comma (column "1" in tables), a better precision (%83.3) than the best

[5] A newspaper corpus, from Reuters

[6] Linguistic information obtained using Freeling (http://garraf.epsevg.upc.es/freeling/)

one obtained for the Basque language. However, the recall was worse than ours: %38.7. We have to take into account that we used less learning attributes with the English corpus and that we did not change the application window chosen for the Basque experiment. Another application window would have been probably more suitable for English. Therefore, we believe that with a few tests we easily would achieve a better recall. These results, anyway, confirm our hypothesis and our diagnosis of the detected problems.

Nevertheless, we think the presented results for the Basque language could be improved. One way would be to use "information gain" techniques in order to carry out the feature selection. On the other hand, we think that more syntactic information, concretely clause splits tags, would be especially beneficial to detect those commas named delimiters by Nunberg (1990).

In fact, our main future research will consist on clause identification. Based on the "accepted theory of the comma", we can assure that a good identification of clauses (together with some significant linguistic information we already have) would enable us to put commas correctly in any text, just implementing some simple rules. Besides, a combination of both methods —learning commas and putting commas after identifying clauses— would probably improve the results even more.

Finally, we contemplate building an ICALL (Intelligent Computer Assisted Language Learning) system to help learners to put commas correctly.

Acknowledgements

We would like to thank all the people who have collaborated in this research: Juan Garzia, Joxe Ramon Etxeberria, Igone Zabala, Juan Carlos Odriozola, Agurtzane Elorduy, Ainara Ondarra, Larraitz Uria and Elisabete Pociello.

This research is supported by the University of the Basque Country (9/UPV00141.226-14601/2002) and the Ministry of Industry of the Basque Government (XUXENG project, OD02UN52).

References

Aduriz I., Aranzabe M., Arriola J., Díaz de Ilarraza A., Gojenola K., Oronoz M., Uria L. 2004. *A Cascaded Syntactic Analyser for Basque Computational Linguistics and Intelligent Text Processing.* 2945 LNCS Series.pg. 124-135. Springer Verlag. Berlin (Germany).

Aldezabal I., Aranzabe M., Arrieta B., Maritxalar M., Oronoz M. 2003. *Toward a punctuation checker for Basque.* Atala Workshop on Punctuation. Paris (France).

Alegria I., Arregi O., Ezeiza N., Fernandez I., Urizar R. 2004. *Design and Development of a Named Entity Recognizer for an Agglutinative Language.* First International Joint Conference on NLP (IJCNLP-04). Workshop on Named Entity Recognition.

Ansa O., Arregi X., Arrieta B., Ezeiza N., Fernandez I., Garmendia A., Gojenola K., Laskurain B., Martínez E., Oronoz M., Otegi A., Sarasola K., Uria L. 2004. *Integrating NLP Tools for Basque in Text Editors.* Workshop on International Proofing Tools and Language Technologies. University of Patras (Greece).

Aranzabe M., Arriola J.M., Díaz de Ilarraza A. 2004. *Towards a Dependency Parser of Basque.* Proceedings of the Coling 2004 Workshop on Recent Advances in Dependency Grammar. Geneva (Switzerland).

Bayraktar M., Say B., Akman V. 1998. *An Analysis of English Punctuation: the special case of comma.* International Journal of Corpus Linguistics 3(1):pp. 33-57. John Benjamins Publishing Company. Amsterdam (The Netherlands).

Beeferman D., Berger A., Lafferty J. 1998. *Cyberpunc: a lightweight punctuation annotation system for speech.* Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, pages 689-692, Seattle (WA).

Brill, E. 1994. *Some Advances in rule-based part of speech tagging.* In Proceedings of the Twelfth National Conference on Artificial Intelligence. Seattle (WA).

Brill, E. 1995. *Transformation-based error-driven learning and natural language processing: a case study in part of speech tagging.* Computational Linguistics 21(4). MIT Press. Cambridge (MA).

Briscoe T., Carroll J. 1995. *Developing and evaluating a probabilistic lr parser of part-of-speech and punctuation labels.* ACL/SIGPARSE 4th international Workshop on Parsing Technologies, Prague / Karlovy Vary (Czech Republic).

Carreras X., Màrquez L. 2003. *Phrase Recognition by Filtering and Ranking with Perceptrons.* Proceedings of the 4th RANLP Conference. Borovets (Bulgaria).

Díaz de Ilarraza A., Gojenola K., Oronoz M. 2005. *Design and Development of a System for the Detection of Agreement Errors in Basque*. CICLing-2005, Sixth International Conference on Intelligent Text Processing and Computational Linguistics. Mexico City (Mexico).

Garzia J. 1997. *Joskera Lantegi.* Herri Arduralaritzaren Euskal Erakundea. Gasteiz, Basque Country (Spain).

Hardt D. 2001. *Comma checking in Danish.* Corpus linguistics. Lancaster (England).

Hill R.L., Murray W.S. 1998. *Commas and Spaces: the Point of Punctuation.* 11th Annual CUNY Conference on Human Sentence Processing. New Brunswick, New Jersey (USA).

Jones B. 1996. *Towards a Syntactic Account of Punctuation*. Proceedings of the 16th International Conference on Computational Linguistics. Copenhagen (Denmark).

Nunberg, G. 1990. *The linguistics of punctuation.* Center for the Study of Language and Information. Leland Stanford Junior University (USA).

Say B., Akman V. 1996. *Information-Based Aspects of Punctuation.* Proceedings ACL/SIGPARSE International Meeting on Punctuation in Computational Linguistics, pages pp. 49-56, Santa Cruz, California (USA).

Tjong Kim Sang E.F. and Buchholz S. 2000. *Introduction to the CoNLL-2000 shared task: chunking.* In proceedings of CoNLL-2000 and LLL-2000. Lisbon (Portugal).

Tjong Kim Sang E.F. and Déjean H. 2001. *Introduction to the CoNLL-2001 shared task: clause identification.* In proceedings of CoNLL-2001. Tolouse (France).

Van Delden S., Gomez F. 2002. *Combining Finite State Automata and a Greedy Learning Algorithm to Determine the Syntactic Roles of Commas.* 14th IEEE International Conference on Tools with Artificial Intelligence. Washington, D.C. (USA)

Zubimendi, J.R. 2004. *Ortotipografia. Estilo liburuaren lehen atala.* Eusko Jaurlaritzaren Argitalpen Zerbitzu Nagusia. Gasteiz, Basque Country (Spain).

A Rote Extractor with Edit Distance-based Generalisation and Multi-corpora Precision Calculation

Enrique Alfonseca[12] **Pablo Castells**[1] **Manabu Okumura**[2] **Maria Ruiz-Casado**[12]

[1]Computer Science Deptartment
Univ. Autónoma de Madrid
Enrique.Alfonseca@uam.es
Pablo.Castells@uam.es
Maria.Ruiz@uam.es

[2]Precision and Intelligence Laboratory
Tokyo Institute of Technology
enrique@lr.pi.titech.ac.jp
oku@pi.titech.ac.jp
maria@lr.pi.titech.ac.jp

Abstract

In this paper, we describe a rote extractor that learns patterns for finding semantic relationships in unrestricted text, with new procedures for pattern generalization and scoring. These include the use of part-of-speech tags to guide the generalization, Named Entity categories inside the patterns, an edit-distance-based pattern generalization algorithm, and a pattern accuracy calculation procedure based on evaluating the patterns on several test corpora. In an evaluation with 14 entities, the system attains a precision higher than 50% for half of the relationships considered.

1 Introduction

Recently, there is an increasing interest in automatically extracting structured information from large corpora and, in particular, from the Web (Craven et al., 1999). Because of the difficulty of collecting annotated data, several procedures have been described that can be trained on unannotated textual corpora (Riloff and Schmelzenbach, 1998; Soderland, 1999; Mann and Yarowsky, 2005). An interesting approach is that of rote extractors (Brin, 1998; Agichtein and Gravano, 2000; Ravichandran and Hovy, 2002), which look for textual contexts that happen to convey a certain relationship between two concepts.

In this paper, we describe some contributions to the training of Rote extractors, including a procedure for generalizing the patterns, and a more complex way of calculating their accuracy. We first introduce the general structure of a rote extractor and its limitations. Next, we describe the proposed modifications (Sections 2, 3 and 4) and the evaluation performed (Section 5).

1.1 Rote extractors

According to the traditional definition of rote extractors (Mann and Yarowsky, 2005), they estimate the probability of a relationship $r(p, q)$ given the surrounding context $A_1pA_2qA_3$. This is calculated, with a training corpus T, as the number of times that two related elements $r(x, y)$ from T appear with that same context $A_1xA_2yA_3$, divided by the total number of times that x appears in that context together with any other word:

$$P(r(p,q)|A_1pA_2qA_3) = \frac{\sum_{x,y \epsilon r} c(A_1xA_2yA_3)}{\sum_{x,z} c(A_1xA_2zA_3)} \quad (1)$$

x is called the *hook*, and y the *target*. In order to train a Rote extractor from the web, this procedure is usually followed (Ravichandran and Hovy, 2002):

1. Select a pair of related elements to be used as seed. For instance, *(Dickens,1812)* for the relationship *birth year*.
2. Submit the query *Dickens AND 1812* to a search engine, and download a number of documents to build the training corpus.
3. Keep all the sentences containing both elements.
4. Extract the set of contexts between them and identify repeated patterns. This may just be the m characters to the left or to the right, (Brin, 1998), the longest common substring of several contexts (Agichtein and Gravano, 2000), or all substrings obtained with a suffix tree constructor (Ravichandran and Hovy, 2002).
5. Download a separate corpus, called *hook corpus*, containing just the hook (in the example, *Dickens*).
6. Apply the previous patterns to the hook corpus, calculate the precision of each pattern

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 9–16,
Sydney, July 2006.

in the following way: the number of times it identifies a target related to the hook divided by the total number of times the pattern appears.

7. Repeat the procedure for other examples of the same relationship.

To illustrate this process, let us suppose that we want to learn patterns to identify birth years. We may start with the pair (*Dickens*, 1812). From the downloaded corpus, we extract sentences such as

Dickens was born in 1812
Dickens (1812 - 1870) was an English writer
Dickens (1812 - 1870) wrote Oliver Twist

The system identifies that the contexts of the last two sentences are very similar and chooses their longest common substring to produce the following patterns:

```
<hook> was born in <target>
<hook> ( <target> - 1870 )
```

In order to measure the precision of the extracted patterns, a new corpus is downloaded using the hook *Dickens* as the only query word, and the system looks for appearances of the patterns in the corpus. For every occurrence in which the hook of the relationship is Dickens, if the target is 1812 it will be deemed correct, and otherwise it will be deemed incorrect (e.g. in *Dickens was born in Portsmouth*).

1.2 Limitations and new proposal

We have identified the following limitations in this algorithm: firstly, to our knowledge, no Rote extractor allows for the insertion of wildcards (e.g. *) in the extracted patterns. Ravichandran and Hovy (2002) have noted that this might be dangerous if the wildcard matches unrestrictedly incorrect sentences. However, we believe that the precision estimation that is performed at the last step of the algorithm, using the *hook corpus*, may be used to rule out the dangerous wildcards while keeping the useful ones.

Secondly, we believe that the procedure for calculating the precision of the patterns may be somewhat unreliable in a few cases. For instance, Ravichandran and Hovy (2002) report the following patterns for the relationships Inventor, Discoverer and Location:

Relation	Prec.	Pattern
Inventor	1.0	<target> 's <hook> and
Inventor	1.0	that <target> 's <hook>
Discoverer	0.91	of <target> 's <hook>
Location	1.0	<target> 's <hook>

In this case, it can be seen that the same pattern (the genitive construction) may be used to indicate several different relationships, apart from the most common use indicating possession. However, they all receive very high precision values. The reason is that the patterns are only evaluated for the same hook for which they were extracted. Let us suppose that we obtain the pattern for Location using the pairs *(New York, Chrysler Building)*. The genitive construction can be extracted from the context *New York's Chrysler Building*. Afterward, when evaluating it, only sentences containing *<target>'s Chrysler Building* are taken into account, which makes it unlikely that the pattern is expressing a relationship other than Location, so the pattern will receive a high precision value.

For our purposes, however, we need to collect patterns for several relations such as *writer-book*, *painter-picture*, *director-film*, *actor-film*, and we want to make sure that the obtained patterns are only applicable to the desired relationship. Patterns like *<target> 's <hook>* are very likely to be applicable to all of these relationships at the same time, so we would like to be able to discard them automatically.

Hence, we propose the following improvements for a Rote extractor:

- A new pattern generalization procedure that allows the inclusion of wildcards in the patterns.
- The combination with Named Entity recognition, so people, locations, organizations and dates are replaced by their entity type in the patterns, in order to increase their degree of generality. This is in line with Mann and Yarowsky (2003)'s modification, consisting in replacing all numbers in the patterns with the symbol ####.
- A new precision calculation procedure, in a way that the patterns obtained for a given relationship are evaluated on the corpus for different relationships, in order to improve the detection of over-general patterns.

2 Proposed pattern generalization procedure

To begin with, for every appearance of a pair of concepts, we extract a context around them. Next, those contexts are generalized to obtain the parts that are shared by several of them. The procedure is detailed in the following subsections.

Birth year:

```
BOS/BOS <hook> (/( <target> -/- number/entity )/) EOS/EOS
BOS/BOS <hook> (/( <target> -/- number/entity )/) British/JJ writer/NN
BOS/BOS <hook> was/VBD born/VBN on/IN the/DT first/JJ of/IN time_expr/entity ,/, <target> ,/, at/IN location/entity ,/, of/IN
BOS/BOS <hook> (/( <target> -/- )/) a/DT web/NN guide/NN
```

Birth place:

```
BOS/BOS <hook> was/VBD born/VBN in/IN <target> ,/, in/IN central/JJ location/entity ,/,
BOS/BOS <hook> was/VBD born/VBN in/IN <target> date/entity and/CC moved/VBD to/TO location/entity
BOS/BOS Artist/NN :/, <hook> -/- <target> ,/, location/entity (/( number/entity -/-
BOS/BOS <hook> ,/, born/VBN in/IN <target> on/IN date/entity ,/, worked/VBN as/IN
```

Author-book:

```
BOS/BOS <hook> author/NN of/IN <target> EOS/EOS
BOS/BOS Odysseus/NNP :/, Based/VBN on/IN <target> ,/, <hook> 's/POS epic/NN from/IN Greek/JJ mythology/NN
BOS/BOS Background/NN on/IN <target> by/IN <hook> EOS/EOS
did/VBD the/DT circumstances/NNS in/IN which/WDT <hook> wrote/VBD "/'' <target> "/'' in/IN number/entity ,/, and/CC
```

Capital-country:

```
BOS/BOS <hook> is/VBZ the/DT capital/NN of/IN <target> location/entity ,/, location/entity correct/JJ time/NN
BOS/BOS The/DT harbor/NN in/IN <hook> ,/, the/DT capital/NN of/IN <target> ,/, is/VBZ number/entity of/IN location/entity
BOS/BOS <hook> ,/, <target> EOS/EOS
BOS/BOS <hook> ,/, <target> -/- organization/entity EOS/EOS
```

Figure 1: Example patterns extracted from the training corpus for each several kinds of relationships.

2.1 Context extraction procedure

After selecting the sentences for each pair of related words in the training set, these are processed with a part-of-speech tagger and a module for Named Entity Recognition and Classification (NERC) that annotates people, organizations, locations, dates, relative temporal expressions and numbers. Afterward, a context around the two words in the pair is extracted, including (a) at most five words to the left of the first word; (b) all the words in between the pair words; (c) at most five words to the right of the second word. The context never jumps over sentence boundaries, which are marked with the symbols BOS (*Beginning of sentence*) and EOS (*End of sentence*). The two related concepts are marked as <hook> and <target>. Figure 1 shows several example contexts extracted for the relationships *birth year*, *birth place*, *writer-book* and *capital-country*.

Furthermore, for each of the entities in the relationship, the system also stores in a separate file the way in which they are annotated in the training corpus: the sequences of part-of-speech tags of every appearance, and the entity type (if marked as such). So, for instance, typical PoS sequences for names of authors are "NNP"[1] (surname) and "NNP NNP" (first name and surname). A typical entity kind for an author is person.

2.2 Generalization pseudocode

In order to identify the portions in common between the patterns, and to generalize them, we apply the following pseudocode (Ruiz-Casado et al., in press):

1. Store all the patterns in a set $\mathcal{P}$.
2. Initialize a set $\mathcal{R}$ as an empty set.
3. While $\mathcal{P}$ is not empty,
   (a) For each possible pair of patterns, calculate the distance between them (described in Section 2.3).
   (b) Take the two patterns with the smallest distance, p_i and p_j.
   (c) Remove them from $\mathcal{P}$, and add them to $\mathcal{R}$.
   (d) Obtain the generalization of both, p_g (Section 2.4).
   (e) If p_g does not have a wildcard adjacent to the hook or the target, add it to $\mathcal{P}$.
4. Return $\mathcal{R}$

At the end, $\mathcal{R}$ contains all the initial patterns and those obtained while generalizing the previous ones. The motivation for step (e) is that, if a pattern contains a wildcard adjacent to either the hook or the target, it will be impossible to know where it starts or ends. For instance, when applying the pattern <hook> wrote * <target> to a text, the wildcard prevents the system from guessing where the title of the book starts.

2.3 Edit distance calculation

So as to calculate the similarity between two patterns, a slightly modified version of the dynamic programming algorithm for *edit-distance* calculation (Wagner and Fischer, 1974) is used. The distance between two patterns A and B is defined as the minimum number of changes (insertion, addition or replacement) that have to be done to the first one in order to obtain the second one.

The calculation is carried on by filling a matrix $\mathcal{M}$, as shown in Figure 2 (left). At the same

[1] All the PoS examples in this paper are done with Penn Treebank labels (Marcus et al., 1993).

A: `wrote the well known novel`
B: `wrote the classic novel`

$\mathcal{M}$	**0**	**1**	**2**	**3**	**4**
0	0	1	2	3	4
1	1	0	1	2	3
2	2	1	0	1	2
3	3	2	1	1	2
4	4	3	2	2	2
5	5	4	3	3	2

$\mathcal{D}$	**0**	**1**	**2**	**3**	**4**
0		I	I	I	I
1	R	**E**	I	I	I
2	R	R	**E**	I	I
3	R	R	R	**U**	I
4	R	R	R	**R**	U
5	R	R	R	R	**E**

Figure 2: Example of the edit distance algorithm. A and B are two word patterns; $\mathcal{M}$ is the matrix in which the edit distance is calculated, and $\mathcal{D}$ is the matrix indicating the choice that produced the minimal distance for each cell in $\mathcal{M}$.

time that we calculate the edit distance matrix, it is possible to fill in another matrix $\mathcal{D}$, in which we record which of the choices was selected at each step: insertion, deletion, replacement or no edition. This will be used later to obtain the generalized pattern. We have used the following four characters:

- `I` means that it is necessary to insert a token in the first pattern to obtain the second one.
- `R` means that it is necessary to remove a token.
- `E` means that the corresponding tokens are equal, so no edition is required.
- `U` means that the corresponding tokens are unequal, so a replacement has to be done.

Figure 2 shows an example for two patterns, A and B, containing respectively 5 and 4 tokens. $\mathcal{M}(5,4)$ has the value 2, indicating the distance between the two complete patterns. For instance, the two editions would be replacing `well` by `classic` and removing `known`.

2.4 Obtaining the generalized pattern

After calculating the edit distance between two patterns A and B, we can use matrix $\mathcal{D}$ to obtain a generalized pattern, which should maintain the common tokens shared by them. The procedure used is the following:

- Every time there is an insertion or a deletion, the generalized pattern will contain a wildcard, indicating that there may be anything in between.
- Every time there is replacement, the generalized pattern will contain a disjunction of both tokens.
- Finally, in the positions where there is no edit operation, the token that is shared between the two patterns is left unchanged.

The patterns in the example will produced the generalized pattern

```
Wrote the well known novel
Wrote the classic    novel
_____________________
Wrote the well|classic * novel
```

The generalization of these two patterns produces one that can match a wide variety of sentences, so we should always take care in order not to over-generalize.

2.5 Considering part-of-speech tags and Named Entities

If we consider the result in the previous example, we can see that the disjunction has been made between an adverb and an adjective, while the other adjective has been deleted. A more natural result, with the same number of editing operations as the previous one, would have been to delete the adverb to obtain the following generalization:

```
Wrote the well known novel
Wrote the classic    novel
_____________________
Wrote the * known|classic novel
```

This is done taking into account part-of-speech tags in the generalization process. In this way, the edit distance has been modified so that a replacement operation can only be done between words of the same part-of-speech.[2] Furthermore, replacements are given an edit distance of 0. This favors the choice of replacements with respect to deletions and insertions. To illustrate this point, the distance between `known|classic/JJ` and `old/JJ`

[2]Note that, although our tagger produces the very detailed PennTreebank labels, we consider that all nouns (NN, NNS, NNP and NNPS) belong to the same part-of-speech class, and the same for adjectives, verbs and adverbs.

Hook	Birth	Death	Birth place	Author of	Director of	Capital of
Charles Dickens	1812	1870	Portsmouth	{Oliver Twist, The Pickwick Papers, Nicholas Nickleby, David Copperfield...}	**None**	**None**
Woody Allen	1935	**None**	Brooklin	**None**	{Bananas, Annie_Hall, Manhattan, ... }	**None**
Luanda	**None**	**None**	**None**	**None**	**None**	Angola

Table 1: Example rows in the input table for the system.

will be set to 0, because both tokens are adjectives. In other words, the d function is redefined as:

$$d(A[i], B[j]) = \begin{cases} 0 & \text{if } PoS(A[i]) = PoS(B[j]) \\ 1 & \text{otherwise} \end{cases} \quad (2)$$

Note that all the entities identified by the NERC module will appear with a PoS tag of *entity*, so it is possible to have a disjunction such as `location|organization/entity` in a generalized pattern (See Figure 1).

3 Proposed pattern scoring procedure

As indicated above, if we measure the precision of the patterns using a *hook corpus*-based approach, the score may be inadvertently increased because they are only evaluated using the same terms with which they were extracted. The approach proposed herein is to take advantage of the fact that we are obtaining patterns for several relationships. Thus, the hook corpora for some of the patterns can be used also to identify errors done by other patterns.

The input of the system now is not just a list of related pairs, but a table including several relationships for the same entities. We may consider it as mini-biographies as in Mann and Yarowsky (2005)'s system. Table 1 shows a few rows in the input table for the system. The cells for which no data is provided have a default value of **None**, which means that anything extracted for that cell will be considered as incorrect.

Although this table can be written by hand, in our experiments we have chosen to build it automatically from the lists of related pairs. The system receives the seed pairs for the relationships, and mixes the information from all of them in a single table. In this way, if *Dickens* appears in the seed list for the *birth year*, *death year*, *birth place* and *writer-book* relationships, four of the cells in its row will be filled in with values, and all the rest will be set to **None**. This is probably a very strict evaluation, because, for all the cells for which there was no value in any of the lists, any result obtained will be judged as incorrect. However, the advantage is that we can study the behavior of the system working with incomplete data.

The new procedure for calculating the patterns' precisions is as follows:

1. For every relationship, and for every *hook*, collect a *hook corpus* from the Internet.
2. Apply the patterns to all the hook corpora collected. Whenever a pattern extracts a relationship from a sentence,
   - If the table does not contain a row for the hook, ignore the result.
   - If the extracted target appears in the corresponding cell in the table, consider it correct.
   - If that cell contained different values, or **None**, consider it incorrect.

For instance, the pattern *<target> 's <hook>* extracted for *director-film* may find, in the Dickens corpus, book titles. Because these titles do not appear in the table as films directed by Dickens, the pattern will be considered to have a low accuracy.

In this step, every pattern that did not apply at least three times in the test corpora was discarded.

4 Pattern application

Finally, given a set of patterns for a particular relation, the procedure for obtaining new pairs is straightforward:

1. For any of the patterns,
2. For each sentence in the test corpus,
   (a) Look for the left-hand-side context in the sentence.
   (b) Look for the middle context.
   (c) Look for the right-hand-side context.
   (d) Take the words in between, and check that either the sequence of part-of-speech tags or the entity type had been

Applied	Prec.	Pattern
3	1.0	BOS/BOS On/IN time_expr/entity TARGET HOOK was/VBD baptized|born/VBN EOS/EOS
15	1.0	"/'' HOOK (/(TARGET -/-
4	1.0	,/, TARGET ,/, */* Eugne|philosopher|playwright|poet/NNP HOOK earned|was/VBD */* at|in/IN
23	1.0	-|--/- HOOK (/(TARGET -/-
12	1.0	AND|and|or/CC HOOK (/(TARGET -/-
48	1.0	By|about|after|by|for|in|of|with/IN HOOK TARGET -/-
4	1.0	On|of|on/IN TARGET ,/, HOOK emigrated|faced|graduated|grew|has|perjured|settled|was/VBD
12	1.0	BOS/BOS HOOK TARGET -|--/-
49	1.0	ABOUT|ALFRED|Amy|Audre|Authors|BY| (...) |teacher|writer/NNPS HOOK (/(TARGET -|--/-
7	1.0	BOS/BOS HOOK (/(born/VBN TARGET)/)
3	1.0	BOS/BOS HOOK ,/, */* ,/, TARGET ,/,
13	1.0	BOS/BOS HOOK ,|:/, TARGET -/-
132	0.98	BOS/BOS HOOK (/(TARGET -|--/-
18	0.94	By|Of|about|as|between|by|for|from|of|on|with/IN HOOK (/(TARGET -/-
33	0.91	BOS/BOS HOOK ,|:/, */* (/(TARGET -|--/-
10	0.9	BOS/BOS HOOK ,|:/, */* ,|:/, TARGET -/-
3	0.67	,|:|;/, TARGET ,|:/, */* Birth|son/NN of/IN */* General|playwright/NNP HOOK ,|;/,
210	0.63	,|:|;/, HOOK (/(TARGET -|--/-
7	0.29	(/(HOOK TARGET)/)

Table 3: Patterns for the relationship *birth year*.

.

Relation	Seeds	Extr.	Gener.	Filt.
Actor-film	133	480	519	10
Writer-book	836	3858	4847	171
Birth-year	492	2520	3220	19
Birth-place	68	681	762	5
Country-capital	36	932	1075	161
Country-president	56	1260	1463	119
Death-year	492	2540	3219	16
Director-film	1530	3126	3668	121
Painter-picture	44	487	542	69
Player-team	110	2903	3514	195

Table 2: Number of seed pairs for each relation, and number of unique patterns after the extraction and the generalization step, and after calculating their accuracy and filtering those that did not apply 3 times on the test corpus.

Relation	Precision	Incl. prec.	Applied
Actor-film	0%	76.84%	95
Writer-book	6.25%	28.13%	32
Birth-year	79.67%	79.67%	477
Birth-place	14.56%	14.56%	103
Country-capital	72.43%	72.43%	599
Country-president	81.40%	81.40%	43
Death-year	96.71%	96.71%	152
Director-film	43.40%	84.91%	53
Painter-picture	-	-	0
Player-team	52.50%	52.50%	120

Table 4: Precision, *inclusion precision* and number of times that a pattern extracted information, when applied to a test corpus.

seen in the training corpus for that role (hook or target). If so, output the relationship.

5 Evaluation and results

The procedure has been tested with 10 different relationships. For each pair in each seed list, a corpus with 500 documents has been collected using Google, from which the patterns are extracted. Table 2 shows the number of patterns obtained. It is interesting to see that for some relations, such as birth-year or birth-place, more than one thousand patterns have been reduced to a few. Table 3 shows the patterns obtained for the relationship *birth-year*. It can also be seen that some of the patterns with good precision contain the wildcard *, which helped extract the correct birth year in roughly 50 occasions. Specially of interest is the last pattern,

(/(HOOK TARGET)/)

which resulted in an accuracy of 0.29 with the procedure here indicated, but which would have obtained an accuracy of 0.54 using the traditional *hook corpus* approach. This is because in other test corpora (e.g. in the one containing soccer players and clubs) it is more frequent to find the name of a person followed by a number that is not his/her birth year, while that did not happen so often in the *birth year* test corpus.

For evaluating the patterns, a new test corpus has been collected for fourteen entities not present in the training corpora, again using Google. The chosen entities are *Robert de Niro* and *Natalie Wood* (actors), *Isaac Asimov* and *Alfred Bester* (writers), *Montevideo* and *Yaounde* (capitals), *Gloria Macapagal Arroyo* and *Hosni Mubarak* (country presidents), *Bernardo Bertolucci* and *Federico Fellini* (directors), *Peter Paul Rubens* and *Paul Gauguin* (painters), and *Jens Lehmann* and *Thierry Henry* (soccer players). Table 4 shows the results obtained for each relationship.

We have observed that, for those relationships in which the target does not belong to a Named

Entity type, it is common for the patterns to extract additional words together with the right target. For example, rather than extracting *The Last Emperor*, the patterns may extract this title together with its rating or its length, the title between quotes, or phrases such as *The classic The Last Emperor*. In the second column in the table, we measured the percentage of times that a correct answer appears inside the extracted target, so these examples would be considered correct. We call this metric *inclusion precision*.

5.1 Comparison to related approaches

Although the above results are not comparable to Mann and Yarowsky (2005), as the corpora used are different, in most cases the precision is equal or higher to that reported there. On the other hand, we have rerun Ravichandran and Hovy (2002)'s algorithm on our corpus. In order to assure a fair comparison, their algorithm has been slightly modified so it also takes into account the part-of-speech sequences and entity types while extracting the hooks and the targets during the rule application. So, for instance, the relationship *birth date* is only extracted between a hook tagged as a person and a target tagged as either a date or a number. The results are shown in Table 5. As can be seen, our procedure seems to perform better for all of the relations except *birth place*. It is interesting to note that, as could be expected, for those targets for which there is no entity type defined (films, books and pictures), Ravichandran and Hovy (2002)'s extracts many errors, because it is not possible to apply the Named Entity Recognizer to clean up the results, and the accuracy remains below 10%. On the other hand, that trend does not seem to affect our system, which had very poor results for *painter-picture*, but reasonably good for *actor-film*.

Other interesting case is that of *birth places*. A manual observation of our generalized patterns shows that they often contain disjunctions of verbs such as that in (1), that detects not just the birth place but also places where the person lived. In this case, Ravichandran and Hovy (2002)'s patterns resulted more precise as they do not contain disjunctions or wildcards.

(1) `HOOK ,/, returned|travelled|born/VBN to|in/IN TARGET`

It is interesting that, among the three relationships with the smaller number of extracted patterns, one of them did not produce any result, and the two others attained a low precision. Therefore, it should be possible to improve the performance of the system if, while training, we augment the training corpora until the number of extracted patterns exceeds a given threshold.

Relation	Our approach	Ravichandran and Hovy's
Actor-film	76.84%	1.71%
Writer-book	28.13%	8.55%
Birth-year	79.67%	49.49%
Birth-place	14.56%	88.66%
Country-capital	72.43%	24.79%
Country-president	81.40%	16.13%
Death-year	96.71%	35.35%
Director-film	84.91%	1.01%
Painter-picture	-	0.85%
Player-team	52.50%	44.44%

Table 5: *Inclusion precision* on the same test corpus for our approach and Ravichandran and Hovy (2002)'s.

6 Related work

Extracting information using Machine Learning algorithms has received much attention since the nineties, mainly motivated by the Message Understanding Conferences (MUC6, 1995; MUC7, 1998). From the mid-nineties, there are systems that learn extraction patterns from partially annotated and unannotated data (Huffman, 1995; Riloff, 1996; Riloff and Schmelzenbach, 1998; Soderland, 1999).

Generalizing textual patterns (both manually and automatically) for the identification of relationships has been proposed since the early nineties (Hearst, 1992), and it has been applied to extending ontologies with hyperonymy and holonymy relationships (Kietz et al., 2000; Cimiano et al., 2004; Berland and Charniak, 1999), with overall precision varying between 0.39 and 0.68. Finkelstein-Landau and Morin (1999) learn patterns for company merging relationships with exceedingly good accuracies (between 0.72 and 0.93).

Rote extraction systems from the web have the advantage that the training corpora can be collected easily and automatically. Several similar approaches have been proposed (Brin, 1998; Agichtein and Gravano, 2000; Ravichandran and Hovy, 2002), with various applications: Question-Answering (Ravichandran and Hovy, 2002), multi-document Named Entity Coreference (Mann and Yarowsky, 2003), and generating

biographical information (Mann and Yarowsky, 2005).

7 Conclusions and future work

We have described here a new procedure for building a rote extractor from the web. Compared to other similar approaches, it addresses several issues: (a) it is able to generate generalized patterns containing wildcards; (b) it makes use of PoS and Named Entity tags during the generalization process; and (c) several relationships are learned and evaluated at the same time, in order to test each one on the test corpora built for the others. The results, measured in terms of precision and *inclusion precision* are very good in most of the cases.

Our system needs an input table, which may seem more complicated to compile that the list of related pairs used by previous approaches, but we have seen that the table can be built automatically from the lists, with no extra work. In any case, the time to build the table is significantly smaller than the time needed to write the extraction patterns manually.

Concerning future work, we are currently trying to improve the estimation of the patterns accuracy for the pruning step. We also plan to apply the obtained patterns in a system for automatically generating biographical knowledge bases from various web corpora.

References

E. Agichtein and L. Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of ICDL*, pages 85–94.

M. Berland and E. Charniak. 1999. Finding parts in very large corpora. In *Proceedings of ACL-99*.

S. Brin. 1998. Extracting patterns and relations from the World Wide Web. In *Proceedings of the WebDB Workshop at the 6th International Conference on Extending Database Technology, EDBT'98*.

P. Cimiano, S. Handschuh, and S. Staab. 2004. Towards the self-annotating web. In *Proceedings of the 13th World Wide Web Conference*, pages 462–471.

M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery. 1999. Learning to construct knowledge bases from the world wide web. *Artificial Intelligence*, 118(1–2):69–113.

M. Finkelstein-Landau and E. Morin. 1999. Extracting semantic relationships between terms: supervised vs. unsupervised methods. In *Workshop on Ontological Engineering on the Global Info. Infrastructure*.

M. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING-92*.

S. Huffman. 1995. Learning information extraction patterns from examples. In *IJCAI-95 Workshop on New Approaches to Learning for NLP*.

J. Kietz, A. Maedche, and R. Volz. 2000. A method for semi-automatic ontology acquisition from a corporate intranet. In *Workshop "Ontologies and text"*.

G. S. Mann and D. Yarowsky. 2003. Unsupervised personal name disambiguation. In *CoNLL-2003*.

G. S. Mann and D. Yarowsky. 2005. Multi-field information extraction and cross-document fusion. In *ACL 2005*.

M. Marcus, B. Santorini, and M.A. Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

MUC6. 1995. *Proceedings of the 6^{th} Message Understanding Conference (MUC-6)*. Morgan Kaufman.

MUC7. 1998. *Proceedings of the 7^{th} Message Understanding Conference (MUC-7)*. Morgan Kaufman.

D. Ravichandran and E. Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of ACL-2002*, pages 41–47.

E. Riloff and M. Schmelzenbach. 1998. An empirical approach to conceptual case frame acquisition. In *Proceedings of WVLC*, pages 49–56.

E. Riloff. 1996. Automatically generating extraction patterns from untagged text. In *AAAI*.

M. Ruiz-Casado, E. Alfonseca, and P. Castells. in press. Automatising the learning of lexical patterns: an application to the enrichment of wordnet by extracting semantic relationships from wikipedia. *Data and Knowledge Engineering*.

S. Soderland. 1999. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34(1–3):233–272.

R. Wagner and M. Fischer. 1974. The string-to-string correction problem. *Journal of Association for Computing Machinery*, 21.

MT Evaluation: Human-like vs. Human Acceptable

Enrique Amigó † , Jesús Giménez ‡ , Julio Gonzalo † , and Lluís Màrquez ‡

† Departamento de Lenguajes y Sistemas Informáticos
Universidad Nacional de Educación a Distancia
Juan del Rosal, 16, E-28040, Madrid
{enrique,julio}@lsi.uned.es

‡ TALP Research Center, LSI Department
Universitat Politècnica de Catalunya
Jordi Girona Salgado, 1–3, E-08034, Barcelona
{jgimenez,lluism}@lsi.upc.edu

Abstract

We present a comparative study on Machine Translation Evaluation according to two different criteria: Human Likeness and Human Acceptability. We provide empirical evidence that there is a relationship between these two kinds of evaluation: Human Likeness implies Human Acceptability but the reverse is not true. From the point of view of automatic evaluation this implies that metrics based on Human Likeness are more reliable for system tuning.

Our results also show that current evaluation metrics are not always able to distinguish between automatic and human translations. In order to improve the descriptive power of current metrics we propose the use of additional syntax-based metrics, and metric combinations inside the QARLA Framework.

1 Introduction

Current approaches to Automatic Machine Translation (MT) Evaluation are mostly based on metrics which determine the quality of a given translation according to its similarity to a given set of reference translations. The commonly accepted criterion that defines the quality of an evaluation metric is its level of correlation with human evaluators. High levels of correlation (Pearson over 0.9) have been attained at the system level (Eck and Hori, 2005). But this is an average effect: the degree of correlation achieved at the sentence level, crucial for an accurate error analysis, is much lower.

We argue that there is two main reasons that explain this fact:

Firstly, current MT evaluation metrics are based on shallow features. Most metrics work only at the lexical level. However, natural languages are rich and ambiguous, allowing for many possible different ways of expressing the same idea. In order to capture this flexibility, these metrics would require a combinatorial number of reference translations, when indeed in most cases only a single reference is available. Therefore, metrics with higher descriptive power are required.

Secondly, there exists, indeed, two different evaluation criteria: (i) Human Acceptability, i.e., to what extent an automatic translation could be considered acceptable by humans; and (ii) Human Likeness, i.e., to what extent an automatic translation could have been generated by a human translator. Most approaches to automatic MT evaluation implicitly assume that both criteria should lead to the same results; but this assumption has not been proved empirically or even discussed.

In this work, we analyze this issue through empirical evidence. First, in Section 2, we investigate to what extent current evaluation metrics are able to distinguish between human and automatic translations (Human Likeness). As individual metrics do not capture such distinction well, in Section 3 we study how to improve the descriptive power of current metrics by means of metric combinations inside the QARLA Framework (Amigó et al., 2005), including a new family of metrics based on syntactic criteria. Second, we claim that the two evaluation criteria (Human Acceptability and Human Likeness) are indeed of a different nature, and may lead to different results (Section 4). However, translations exhibiting a high level of Human Likeness obtain good results in human judges. Therefore, automatic evaluation metrics based on similarity to references should be

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 17–24,
Sydney, July 2006.

optimized over their capacity to represent Human Likeness. See conclusions in Section 5.

2 Descriptive Power of Standard Metrics

In this section we perform a simple experiment in order to measure the descriptive power of current state-of-the-art metrics, i.e., their ability to capture the features which characterize human translations with respect to automatic ones.

2.1 Experimental Setting

We use the data from the *Openlab 2006* Initiative[1] promoted by the TC-STAR Consortium[2]. This test suite is entirely based on European Parliament Proceedings[3], covering April 1996 to May 2005. We focus on the Spanish-to-English translation task. For the purpose of evaluation we use the development set which consists of 1008 sentences. However, due to lack of available MT outputs for the whole set we used only a subset of 504 sentences corresponding to the first half of the development set. Three human references per sentence are available.

We employ ten system outputs; nine are based on Statistical Machine Translation (SMT) systems (Giménez and Màrquez, 2005; Crego et al., 2005), and one is obtained from the free Systran[4] on-line rule-based MT engine. Evaluation results have been computed by means of the IQ$_{\text{MT}}$[5] Framework for Automatic MT Evaluation (Giménez and Amigó, 2006).

We have selected a representative set of 22 metric variants corresponding to six different families: BLEU (Papineni et al., 2001), NIST (Doddington, 2002), GTM (Melamed et al., 2003), mPER (Leusch et al., 2003), mWER (Nießen et al., 2000) and ROUGE (Lin and Och, 2004a).

2.2 Measuring Descriptive Power of Evaluation Metrics

Our main assumption is that if an evaluation metric is able to characterize human translations, then, human references should be closer to each other than automatic translations to other human references. Based on this assumption we introduce two measures (ORANGE and KING) which analyze the descriptive power of evaluation metrics from diferent points of view.

ORANGE Measure

ORANGE compares automatic and manual translations one-on-one. Let A and R be the sets of automatic and reference translations, respectively, and $x(a, R)$ an evaluation metric which outputs the quality of an automatic translation $a \in A$ by comparison to R. ORANGE measures the descriptive power as the probability that a human reference r is more similar than an automatic translation a to the rest of human references:

$$ORANGE_{A,R}(x) =$$
$$P(r \in R, a \in A : x(r, R-\{r\}) \geq x(a, R-\{r\}))$$

ORANGE was introduced by Lin and Och (2004b)[6] for the meta-evaluation of MT evaluation metrics. The $ORANGE$ measure provides information about the average behavior of automatic and manual translations regarding an evaluation metric.

KING Measure

However, ORANGE does not provide information about how many manual translations are discernible from automatic translations. The $KING$ measure complements the ORANGE, tackling these two issues by universally quantifying on variable a:

$$KING_{A,R}(x) =$$
$$P(r \in R, \forall a \in A : x(r, R-\{r\}) \geq x(a, R-\{r\}))$$

KING represents the probability that, for a given evaluation metric, a human reference is more similar to the rest of human references than *any* automatic translation [7].

KING does not depend on the distribution of automatic translations, and identifies the cases for

[1] http://tc-star.itc.it/openlab2006/

[2] http://www.tc-star.org/

[3] http://www.europarl.eu.int/

[4] http://www.systransoft.com.

[5] The IQ$_{\text{MT}}$ Framework may be freely downloaded at http://www.lsi.upc.edu/~nlp/IQMT.

[6] They defined this measure as the average rank of the reference translations within the combined machine and reference translations list.

[7] Originally KING is defined over the evaluation metric QUEEN, satisfying some restrictions which are not relevant in our context (Amigó et al., 2005).

which the given metric has been able to discern human translations from automatic ones. That is, it measures how many manual translations can be used as gold-standard for system evaluation/improvement purposes.

2.3 Results

Figure 1 shows the descriptive power, in terms of the ORANGE and KING measures, over the test set described in Subsection 2.1.

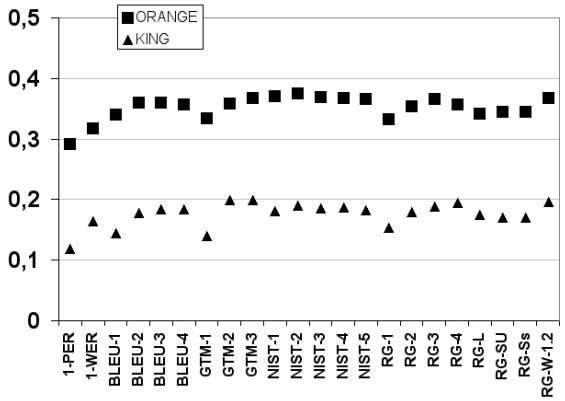


Figure 1: ORANGE and KING values for standard metrics.

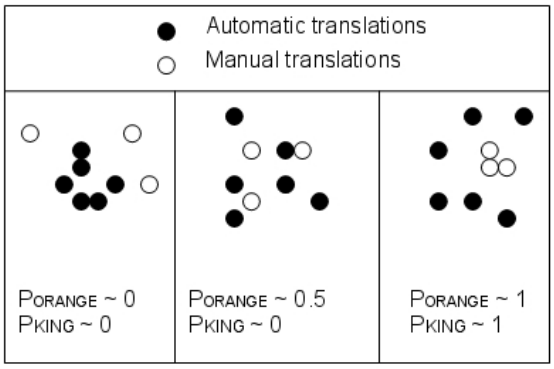


Figure 2: ORANGE and KING behavior.

ORANGE Results

All values of the ORANGE measure are lower than 0.5, which is the ORANGE value that a random metric would obtain (see central representation in Figure 2). This is a rather counterintuitive result. A reasonable explanation, however, is that automatic translations behave as centroids with respect to human translations, because they somewhat average the vocabulary distribution in the manual references; as a result, automatic translations are closer to each manual summary than manual summaries to each other (see leftmost representation in Figure 2).

In other words, automatic translations tend to share (lexical) features with most of the references, but not to match exactly any of them. This is a combined effect of:

- The nature of MT systems, mostly statistical, which compute their estimates based on the number of occurrences of words, tending to rely more on events that occur more often. Consequently, automatic translations typically consist of frequent words, which are likely to appear in most of the references.
- The shallowness of current metrics, which are not able to identify the common properties of manual translations with regard to automatic translations.

KING Results

KING values, on the other hand, are slightly higher than the value that a random metric would obtain ($\frac{1}{|A|} = 0.1$). This means that every standard metric is able to discriminate a certain number of manual translations from the set of automatic translations; for instance, GTM-3 identifies 19% of the manual references. For the remaining 81% of the test cases, however, GTM-3 cannot make the distinction, and therefore cannot be used to detect and improve weaknesses of the automatic MT systems.

These results provide an explanation for the low correlation between automatic evaluation metrics and human judgements at the sentence level. The necessary conclusion is that new metrics with higher descriptive power are required.

3 Improving Descriptive Power

The design of a metric that is able to capture all the linguistic aspects that distinguish human translations from automatic ones is a difficult path to trace. We approach this challenge by following a 'divide and conquer' strategy. We suggest to build a set of specialized similarity metrics devoted to the evaluation of partial aspects of MT quality. The challenge is then how to combine a set of similarity metrics into a single evaluation measure of

MT quality. The QARLA framework provides a solution for this challenge.

3.1 Similarity Metric Combinations inside QARLA

The QARLA Framework permits to combine several similarity metrics into a single quality measure (QUEEN). Besides considering the similarity of automatic translations to human references, the QUEEN measure additionally considers the distribution of similarities among human references.

The QUEEN measure operates under the assumption that a good translation must be similar to human references (R) according to all similarity metrics. QUEEN(a) is defined as the probability, over $R \times R \times R$, that for every metric x in a given metric set X the automatic translation a is more similar to a human reference than two other references to each other:

$$\text{QUEEN}_{X,R}(a) = P(\forall x \in X : x(a,r) \geq x(r',r''))$$

where a is the automatic translation being evaluated, $\langle r, r', r'' \rangle$ are three different human references in R, and $x(a,r)$ stands for the similarity of r to a.

In the case of Openlab data, we can count only on three human references per sentence. In order to increase the number of samples for QUEEN estimation we can use reference similarities $x(r',r'')$ between manual translation pairs from other sentences, assuming that the distances between manual references are relatively stable across examples.

3.2 Similarity Metrics

We begin by defining a set of 22 similarity metrics taken from the list of standard evaluation metrics in Subsection 2.1. Evaluation metrics can be tuned into similarity metrics simply by considering only one reference when computing its value.

Secondly, we explore the possibility of designing complementary similarity metrics that exploit linguistic information at levels further than lexical. Inspired in the work by Liu and Gildea (2005), who introduced a series of metrics based on constituent/dependency syntactic matching, we have designed three subgroups of syntactic similarity metrics. To compute them, we have used the dependency trees provided by the MINIPAR dependency parser (Lin, 1998). These metrics compute the level of word overlapping (unigram precision/recall) between dependency trees associated to automatic and reference translations, from three different points of view:

TREE-X overlapping between the words hanging from non-terminal nodes of type X of the tree. For instance, the metric TREE_PRED reflects the proportion of word overlapping between subtrees of type 'pred' (predicate of a clause).

GRAM-X overlapping between the words with the grammatical category X. For instance, the metric GRAM_A reflects the proportion of word overlapping between terminal nodes of type 'A' (Adjective/Adverbs).

LEVEL-X overlapping between the words hanging at a certain level X of the tree, or deeper. For instance, LEVEL-1 would consider overlapping between all the words in the sentences.

In addition, we also consider three coarser metrics, namely TREE, GRAM and LEVEL, which correspond to the average value of the finer metrics corresponding to each subfamily.

3.3 Metric Set Selection

We can compute KING over combinations of metrics by directly replacing the similarity metric $x(a,r)$ with the QUEEN measure. This corresponds exactly to the KING measure used in QARLA:

$$\text{KING}_{A,R}(X) = P(r \in R, \forall a \in A : \text{QUEEN}_{X,R-\{r\}}(r) \geq \text{QUEEN}_{X,R-\{r\}}(a))$$

KING represents the probability that, for a given set of human references R, and a set of metrics X, the QUEEN quality of a human reference is greater than the QUEEN quality of any automatic translation in A.

The similarity metrics based on standard evaluation measures together with the two new families of similarity metrics form a set of 104 metrics. Our goal is to obtain the subset of metrics with highest descriptive power; for this, we rely on the KING probability. A brute force exploration of all possible metric combinations is not viable. In order to

perform an approximate search for a local maximum in KING over all the possible metric combinations defined by X, we have used the following greedy heuristic:

1. Individual metrics are ranked by their KING value.

2. In decreasing rank order, metrics are individually added to the set of optimal metrics if, and only if, the global KING is increased.

After applying the algorithm we have obtained the optimal metric set:

{GTM-1, NIST-2, GRAM_A, GRAM_N, GRAM_AUX, GRAM_BE, TREE, TREE_AUX, TREE_PNMOD, TREE_PRED, TREE_REL, TREE_S and TREE_WHN}

which has a KING value of 0.29. This is significantly higher than the maximum KING obtained by any individual standard metric (which was 0.19 for GTM-3).

As to the probability ORANGE that a reference translation attains a higher score than an automatic translation, this metric set obtains a value of 0.49 vs. 0.42. This means that still the metrics are, on average, unable to discriminate between human references and automatic translations. However, the proportion of sentences for which the metrics are able to discriminate (KING value) is significantly higher.

The metric set with highest descriptive power contains metrics at different linguistic levels. For instance, GTM-1 and NIST-2 reward n-gram matches at the lexical level. GRAM_A, GRAM_N, GRAM_AUX and GRAM_BE capture word overlapping for nouns, auxiliary verbs, adjectives and adverbs, and auxiliary uses of the verb 'to be', respectively. TREE, TREE_AUX, TREE_PNMOD, TREE_PRED, TREE_REL, TREE_S and TREE_WHN reward lexical overlapping over different types of dependency subtrees: surface subjects, relative clauses, predicates, auxiliary verbs, postnominal modifiers, and whn-elements at C-spec positions, respectively.

These results are a clear indication that features from several linguistic levels are useful for the characterization of human translations.

4 Human-like vs. Human Acceptable

In this section we analyze the relationship between the two different kinds of MT evaluation presented: (i) the ability of MT systems to generate human-like translations, and (ii) the ability of MT systems to generate translations that look acceptable to human judges.

4.1 Experimental Setting

The ideal test set to study this dichotomy inside the QARLA Framework would consist of a large number of human references per sentence, and automatic outputs generated by heterogeneous MT systems.

4.2 Descriptive Power vs. Correlation with Human Judgements

We use the data and results from the IWSLT04 Evaluation Campaign[8]. We focus on the evaluation of the Chinese-to-English (CE) translation task, in which a set of 500 short sentences from the Basic Travel Expressions Corpus (BTEC) were translated (Akiba et al., 2004). For purposes of automatic evaluation, 16 reference translations and outputs by 20 different MT systems are available for each sentence. Moreover, each of these outputs was evaluated by three judges on the basis of adequacy and fluency (LDC, 2002). In our experiments we consider the sum of adequacy and fluency assessments.

However, the BTEC corpus has a serious drawback: sentences are very short (8 word length in average). In order to consider a sentence adequate we are practically forcing it to match exactly some of the human references. To alleviate this effect we selected sentences consisting of at least ten words. A total of 94 sentences (of 13 words length in average) satisfied this constraint.

Figure 3 shows, for all metrics, the relationship between the power of characterization of human references (KING, horizontal axis) and the correlation with human judgements (Pearson correlation, vertical axis). Data are plotted in three different groups: original standard metrics, single metrics inside QARLA (QUEEN measure), and the optimal metric combination according to KING. The optimal set is:

{GRAM_N, LEVEL_2, LEVEL_4, NIST-1, NIST-3, NIST-4, and 1-WER}

This set suggests that all kinds of n-grams play an important role in the characterization of human

[8] http://www.slt.atr.co.jp/IWSLT2004/

translations. The metric GRAM_N reflects the importance of noun translations. Unlike the Openlab corpus, levels of the dependency tree (LEVEL_2 and LEVEL_4) are descriptive features, but dependency relations are not (TREE metrics). This is probably due to the small average sentence length in IWSLT.

Metrics exhibiting a high level of correlation outside QARLA, such as NIST-3, also exhibit a high descriptive power (KING). There is also a tendency for metrics with a KING value around 0.6 to concentrate at a level of Pearson correlation around 0.5.

But the main point is the fact that the QUEEN measure obtained by the metric combination with highest KING does not yield the highest level of correlation with human assessments, which is obtained by standard metrics outside QARLA (0.5 vs. 0.7).

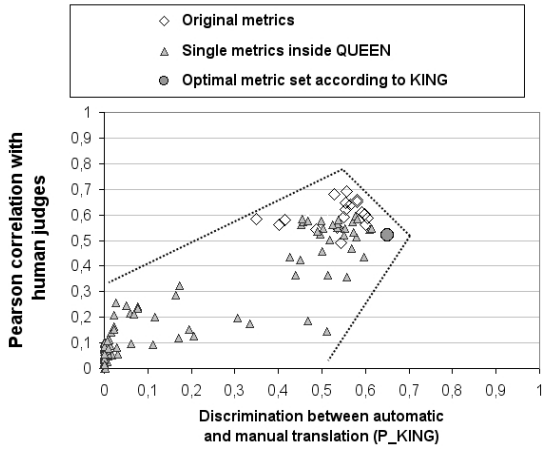


Figure 3: Human characterization vs. correlation with human judgements for IWSLT'04 CE translation task.

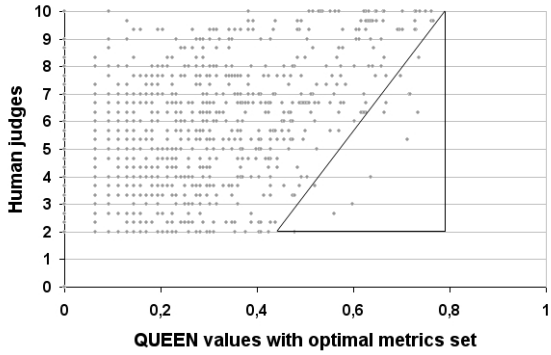


Figure 4: QUEEN values vs. human judgements for IWSLT'04 CE translation task.

4.3 Human Judgements vs. Similarity to References

In order to explain the above results, we have analyzed the relationship between human assessments and the QUEEN values obtained by the best combination of metrics for every individual translation.

Figure 4 shows that high values of QUEEN (i.e., similarity to references) imply high values of human judgements. But the reverse is not true. There are translations acceptable to a human judge but not similar to human translations according to QUEEN. This fact can be understood by inspecting a few particular cases. Table 1 shows two cases of translations exhibiting a very low QUEEN value and very high human judgment score. The two cases present the same kind of problem: there exists some word or phrase absent from all human references. In the first example, the automatic translation uses the expression "seats" to make a reservation, where humans invariably choose "table". In the second example, the automatic translation users "rack" as the place to put a bag, while humans choose "overhead bin", "overhead compartment", but never "rack".

Therefore, the QUEEN measure discriminates these automatic translations regarding to all human references, thus assigning them a low value. However, human judges find the translation still acceptable and informative, although not strictly human-like.

These results suggest that inside the set of human acceptable translations, which includes human-like translations, there is also a subset of translations unlikely to have been produced by a human translator. This is a drawback of MT evaluation based on human references when the evaluation criteria is Human Acceptability. The good news are that when Human Likeness increases, Human Acceptability increases as well.

5 Conclusions

We have analyzed the ability of current MT evaluation metrics to characterize human translations (as opposed to automatic translations), and the relationship between MT evaluation based on Human Acceptability and Human Likeness.

The first conclusion is that, over a limited number of references, standard metrics are unable to identify the features that characterize human translations. Instead, systems behave as centroids with

respect to human references. This is due, among other reasons, to the combined effect of the shallowness of current MT evaluation metrics (mostly lexical), and the fact that the choice of lexical items is mostly based on statistical methods. We suggest two complementary ways of solving this problem. First, we introduce a new family of syntax-based metrics covering partial aspects of MT quality. Second, we use the QARLA Framework to combine multiple metrics into a single measure of quality. In the future we will study the design of new metrics working at different linguistic levels. For instance, we are currently developing a new family of metrics based on shallow parsing (i.e., part-of-speech, lemma, and chunk information).

Second, our results suggest that there exists a clear relation between the two kinds of MT evaluation described. While Human Likeness is a sufficient condition to get Human Acceptability, Human Acceptability does not guarantee Human Likeness. Human judges may consider acceptable automatic translations that would never be generated by a human translator.

Considering these results, we claim that improving metrics according to their descriptive power (Human Likeness) is more reliable than improving metrics based on correlation with human judges. First, because this correlation is not granted, since automatic metrics are based on similarity to models. Second, because high Human Likeness ensures high scores from human judges.

References

Yasuhiro Akiba, Marcello Federico, Noriko Kando, Hiromi Nakaiwa, Michael Paul, and Jun'ichi Tsujii. 2004. Overview of the IWSLT04 Evaluation Campaign. In *Proceedings of the International Workshop on Spoken Language Translation*, pages 1–12, Kyoto, Japan.

Enrique Amigó, Julio Gonzalo, Anselmo Peñas, and Felisa Verdejo. 2005. QARLA: a Framework for the Evaluation of Automatic Sumarization. In *Proceedings of the 43th Annual Meeting of the Association for Computational Linguistics*, Michigan, June. Association for Computational Linguistics.

J.M. Crego, Costa jussà M.R., J.B. Mariño, and Fonollosa J.A.R. 2005. Ngram-based versus Phrase-based Statistical Machine Translation. In *Proceedings of the International Workshop on Spoken Language Technology (IWSLT'05)*.

George Doddington. 2002. Automatic Evaluation of Machine Translation Quality Using N-gram Co-Occurrence Statistics. In *Proceedings of the 2nd Internation Conference on Human Language Technology*, pages 138–145.

Matthias Eck and Chiori Hori. 2005. Overview of the IWSLT 2005 Evaluation Campaign. In *Proceedings of the International Workshop on Spoken Language Translation*, Carnegie Mellon University, Pittsburgh, PA.

Jesús Giménez and Enrique Amigó. 2006. IQMT: A Framework for Automatic Machine Translation Evaluation. In *Proceedings of the 5th LREC*.

Jesús Giménez and Lluís Màrquez. 2005. Combining Linguistic Data Views for Phrase-based SMT. In *Proceedings of the Workshop on Building and Using Parallel Texts, ACL*.

LDC. 2002. Linguistic Data Annotation Specification: Assessment of Fluency and Adequacy in Chinese-English Translations Revision 1.0. Technical report, Linguistic Data Consortium. http://-www.ldc.upenn.edu/Projects/TIDES/Translation/-TransAssess02.pdf.

G. Leusch, N. Ueffing, and H. Ney. 2003. A Novel String-to-String Distance Measure with Applications to Machine Translation Evaluation. In *Proceedings of MT Summit IX*.

Chin-Yew Lin and Franz Josef Och. 2004a. Automatic Evaluation of Machine Translation Quality Using Longest Common Subsequence and Skip-Bigram Statics. In *Proceedings of ACL*.

Chin-Yew Lin and Franz Josef Och. 2004b. ORANGE: a Method for Evaluating Automatic Evaluation Metrics for Machine Translation. In *Proceedings of COLING*.

Dekang Lin. 1998. Dependency-based Evaluation of MINIPAR. In *Proceedings of the Workshop on the Evaluation of Parsing Systems*.

Ding Liu and Daniel Gildea. 2005. Syntactic Features for Evaluation of Machine Translation. In *Proceedings of ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*.

I. Dan Melamed, Ryan Green, and Joseph P. Turian. 2003. Precision and Recall of Machine Translation. In *Proceedings of HLT/NAACL*.

S. Nießen, F.J. Och, G. Leusch, and H. Ney. 2000. Evaluation Tool for Machine Translation: Fast Evaluation for MT Research. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. Bleu: a method for automatic evaluation of machine translation, IBM Research Report, RC22176. Technical report, IBM T.J. Watson Research Center.

Automatic Translation:	my name is endo i ’ve reserved **seats** for nine o’clock
Human Reference 1:	this is endo i booked a **table** at nine o’clock
2:	i reserved a **table** for nine o’clock and my name is endo
3:	my name is endo and i made a reservation for a **table** at nine o’clock
4:	i am endo and i have a reservation for a **table** at nine pm
5:	my name is endo and i booked a **table** at nine o’clock
6:	this is endo i reserved a **table** for nine o’clock
7:	my name is endo and i reserved a **table** with you for nine o’clock
8:	i ’ve booked a **table** under endo for nine o’clock
9:	my name is endo and i have a **table** reserved for nine o’clock
10:	i ’m endo and i have a reservation for a **table** at nine o’clock
11:	my name is endo and i reserved a **table** for nine o’clock
12:	the name is endo and i have a reservation for nine
13:	i have a **table** reserved for nine under the name of endo
14:	hello my name is endo i reserved a **table** for nine o’clock
15:	my name is endo and i have a **table** reserved for nine o’clock
16:	my name is endo and i made a reservation for nine o’clock

Automatic Translation:	could you help me put my bag on the **rack** please
Human Reference 1:	could you help me put my bag in the **overhead bin**
2:	can you help me to get my bag into the **overhead bin**
3:	would you give me a hand with getting my bag into the **overhead bin**
4:	would you mind assisting me to put my bag into the **overhead bin**
5:	could you give me a hand putting my bag in the **overhead compartment**
6:	please help me put my bag in the **overhead bin**
7:	would you mind helping me put my bag in the **overhead compartment**
8:	do you mind helping me put my bag in the **overhead compartment**
9:	could i get a hand with putting my bag in the **overhead compartment**
10:	could i ask you to help me put my bag in the **overhead compartment**
11:	please help me put my bag in the **overhead bin**
12:	would you mind helping me put my bag in the **overhead compartment**
13:	i ’d like you to help me put my bag in the **overhead compartment**
14:	would you mind helping get my bag up into the **overhead storage compartment**
15:	may i get some assistance getting my bag into the **overhead storage compartment**
16:	please help me put my into the **overhead storage compartment**

Table 1: Automatic translations with high score in human judgements and low QUEEN value.

The Effect of Corpus Size in Combining Supervised and Unsupervised Training for Disambiguation

Michaela Atterer
Institute for NLP
University of Stuttgart
atterer@ims.uni-stuttgart.de

Hinrich Schütze
Institute for NLP
University of Stuttgart
hinrich@hotmail.com

Abstract

We investigate the effect of corpus size in combining supervised and unsupervised learning for two types of attachment decisions: relative clause attachment and prepositional phrase attachment. The supervised component is Collins' parser, trained on the Wall Street Journal. The unsupervised component gathers lexical statistics from an unannotated corpus of newswire text. We find that the combined system only improves the performance of the parser for small training sets. Surprisingly, the size of the unannotated corpus has little effect due to the noisiness of the lexical statistics acquired by unsupervised learning.

1 Introduction

The best performing systems for many tasks in natural language processing are based on supervised training on annotated corpora such as the Penn Treebank (Marcus et al., 1993) and the prepositional phrase data set first described in (Ratnaparkhi et al., 1994). However, the production of training sets is expensive. They are not available for many domains and languages. This motivates research on combining supervised with unsupervised learning since unannotated text is in ample supply for most domains in the major languages of the world. The question arises how much annotated and unannotated data is necessary in combination learning strategies. We investigate this question for two attachment ambiguity problems: relative clause (RC) attachment and prepositional phrase (PP) attachment. The supervised component is Collins' parser (Collins, 1997), trained on the Wall Street Journal. The unsupervised component gathers lexical statistics from an unannotated corpus of newswire text.

The sizes of both types of corpora, annotated and unannotated, are of interest. We would expect that large annotated corpora (training sets) tend to make the additional information from unannotated corpora redundant. This expectation is confirmed in our experiments. For example, when using the maximum training set available for PP attachment, performance decreases when "unannotated" lexical statistics are added.

For unannotated corpora, we would expect the opposite effect. The larger the unannotated corpus, the better the combined system should perform. While there is a general tendency to this effect, the improvements in our experiments reach a plateau quickly as the unlabeled corpus grows, especially for PP attachment. We attribute this result to the noisiness of the statistics collected from unlabeled corpora.

The paper is organized as follows. Sections 2, 3 and 4 describe data sets, methods and experiments. Section 5 evaluates and discusses experimental results. Section 6 compares our approach to prior work. Section 7 states our conclusions.

2 Data Sets

The unlabeled corpus is the Reuters RCV1 corpus, about 80,000,000 words of newswire text (Lewis et al., 2004). Three different subsets, corresponding to roughly 10%, 50% and 100% of the corpus, were created for experiments related to the size of the unannotated corpus. (Two weeks after Aug 5, 1997, were set apart for future experiments.)

The labeled corpus is the Penn Wall Street Journal treebank (Marcus et al., 1993). We

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 25–32,
Sydney, July 2006.

created the 5 subsets shown in Table 1 for experiments related to the size of the annotated corpus.

unlabeled R	
100%	20/08/1996–05/08/1997 (351 days)
50%	20/08/1996–17/02/1997 (182 days)
10%	20/08/1996–24/09/1996 (36 days)
labeled WSJ	
50%	sections 00–12 (23412 sentences)
25%	lines 1 – 292960 (11637 sentences)
5%	lines 1 – 58284 (2304 sentences)
1%	lines 1 – 11720 (500 sentences)
0.05%	lines 1 – 611 (23 sentences)

Table 1: Corpora used for the experiments: unlabeled Reuters (R) corpus for attachment statistics, labeled Penn treebank (WSJ) for training the Collins parser.

The test set, sections 13-24, is larger than in most studies because a single section does not contain a sufficient number of RC attachment ambiguities for a meaningful evaluation.

which-clauses subset	highA	lowA	total
develop set (sec 00-12)	71	211	282
test set (sec 13-24)	71	193	264
PP subset	verbA	nounA	total
develop set (sec 00-12)	5927	6560	12487
test set (sec 13-24)	5930	6273	12203

Table 2: RC and PP attachment ambiguities in the Penn Treebank. Number of instances with high attachment (highA), low attachment (lowA), verb attachment (verbA), and noun attachment (nounA) according to the gold standard.

All instances of RC and PP attachments were extracted from development and test sets, yielding about 250 RC ambiguities and 12,000 PP ambiguities per set (Table 2). An RC attachment ambiguity was defined as a sentence containing the pattern `NP1 Prep NP2 which`. For example, the relative clause in Example 1 can either attach to *mechanism* or to *System.*

(1) ... the exchange-rate mechanism of the European Monetary System, which links the major EC currencies.

A PP attachment ambiguity was defined as a subtree matching either [VP [NP PP]] or [VP NP PP]. An example of a PP attachment ambiguity is Example 2 where either the approval or the transaction is performed by written consent.

(2) ...a majority ...have approved the transaction by written consent ...

Both data sets are available for download (Web Appendix, 2006). We did not use the PP data set described by (Ratnaparkhi et al., 1994) because we are using more context than the limited context available in that set (see below).

3 Methods

Collins parser. Our baseline method for ambiguity resolution is the Collins parser as implemented by Bikel (Collins, 1997; Bikel, 2004). For each ambiguity, we check whether the attachment ambiguity is resolved correctly by the 5 parsers corresponding to the different training sets. If the attachment ambiguity is not recognized (e.g., because parsing failed), then the corresponding ambiguity is excluded for that instance of the parser. As a result, the size of the effective test set varies from parser to parser (see Table 4).

Minipar. The unannotated corpus is analyzed using minipar (Lin, 1998), a partial dependency parser. The corpus is parsed and all extracted dependencies are stored for later use. Dependencies in ambiguous PP attachments (those corresponding to [VP NP PP] and [VP [NP PP]] subtrees) are not indexed. An experiment with indexing both alternatives for ambiguous structures yielded poor results. For example, indexing both alternatives will create a large number of spurious verb attachments of *of*, which in turn will result in incorrect high attachments by our disambiguation algorithm.

For relative clauses, no such filtering is necessary. For example, spurious subject-verb dependencies due to RC ambiguities are rare compared to a large number of subject-verb dependencies that can be extracted reliably.

Inverted index. Dependencies extracted by minipar are stored in an inverted index (Witten et al., 1999), implemented in Lucene (Lucene, 2006). For example, "john subj buy", the analysis returned by minipar for *John buys*, is stored as "john buy john<subj

subj<buy john<subj<buy". All words, dependencies and partial dependencies of a sentence are stored together as one document. This storage mechanism enables fast on-line queries for lexical and dependency statistics, e.g., how many sentences contain the dependency "john subj buy", how often does *john* occur as a subject, how often does *buy* have *john* as a subject and *car* as an object etc. Query results are approximate because double occurrences are only counted once and structures giving rise to the same set of dependencies (*a piece of a tile of a roof of a house* vs. *a piece of a roof of a tile of a house*) cannot be distinguished. We believe that an inverted index is the most efficient data structure for our purposes. For example, we need not compute expensive joins as would be required in a database implementation. Our long-term goal is to use this inverted index of dependencies as a versatile component of NLP systems in analogy to the increasingly important role of search engines for association and word count statistics in NLP.

A total of three inverted indexes were created, one each for the 10%, 50% and 100% Reuters subset.

Lattice-Based Disambiguation. Our disambiguation method is Lattice-Based Disambiguation (LBD, (Atterer and Schütze, 2006)). We formalize a possible attachment as a triple $< R, i, X >$ where X is (the parse of) a phrase with two or more possible attachment nodes in a sentence S, i is one of these attachment nodes and R is (the relevant part of a parse of) S with X removed. For example, the two attachments in Example 2 are represented as the triples:
$<$ approved$_{i_1}$ the transaction$_{i_2}, i_1,$ by consent $>$,
$<$ approved$_{i_1}$ the transaction$_{i_2}, i_2,$ by consent $>$.

We decide between attachment possibilities based on pointwise mutual information, the well-known measure of how surprising it is to see R and X together given their individual frequencies:

$$\mathrm{MI}(< R, i, X >) = \log_2 \frac{P(<R,i,X>)}{P(R)P(X)}$$
$$\text{for } P(< R, i, X >), P(R), P(X) \neq 0$$
$$\mathrm{MI}(< R, i, X >) = 0 \quad \text{otherwise}$$

where the probabilities of the dependency structures $< R, i, X >$, R and X are estimated on the unlabeled corpus by querying the inverted index. Unfortunately, these structures will often not occur in the corpus. If this is the case we back off to generalizations of R and X. The generalizations form a lattice as shown in Figure 1 for PP attachment. For example, MN:pMN corresponds to *commercial transaction by unanimous consent*, N:pM to *transaction by unanimous* etc. For 0:p we compute MI of the two events "noun attachment" and "occurrence of p". Points in the lattice in Figure 1 are created by successive elimination of material from the complete context R:X. A child c directly dominated by a parent p is created by removing exactly one contextual element from p, either on the right side (the attachment phrase) or on the left side (the attachment node). For RC attachment, generalizations other than elimination are introduced such as the replacement of a proper noun (e.g., *Canada*) by its category (*country*) (see below).

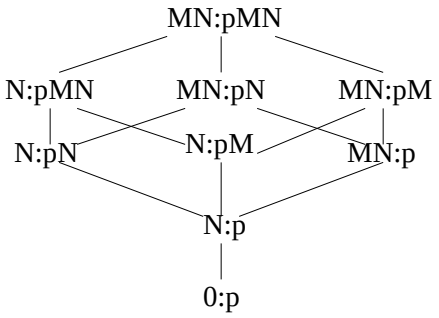


Figure 1: Lattice of pairs of potential attachment site (NP) and attachment phrase (PP). M: premodifying adjective or noun (upper or lower NP), N: head noun (upper or lower NP), p: Preposition.

The MI of each point in the lattice is computed. We then take the maximum over all MI values of the lattice as a measure of the *affinity* of attachment phrase and attachment node. The intuition is that we are looking for the strongest evidence available for the attachment. The strongest evidence is often not provided by the most specific context (MN:pMN in the example) since contextual elements like modifiers will only add noise to the attachment decision in some cases. The actual syntactic disambiguation is performed by computing the affinity (maximum over MI values in the lattice) for each possible attachment and selecting the attachment with highest affinity. (The

default attachment is selected if the two values are equal.) The second lattice for PP attachment, the lattice for attachment to the verb, has a structure identical to Figure 1, but the attachment node is SV instead of MN, where S denotes the subject and V the verb. So the supremum of that lattice is SV:pMN and the infimum is 0:p (which in this case corresponds to the MI of verb attachment and occurrence of the preposition).

LBD is motivated by the desire to use as much context as possible for disambiguation. Previous work on attachment disambiguation has generally used less context than in this paper (e.g., modifiers have not been used for PP attachment). No change to LBD is necessary if the lattice of contexts is extended by adding additional contextual elements (e.g., the preposition between the two attachment nodes in RC, which we do not consider in this paper).

4 Experiments

The Reuters corpus was parsed with minipar and all dependencies were extracted. Three inverted indexes were created, corresponding to 10%, 50% and 100% of the corpus.[1] Five parameter sets for the Collins parser were created by training it on the WSJ training sets in Table 1. Sentences with attachment ambiguities in the WSJ corpus were parsed with minipar to generate Lucene queries. (We chose this procedure to ensure compatibility of query and index formats.) The Lucene queries were run on the three indexes. LBD disambiguation was then applied based on the statistics returned by the queries. LBD results are applied to the output of the Collins parser by simply replacing all attachment decisions with LBD decisions.

4.1 RC attachment

The lattice for LBD in RC attachment is shown in Figure 2. When disambiguating an RC attachment, two instances of the lattice are formed, one for NP1 and one for NP2 in `NP1 Prep NP2 RC`. Figure 2 shows the maximum possible lattice. If contextual elements are not present in a context (e.g., a modifier), then the lattice will be smaller. The supremum of the lattice corresponds to a query that includes the entire NP (including modifying adjectives and nouns)[2], the verb and its object. Example: *exchange_rate<nn<mechanim && mechanism<subj<link && currency<obj<link.*

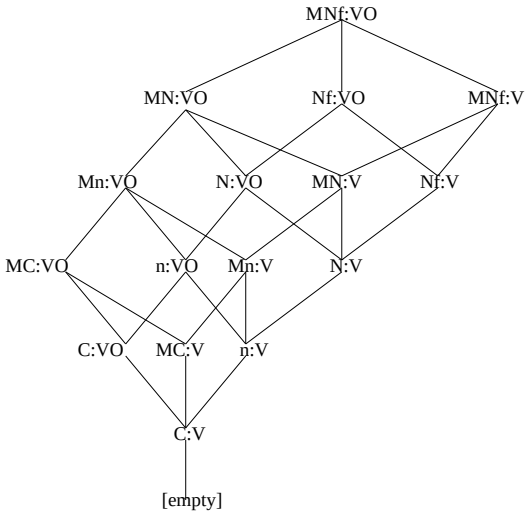


Figure 2: Lattice of pairs of potential attachment site NP and relative clause X. M: premodifying adjective or noun, Nf: head noun with lexical modifiers, N: head noun only, n: head noun in lower case, C: class of NP, V: verb in relative clause, O: object of verb in the relative clause.

To generalize contexts in the lattice, the following generalization operations are employed:

- strip the NP of the modifying adjective/noun (*weekly report → report*)
- use only the head noun of the NP (*Catastrophic Care Act → Act*)
- use the head noun in lower case (*Act → act*)
- for named entities use a hypernym of the NP (*American Bell Telephone Co. → company*)
- strip the object from X (*company have subsidiary → company have*)

The most important dependency for disam-

[1] In fact, two different sets of inverted indexes were created, one each for PP and RC disambiguation. The RC index indexes all dependencies, including ambiguous PP dependencies. Computing the RC statistics on the PP index should not affect the RC results presented here, but we didn't have time to confirm this experimentally for this paper.

[2] From the minipar output, we use all adjectives that modify the NP via the relation *mod*, and all nouns that modify the NP via the relation *nn*.

biguation is the noun-verb link, but the other dependencies also improve the accuracy of disambiguation (Atterer and Schütze, 2006). For example, light verbs like *make* and *have* only provide disambiguation information when their objects are also considered.

Downcasing and hypernym generalizations were used because proper nouns often cause sparse data problems. Named entity classes were identified with LingPipe (LingPipe, 2006). Named entities identified as companies or organizations are replaced with *company* in the query. Locations are replaced with *country*. Persons block RC attachment because *which*-clauses do not attach to person names, resulting in an attachment of the RC to the other NP.

query	MI
+exchange_rate⟨nn⟨mechanism +mechanism⟨subj⟨link +currency⟨obj⟨link	12.2
+exchange_rate⟨nn⟨mechanism +mechanism⟨subj⟨link	4.8
+mechanism⟨subj⟨link +currency⟨obj⟨link	10.2
mechanism⟨subj⟨link	3.4
+European_Monetary_System⟨subj⟨link +currency⟨obj⟨link	0
+System⟨subj⟨link +currency⟨obj⟨link	0
European_Monetary_System⟨subj⟨link	0
System⟨subj⟨link	0
+system⟨subj⟨link +currency⟨obj⟨link	0
system⟨subj⟨link	1.2
+company⟨subj⟨link +currency⟨obj⟨link	0
company⟨subj⟨link	-1.1
empty	3

Table 3: Queries for computing high attachment (above) and low attachment (below) for Example 1.

Table 3 shows queries and mutual information values for Example 1. The highest values are 12.2 for high attachment (*mechanism*) and 3 for low attachment (*System*). The algorithm therefore selects high attachment.

The value 3 for low attachment is the default value for the empty context. This value reflects the bias for low attachment: the majority of relative clauses are attached low. If all MI-values are zero or otherwise low, this procedure will automatically result in low attachment.[3]

[3]We experimented with a number of values (2, 3, and 4) on the development set. Accuracy of the algorithm was best for a value of 3. The results presented here differ slightly from those in (Atterer and Schütze, 2006) due to a coding error.

Decision list. For increased accuracy, LBD is embedded in the following decision list.

1. If minipar has already chosen high attachment, choose high attachment (this only occurs if `NP1 Prep NP2` is a named entity).
2. If there is agreement between the verb and only one of the NPs, attach to this NP.
3. If one of the NPs is in a list of person entities, attach to the other NP.[4]
4. If possible, use LBD.
5. If none of the above strategies was successful (e.g. in the case of parsing errors), attach low.

4.2 PP attachment

The two lattices for LBD applied to PP attachment were described in Section 3 and Figure 1. The only generalization operation used in these two lattices is elimination of contextual elements (in particular, there is no downcasing and named entity recognition). Note that in RC attachment, we compare affinities of two instances of the same lattice (the one shown in Figure 2). In PP attachment, we compare affinities of two different lattices since the two attachment points (verb vs. noun) are different. The basic version of LBD (with the untuned default value 0 and without decision lists) was used for PP attachment.

5 Evaluation and Discussion

Evaluation results are shown in Table 4. The lines marked *LBD* evaluate the performance of LBD separately (without Collins' parser). LBD is significantly better than the baseline for PP attachment ($p < 0.001$, all tests are χ^2 tests). LBD is also better than baseline for RC attachment, but this result is not significant due to the small size of the data set (264). Note that the baseline for PP attachment is 51.4% as indicated in the table (upper right corner of PP table), but that the baseline for RC attachment is 73.1%. The difference between 73.1% and 76.1% (upper right corner of RC table) is due to the fact that for RC attachment LBD proper is embedded in a decision list. The decision list alone, with an

[4]This list contains 136 entries and was semiautomatically computed from the Reuters corpus: Antecedents of *who* relative clauses were extracted, and the top 200 were filtered manually.

RC attachment						
Train data	#	Coll. only	100% R	50% R	10% R	0% R
LBD	264		78.4%	78.0%	76.9%	76.1%
50%	251	71.7%	78.5%	78.1%	76.9%	76.1%
25%	250	70.0%	78.0%	77.6%	76.4%	76.4%
5%	238	68.9%	78.2%	77.7%	76.9%	76.1%
1%	245	67.8%	78.8%	78.4%	77.1%	76.7%
0.05%	194	60.8%	76.8%	76.3%	75.8%	73.7%

PP attachment						
Train data	#	Coll. only	100% R	50% R	10% R	0% R
LBD	12203		73.4%	73.4%	73.0%	51.4%
50%	11953	82.8%	73.6%	73.6%	73.2%	51.7%
25%	11950	81.5%	73.6%	73.7%	73.3%	51.7%
5%	11737	77.4%	74.1%	74.2%	73.7%	52.3%
1%	11803	72.9%	73.6%	73.6%	73.2%	51.6%
0.05%	8486	58.0%	73.9%	73.8%	74.0%	52.8%

Table 4: Experimental results. Results for LBD (without Collins) are given in the first lines. # is the size of the test set. The baselines are 73.1% (RC) and 51.4% (PP). The combined method performs better for small training sets. There is no significant difference between 10%, 50% and 100% for the combination method ($p < 0.05$).

unlabeled corpus of size 0, achieves a performance of 76.1%.

The bottom five lines of each table evaluate combinations of a parameter set trained on a subset of WSJ (0.05% – 50%) and a particular size of the unlabeled corpus (100% – 0%). In addition, the third column gives the performance of Collins' parser without LBD. Recall that test set size (second column) varies because we discard a test instance if Collins' parser does not recognize that there is an ambiguity (e.g., because of a parse failure). As expected, performance increases as the size of the training set grows, e.g., from 58.0% to 82.8% for PP attachment.

The combination of Collins and LBD is consistently better than Collins for RC attachment (not statistically significant due to the size of the data set). However, this is not the case for PP attachment. Due to the good performance of Collins' parser for even small training sets, the combination is only superior for the two smallest training sets (significant for the smallest set, $p < 0.001$).

The most surprising result of the experiments is the small difference between the three unlabeled corpora. There is no clear pattern in the data for PP attachment and only a small effect for RC attachment: an increase between 1% and 2% when corpus size is increased from 10% to 100%.

We performed an analysis of a sample of incorrectly attached PPs to investigate why unlabeled corpus size has such a small effect. We found that the noisiness of the statistics extracted from Reuters were often responsible for attachment errors. The noisiness is caused by our filtering strategy (ambiguous PPs are not used, resulting in undercounting), by the approximation of counts by Lucene (Lucene overcounts and undercounts as discussed in Section 3) and by minipar parse errors. Parse errors are particularly harmful in cases like *the impact it would have on prospects*, where, due to the extraction of the NP *impact*, minipar attaches the PP to the verb. We did not filter out these more complex ambiguous cases. Finally, the two corpora are from distinct sources and from distinct time periods (early nineties vs. mid-nineties). Many topic- and time-specific dependencies can only be mined from more similar corpora.

The experiments reveal interesting differences between PP and RC attachment. The dependencies used in RC disambiguation rarely occur in an ambiguous context (e.g., most subject-verb dependencies can be reliably extracted). In contrast, a large proportion of the dependencies needed in PP disambiguation (verb-prep and noun-prep dependencies) do occur in ambiguous contexts. Another difference is that RC attachment is syntactically more complex. It interacts with agreement, passive and long-distance depen-

dencies. The algorithm proposed for RC applies grammatical constraints successfully. A final difference is that the baseline for RC is much higher than for PP and therefore harder to beat.[5]

An innovation of our disambiguation system is the use of a search engine, lucene, for serving up dependency statistics. The advantage is that counts can be computed quickly and dynamically. New text can be added on an ongoing basis to the index. The updated dependency statistics are immediately available and can benefit disambiguation performance. Such a system can adapt easily to new topics and changes over time. However, this architecture negatively affects accuracy. The unsupervised approach of (Hindle and Rooth, 1993) achieves almost 80% accuracy by using partial dependency statistics to disambiguate ambiguous sentences in the unlabeled corpus. Ambiguous sentences were excluded from our index to make index construction simple and efficient. Our larger corpus (about 6 times as large as Hindle et al.'s) did not compensate for our lower-quality statistics.

6 Related Work

Other work combining supervised and unsupervised learning for parsing includes (Charniak, 1997), (Johnson and Riezler, 2000), and (Schmid, 2002). These papers present integrated formal frameworks for incorporating information learned from unlabeled corpora, but they do not explicitly address PP and RC attachment. The same is true for uncorrected colearning in (Hwa et al., 2003).

Conversely, no previous work on PP and RC attachment has integrated specialized ambiguity resolution into parsing. For example, (Toutanova et al., 2004) present one of the best results achieved so far on the WSJ PP set: 87.5%. They also integrate supervised and unsupervised learning. But to our knowledge, the relationship to parsing has not been explored before – even though application to parsing is the stated objective of most work on PP attachment.

[5]However, the baseline is similarly high for the PP problem if the most likely attachment is chosen per preposition: 72.2% according to (Collins and Brooks, 1995).

With the exception of (Hindle and Rooth, 1993), most unsupervised work on PP attachment is based on superficial analysis of the unlabeled corpus without the use of partial parsing (Volk, 2001; Calvo et al., 2005). We believe that dependencies offer a better basis for reliable disambiguation than cooccurrence and fixed-phrase statistics. The difference to (Hindle and Rooth, 1993) was discussed above with respect to analysing the unlabeled corpus. In addition, the decision procedure presented here is different from Hindle et al.'s. LBD uses more context and can, in principle, accommodate arbitrarily large contexts. However, an evaluation comparing the performance of the two methods is necessary.

The LBD model can be viewed as a backoff model that combines estimates from several "backoffs". In a typical backoff model, there is a single more general model to back off to. (Collins and Brooks, 1995) also present a model with multiple backoffs. One of its variants computes the estimate in question as the average of three backoffs. In addition to the maximum used here, testing other combination strategies for the MI values in the lattice (e.g., average, sum, frequency-weighted sum) would be desirable. In general, MI has not been used in a backoff model before as far as we know.

Previous work on relative clause attachment has been supervised (Siddharthan, 2002a; Siddharthan, 2002b; Yeh and Vilain, 1998).[6] (Siddharthan, 2002b)'s accuracy for RC attachment is 76.5%.[7]

7 Conclusion

Previous work on specific types of ambiguities (like RC and PP) has not addressed the integration of specific resolution algorithms into a generic statistical parser. In this paper, we have shown for two types of ambiguities, relative clause and prepositional phrase attachment ambiguity, that integration into a statistical parser is possible and that the com-

[6]Strictly speaking, our experiments were not completely unsupervised since the default value and the most frequent attachment were determined based on the development set.

[7]We attempted to recreate Siddharthan's training and test sets, but were not able to based on the description in the paper and email communication with the author.

bined system performs better than either component by itself. However, for PP attachment this only holds for small training set sizes. For large training sets, we could only show an improvement for RC attachment.

Surprisingly, we only found a small effect of the size of the unlabeled corpus on disambiguation performance due to the noisiness of statistics extracted from raw text. Once the unlabeled corpus has reached a certain size (5-10 million words in our experiments) combined performance does not increase further.

The results in this paper demonstrate that the baseline of a state-of-the-art lexicalized parser for specific disambiguation problems like RC and PP is quite high compared to recent results for stand-alone PP disambiguation. For example, (Toutanova et al., 2004) achieve a performance of 87.6% for a training set of about 85% of WSJ. That number is not that far from the 82.8% achieved by Collins' parser in our experiments when trained on 50% of WSJ. Some of the supervised approaches to PP attachment may have to be reevaluated in light of this good performance of generic parsers.

References

Michaela Atterer and Hinrich Schütze. 2006. A lattice-based framework for enhancing statistical parsers with information from unlabeled corpora. In *CoNLL*.

Daniel M. Bikel. 2004. Intricacies of Collins' parsing model. *Computational Linguistics*, 30(4):479–511.

Hiram Calvo, Alexander Gelbukh, and Adam Kilgarriff. 2005. Distributional thesaurus vs. WordNet: A comparison of backoff techniques for unsupervised PP attachment. In *CICLing*.

Eugene Charniak. 1997. Statistical parsing with a context-free grammar and word statistics. In *AAAI/IAAI*, pages 598–603.

Michael Collins and James Brooks. 1995. Prepositional attachment through a backed-off model. In *Third Workshop on Very Large Corpora*. Association for Computational Linguistics.

Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *ACL*.

Donald Hindle and Mats Rooth. 1993. Structural ambiguity and lexical relations. *Computational Linguistics*, 19(1):103–120.

Rebecca Hwa, Miles Osborne, Anoop Sarkar, and Mark Steedman. 2003. Corrected co-training for statistical parsers. In *Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining, ICML*.

Mark Johnson and Stefan Riezler. 2000. Exploiting auxiliary distributions in stochastic unification-based grammars. In *NAACL*.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. RCV1: A new benchmark collection for text categorization research. *J. Mach. Learn. Res.*, 5.

Dekang Lin. 1998. Dependency-based evaluation of MINIPAR. In *Workshop on the Evaluation of Parsing Systems*, Granada, Spain.

LingPipe. 2006. http://www.alias-i.com/lingpipe/.

Lucene. 2006. http://lucene.apache.org.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large natural language corpus of English: the Penn treebank. *Computational Linguistics*, 19:313–330.

Adwait Ratnaparkhi, Jeff Reynar, and Salim Roukos. 1994. A maximum entropy model for prepositional phrase attachment. In *HLT*.

Helmut Schmid. 2002. Lexicalization of probabilistic grammars. In *Coling*.

Advaith Siddharthan. 2002a. Resolving attachment and clause boundary ambiguities for simplifying relative clause constructs. In *Student Research Workshop, ACL*.

Advaith Siddharthan. 2002b. Resolving relative clause attachment ambiguities using machine learning techniques and wordnet hierarchies. In *4th Discourse Anaphora and Anaphora Resolution Colloquium*.

Kristina Toutanova, Christopher D. Manning, and Andrew Y. Ng. 2004. Learning random walk models for inducing word dependency distributions. In *ICML*.

Martin Volk. 2001. Exploiting the WWW as a corpus to resolve pp attachment ambiguities. In *Corpus Linguistics 2001*.

Web Appendix. 2006. http://www.ims.uni-stuttgart.de/∼schuetze/colingacl06/apdx.html.

Ian H. Witten, Alistair Moffat, and Timothy C. Bell. 1999. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufman.

Alexander S. Yeh and Marc B. Vilain. 1998. Some properties of preposition and subordinate conjunction attachments. In *Coling*.

A Phrase-based Statistical Model for SMS Text Normalization

AiTi Aw, Min Zhang, Juan Xiao, Jian Su
Institute of Infocomm Research
21 Heng Mui Keng Terrace
Singapore 119613
{aaiti,mzhang,stuxj,sujian}@i2r.a-star.edu.sg

Abstract

Short Messaging Service (SMS) texts behave quite differently from normal written texts and have some very special phenomena. To translate SMS texts, traditional approaches model such irregularities directly in Machine Translation (MT). However, such approaches suffer from customization problem as tremendous effort is required to adapt the language model of the existing translation system to handle SMS text style. We offer an alternative approach to resolve such irregularities by normalizing SMS texts before MT. In this paper, we view the task of SMS normalization as a translation problem from the SMS language to the English language [1] and we propose to adapt a phrase-based statistical MT model for the task. Evaluation by 5-fold cross validation on a parallel SMS normalized corpus of 5000 sentences shows that our method can achieve 0.80702 in BLEU score against the baseline BLEU score 0.6958. Another experiment of translating SMS texts from English to Chinese on a separate SMS text corpus shows that, using SMS normalization as MT preprocessing can largely boost SMS translation performance from 0.1926 to 0.3770 in BLEU score.

1 Motivation

SMS translation is a mobile Machine Translation (MT) application that translates a message from one language to another. Though there exists many commercial MT systems, direct use of such systems fails to work well due to the special phenomena in SMS texts, e.g. the unique relaxed and creative writing style and the frequent use of unconventional and not yet standardized short-forms. Direct modeling of these special phenomena in MT requires tremendous effort. Alternatively, we can normalize SMS texts into grammatical texts before MT. In this way, the traditional MT is treated as a "black-box" with little or minimal adaptation. One advantage of this pre-translation normalization is that the diversity in different user groups and domains can be modeled separately without accessing and adapting the language model of the MT system for each SMS application. Another advantage is that the normalization module can be easily utilized by other applications, such as SMS to voicemail and SMS-based information query.

In this paper, we present a phrase-based statistical model for SMS text normalization. The normalization is visualized as a translation problem where messages in the SMS language are to be translated to normal English using a similar phrase-based statistical MT method (Koehn et al., 2003). We use IBM's BLEU score (Papineni et al., 2002) to measure the performance of SMS text normalization. BLEU score computes the similarity between two sentences using n-gram statistics, which is widely-used in MT evaluation. A set of parallel SMS messages, consisting of 5000 raw (un-normalized) SMS messages and their manually normalized references, is constructed for training and testing. Evaluation by 5-fold cross validation on this corpus shows that our method can achieve accuracy of 0.80702 in BLEU score compared to the baseline system of 0.6985. We also study the impact of our SMS text normalization on the task of SMS translation. The experiment of translating SMS texts from English to Chinese on a corpus comprising 402 SMS texts shows that, SMS normalization as a preprocessing step of MT can boost the translation performance from 0.1926 to 0.3770 in BLEU score.

The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 summarizes the characteristics of English SMS texts. Section 4 discusses our method and Section 5 reports our experiments. Section 6 concludes the paper.

2 Related Work

There is little work reported on SMS normalization and translation. Bangalore et al. (2002) used

[1] This paper only discusses English SMS text normalization.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 33–40,
Sydney, July 2006.

a consensus translation technique to bootstrap parallel data using off-the-shelf translation systems for training a hierarchical statistical translation model for general domain instant messaging used in Internet chat rooms. Their method deals with the special phenomena of the instant messaging language (rather than the SMS language) in each individual MT system. Clark (2003) proposed to unify the process of tokenization, segmentation and spelling correction for normalization of general noisy text (rather than SMS or instant messaging texts) based on a noisy channel model at the character level. However, results of the normalization are not reported. Aw et al. (2005) gave a brief description on their input pre-processing work for an English-to-Chinese SMS translation system using a word-group model. In addition, in most of the commercial SMS translation applications [2], SMS lingo (i.e., SMS short form) dictionary is provided to replace SMS short-forms with normal English words. Most of the systems do not handle OOV (out-of-vocabulary) items and ambiguous inputs. Following compares SMS text normalization with other similar or related applications.

2.1 SMS Normalization versus General Text Normalization

General text normalization deals with Non-Standard Words (NSWs) and has been well-studied in text-to-speech (Sproat et al., 2001) while SMS normalization deals with Non-Words (NSs) or lingoes and has seldom been studied before. NSWs, such as digit sequences, acronyms, mixed case words (WinNT, SunOS), abbreviations and so on, are grammatically correct in linguistics. However lingoes, such as "*b4*" (*before*) and "*bf*" (*boyfriend*), which are usually self-created and only accepted by young SMS users, are not yet formalized in linguistics. Therefore, the special phenomena in SMS texts impose a big challenge to SMS normalization.

2.2 SMS Normalization versus Spelling Correction Problem

Intuitively, many would regard SMS normalization as a spelling correction problem where the lingoes are erroneous words or non-words to be replaced by English words. Researches on spelling correction centralize on typographic and cognitive/orthographic errors (Kukich, 1992) and use approaches (M.D. Kernighan, Church and Gale, 1991) that mostly model the edit operations using distance measures (Damerau 1964; Levenshtein 1966), specific word set confusions (Golding and Roth, 1999) and pronunciation modeling (Brill and Moore, 2000; Toutanova and Moore, 2002). These models are mostly character-based or string-based without considering the context. In addition, the author might not be aware of the errors in the word introduced during the edit operations, as most errors are due to mistype of characters near to each other on the keyboard or homophones, such as "poor" or "pour".

In SMS, errors are not isolated within word and are usually not surrounded by clean context. Words are altered deliberately to reflect sender's distinct creation and idiosyncrasies. A character can be deleted on purpose, such as "*wat*" (*what*) and "*hv"* (*have*). It also consists of short-forms such as "*b4*" (*before*), "*bf*" (*boyfriend*). In addition, normalizing SMS text might require the context to be spanned over more than one lexical unit such as "*lemme*" (*let me*), "*ur*" (*you are*) etc. Therefore, the models used in spelling correction are inadequate for providing a complete solution for SMS normalization.

2.3 SMS Normalization versus Text Paraphrasing Problem

Others may regard SMS normalization as a paraphrasing problem. Broadly speaking, paraphrases capture core aspects of variability in language, by representing equivalencies between different expressions that correspond to the same meaning. In most of the recent works (Barzilay and McKeown, 2001; Shimohata, 2002), they are acquired (semi-) automatically from large comparable or parallel corpora using lexical and morpho-syntactic information.

Text paraphrasing works on clean texts in which contextual and lexical-syntactic features can be extracted and used to find "approximate conceptual equivalence". In SMS normalization, we are dealing with non-words and "ungrammatically" sentences with the purpose to normalize or standardize these words and form better sentences. The SMS normalization problem is thus different from text paraphrasing. On the other hand, it bears some similarities with MT as we are trying to "convert" text from one language to another. However, it is a simpler problem as most of the time; we can find the same word in both the source and target text, making alignment easier.

[2] http://www.etranslator.ro and http://www.transl8bit.com

3 Characteristics of English SMS

Our corpus consists of 55,000 messages collected from two sources, a SMS chat room and correspondences between university students. The content is mostly related to football matches, making friends and casual conversations on "how, what and where about". We summarize the text behaviors into two categories as below.

3.1 Orthographic Variation

The most significant orthographic variant in SMS texts is in the use of non-standard, self-created short-forms. Usually, sender takes advantage of phonetic spellings, initial letters or number homophones to mimic spoken conversation or shorten words or phrases (*hw* vs. *homework* or *how*, *b4* vs. *before*, *cu* vs. *see you*, *2u* vs. *to you*, *oic* vs. *oh I see*, etc.) in the attempt to minimize key strokes. In addition, senders create a new form of written representation to express their oral utterances. Emotions, such as ":(" symbolizing sad, ":)" symbolizing smiling, ":()" symbolizing shocked, are representations of body language. Verbal effects such as "*hehe*" for laughter and emphatic discourse particles such as "*lor*", "*lah*", "*meh*" for colloquial English are prevalent in the text collection.

The loss of "alpha-case" information posts another challenge in lexical disambiguation and introduces difficulty in identifying sentence boundaries, proper nouns, and acronyms. With the flexible use of punctuation or not using punctuation at all, translation of SMS messages without prior processing is even more difficult.

3.2 Grammar Variation

SMS messages are short, concise and convey much information within the limited space quota (160 letters for English), thus they tend to be implicit and influenced by pragmatic and situation reasons. These inadequacies of language expression such as deletion of articles and subject pronoun, as well as problems in number agreements or tenses make SMS normalization more challenging. Table 1 illustrates some orthographic and grammar variations of SMS texts.

3.3 Corpus Statistics

We investigate the corpus to assess the feasibility of replacing the lingoes with normal English words and performing limited adjustment to the text structure. Similarly to Aw et al. (2005), we focus on the three major cases of transformation as shown in the corpus: (1) replacement of OOV words and non-standard SMS lingoes; (2) removal of slang and (3) insertion of auxiliary or copula verb and subject pronoun.

Phenomena	Messages
1. Dropping '?' at the end of question	*btw, wat is ur view* (*By the way, what is your view?*)
2. Not using any punctuation at all	*Eh speak english mi malay not tt good* (*Eh, speak English! My Malay is not that good.*)
3. Using spelling/punctuation for emphasis	*goooooood Sunday morning !!!!!!* (*Good Sunday morning!*)
4. Using phonetic spelling	*dat iz enuf* (*That is enough*)
5. Dropping vowel	*i hv cm to c my luv.* (*I have come to see my love.*)
6. Introducing local flavor	*yar lor where u go juz now* (*yes, where did you go just now?*)
7. Dropping verb	*I hv 2 go. Dinner w parents.* (*I have to go. Have dinner with parents.*)

Table 1. Examples of SMS Messages

Transformation	Percentage (%)
Insertion	8.09
Deletion	5.48
Substitution	86.43

Table 2. Distribution of Insertion, Deletion and Substitution Transformation.

Substitution	Deletion	Insertion
u -> you	*m*	*are*
2 → to	*lah*	*am*
n → and	*t*	*is*
r → are	*ah*	*you*
ur →your	*leh*	*to*
dun → don't	*1*	*do*
man → manchester	*huh*	*a*
no → number	*one*	*in*
intro → introduce	*lor*	*yourself*
wat → what	*ahh*	*will*

Table 3. Top 10 Most Common Substitution, Deletion and Insertion

Table 2 shows the statistics of these transformations based on 700 messages randomly selected, where 621 (88.71%) messages required

normalization with a total of 2300 transformations. Substitution accounts for almost 86% of all transformations. Deletion and substitution make up the rest. Table 3 shows the top 10 most common transformations.

4 SMS Normalization

We view the SMS language as a variant of English language with some derivations in vocabulary and grammar. Therefore, we can treat SMS normalization as a MT problem where the SMS language is to be translated to normal English. We thus propose to adapt the statistical machine translation model (Brown et al., 1993; Zens and Ney, 2004) for SMS text normalization. In this section, we discuss the three components of our method: modeling, training and decoding for SMS text normalization.

4.1 Basic Word-based Model

The SMS normalization model is based on the source channel model (Shannon, 1948). Assuming that an English sentence *e*, of length N is "corrupted" by a noisy channel to produce a SMS message *s*, of length M, the English sentence *e*, could be recovered through a posteriori distribution for a channel target text given the source text $P(s \mid e)$, and a prior distribution for the channel source text $P(e)$.

$$\begin{aligned} \hat{e}_1^N &= \arg\max_{e_1^N} \left\{ P(e_1^N \mid s_1^M) \right\} \\ &= \arg\max_{e_1^N} \left\{ P(s_1^M \mid e_1^N) \bullet P(e_1^N) \right\} \end{aligned} \tag{1}$$

Assuming that one SMS word is mapped exactly to one English word in the channel model $P(s \mid e)$ under an alignment A, we need to consider only two types of probabilities: the alignment probabilities denoted by $P(m \mid a_m)$ and the lexicon mapping probabilities denoted by $P(s_m \mid e_{a_m})$ (Brown et al. 1993). The channel model can be written as in the following equation where *m* is the position of a word in *s* and a_m its alignment in e.

$$\begin{aligned} P(s_1^M \mid e_1^N) &= \sum_A P(s_1^M, A \mid e_1^N) \\ &= \sum_A P(A \mid e_1^N) \bullet P(s_1^M \mid A, e_1^N) \\ &\approx \sum_A \left(\prod_{m=1}^{M} \left\{ P(m \mid a_m) \bullet P(s_m \mid e_{a_m}) \right\} \right) \end{aligned} \tag{2}$$

If we include the word "null" in the English vocabulary, the above model can fully address the deletion and substitution transformations, but inadequate to address the insertion transformation. For example, the lingoes *"duno"*, *"ysnite"* have to be normalized using an insertion transformation to become *"don't know"* and *"yesterday night"*. Moreover, we also want the normalization to have better lexical affinity and linguistic equivalent, thus we extend the model to allow many words to many words alignment, allowing a sequence of SMS words to be normalized to a sequence of contiguous English words. We call this updated model a phrase-based normalization model.

4.2 Phrase-based Model

Given an English sentence *e* and SMS sentence *s*, if we assume that *e* can be decomposed into K phrases with a segmentation T, such that each phrase $\tilde{e}_k$ in *e* can be corresponded with one phrase $\tilde{s}_k$ in *s*, we have $e_1^N = \tilde{e}_1 \ldots \tilde{e}_k \ldots \tilde{e}_K$ and $s_1^M = \tilde{s}_1 \ldots \tilde{s}_k \ldots \tilde{s}_K$. The channel model can be rewritten in equation (3).

$$\begin{aligned} P(s_1^M \mid e_1^N) &= \sum_T P(s_1^M, T \mid e_1^N) \\ &= \sum_T P(T \mid e_1^N) \bullet P(s_1^M \mid T, e_1^N) \\ &= \sum_T P(T \mid e_1^N) \bullet P(\tilde{s}_1^K \mid \tilde{e}_1^K) \\ &\approx \max_T \left\{ P(T \mid e_1^N) \bullet P(\tilde{s}_1^K \mid \tilde{e}_1^K) \right\} \end{aligned} \tag{3}$$

This is the basic function of the channel model for the phrase-based SMS normalization model, where we used the maximum approximation for the sum over all segmentations. Then we further decompose the probability $P(\tilde{s}_1^K \mid \tilde{e}_1^K)$ using a phrase alignment $\tilde{A}$ as done in the previous word-based model.

$$\begin{aligned} P(\tilde{s}_1^K \mid \tilde{e}_1^K) &= \sum_{\tilde{A}} P(\tilde{s}_1^K, \tilde{A} \mid \tilde{e}_1^K) \\ &= \sum_{\tilde{A}} \left\{ P(\tilde{A} \mid \tilde{e}_1^K) \bullet P(\tilde{s}_1^K \mid \tilde{A}, \tilde{e}_1^K) \right\} \\ &= \sum_{\tilde{A}} \left(\prod_{k=1}^{K} \left\{ P(k \mid \tilde{a}_k) \bullet P(\tilde{s}_k \mid \tilde{s}_1^{k-1}, \tilde{e}_{\tilde{a}_1}^{\tilde{a}_k}) \right\} \right) \\ &\approx \sum_{\tilde{A}} \left(\prod_{k=1}^{K} \left\{ P(k \mid \tilde{a}_k) \bullet P(\tilde{s}_k \mid \tilde{e}_{\tilde{a}_k}) \right\} \right) \end{aligned} \tag{4}$$

We are now able to model the three transformations through the normalization pair $(\tilde{s}_k, \tilde{e}_{\tilde{a}_k})$,

with the mapping probability $P(\tilde{s}_k \mid \tilde{e}_{\tilde{a}_k})$. The followings show the scenarios in which the three transformations occur.

Insertion	$\lvert\tilde{s}_k\rvert < \lvert\tilde{e}_{\tilde{a}_k}\rvert$
Deletion	$\tilde{e}_{\tilde{a}_k}$ = null
Substitution	$\lvert\tilde{s}_k\rvert = \lvert\tilde{e}_{\tilde{a}_k}\rvert$

The statistics in our training corpus shows that by selecting appropriate phrase segmentation, the position re-ordering at the phrase level occurs rarely. It is not surprising since most of the English words or phrases in normal English text are replaced with lingoes in SMS messages without position change to make SMS text short and concise and to retain the meaning. Thus we need to consider only monotone alignment at phrase level, i.e., $k = \tilde{a}_k$, as in equation (4). In addition, the word-level reordering within phrase is learned during training. Now we can further derive equation (4) as follows:

$$\begin{aligned} P(\tilde{s}_1^K \mid \tilde{e}_1^K) &\approx \sum_{\tilde{A}} \left(\prod_{k=1}^{K} \left\{ P(k \mid \tilde{a}_k) \bullet P(\tilde{s}_k \mid \tilde{e}_{\tilde{a}_k}) \right\} \right) \\ &\approx \prod_{k=1}^{K} P(\tilde{s}_k \mid \tilde{e}_k) \end{aligned} \quad (5)$$

The mapping probability $P(\tilde{s}_k \mid \tilde{e}_k)$ is estimated via relative frequencies as follows:

$$P(\tilde{s}_k \mid \tilde{e}_k) = \frac{N(\tilde{s}_k, \tilde{e}_k)}{\sum_{\tilde{s}_k'} N(\tilde{s}_k', \tilde{e}_k)} \quad (6)$$

Here, $N(\tilde{s}_k, \tilde{e}_k)$ denotes the frequency of the normalization pair $(\tilde{s}_k, \tilde{e}_k)$.

Using a bigram language model and assuming Bayes decision rule, we finally obtain the following search criterion for equation (1).

$$\begin{aligned} \hat{e}_1^N &= \arg\max_{e_1^N} \left\{ P(e_1^N) \bullet P(s_1^M \mid e_1^N) \right\} \\ &\approx \arg\max_{e_1^N} \left\{ \prod_{n=1}^{N} P(e_n \mid e_{n-1}) \right. \\ &\qquad \left. \bullet \max_T \left\{ P(T \mid e_1^N) \bullet \prod_{k=1}^{K} P(\tilde{s}_k \mid \tilde{e}_k) \right\} \right\} \\ &\approx \arg\max_{e_1^N, T} \left\{ \prod_{n=1}^{N} P(e_n \mid e_{n-1}) \bullet \prod_{k=1}^{K} P(\tilde{s}_k \mid \tilde{e}_k) \right\} \end{aligned} \quad (7)$$

For the above equation, we assume the segmentation probability $P(T \mid e_1^N)$ to be constant. Finally, the SMS normalization model consists of two sub-models: **a word-based language model** (LM), characterized by $P(e_n \mid e_{n-1})$ and **a phrase-based lexical mapping model** (channel model), characterized by $P(\tilde{s}_k \mid \tilde{e}_k)$.

4.3 Training Issues

For the phrase-based model training, the sentence-aligned SMS corpus needs to be aligned first at the phrase level. The maximum likelihood approach, through EM algorithm and Viterbi search (Dempster et al., 1977) is employed to infer such an alignment. Here, we make a reasonable assumption on the alignment unit that a single SMS word can be mapped to a sequence of contiguous English words, but not vice verse. The EM algorithm for phrase alignment is illustrated in Figure 1 and is formulated by equation (8).

The Expectation-Maximization Algorithm

(1) Bootstrap initial alignment using orthographic similarities
(2) Expectation: Update the joint probabilities $P(\tilde{s}_k, \tilde{e}_k)$
(3) Maximization: Apply the joint probabilities $P(\tilde{s}_k, \tilde{e}_k)$ to get new alignment using Viterbi search algorithm
(4) Repeat (2) to (3) until alignment converges
(5) Derive normalization pairs from final alignment

Figure 1. Phrase Alignment Using EM Algorithm

$$\hat{\gamma}_{<\tilde{s}_k, \tilde{e}_k>} = \arg\max_{\hat{\gamma}_{<\tilde{s}_k, \tilde{e}_k>}} \prod_{k=1}^{K} P(\tilde{s}_k, \tilde{e}_k \mid s_1^M, e_1^N) \quad (8)$$

The alignment process given in equation (8) is different from that of normalization given in equation (7) in that, here we have an aligned input sentence pair, s_1^M and e_1^N. The alignment process is just to find the alignment segmentation $\hat{\gamma}_{<\tilde{s}_k, \tilde{e}_k>} = <\tilde{s}_k, \tilde{e}_k>_{k=1,K}$ between the two sentences that maximizes the joint probability. Therefore, in step (2) of the EM algorithm given at Figure 1, only the joint probabilities $P(\tilde{s}_k, \tilde{e}_k)$ are involved and updated.

Since EM may fall into local optimization, in order to speed up convergence and find a nearly global optimization, a string matching technique is exploited at the initialization step to identify the most probable normalization pairs. The or-

thographic similarities captured by edit distance and a SMS lingo dictionary[3] which contains the commonly used short-forms are first used to establish phrase mapping boundary candidates. Heuristics are then exploited to match tokens within the pairs of boundary candidates by trying to combine consecutive tokens within the boundary candidates if the numbers of tokens do not agree.

Finally, a filtering process is carried out to manually remove the low-frequency noisy alignment pairs. Table 4 shows some of the extracted normalization pairs. As can be seen from the table, our algorithm discovers ambiguous mappings automatically that are otherwise missing from most of the lingo dictionary.

$(\tilde{s},\tilde{e})$	$\log P(\tilde{s}\mid\tilde{e})$
(2, 2)	0
(2, to)	-0.579466
(2, too)	-0.897016
(2, null)	-2.97058
(4, 4)	0
(4, for)	-0.431364
(4, null)	-3.27161
(w, who are)	-0.477121
(w, with)	-0.764065
(w, who)	-1.83885
(dat, that)	-0.726999
(dat, date)	-0.845098
(tmr, tomorrow)	-0.341514

Table 4. Examples of normalization pairs

Given the phrase-aligned SMS corpus, the lexical mapping model, characterized by $P(\tilde{s}_k \mid \tilde{e}_k)$, is easily to be trained using equation (6). Our n-gram LM $P(e_n \mid e_{n-1})$ is trained on English Gigaword provided by LDC using SRILM language modeling toolkit (Stolcke, 2002). Backoff smoothing (Jelinek, 1991) is used to adjust and assign a non-zero probability to the unseen words to address data sparseness.

4.4 Monotone Search

Given an input s, the search, characterized in equation (7), is to find a sentence e that maximizes $P(s \mid e)\bullet P(e)$ using the normalization model. In this paper, the maximization problem in equation (7) is solved using a monotone search, implemented as a Viterbi search through dynamic programming.

5 Experiments

The aim of our experiment is to verify the effectiveness of the proposed statistical model for SMS normalization and the impact of SMS normalization on MT.

A set of 5000 parallel SMS messages, which consists of raw (un-normalized) SMS messages and reference messages manually prepared by two project members with inter-normalization agreement checked, was prepared for training and testing. For evaluation, we use IBM's BLEU score (Papineni et al., 2002) to measure the performance of the SMS normalization. BLEU score measures the similarity between two sentences using n-gram statistics with a penalty for too short sentences, which is already widely-used in MT evaluation.

Setup	**BLEU score (3-gram)**
Raw SMS without Normalization	0.5784
Dictionary Look-up plus Frequency	0.6958
Bi-gram Language Model Only	0.7086

Table 5. Performance of different setups of the baseline experiments on the 5000 parallel SMS messages

5.1 Baseline Experiments: Simple SMS Lingo Dictionary Look-up and Using Language Model Only

The baseline experiment is to moderate the texts using a lingo dictionary comprises 142 normalization pairs, which is also used in bootstrapping the phrase alignment learning process.

Table 5 compares the performance of the different setups of the baseline experiments. We first measure the complexity of the SMS normalization task by directly computing the similarity between the raw SMS text and the normalized English text. The 1st row of Table 5 reports the similarity as 0.5784 in BLEU score, which implies that there are quite a number of English word 3-gram that are common in the raw and normalized messages. The 2nd experiment is carried out using only simple dictionary look-up.

[3] The entries are collected from various websites such as http://www.handphones.info/sms-dictionary/sms-lingo.php, and http://www.funsms.net/sms_dictionary.htm, etc.

Lexical ambiguity is addressed by selecting the highest-frequency normalization candidate, i.e., only unigram LM is used. The performance of the 2nd experiment is 0.6958 in BLEU score. It suggests that the lingo dictionary plus the unigram LM is very useful for SMS normalization. Finally we carry out the 3rd experiment using dictionary look-up plus bi-gram LM. Only a slight improvement of 0.0128 (0.7086-0.6958) is obtained. This is largely because the English words in the lingo dictionary are mostly high-frequency and commonly-used. Thus bi-gram does not show much more discriminative ability than unigram without the help of the phrase-based lexical mapping model.

5.2 Using Phrase-based Model

We then conducted the experiment using the proposed method (Bi-gram LM plus a phrase-based lexical mapping model) through a five-fold cross validation on the 5000 parallel SMS messages. Table 6 shows the results. An average score of 0.8070 is obtained. Compared with the baseline performance in Table 5, the improvement is very significant. It suggests that the phrase-based lexical mapping model is very useful and our method is effective for SMS text normalization. Figure 2 is the learning curve. It shows that our algorithm converges when training data is increased to 3000 SMS parallel messages. This suggests that our collected corpus is representative and enough for training our model. Table 7 illustrates some examples of the normalization results.

5-fold cross validation	BLEU score (3-gram)
Setup 1	0.8023
Setup 2	0.8236
Setup 3	0.8071
Setup 4	0.8113
Setup 5	0.7908
Ave.	0.8070

Table 6. Normalization results for 5-fold cross validation test

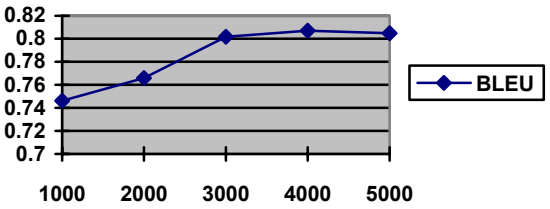


Figure 2. Learning Curve

Experimental result analysis reveals that the strength of our model is in its ability to disambiguate mapping as in "*2*" to "*two*" or "*to*" and "*w*" to "w*ith*" or "*who*". Error analysis shows that the challenge of the model lies in the proper insertion of subject pronoun and auxiliary or copula verb, which serves to give further semantic information about the main verb, however this requires significant context understanding. For example, a message such as "*u smart*" gives little clues on whether it should be normalized to "*Are you smart*?" or "*You are smart*." unless the full conversation is studied.

Takako w r u?
Takako who are you?
Im in ns, lik soccer, clubbin hangin w frenz! Wat bout u mee?
I'm in ns, like soccer, clubbing hanging with friends! What about you?
fancy getting excited w others' boredom
Fancy getting excited with others' boredom
If u ask me b4 he ask me then i'll go out w u all lor. N u still can act so real.
If you ask me before he asked me then I'll go out with you all. And you still can act so real.
Doing nothing, then u not having dinner w us?
Doing nothing, then you do not having dinner with us?
Aiyar sorry lor forgot 2 tell u... Mtg at 2 pm.
Sorry forgot to tell you... Meeting at two pm.
tat's y I said it's bad dat all e gals know u... Wat u doing now?
That's why I said it's bad that all the girls know you... What you doing now?

Table 7. Examples of Normalization Results

5.3 Effect on English-Chinese MT

An experiment was also conducted to study the effect of normalization on MT using 402 messages randomly selected from the text corpus. We compare three types of SMS message: raw SMS messages, normalized messages using simple dictionary look-up and normalized messages using our method. The messages are passed to two different English-to-Chinese translation systems provided by Systran[4] and Institute for Infocomm Research[5](I^{2}R) separately to produce three sets of translation output. The translation quality is measured using 3-gram cumulative BLEU score against two reference messages. 3-gram is

[4] http://www.systranet.com/systran/net
[5] http://nlp.i2r.a-star.edu.sg/techtransfer.html

used as most of the messages are short with average length of seven words. Table 8 shows the details of the BLEU scores. We obtain an average of 0.3770 BLEU score for normalized messages against 0.1926 for raw messages. The significant performance improvement suggests that preprocessing of normalizing SMS text using our method before MT is an effective way to adapt a general MT system to SMS domain.

	I^2R	**Systran**	**Ave.**
Raw Message	0.2633	0.1219	0.1926
Dict Lookup	0.3485	0.1690	0.2588
Normalization	0.4423	0.3116	0.3770

Table 8. SMS Translation BLEU score with or without SMS normalization

6 Conclusion

In this paper, we study the differences among SMS normalization, general text normalization, spelling check and text paraphrasing, and investigate the different phenomena of SMS messages. We propose a phrase-based statistical method to normalize SMS messages. The method produces messages that collate well with manually normalized messages, achieving 0.8070 BLEU score against 0.6958 baseline score. It also significantly improves SMS translation accuracy from 0.1926 to 0.3770 in BLEU score without adjusting the MT model.

This experiment results provide us with a good indication on the feasibility of using this method in performing the normalization task. We plan to extend the model to incorporate mechanism to handle missing punctuation (which potentially affect MT output and are not being taken care at the moment), and making use of pronunciation information to handle OOV caused by the use of phonetic spelling. A bigger data set will also be used to test the robustness of the system leading to a more accurate alignment and normalization.

References

A.T. Aw, M. Zhang, Z.Z. Fan, P.K. Yeo and J. Su. 2005. *Input Normalization for an English-to-Chinese SMS Translation System.* MT Summit-2005

S. Bangalore, V. Murdock and G. Riccardi. 2002. *Bootstrapping Bilingual Data using Consensus Translation for a Multilingual Instant Messaging System.* COLING-2002

R. Barzilay and K. R. McKeown. 2001. *Extracting paraphrases from a parallel corpus*. ACL-2001

E. Brill and R. C. Moore. 2000. *An Improved Error Model for Noisy Channel Spelling Correction.* ACL-2000

P. F. Brown, S. D. Pietra, V. D. Pietra and R. Mercer. 1993. *The Mathematics of Statistical Machine Translation: Parameter Estimation*. Computational Linguistics: 19(2)

A. Clark. 2003. *Pre-processing very noisy text.* In Proceedings of Workshop on Shallow Processing of Large Corpora, Lancaster, 2003

F. J. Damerau. 1964. *A technique for computer detection and correction of spelling errors*. Communications ACM 7, 171-176

A.P. Dempster, N.M. Laird and D.B. Rubin. 1977. *Maximum likelihood from incomplete data via the EM algorithm*, Journal of the Royal Statistical Society, Series B, Vol. 39, 1-38

A. Golding and D. Roth. 1999. *A Winnow-Based Approach to Spelling Correction*. Machine Learning 34: 107-130

F. Jelinek. 1991. *Self-organized language modeling for speech recognition*. In A. Waibel and K.F. Lee, editors, Readings in Speech Recognition, pages 450-506. Morgan Kaufmann, 1991

M. D. Kernighan, K Church and W. Gale. 1990. *A spelling correction program based on a noisy channel model.* COLING-1990

K. Kukich. 1992. *Techniques for automatically correcting words in text*. ACM Computing Surveys, 24(4):377-439

K. A. Papineni, S. Roukos, T. Ward and W. J. Zhu. 2002. *BLEU : a Method for Automatic Evaluation of Machine Translation*. ACL-2002

P. Koehn, F.J. Och and D. Marcu. 2003. *Statistical Phrase-Based Translation*. HLT-NAACL-2003

C. Shannon. 1948. *A mathematical theory of communication*. Bell System Technical Journal 27(3): 379-423

M. Shimohata and E. Sumita 2002. *Automatic Paraphrasing Based on Parallel Corpus for Normalization*. LREC-2002

R. Sproat, A. Black, S. Chen, S. Kumar, M. Ostendorf and C. Richards. 2001. *Normalization of Non-Standard Words*. Computer Speech and Language, 15(3):287-333

A. Stolcke. 2002. SRILM – *An extensible language modeling toolkit*. ICSLP-2002

K. Toutanova and R. C. Moore. 2002. *Pronunciation Modeling for Improved Spelling Correction.* ACL-2002

R. Zens and H. Ney. 2004. *Improvements in Phrase-Based Statistical MT*. HLT-NAALL-2004

Evaluating the Accuracy of an Unlexicalized Statistical Parser on the PARC DepBank

Ted Briscoe
Computer Laboratory
University of Cambridge

John Carroll
School of Informatics
University of Sussex

Abstract

We evaluate the accuracy of an unlexicalized statistical parser, trained on 4K treebanked sentences from balanced data and tested on the PARC DepBank. We demonstrate that a parser which is competitive in accuracy (without sacrificing processing speed) can be quickly tuned without reliance on large in-domain manually-constructed treebanks. This makes it more practical to use statistical parsers in applications that need access to aspects of predicate-argument structure. The comparison of systems using DepBank is not straightforward, so we extend and validate DepBank and highlight a number of representation and scoring issues for relational evaluation schemes.

1 Introduction

Considerable progress has been made in accurate statistical parsing of realistic texts, yielding rooted, hierarchical and/or relational representations of full sentences. However, much of this progress has been made with systems based on large lexicalized probabilistic context-free like (PCFG-like) models trained on the Wall Street Journal (WSJ) subset of the Penn TreeBank (PTB). Evaluation of these systems has been mostly in terms of the PARSEVAL scheme using tree similarity measures of (labelled) precision and recall and crossing bracket rate applied to section 23 of the WSJ PTB. (See e.g. Collins (1999) for detailed exposition of one such very fruitful line of research.)

We evaluate the comparative accuracy of an unlexicalized statistical parser trained on a smaller treebank and tested on a subset of section 23 of the WSJ using a relational evaluation scheme. We demonstrate that a parser which is competitive in accuracy (without sacrificing processing speed) can be quickly developed without reliance on large in-domain manually-constructed treebanks. This makes it more practical to use statistical parsers in diverse applications needing access to aspects of predicate-argument structure.

We define a lexicalized statistical parser as one which utilizes probabilistic parameters concerning lexical subcategorization and/or bilexical relations over tree configurations. Current lexicalized statistical parsers developed, trained and tested on PTB achieve a labelled F_1-score – the harmonic mean of labelled precision and recall – of around 90%. Klein and Manning (2003) argue that such results represent about 4% absolute improvement over a carefully constructed unlexicalized PCFG-like model trained and tested in the same manner.[1] Gildea (2001) shows that WSJ-derived bilexical parameters in Collins' (1999) Model 1 parser contribute less than 1% to parse selection accuracy when test data is in the same domain, and yield no improvement for test data selected from the Brown Corpus. Bikel (2004) shows that, in Collins' (1999) Model 2, bilexical parameters contribute less than 0.5% to accuracy on in-domain data while lexical subcategorization-like parameters contribute just over 1%.

Several alternative relational evaluation schemes have been developed (e.g. Carroll *et al.*, 1998; Lin, 1998). However, until recently, no WSJ data has been carefully annotated to support relational evaluation. King *et al.* (2003) describe the PARC 700 Dependency Bank (hereinafter *DepBank*), which consists of 700 WSJ sentences randomly drawn from section 23. These sentences have been annotated with syntactic features and with bilexical head-dependent relations derived from the F-structure representation of Lexical Functional Grammar (LFG). DepBank facilitates

[1] Klein and Manning retained some functional tag information from PTB, so it could be argued that their model remains 'mildly' lexicalized since functional tags encode some subcategorization information.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 41–48,
Sydney, July 2006.

comparison of PCFG-like statistical parsers developed from the PTB with other parsers whose output is not designed to yield PTB-style trees, using an evaluation which is closer to the protypical parsing task of recovering predicate-argument structure.

Kaplan *et al.* (2004) compare the accuracy and speed of the PARC XLE Parser to Collins' Model 3 parser. They develop transformation rules for both, designed to map native output to a subset of the features and relations in DepBank. They compare performance of a grammatically cut-down and complete version of the XLE parser to the publically available version of Collins' parser. One fifth of DepBank is held out to optimize the speed and accuracy of the three systems. They conclude from the results of these experiments that the cut-down XLE parser is two-thirds the speed of Collins' Model 3 but 12% more accurate, while the complete XLE system is 20% more accurate but five times slower. F_1-score percentages range from the mid- to high-70s, suggesting that the relational evaluation is harder than PARSEVAL.

Both Collins' Model 3 and the XLE Parser use lexicalized models for parse selection trained on the rest of the WSJ PTB. Therefore, although Kaplan *et al.* demonstrate an improvement in accuracy at some cost to speed, there remain questions concerning viability for applications, at some remove from the financial news domain, for which substantial treebanks are not available. The parser we deploy, like the XLE one, is based on a manually-defined feature-based unification grammar. However, the approach is somewhat different, making maximal use of more generic structural rather than lexical information, both within the grammar and the probabilistic parse selection model. Here we compare the accuracy of our parser with Kaplan *et al.*'s results, by repeating their experiment with our parser. This comparison is not straightforward, given both the system-specific nature of some of the annotation in DepBank and the scoring reported. We, therefore, extend DepBank with a set of grammatical relations derived from our own system output and highlight how issues of representation and scoring can affect results and their interpretation.

In §2, we describe our development methodology and the resulting system in greater detail. §3 describes the extended Depbank that we have developed and motivates our additions. §2.4 discusses how we trained and tuned our current system and describes our limited use of information derived from WSJ text. §4 details the various experiments undertaken with the extended DepBank and gives detailed results. §5 discusses these results and proposes further lines of research.

2 Unlexicalized Statistical Parsing

2.1 System Architecture

Both the XLE system and Collins' Model 3 preprocess textual input before parsing. Similarly, our baseline system consists of a pipeline of modules. First, text is tokenized using a deterministic finite-state transducer. Second, tokens are part-of-speech and punctuation (PoS) tagged using a 1st-order Hidden Markov Model (HMM) utilizing a lexicon of just over 50K words and an unknown word handling module. Third, deterministic morphological analysis is performed on each token-tag pair with a finite-state transducer. Fourth, the lattice of lemma-affix-tags is parsed using a grammar over such tags. Finally, the *n*-best parses are computed from the parse forest using a probabilistic parse selection model conditioned on the structural parse context. The output of the parser can be displayed as syntactic trees, and/or factored into a sequence of bilexical grammatical relations (GRs) between lexical heads and their dependents.

The full system can be extended in a variety of ways – for example, by pruning PoS tags but allowing multiple tag possibilities per word as input to the parser, by incorporating lexical subcategorization into parse selection, by computing GR weights based on the proportion and probability of the *n*-best analyses yielding them, and so forth – broadly trading accuracy and greater domain-dependence against speed and reduced sensitivity to domain-specific lexical behaviour (Briscoe and Carroll, 2002; Carroll and Briscoe, 2002; Watson *et al.*, 2005; Watson, 2006). However, in this paper we focus exclusively on the baseline unlexicalized system.

2.2 Grammar Development

The grammar is expressed in a feature-based, unification formalism. There are currently 676 phrase structure rule schemata, 15 feature propagation rules, 30 default feature value rules, 22 category expansion rules and 41 feature types which together define 1124 compiled phrase structure rules in which categories are represented as sets of fea-

tures, that is, attribute-value pairs, possibly with variable values, possibly bound between mother and one or more daughter categories. 142 of the phrase structure schemata are manually identified as peripheral rather than core rules of English grammar. Categories are matched using fixed-arity term unification at parse time.

The lexical categories of the grammar consist of feature-based descriptions of the 149 PoS tags and 13 punctuation tags (a subset of the CLAWS tagset, see e.g. Sampson, 1995) which constitute the preterminals of the grammar. The number of distinct lexical categories associated with each preterminal varies from 1 for some function words through to around 35 as, for instance, tags for main verbs are associated with a VSUBCAT attribute taking 33 possible values. The grammar is designed to enumerate possible valencies for predicates by including separate rules for each pattern of possible complementation in English. The distinction between arguments and adjuncts is expressed by adjunction of adjuncts to maximal projections ($XP \rightarrow XP\ Adjunct$) as opposed to government of arguments (i.e. arguments are sisters within *X1* projections; $X1 \rightarrow X0\ Arg1 \ldots ArgN$).

Each phrase structure schema is associated with one or more GR specifications which can be conditioned on feature values instantiated at parse time and which yield a rule-to-rule mapping from local trees to GRs. The set of GRs associated with a given derivation define a connected, directed graph with individual nodes representing lemma-affix-tags and arcs representing named grammatical relations. The encoding of this mapping within the grammar is similar to that of F-structure mapping in LFG. However, the connected graph is not constructed and completeness and coherence constraints are not used to filter the phrase structure derivation space.

The grammar finds at least one parse rooted in the start category for 85% of the Susanne treebank, a 140K word balanced subset of the Brown Corpus, which we have used for development (Sampson, 1995). Much of the remaining data consists of phrasal fragments marked as independent text sentences, for example in dialogue. Grammatical coverage includes the majority of construction types of English, however the handling of some unbounded dependency constructions, particularly comparatives and equatives, is limited because of the lack of fine-grained subcategorization information in the PoS tags and by the need to balance depth of analysis against the size of the derivation space. On the Susanne corpus, the geometric mean of the number of analyses for a sentence of length n is 1.31^n. The microaveraged F_1-score for GR extraction on held-out data from Susanne is 76.5% (see section 4.2 for details of the evaluation scheme).

The system has been used to analyse about 150 million words of English text drawn primarily from the PTB, TREC, BNC, and Reuters RCV1 datasets in connection with a variety of projects. The grammar and PoS tagger lexicon have been incrementally improved by manually examining cases of parse failure on these datasets. However, the effort invested amounts to a few days' effort for each new dataset as opposed to the main grammar development effort, centred on Susanne, which has extended over some years and now amounts to about 2 years' effort (see Briscoe, 2006 for further details).

2.3 Parser

To build the parsing module, the unification grammar is automatically converted into an atomic-categoried context free 'backbone', and a non-deterministic LALR(1) table is constructed from this, which is used to drive the parser. The residue of features not incorporated into the backbone are unified on each rule application (reduce action). In practice, the parser takes average time roughly quadratic in the length of the input to create a packed parse forest represented as a graph-structured stack. The statistical disambiguation phase is trained on Susanne treebank bracketings, producing a probabilistic generalized LALR(1) parser (e.g. Inui *et al.*, 1997) which associates probabilities with alternative actions in the LR table.

The parser is passed as input the sequence of most probable lemma-affix-tags found by the tagger. During parsing, probabilities are assigned to subanalyses based on the the LR table actions that derived them. The n-best (i.e. most probable) parses are extracted by a dynamic programming procedure over subanalyses (represented by nodes in the parse forest). The search is efficient since probabilities are associated with single nodes in the parse forest and no weight function over ancestor or sibling nodes is needed. Probabilities capture structural context, since nodes in

the parse forest partially encode a configuration of the graph-structured stack and lookahead symbol, so that, unlike a standard PCFG, the model discriminates between derivations which only differ in the order of application of the same rules and also conditions rule application on the PoS tag of the lookahead token.

When there is no parse rooted in the start category, the parser returns a connected sequence of partial parses which covers the input based on subanalysis probability and a preference for longer and non-lexical subanalysis combinations (e.g. Kiefer *et al.*, 1999). In these cases, the GR graph will not be fully connected.

2.4 Tuning and Training Method

The HMM tagger has been trained on 3M words of balanced text drawn from the LOB, BNC and Susanne corpora, which are available with hand-corrected CLAWS tags. The parser has been trained from 1.9K trees for sentences from Susanne that were interactively parsed to manually obtain the correct derivation, and also from 2.1K further sentences with unlabelled bracketings derived from the Susanne treebank. These bracketings guide the parser to one or possibly several closely-matching derivations and these are used to derive probabilities for the LR table using (weighted) Laplace estimation. Actions in the table involving rules marked as peripheral are assigned a uniform low prior probability to ensure that derivations involving such rules are consistently lower ranked than those involving only core rules.

To improve performance on WSJ text, we examined some parse failures from sections other than section 23 to identify patterns of consistent failure. We then manually modified and extended the grammar with a further 6 rules, mostly to handle cases of indirect and direct quotation that are very common in this dataset. This involved 3 days' work. Once completed, the parser was retrained on the original data. A subsequent limited inspection of top-ranked parses led us to disable 6 existing rules which applied too freely to the WSJ text; these were designed to analyse auxiliary ellipsis which appears to be rare in this genre. We also catalogued incorrect PoS tags from WSJ parse failures and manually modified the tagger lexicon where appropriate. These modifications mostly consisted of adjusting lexical probabilities of extant entries with highly-skewed distributions. We also added some tags to extant entries for infrequent words. These modifications took a further day. The tag transition probabilities were not reestimated. Thus, we have made no use of the PTB itself and only limited use of WSJ text.

This method of grammar and lexicon development incrementally improves the overall performance of the system averaged across all the datasets that it has been applied to. It is very likely that retraining the PoS tagger on the WSJ and retraining the parser using PTB would yield a system which would perform more effectively on DepBank. However, one of our goals is to demonstrate that an unlexicalized parser trained on a modest amount of annotated text from other sources, coupled to a tagger also trained on generic, balanced data, can perform competitively with systems which have been (almost) entirely developed and trained using PTB, whether or not these systems deploy hand-crafted grammars or ones derived automatically from treebanks.

3 Extending and Validating DepBank

DepBank was constructed by parsing the selected section 23 WSJ sentences with the XLE system and outputting syntactic features and bilexical relations from the F-structure found by the parser. These features and relations were subsequently checked, corrected and extended interactively with the aid of software tools (King *et al.*, 2003).

The choice of relations and features is based quite closely on LFG and, in fact, overlaps substantially with the GR output of our parser. Figure 1 illustrates some DepBank annotations used in the experiment reported by Kaplan *et al.* and our hand-corrected GR output for the example *Ten of the nation's governors meanwhile called on the justices to reject efforts to limit abortions*. We have kept the GR representation simpler and more readable by suppressing lemmatization, token numbering and PoS tags, but have left the DepBank annotations unmodified.

The example illustrates some differences between the schemes. For instance, the **subj** and **ncsubj** relations overlap as both annotations contain such a relation between *call(ed)* and *Ten*), but the GR annotation also includes this relation between *limit* and *effort(s)* and *reject* and *justice(s)*, while DepBank links these two verbs to a variable **pro**. This reflects a difference of philosophy about

DepBank:
```
obl(call~0, on~2)
stmt_type(call~0, declarative)
subj(call~0, ten~1)
tense(call~0, past)
number_type(ten~1, cardinal)
obl(ten~1, governor~35)
obj(on~2, justice~30)
obj(limit~7, abortion~15)
subj(limit~7, pro~21)
obj(reject~8, effort~10)
subj(reject~8, pro~27)
adegree(meanwhile~9, positive)
num(effort~10, pl)
xcomp(effort~10, limit~7)
```

GR:
```
(ncsubj called Ten _)
(ncsubj reject justices _)
(ncsubj limit efforts _)
(iobj called on)
(xcomp to called reject)
(dobj reject efforts)
(xmod to efforts limit)
(dobj limit abortions)
(dobj on justices)
(det justices the)
(ta bal governors meanwhile)
(ncmod poss governors nation)
(iobj Ten of)
(dobj of governors)
(det nation the)
```

Figure 1: DepBank and GR annotations.

resolution of such 'understood' relations in different constructions. Viewed as output appropriate to specific applications, either approach is justifiable. However, for evaluation, these DepBank relations add little or no information not already specified by the **xcomp** relations in which these verbs also appear as dependents. On the other hand, DepBank includes an **adjunct** relation between *meanwhile* and *call(ed)*, while the GR annotation treats *meanwhile* as a text adjunct (**ta**) of *governors*, delimited by balanced commas, following Nunberg's (1990) text grammar but conveying less information here.

There are also issues of incompatible tokenization and lemmatization between the systems and of differing syntactic annotation of similar information, which lead to problems mapping between our GR output and the current DepBank. Finally, differences in the linguistic intuitions of the annotators and errors of commission or omission on both sides can only be uncovered by manual comparison of output (e.g. **xmod** vs. **xcomp** for *limit efforts* above). Thus we reannotated the DepBank sentences with GRs using our current system, and then corrected and extended this annotation utilizing a software tool to highlight differences between the extant annotations and our own.[2] This exercise, though time-consuming, uncovered problems in both annotations, and yields a doubly-annotated and potentially more valuable resource in which annotation disagreements over complex attachment decisions, for instance, can be inspected.

The GR scheme includes one feature in DepBank (**passive**), several splits of relations in DepBank, such as **adjunct**, adds some of DepBank's featural information, such as **subord_form**, as a subtype slot of a relation (**ccomp**), merges DepBank's **obl**ique with **iobj**, and so forth. But it does not explicitly include all the features of DepBank or even of the reduced set of semantically-relevant features used in the experiments and evaluation reported in Kaplan *et al.*. Most of these features can be computed from the full GR representation of bilexical relations between numbered lemma-affix-tags output by the parser. For instance, **num** features, such as the plurality of *justices* in the example, can be computed from the full **det** GR `(det justice+s_NN2:4 the_AT:3)` based on the CLAWS tag (NN**2** indicating 'plural') selected for output. The few features that cannot be computed from GRs and CLAWS tags directly, such as **stmt_type**, could be computed from the derivation tree.

4 Experiments

4.1 Experimental Design

We selected the same 560 sentences as test data as Kaplan *et al.*, and all modifications that we made to our system (see §2.4) were made on the basis of (very limited) information from other sections of WSJ text.[3] We have made no use of the further 140 held out sentences in DepBank. The results we report below are derived by choosing the most probable tag for each word returned by the PoS tagger and by choosing the unweighted GR set returned for the most probable parse with no lexical information guiding parse ranking.

4.2 Results

Our parser produced rooted sentential analyses for 84% of the test items; actual coverage is higher

[2] The new version of DepBank along with evaluation software is included in the current RASP distribution: www.informatics.susx.ac.uk/research/nlp/rasp

[3] The PARC group kindly supplied us with the experimental data files they used to facilitate accurate reproduction of this experiment.

Relation	Precision	Recall	F_1	*P R F_1 Relation*
mod	75.4	71.2	73.3	
ncmod	72.9	67.9	70.3	
xmod	47.7	45.5	46.6	
cmod	51.4	31.6	39.1	
pmod	30.8	33.3	32.0	
det	88.7	91.1	89.9	
arg_mod	71.9	67.9	69.9	
arg	76.0	73.4	74.6	
subj	80.1	66.6	72.7	*73 73 73*
ncsubj	80.5	66.8	73.0	
xsubj	50.0	28.6	36.4	
csubj	20.0	50.0	28.6	
subj_or_dobj	82.1	74.9	78.4	
comp	74.5	76.4	75.5	
obj	78.4	77.9	78.1	
dobj	83.4	81.4	82.4	*75 75 75 obj*
obj2	24.2	38.1	29.6	*42 36 39 obj-theta*
iobj	68.2	68.1	68.2	*64 83 72 obl*
clausal	63.5	71.6	67.3	
xcomp	75.0	76.4	75.7	*74 73 74*
ccomp	51.2	65.6	57.5	*78 64 70 comp*
pcomp	69.6	66.7	68.1	
aux	92.8	90.5	91.6	
conj	71.7	71.0	71.4	*68 62 65*
ta	39.1	48.2	43.2	
passive	93.6	70.6	80.5	*80 83 82*
adegree	89.2	72.4	79.9	*81 72 76*
coord_form	92.3	85.7	88.9	*92 93 93*
num	92.2	89.8	91.0	*86 87 86*
number_type	86.3	92.7	89.4	*96 95 96*
precoord_form	100.0	16.7	28.6	*100 50 67*
pron_form	92.1	91.9	92.0	*88 89 89*
prt_form	71.1	58.7	64.3	*72 65 68*
subord_form	60.7	48.1	53.6	
macroaverage	69.0	63.4	66.1	
microaverage	81.5	78.1	79.7	*80 79 79*

Table 1: Accuracy of our parser, and where roughly comparable, the XLE as reported by King *et al.*

than this since some of the test sentences are elliptical or fragmentary, but in many cases are recognized as single complete constituents. Kaplan *et al.* report that the complete XLE system finds rooted analyses for 79% of section 23 of the WSJ but do not report coverage just for the test sentences. The XLE parser uses several performance optimizations which mean that processing of subanalyses in longer sentences can be curtailed or preempted, so that it is not clear what proportion of the remaining data is outside grammatical coverage.

Table 1 shows accuracy results for each individual relation and feature, starting with the GR bilexical relations in the extended DepBank and followed by most DepBank features reported by Kaplan *et al.*, and finally overall macro- and microaverages. The macroaverage is calculated by taking the average of each measure for each individual relation and feature; the microaverage measures are calculated from the counts for all relations and features.[4] Indentation of GRs shows degree of specificity of the relation. Thus, **mod** scores are microaveraged over the counts for the five fully specified modifier relations listed immediately after it in Table 1. This allows comparison of overall accuracy on modifiers with, for instance overall accuracy on **arg**uments. Figures in italics to the right are discussed in the next section.

Kaplan *et al.*'s microaveraged scores for Collins' Model 3 and the cut-down and complete versions of the XLE parser are given in Table 2, along with the microaveraged scores for our parser from Table 1. Our system's accuracy results (evaluated on the reannotated DepBank) are better than those for Collins and the cut-down XLE, and very similar overall to the complete XLE (evaluated on DepBank). Speed of processing is also very competitive.[5] These results demonstrate that a statistical parser with roughly state-of-the-art accuracy can be constructed without the need for large in-domain treebanks. However, the performance of the system, as measured by microraveraged F_1-score on GR extraction alone, has declined by 2.7% over the held-out Susanne data, so even the unlexicalized parser is by no means domain-independent.

4.3 Evaluation Issues

The DepBank **num** feature on nouns is evaluated by Kaplan *et al.* on the grounds that it is semantically-relevant for applications. There are over 5K **num** features in DepBank so the overall microaveraged scores for a system will be significantly affected by accuracy on **num**. We expected our system, which incorporates a tagger with good empirical (97.1%) accuracy on the test data, to recover this feature with 95% accuracy or better, as it will correlate with tags NNx1 and NNx2 (where 'x' represents zero or more capitals in the CLAWS

[4] We did not compute the remaining DepBank features **stmt_type**, **tense**, **prog** or **perf** as these rely on information that can only be extracted from the derivation tree rather than the GR set.

[5] Processing time for our system was 61 seconds on one 2.2GHz Opteron CPU (comprising tokenization, tagging, morphology, and parsing, including module startup overheads). Allowing for slightly different CPUs, this is 2.5–10 times faster than the Collins and XLE parsers, as reported by Kaplan *et al.*

System	Eval corpus	Precision	Recall	F_1
Collins	DepBank	78.3	71.2	74.6
Cut-down XLE	DepBank	79.1	76.2	77.6
Complete XLE	DepBank	79.4	79.8	79.6
Our system	DepBank/GR	81.5	78.1	79.7

Table 2: Microaveraged overall scores from Kaplan *et al.* and for our system.

tagset). However, DepBank treats the majority of prenominal modifiers as adjectives rather than nouns and, therefore, associates them with an **adegree** rather than a **num** feature. The PoS tag selected depends primarily on the relative lexical probabilities of each tag for a given lexical item recorded in the tagger lexicon. But, regardless of this lexical decision, the correct GR is recovered, and neither **adegree(positive)** or **num(sg)** add anything semantically-relevant when the lexical item is a nominal premodifier. A strategy which only provided a **num** feature for nominal heads would be both more semantically-relevant and would also yield higher precision (95.2%). However, recall (48.4%) then suffers against DepBank as noun premodifiers have a **num** feature. Therefore, in the results presented in Table 1 we have not counted cases where either DepBank or our system assign a premodifier **adegree(positive)** or **num(sg)**.

There are similar issues with other DepBank features and relations. For instance, the form of a subordinator with clausal complements is annotated as a relation between verb and subordinator, while there is a separate **comp** relation between verb and complement head. The GR representation adds the subordinator as a subtype of **ccomp** recording essentially identical information in a single relation. So evaluation scores based on aggregated counts of correct decisions will be doubled for a system which structures this information as in DepBank. However, reproducing the exact DepBank **subord_form** relation from the GR **ccomp** one is non-trivial because DepBank treats modal auxiliaries as syntactic heads while the GR-scheme treats the main verb as head in all **ccomp** relations. We have not attempted to compensate for any further such discrepancies other than the one discussed in the previous paragraph. However, we do believe that they collectively damage scores for our system.

As King *et al.* note, it is difficult to identify such informational redundancies to avoid double-counting and to eradicate all system specific biases. However, reporting precision, recall and F_1-scores for each relation and feature separately and microaveraging these scores on the basis of a hierarchy, as in our GR scheme, ameliorates many of these problems and gives a better indication of the strengths and weaknesses of a particular parser, which may also be useful in a decision about its usefulness for a specific application. Unfortunately, Kaplan *et al.* do not report their results broken down by relation or feature so it is not possible, for example, on the basis of the arguments made above, to choose to compare the performance of our system on **ccomp** to theirs for **comp**, ignoring **subord_form**. King *et al.* do report individual results for selected features and relations from an evaluation of the complete XLE parser on all 700 DepBank sentences with an almost identical overall microaveraged F_1 score of 79.5%, suggesting that these results provide a reasonably accurate idea of the XLE parser's relative performance on different features and relations. Where we believe that the information captured by a DepBank feature or relation is roughly comparable to that expressed by a GR in our extended DepBank, we have included King *et al.*'s scores in the rightmost column in Table 1 for comparison purposes. Even if these features and relations were drawn from the same experiment, however, they would still not be *exactly* comparable. For instance, as discussed in §3 nearly half (just over 1K) the DepBank **subj** relations include **pro** as one element, mostly double counting a corresponding **xcomp** relation. On the other hand, our **ta** relation syntactically underspecifies many DepBank **adjunct** relations. Nevertheless, it is possible to see, for instance, that while both parsers perform badly on second objects ours is worse, presumably because of lack of lexical subcategorization information.

5 Conclusions

We have demonstrated that an unlexicalized parser with minimal manual modification for WSJ text – but no tuning of performance to optimize on this dataset alone, and no use of PTB – can achieve accuracy competitive with parsers employing lexicalized statistical models trained on PTB.

We speculate that we achieve these results because our system is engineered to make minimal use of lexical information both in the grammar and in parse ranking, because the grammar has been developed to constrain ambiguity despite this lack of lexical information, and because we can compute the full packed parse forest for all the test sentences efficiently (without sacrificing speed of processing with respect to other statistical parsers). These advantages appear to effectively offset the disadvantage of relying on a coarser, purely structural model for probabilistic parse selection. In future work, we hope to improve the accuracy of the system by adding lexical information to the statistical parse selection component without exploiting in-domain treebanks.

Clearly, more work is needed to enable more accurate, informative, objective and wider comparison of extant parsers. More recent PTB-based parsers show small improvements over Collins' Model 3 using PARSEVAL, while Clark and Curran (2004) and Miyao and Tsujii (2005) report 84% and 86.7% F_1-scores respectively for their own relational evaluations on section 23 of WSJ. However, it is impossible to meaningfully compare these results to those reported here. The reannotated DepBank potentially supports evaluations which score according to the degree of agreement between this and the original annotation and/or development of future consensual versions through collaborative reannotation by the research community. We have also highlighted difficulties for relational evaluation schemes and argued that presenting individual scores for (classes of) relations and features is both more informative and facilitates system comparisons.

6 References

Bikel, D.. 2004. Intricacies of Collins' parsing model, *Computational Linguistics*, 30(4):479–512.

Briscoe, E.J.. 2006. *An introduction to tag sequence grammars and the RASP system parser,* University of Cambridge, Computer Laboratory Technical Report 662.

Briscoe, E.J. and J. Carroll. 2002. Robust accurate statistical annotation of general text. In *Proceedings of the 3rd Int. Conf. on Language Resources and Evaluation (LREC)*, Las Palmas, Gran Canaria. 1499–1504.

Carroll, J. and E.J. Briscoe. 2002. High precision extraction of grammatical relations. In *Proceedings of the 19th Int. Conf. on Computational Linguistics (COLING)*, Taipei, Taiwan. 134–140.

Carroll, J., E. Briscoe and A. Sanfilippo. 1998. Parser evaluation: a survey and a new proposal. In *Proceedings of the 1st International Conference on Language Resources and Evaluation*, Granada, Spain. 447–454.

Clark, S. and J. Curran. 2004. The importance of supertagging for wide-coverage CCG parsing. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-04)*, Geneva, Switzerland. 282–288.

Collins, M.. 1999. *Head-driven Statistical Models for Natural Language Parsing*. PhD Dissertation, Computer and Information Science, University of Pennsylvania.

Gildea, D.. 2001. Corpus variation and parser performance. In *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP'01)*, Pittsburgh, PA.

Inui, K., V. Sornlertlamvanich, H. Tanaka and T. Tokunaga. 1997. A new formalization of probabilistic GLR parsing. In *Proceedings of the 5th International Workshop on Parsing Technologies (IWPT'97)*, Boston, MA. 123–134.

Kaplan, R., S. Riezler, T. H. King, J. Maxwell III, A. Vasserman and R. Crouch. 2004. Speed and accuracy in shallow and deep stochastic parsing. In *Proceedings of the HLT Conference and the 4th Annual Meeting of the North American Chapter of the ACL (HLT-NAACL'04)*, Boston, MA.

Kiefer, B., H-U. Krieger, J. Carroll and R. Malouf. 1999. A bag of useful techniques for efficient and robust parsing. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, University of Maryland. 473–480.

King, T. H., R. Crouch, S. Riezler, M. Dalrymple and R. Kaplan. 2003. The PARC700 Dependency Bank. In *Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora (LINC-03)*, Budapest, Hungary.

Klein, D. and C. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, Sapporo, Japan. 423–430.

Lin, D.. 1998. Dependency-based evaluation of MINIPAR. In *Proceedings of the Workshop at LREC'98 on The Evaluation of Parsing Systems*, Granada, Spain.

Manning, C. and H. Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA.

Miyao, Y. and J. Tsujii. 2005. Probabilistic disambiguation models for wide-coverage HPSG parsing. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, Ann Arbor, MI. 83–90.

Nunberg, G.. 1990. *The Linguistics of Punctuation*. CSLI Lecture Notes 18, Stanford, CA.

Sampson, G.. 1995. *English for the Computer*. Oxford University Press, Oxford, UK.

Watson, R.. 2006. Part-of-speech tagging models for parsing. In *Proceedings of the 9th Conference of Computational Linguistics in the UK (CLUK'06)*, Open University, Milton Keynes.

Watson, R., J. Carroll and E.J. Briscoe. 2005. Efficient extraction of grammatical relations. In *Proceedings of the 9th Int. Workshop on Parsing Technologies (IWPT'05)*, Vancouver, Ca..

N Semantic Classes are Harder than Two

Ben Carterette[*]
CIIR
University of Massachusetts
Amherst, MA 01003
carteret@cs.umass.edu

Rosie Jones
Yahoo! Research
3333 Empire Ave.
Burbank, CA 91504
jonesr@yahoo-inc.com

Wiley Greiner[*]
Los Angeles Software Inc.
1329 Pine Street
Santa Monica, CA 90405
w.greiner@lasoft.com

Cory Barr
Yahoo! Research
3333 Empire Ave.
Burbank, CA 91504
barrc@yahoo-inc.com

Abstract

We show that we can automatically classify semantically related phrases into 10 classes. Classification robustness is improved by training with multiple sources of evidence, including within-document cooccurrence, HTML markup, syntactic relationships in sentences, substitutability in query logs, and string similarity. Our work provides a benchmark for automatic *n-way* classification into WordNet's semantic classes, both on a TREC news corpus and on a corpus of substitutable search query phrases.

1 Introduction

Identifying semantically related phrases has been demonstrated to be useful in information retrieval (Anick, 2003; Terra and Clarke, 2004) and sponsored search (Jones et al., 2006). Work on semantic entailment often includes lexical entailment as a subtask (Dagan et al., 2005).

We draw a distinction between the task of identifying terms which are topically related and identifying the specific semantic class. For example, the terms "dog", "puppy", "canine", "schnauzer", "cat" and "pet" are highly related terms, which can be identified using techniques that include distributional similarity (Lee, 1999) and within-document cooccurrence measures such as pointwise mutual information (Turney et al., 2003). These techniques, however, do not allow us to distinguish the more specific relationships:

- hypernym(dog,puppy)
- hyponym(dog,canine)
- coordinate(dog,cat)

Lexical resources such as WordNet (Miller, 1995) are extremely useful, but are limited by being manually constructed. They do not contain semantic class relationships for the many new terms we encounter in text such as web documents, for example "mp3 player" or "ipod". We can use WordNet as training data for such classification to the extent that the training on pairs found in WordNet and testing on pairs found outside WordNet provides accurate generalization.

We describe a set of features used to train n-way supervised machine-learned classification of semantic classes for arbitrary pairs of phrases. Redundancy in the sources of our feature information means that we are able to provide coverage over an extremely large vocabulary of phrases. We contrast this with techniques that require parsing of natural language sentences (Snow et al., 2005) which, while providing reasonable performance, can only be applied to a restricted vocabulary of phrases cooccuring in sentences.

Our contributions are:

- Demonstration that binary classification removes the difficult cases of classification into closely related semantic classes
- Demonstration that dependency parser paths are inadequate for semantic classification into 7 WordNet classes on TREC news corpora
- A benchmark of 10-class semantic classification over highly substitutable query phrases
- Demonstration that training a classifier using WordNet for labeling does not generalize well to query pairs
- Demonstration that much of the performance in classification can be attained using only

[*]This work was carried out while these authors were at Yahoo! Research.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 49–56,
Sydney, July 2006.

syntactic features

- A learning curve for classification of query phrase pairs that suggests the primary bottleneck is manually labeled training instances: we expect our benchmark to be surpassed.

2 Relation to Previous Work

Snow et al. (2005) demonstrated binary classification of hypernyms and non-hypernyms using WordNet (Miller, 1995) as a source of training labels. Using dependency parse tree paths as features, they were able to generalize from WordNet labelings to human labelings.

Turney et al. (2003) combined features to answer multiple-choice synonym questions from the TOEFL test and verbal analogy questions from the SAT college entrance exam. The multiple-choice questions typically do not consist of multiple closely related terms. A typical example is given by Turney:

- hidden:: (a) laughable (c) ancient (b) veiled (d) revealed

Note that only (b) and (d) are at all related to the term, so the algorithm only needs to distinguish antonyms from synonyms, not synonyms from say hypernyms.

We use as input phrase pairs recorded in query logs that web searchers substitute during search sessions. We find much more closely related phrases:

- hidden:: (a) secret (b) hidden camera (c) hidden cam (d) spy (e) hiden (f) voyeur (g) hide

This set contains a context-dependent synonym, topically related verbs and nouns, and a spelling correction. All of these could cooccur on web pages, so simple cooccurrence statistics may not be sufficient to classify each according to the semantic type.

We show that the techniques used to perform binary semantic classification do not work as well when extended to a full n-way semantic classification. We show that using a variety of features performs better than any feature alone.

3 Identifying Candidate Phrases for Classification

In this section we introduce the two data sources we use to extract sets of candidate related phrases for classification: a TREC-WordNet intersection and query logs.

3.1 Noun-Phrase Pairs Cooccuring in TREC News Sentences

The first is a data-set derived from TREC news corpora and WordNet used in previous work for binary semantic class classification (Snow et al., 2005). We extract two sets of candidate-related pairs from these corpora, one restricted and one more complete set.

Snow et al. obtained training data from the intersection of noun-phrases cooccuring in sentences in a TREC news corpus and those that can be labeled unambiguously as hypernyms or non-hypernyms using WordNet. We use a restricted set since instances selected in the previous work are a subset of the instances one is likely to encounter in text. The pairs are generally either related in one type of relationship, or completely unrelated.

In general we may be able to identify related phrases (for example with distributional similarity (Lee, 1999)), but would like to be able to automatically classify the related phrases by the type of the relationship. For this task we identify a larger set of candidate-related phrases.

3.2 Query Log Data

To find phrases that are similar or substitutable for web searchers, we turn to logs of user search sessions. We look at *query reformulations*: a pair of successive queries issued by a single user on a single day. We collapse repeated searches for the same terms, as well as query pair sequences repeated by the same user on the same day.

3.2.1 Substitutable Query Segments

Whole queries tend to consist of several concepts together, for example "new york | maps" or "britney spears | mp3s". We identify segments or phrases using a measure over adjacent terms similar to mutual information. Substitutions occur at the level of segments. For example, a user may initially search for "britney spears | mp3s", then search for "britney spears | music". By aligning query pairs with a single substituted segment, we generate pairs of phrases which a user has substituted. In this example, the phrase "mp3s" was substituted by the phrase "music".

Aggregating substitutable pairs over millions of users and millions of search sessions, we can calculate the probability of each such rewrite, then

test each pair for statistical significance to eliminate phrase rewrites which occurred in a small number of sessions, perhaps by chance. To test for statistical significance we use the pair independence likelihood ratio, or log-likelihood ratio, test. This metric tests the hypothesis that the probability of phrase β is the same whether phrase α has been seen or not by calculating the likelihood of the observed data under a binomial distribution using probabilities derived using each hypothesis (Dunning, 1993).

$$log\lambda = log\frac{L\left(P(\beta|\alpha) = P(\beta|\neg\alpha)\right)}{L\left(P(\beta|\alpha) \neq P(\beta|\neg\alpha)\right)}$$

A high negative value for λ suggests a strong dependence between query α and query β.

4 Labeling Phrase Pairs for Supervised Learning

We took a random sample of query segment substitutions from our query logs to be labeled. The sampling was limited to pairs that were frequent substitutions for each other to ensure a high probability of the segments having some relationship.

4.1 WordNet Labeling

WordNet is a large lexical database of English words. In addition to defining several hundred thousand words, it defines *synonym sets*, or *synsets*, of words that represent some underlying lexical concept, plus relationships between synsets. The most frequent relationships between noun-phrases are *synonym*, *hyponym*, *hypernym*, and *coordinate*, defined in Table 1. We also may use *meronym* and *holonym*, defined as the PART-OF relationship.

We used WordNet to automatically label the subset of our sample for which both phrases occur in WordNet. Any sense of the first segment having a relationship to any sense of the second would result in the pair being labeled. Since WordNet contains many other relationships in addition to those listed above, we group the rest into the *other* category. If the segments had no relationship in WordNet, they were labeled *no relationship*.

4.2 Segment Pair Labels

Phrase pairs passing a statistical test are common reformulations, but can be of many semantic types. Rieh and Xie (2001) categorized types of query reformulations, defining 10 general categories: *specification, generalization, synonym, parallel movement, term variations, operator usage, error correction, general resource, special resource,* and *site URLs*. We redefine these slightly to apply to query segments. The summary of the definitions is shown in Table 1, along with the distribution in the data of pairs passing the statistical test.

4.2.1 Hand Labeling

More than 90% of phrases in query logs do not appear in WordNet due to being spelling errors, web site URLs, proper nouns of a temporal nature, etc. Six annotators labeled $2,463$ segment pairs selected randomly from our sample. Annotators agreed on the label of 78% of pairs, with a Kappa statistic of $.74$.

5 Automatic Classification

We wish to perform supervised classification of pairs of phrases into semantic classes. To do this, we will assign features to each pair of phrases, which may be predictive of their semantic relationship, then use a machine-learned classifier to assign weights to these features. In Section 7 we will look at the learned weights and discuss which features are most significant for identifying which semantic classes.

5.1 Features

Features for query substitution pairs are extracted from query logs and web pages.

5.1.1 Web Page / Document Features

We submit the two segments to a web search engine as a conjunctive query and download the top 50 results. Each result is converted into an HTML Document Object Model (DOM) tree and segmented into sentences.

Dependency Tree Paths The path from the first segment to the second in a dependency parse tree generated by MINIPAR (Lin, 1998) from sentences in which both segments appear. These were previously used by Snow et al. (2005). These features were extracted from web pages in all experiments, except where we identify that we used TREC news stories (the same data as used by Snow et al.).

HTML Paths The paths from DOM tree nodes the first segment appears in to nodes the second segment appears in. The value is the number of times the path occurs with the pair.

Class	Description	Example	%
synonym	one phrase can be used in place of the other without loss in meaning	*low cost; cheap*	4.2
hypernym	X is a hypernym of Y if and only if Y is a X	*muscle car; mustang*	2.0
hyponym	X is a hyponym of Y if and only if X is a Y (inverse of hypernymy)	*lotus; flowers*	2.0
coordinate	there is some Z such that X and Y are both Zs	*aquarius; gemini*	13.9
generalization	X is a generalization of Y if X contains less information about the topic	*lyrics; santana lyrics*	4.8
specialization	X is a specification of Y if X contains more information about the topic	*credit card; card*	4.7
spelling change	spelling errors, typos, punctuation changes, spacing changes	*peopl; people*	14.9
stemmed form	X and Y have the same lemmas	*ant; ants*	3.4
URL change	X and Y are related and X or Y is a URL	*alliance; alliance.com*	29.8
other relationship	X and Y are related in some other way	*flagpoles; flags*	9.8
no relationship	X and Y are not related in any obvious way	*crypt; tree*	10.4

Table 1: Semantic relationships between phrases rewritten in query reformulation sessions, along with their prevalence in our data.

Lexico-syntactic Patterns (Hearst, 1992) A substring occurring between the two segments extracted from text in nodes in which both segments appear. In the example fragment "authors such as Shakespeare", the feature is "such as" and the value is the number of times the substring appears between "author" and "Shakespeare".

5.1.2 Query Pair Features

Table 2 summarizes features that are induced from the query strings themselves or calculated from query log data.

5.2 Additional Training Pairs

We can double our training set by adding for each pair u_1, u_2 a new pair u_2, u_1. The class of the new pair is the same as the old in all cases but *hypernym*, *hyponym*, *specification*, and *generalization*, which are inverted. Features are reversed from $f(u_1, u_2)$ to $f(u_2, u_1)$.

A pair and its inverse have different sets of features, so splitting the set randomly into training and testing sets should not result in resubstitution error. Nonetheless, we ensure that a pair and its inverse are not separated for training and testing.

5.3 Classifier

For each class we train a binary one-vs.-all linear-kernel support vector machine (SVM) using the optimization algorithm of Keerthi and DeCoste (2005).

5.3.1 Meta-Classifier

For n-class classification, we calibrate SVM scores to probabilities using the method described by Platt (2000). This gives us $P(class|pair)$ for each pair. The final classification for a pair is $argmax_{class} P(class|pair)$.

Source	Snow (NIPS 2005)	Experiment
Task	binary hypernym	binary hypernym
Data	WordNet-TREC	WordNet-TREC
Instance Count	752,311	752,311
Features	minipar paths	minipar paths
Feature Count	69,592	69,592
Classifier	logistic Regression	linear SVM
maxF	0.348	**0.453**

Table 3: Snow et al's (2005) reported performance using linear regression, and our reproduction of the same experiment, using a support vector machine (SVM).

5.3.2 Evaluation

Binary classifiers are evaluated by ranking instances by classification score and finding the Max F1 (the harmonic mean of precision and recall; ranges from 0 to 1) and area under the ROC curve (AUC; ranges from 0.5 to 1 with at least 0.8 being "good"). The meta-classifier is evaluated by precision and recall of each class and classification accuracy of all instances.

6 Experiments

6.1 Baseline Comparison to Snow et al.'s Previous Hypernym Classification on WordNet-TREC data

Snow et al. (2005) evaluated binary classification of noun-phrase pairs as *hypernyms* or *non-hypernyms*. When training and testing on WordNet-labeled pairs from TREC sentences, they report classifier Max F of 0.348, using dependency path features and logistic regression. To justify our choice of an SVM for classification, we replicated their work. Snow et al. provided us with their data. With our SVM we achieved a Max F of 0.453, 30% higher than they reported.

6.2 Extending Snow et al.'s WordNet-TREC Binary Classification to N Classes

Snow et al. select pairs that are "Known Hypernyms" (the first sense of the first word is a hy-

Feature	Description
Levenshtein Distance	# character insertions/deletions/substitutions to change query α to query β (Levenshtein, 1966).
Word Overlap Percent	# words the two queries have in common, divided by num. words in the longer query.
Possible Stem	1 if the two segments stem to the same root using the Porter stemmer.
Substring Containment	1 if the first segment is a substring of the second.
Is URL	1 if either segment matches a handmade URL regexp.
Query Pair Frequency	# times the pair was seen in the entire unlabeled corpus of query pairs.
Log Likelihood Ratio	The Log Likelihood Ratio described in Section 3.2.1 Formula 3.2.1
Dice and Jaccard Coefficients	Measures of the similarity of substitutes for and by the two phrases.

Table 2: Syntactic and statistical features over pairs of phrases.

ponym of the first sense of the second and both have no more than one tagged sense in the Brown corpus) and "Known Non-Hypernyms" (no sense of the first word is a hyponym of any sense of the second). We wished to test whether making the classes less cleanly separable would affect the results, and also whether we could use these features for n-way classification.

From the same TREC corpus we extracted *known synonym, known hyponym, known coordinate, known meronym,* and *known holonym* pairs. Each of these classes is defined analogously to the *known hypernym* class; we selected these six relationships because they are the six most common. A pair is labeled *known no-relationship* if no sense of the first word has any relationship to any sense of the second word. The class distribution was selected to match as closely as possible that observed in query logs. We labeled 50,000 pairs total.

Results are shown in Table 4(a). Although AUC is fairly high for all classes, MaxF is low for all but two. MaxF has degraded quite a bit for hypernyms from Table 3. Removing all instances except hypernym and no relationship brings MaxF up to 0.45, suggesting that the additional classes make it harder to separate hypernyms.

Metaclassifier accuracy is very good, but this is due to high recall of *no relationship* and *coordinate* pairs: more than 80% of instances with some relationship are predicted to be coordinates, and most of the rest are predicted no relationship. It seems that we are only distinguishing between *no* vs. *some* relationship.

The size of the *no relationship* class may be biasing the results. We removed those instances, but performance of the n-class classifier did not improve (Table 4(b)). MaxF of binary classifiers did improve, even though AUC is much worse.

6.3 N-Class Classification of Query Pairs

We now use query pairs rather than TREC pairs.

6.3.1 Classification Using Only Dependency Paths

We first limit features to dependency paths in order to compare to the prior results. Dependency paths cannot be obtained for all query phrase pairs, since the two phrases must appear in the same sentence together. We used only the pairs for which we could get path features, about 32% of the total.

Table 5(a) shows results of binary classification and metaclassification on those instances using dependency path features only. We can see that dependency paths do not perform very well on their own: most instances are assigned to the "coordinate" class that comprises a plurality of instances.

A comparison of Tables 5(a) and 4(a) suggests that classifying query substitution pairs is harder than classifying TREC phrases.

Table 5(b) shows the results of binary classification and metaclassification on the same instances using all features. Using all features improves performance dramatically on each individual binary classifier as well as the metaclassifier.

6.3.2 Classification on All Query Pairs Using All Features

We now expand to all of our hand-labeled pairs. Table 6(a) shows results of binary and meta classification; Figure 1 shows precision-recall curves for 10 binary classifiers (excluding URLs). Our classifier does quite well on every class but hypernym and hyponym. These two make up a very small percentage of the data, so it is not surprising that performance would be so poor.

The metaclassifier achieved 71% accuracy. This is significantly better than random or majority-class baselines, and close to our 78% interannotator agreement. Thresholding the metaclassifier to pairs with greater than .5 max class probability (68% of instances) gives 85% accuracy.

Next we wish to see how much of the performance can be maintained without using the com-

class	binary maxF	binary AUC	n-way prec	n-way rec	data %
no rel	.980	.986	.979	.985	80.0
synonym	.028	**.856**	0	0	0.3
hypernym	.185	**.888**	.512	.019	2.1
hyponym	.193	**.890**	.462	.016	2.1
coordinate	.808	**.971**	.714	.931	14.8
meronym	.158	**.905**	**.615**	.050	0.3
holonym	.120	**.883**	**.909**	.062	0.3
metaclassifier accuracy			**.927**		

(a) All seven WordNet classes. The high accuracy is mostly due to high recall of *no rel* and *coordinate* classes.

binary maxF	binary AUC	n-way prec	n-way rec	data %
–	–	–	–	0
.086	.683	0	0	1.7
.337	.708	**.563**	**.077**	10.6
.341	.720	**.527**	**.080**	10.6
.857	.737	**.757**	**.986**	74.1
.251	.777	.500	**.068**	1.5
.277	.767	.522	**.075**	1.5
–		.749		

(b) Removing *no relationship* instances improves MaxF and recall of all classes, but performance is generally worse.

Table 4: Performance of 7 binary classifier and metaclassifiers on phrase-pairs cooccuring in TREC data labeled with WordNet classes, using minipar dependency features. These features do not seem to be adequate for distinguishing classes other than *coordinate* and *no-relationship*.

class	binary maxf	binary auc	n-way prec	n-way rec
no rel	.281	.611	.067	.006
synonym	.269	.656	.293	.167
hypernym	.140	.626	0	0
hyponym	.121	.610	0	0
coordinate	.506	.760	.303	**.888**
spelling	.288	.677	.121	.022
stemmed	.571	.834	.769	.260
URL	.742	.919	.767	.691
generalization	.082	.547	0	0
specification	.085	.528	0	0
other	.393	.681	.384	.364
metaclassifier accuracy			.385	

(a) Dependency tree paths only.

binary maxf	binary auc	n-way prec	n-way rec	data %	data % full
.602	**.883**	**.639**	**.497**	10.6	3.5
.477	**.851**	**.571**	**.278**	4.5	1.5
.167	**.686**	**.125**	**.017**	3.7	1.2
.136	**.660**	0	0	3.7	1.2
.747	**.935**	**.624**	.862	21.0	6.9
.814	**.970**	**.703**	**.916**	11.0	3.6
.781	**.972**	**.788**	**.675**	4.8	1.6
1	**1**	**1**	**1**	16.2	5.3
.490	**.883**	**.489**	**.393**	3.5	1.1
.584	**.854**	**.600**	**.589**	3.5	1.1
.641	**.895**	**.603**	**.661**	17.5	5.7
–		.692		—	

(b) All features.

Table 5: Binary and metaclassifier performance on the 32% of hand-labeled instances with dependency path features. Adding all our features significantly improves performance over just using dependency paths.

putationally expensive syntactic parsing of dependency paths. To estimate the marginal gain of the other features over the dependency paths, we excluded the latter features and retrained our classifiers. Results are shown in Table 6(b). Even though binary and meta-classifier performance decreases on all classes but generalizations and specifications, much of the performance is maintained.

Because URL changes are easily identifiable by the **IsURL** feature, we removed those instances and retrained the classifiers. Results are shown in Table 6(c). Although overall accuracy is worse, individual class performance is still high, allowing us to conclude our results are not only due to the ease of classifying URLs.

We generated a learning curve by randomly sampling instances, training the binary classifiers on that subset, and training the metaclassifier on the results of the binary classifiers. The curve is shown in Figure 2. With 10% of the instances, we have a metaclassifier accuracy of 59%; with 100% of the data, accuracy is 71%. Accuracy shows no sign of falling off with more instances.

6.4 Training on WordNet-Labeled Pairs Only

Figure 2 implies that more labeled instances will lead to greater accuracy. However, manually labeled instances are generally expensive to obtain. Here we look to other sources of labeled instances for additional training pairs.

6.4.1 Training and Testing on WordNet

We trained and tested five classifiers using 10-fold cross validation on our set of WordNet-labeled query segment pairs. Results for each class are shown in Table 7. We seem to have regressed to predicting *no* vs. *some* relationship.

Because these results are not as good as the human-labeled results, we believe that some of our performance must be due to peculiarities of our data. That is not unexpected: since words that appear in WordNet are very common, features are much noisier than features associated with query entities that are often structured within web pages.

	binary		n-way		binary		n-way		data	binary		n-way	
class	maxf	auc	prec	rec	maxf	auc	prec	rec	%	maxf	auc	prec	rec
no rel	**.531**	**.878**	**.616**	**.643**	.466	.764	.549	.482	10.4	.512	.808	.502	.486
synonym	**.355**	**.820**	**.506**	**.212**	.351	.745	.493	.178	4.2	.350	.759	.478	.212
hypernym	**.173**	**.821**	.100	**.020**	.133	.728	0	0	2.0	.156	.710	**.250**	.020
hyponym	.173	**.797**	.059	.010	.163	.733	0	0	2.0	**.187**	.739	**.125**	**.020**
coordinate	**.635**	**.921**	**.590**	.703	.539	.832	.565	.732	13.9	.634	.885	.587	**.706**
spelling	**.778**	**.960**	.625	.904	.723	.917	**.628**	.902	14.9	.774	.939	.617	**.906**
stemmed	.703	**.973**	.786	.589	.656	.964	.797	.583	3.4	**.717**	.967	**.802**	**.601**
URL	1	1	1	1	1	1	1	1	29.8	–	–	–	–
generalization	.565	**.916**	.575	.483	.492	.852	**.604**	.604	4.8	**.581**	.885	.598	**.634**
specification	.661	**.926**	.652	.506	.578	.869	**.670**	**.644**	4.7	**.665**	.906	.657	.468
other	**.539**	**.898**	**.575**	**.483**	.436	.790	.550	.444	9.8	.529	.847	.559	.469
metaclassifier accuracy			.714		–		.714			–		.587	

(a) All features. (b) Dependency path features removed. (c) URL class removed.

Table 6: Binary and metaclassifier performance on all classes and all hand-labeled instances. Table (a) provides a benchmark for 10-class classification over highly substitutable query phrases. Table (b) shows that a lot of our performance can be achieved without computationally-expensive parsing.

	binary		meta		data
class	maxf	auc	prec	rec	%
no rel	.758	.719	.660	.882	57.8
synonym	.431	.901	.617	.199	2.4
hypernym	.284	.803	.367	.061	1.8
hyponym	.212	.804	.415	.056	1.6
coordinate	.588	.713	.615	.369	35.5
other	.206	.739	.375	.019	0.8
metaclassifier accuracy			.648		

Table 7: Binary and metaclassifier performance on WordNet-labeled instances with all features.

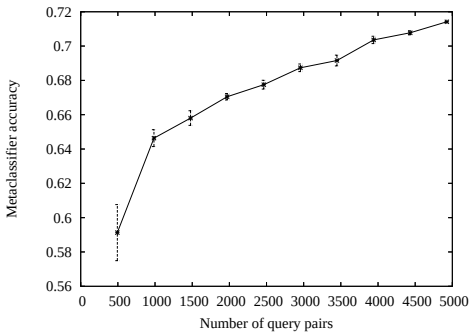


Figure 2: Meta-classifier accuracy as a function of number of labeled instances for training.

	binary		meta		data
class	maxf	auc	prec	rec	%
no rel	.525	.671	.485	.354	31.9
synonym	.381	.671	.684	.125	13.0
hypernym	.211	.605	0	0	6.2
hyponym	.125	.501	0	0	6.2
coordinate	.623	.628	.485	.844	42.6
metaclassifier accuracy			.490		

Table 8: Training on WordNet-labeled pairs and testing on hand-labeled pairs. Classifiers trained on WordNet do not generalize well.

6.4.2 Training on WordNet, Testing on WordNet and Hand-Labeled Pairs

We took the five classes for which human and WordNet definitions agreed (*synonyms, coordinates, hypernyms, hyponyms,* and *no relationship*) and trained classifiers on all WordNet-labeled instances. We tested the classifiers on human-labeled instances from just those five classes. Results are shown in Table 8. Performance was not very good, reinforcing the idea that while our features can distinguish between query segments, they cannot distinguish between common words.

7 Discussion

Almost all high-weighted features are either HTML paths or query log features; these are the ones that are easiest to obtain. Many of the highest-weight HTML tree features are symmetric, e.g. both words appear in cells of the same table, or as items in the same list. Here we note a selection of the more interesting predictors.

synonym —"X or Y" expressed as a dependency path was a high-weight feature.

hyper/hyponym —"Y and other X" as a dependency path has highest weight. An interesting feature is X in a table cell and Y appearing in text outside but nearby the table.

sibling —many symmetric HTML features. "X to the Y" as in "80s to the 90s". "X and Y", "X, Y, and Z" highly-weighted minipar paths.

general/specialization —the top three features are substring containment, word subset difference count, and prefix overlap.

spelling change —many negative features, indi-

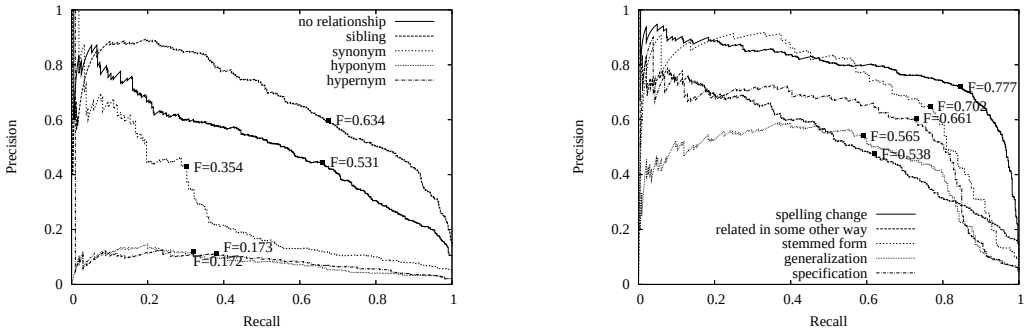


Figure 1: Precision-recall curves for 10 binary classifiers on all hand-labeled instances with all features.

cating that two words that cooccur in a web page are *not* likely to be spelling differences.

other —many symmetric HTML features. Two words emphasized in the same way (e.g. both bolded) may indicate some relationship.

none —many asymmetric HTML features, e.g. one word in a blockquote, the other bolded in a different paragraph. Dice coefficient is a good negative features.

8 Conclusion

We have provided the first benchmark for n-class semantic classification of highly substitutable query phrases. There is much room for improvement, and we expect that this baseline will be surpassed.

Acknowledgments

Thanks to Chris Manning and Omid Madani for helpful comments, to Omid Madani for providing the classification code, to Rion Snow for providing the hypernym data, and to our labelers.

This work was supported in part by the CIIR and in part by the Defense Advanced Research Projects Agency (DARPA) under contract number HR001-06-C-0023. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

References

Peter G. Anick. 2003. Using terminological feedback for web search refinement: a log-based study. In *SIGIR 2003*, pages 88–95.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *PASCAL Challenges Workshop on Recognising Textual Entailment*.

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

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of Coling 1992*, pages 539–545.

Rosie Jones, Benjamin Rey, Omid Madani, and Wiley Greiner. 2006. Generating query substitutions. In *15th International World Wide Web Conference (WWW-2006)*, Edinburgh.

Sathiya Keerthi and Dennis DeCoste. 2005. A modified finite newton method for fast solution of large scale linear svms. *Journal of Machine Learning Research*, 6:341–361, March.

Lillian Lee. 1999. Measures of distributional similarity. In *37th Annual Meeting of the Association for Computational Linguistics*, pages 25–32.

V. I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Cybernetics and Control Theory*, 10(8):707–710. Original in *Doklady Akademii Nauk SSSR* 163(4): 845–848 (1965).

Dekang Lin. 1998. Dependency-based evaluation of minipar. In *Workshop on the Evaluation of Parsing Systems*.

George A. Miller. 1995. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41.

J. Platt. 2000. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. pages 61–74.

Soo Young Rieh and Hong Iris Xie. 2001. Patterns and sequences of multiple query reformulations in web searching: A preliminary study. In *Proceedings of the 64th Annual Meeting of the American Society for Information Science and Technology Vol. 38*, pages 246–255.

Rion Snow, Dan Jurafsky, and Andrew Y. Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. In *Proceedings of the Nineteenth Annual Conference on Neural Information Processing Systems (NIPS 2005)*.

Egidio Terra and Charles L. A. Clarke. 2004. Scoring missing terms in information retrieval tasks. In *CIKM 2004*, pages 50–58.

P.D Turney, M.L. Littman, J. Bigham, and V. Shnayder, 2003. *Recent Advances in Natural Language Processing III: Selected Papers from RANLP 2003*, chapter Combining independent modules in lexical multiple-choice problems, pages 101–110. John Benjamins.

Towards Conversational QA: Automatic Identification of Problematic Situations and User Intent *

Joyce Y. Chai **Chen Zhang** **Tyler Baldwin**
Department of Computer Science and Engineering
Michigan State University
East Lansing, MI 48824
{jchai, zhangch6, baldwi96}@cse.msu.edu

Abstract

To enable conversational QA, it is important to examine key issues addressed in conversational systems in the context of question answering. In conversational systems, understanding user intent is critical to the success of interaction. Recent studies have also shown that the capability to automatically identify problematic situations during interaction can significantly improve the system performance. Therefore, this paper investigates the new implications of user intent and problematic situations in the context of question answering. Our studies indicate that, in basic interactive QA, there are different types of user intent that are tied to different kinds of system performance (e.g., problematic/error free situations). Once users are motivated to find specific information related to their information goals, the interaction context can provide useful cues for the system to automatically identify problematic situations and user intent.

1 Introduction

Interactive question answering (QA) has been identified as one of the important directions in QA research (Burger et al., 2001). One ultimate goal is to support intelligent conversation between a user and a QA system to better facilitate user information needs. However, except for a few systems that use dialog to address complex questions (Small et al., 2003; Harabagiu et al., 2005), the general dialog capabilities have been lacking in most question answering systems. To move towards conversational QA, it is important to examine key issues relevant to conversational systems in the context of interactive question answering.

This paper focuses on two issues related to conversational QA. The first issue is concerned with user intent. In conversational systems, understanding user intent is the key to the success of the interaction. In the context of interactive QA, one question is what type of user intent should be captured. Unlike most dialog systems where user intent can be characterized by dialog acts such as *question*, *reply*, and *statement*, in interactive QA, user inputs are already in the form of *question*. Then the problems become whether there are different types of intent behind these questions that should be handled differently by a QA system and how to automatically identify them.

The second issue is concerned with problematic situations during interaction. In spoken dialog systems, many problematic situations could arise from insufficient speech recognition and language understanding performance. Recent work has shown that the capability to automatically identify problematic situations (e.g., speech recognition errors) can help control and adapt dialog strategies to improve performance (Litman and Pan, 2000). Similarly, QA systems also face challenges of technology limitation from language understanding and information retrieval. Thus one question is, in the context of interactive QA, how to characterize problematic situations and automatically identify them when they occur.

In interactive QA, these two issues are intertwined. Questions formed by a user not only depend on his/her information goals, but are also influenced by the answers from the system. Problematic situations will impact user intent in the

*This work was partially supported by IIS-0347548 from the National Science Foundation.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 57–64,
Sydney, July 2006.

follow-up questions, which will further influence system performance. Both the awareness of problematic situations and understanding of user intent will allow QA systems to adapt better strategies during interaction and move towards intelligent conversational QA.

To address these two questions, we conducted a user study where users interacted with a *controlled* QA system to find information of interest. These controlled studies allowed us to focus on the interaction aspect rather than information retrieval or answer extraction aspects. Our studies indicate that in *basic interactive QA* where users always ask questions and the system always provides some kind of answers, there are different types of user intent that are tied to different kinds of system performance (e.g., problematic/error free situations). Once users are motivated to find specific information related to their information goals, the interaction context can provide useful cues for the system to automatically identify problematic situations and user intent.

2 Related Work

Open domain question answering (QA) systems are designed to automatically locate answers from large collections of documents to users' natural language questions. In the past few years, automated question answering techniques have advanced tremendously, partly motivated by a series of evaluations conducted at the Text Retrieval Conference (TREC) (Voorhees, 2001; Voorhees, 2004). To better facilitate user information needs, recent trends in QA research have shifted towards complex, context-based, and interactive question answering (Voorhees, 2001; Small et al., 2003; Harabagiu et al., 2005). For example, NIST initiated a special task on context question answering in TREC 10 (Voorhees, 2001), which later became a regular task in TREC 2004 (Voorhees, 2004) and 2005. The motivation is that users tend to ask a sequence of related questions rather than isolated single questions to satisfy their information needs. Therefore, the context QA task was designed to investigate the system capability to track context through a series of questions. Based on context QA, some work has been done to identify clarification relations between questions (Boni and Manandhar, 2003). However context QA is different from interactive QA in that context questions are specified ahead of time rather than incrementally as in an interactive setting.

Interactive QA has been applied to process complex questions. For analytical and non-factual questions, it is hard to anticipate answers. Clarification dialogues can be applied to negotiate with users about the intent of their questions (Small et al., 2003). Recently, an architecture for interactive question answering has been proposed based on a notion of predictive questioning (Harabagiu et al., 2005). The idea is that, given a complex question, the system can automatically identify a set of potential follow-up questions from a large collection of question-answer pairs. The empirical results have shown the system with predictive questioning is more efficient and effective for users to accomplish information seeking tasks in a particular domain (Harabagiu et al., 2005).

The work reported in this paper addresses a different aspect of interactive question answering. Both issues raised earlier (Section 1) are inspired by earlier work on intelligent conversational systems. Automated identification of user intent has played an important role in conversational systems. Tremendous amounts of work has focused on this aspect (Stolcke et al., 2000). To improve dialog performance, much effort has also been put on techniques to automatically detect errors during interaction. It has shown that during human machine dialog, there are sufficient cues for machines to automatically identify error conditions (Levow, 1998; Litman et al., 1999; Hirschberg et al., 2001; Walker et al., 2002). The awareness of erroneous situations can help systems make intelligent decisions about how to best guide human partners through the conversation and accomplish the tasks. Motivated by these earlier studies, the goal of this paper is to investigate whether these two issues can be applied in question answering to facilitate intelligent conversational QA.

3 User Studies

We conducted a user study to collect data concerning user behavior in a basic interactive QA setting. We are particularly interested in how users respond to different system performance and its implication in identifying problematic situations and user intent. As a starting point, we characterize system performance as either *problematic*, which indicates the answer has some problem, or *error-free*, which indicates the answer is correct. In this section, we first describe the methodology

and the system used in this effort and then discuss the observed user behavior and its relation to problematic situations and user intent.

3.1 Methodology and System

The system used in our experiments has a user interface that takes a natural language question and presents an answer passage. Currently, our interface only presents to the user the top one retrieved result. This simplification on one hand helps us focus on the investigation of user responses to different system performances and on the other hand represents a possible situation where a list of potential answers may not be practical (e.g., through PDA or telephone line).

We implemented a Wizard-of-Oz (WOZ) mechanism in the interaction loop to control and simulate problematic situations. Users were not aware of the existence of this human wizard and were led to believe they were interacting with a real QA system. This *controlled* setting allowed us to focus on the interaction aspect rather than information retrieval or answer extraction aspect of question answering. More specifically, during interaction after each question was issued, a random number generator was used to decide if a problematic situation should be introduced. If the number indicated no, the wizard would retrieve a passage from a database with correct question/answer pairs. Note that in our experiments we used specific task scenarios (described later), so it was possible to anticipate user information needs and create this database. If the number indicated that a problematic situation should be introduced, then the Lemur retrieval engine [1] was used on the AQUAINT collection to retrieve the answer. Our assumption is that AQUAINT data are not likely to provide an exact answer given our specific scenarios, but they can provide a passage that is most related to the question. The use of the random number generator was to control the ratio between the occurrence of problematic situations and error-free situations. In our initial investigation, since we are interested in observing user behavior in problematic situations, we set the ratio as 50/50. In our future work, we will vary this ratio (e.g., 70/30) to reflect the performance of state-of-the-art factoid QA and investigate the implication of this ratio in automated performance assessment.

[1] http://www-2.cs.cmu.edu/ lemur/

3.2 Experiments

Eleven users participated in our study. Each user was asked to interact with our system to complete information seeking tasks related to four specific scenarios: *the 2004 presidential debates*, *Tom Cruise*, *Hawaii*, and *Pompeii*. The experimental scenarios were further divided into two types: structured and unstructured. In the structured task scenarios (for topics *Tom Cruise* and *Pompeii*), users had to fill in blanks on a diagram pertaining to the given topic. Using the diagram was to avoid the influence of these scenarios on the language formation of the relevant questions. Because users must find certain information, they were constrained in the range of questions in which they could ask, but not the way they ask those questions. The task was completed when all of the blanks on the diagram were filled. The structured scenarios were designed to mimic the real information seeking practice in which users have real motivation to find specific information related to their information goals. In the unstructured scenarios (for topics *the 2004 presidential debates* and *Hawaii*), users were given a general topic to investigate, but were not required to find specific information. This gave the user the ability to ask a much wider range of questions than the structured scenarios. Users were generally in an exploration mode when performing these unstructured tasks. They were not motivated to find specific information and were content with any information provided by the system. In our view, the unstructured scenarios are less representative of the true information seeking situations.

3.3 Observations and Analysis

From our studies, a total of 44 interaction sessions with 456 questions were collected. Figure 1 shows an example of a fragment of interaction related to *Tom Cruise*. In this example, both problematic situations applied to answers (e.g., *Problematic* and *Error-Free*) and user intent (described later) applied to questions are annotated.

There are several observations from this data. First, questions formed during interactive QA tend to be self-contained and free of definite noun phrases, pronouns, or ellipsis. Only one question in the entire data set has a pronoun (i.e., *What are the best movies with Tom Cruise in them?*). Even in this case, the pronoun *them* did not refer to any entities that occurred previously in the

#	Question/Answer	Annotation
Q1	When was Tom Cruise born?	
A1	… Based on the memoir of combat veteran Ron Kovic, the film stars Tom Cruise as Kovic, whose gunshot wound in Vietnam left him paralyzed from the chest down. ….a powerfully intimate portrait that unfolds on an epic scale, *Born on the Fourth of July* is arguably Stone's best film (if you can forgive its often strident tone), …..	**Proble-matic**
Q2	What year was Tom Cruise born?	*Re-try*
A2	…Thomas Cruise Mapother IV was born on the *3rd of July, 1962* (eerily similar to his film Born on the 4th of July), in Syracuse, New York. He was the only boy of four children….	**Error-Free**
Q3	What does Tom Cruise do for a living?	*Continue*
A3	…you get a very nice role that shows you differently, not the heavy, and you're working with a George Clooney or a Tom Cruise or a Nicolas Cage or a Martin Scorsese. I can live with that …	**Proble-matic**
Q4	What is Tom Cruise's profession?	*Re-try*
A4	…So we may agree that Cruise's professional standing accounts for some measure of his fame….	**Proble-matic**
Q5	What are the best movies with Tom Cruise in them?	*Switch*
A5	…Cruise proved his dramatic talents in the 1988 drama Rain Man, where he co-starred with Oscar-winner Dustin Hoffman. Oliver Stone's Born on the Fourth of July (1989) earned him a Best Actor Oscar nomination for his hard-hitting portrayal of anti-war activist Ron Kovic….	**Error-Free**

Figure 1: An example fragment of interaction

QA process. This phenomenon could be caused by how the answers are presented. Unlike specific answer entities, the answer passages provided by our system do not support the natural use of referring expressions in the follow-up questions. Another possible explanation could be that in an interactive environment, users seem to be more aware of the potential limitation of a computer system and thus tend to specify self-contained questions in a hope to reduce the system's inference load.

The second observation is about user behavior in response to different system performances (i.e., problematic or error-free situations). We were hoping to see different strategies users might apply to deal with the problematic situations. However, based on the data, we found that when a problem occurred, users either rephrased their questions (i.e., the same question expressed in a different way) or gave up the question and went on specifying a new question. (Here we use *Rephrase* and *New* to denote these two kinds of behaviors.) We have not observed any sub-dialogs initiated by the user to clarify a previous question or answer. One possible explanation is that the current investigation was conducted in a basic interactive mode where the system was only capable of providing some sort of answers. This may limit users' expectation in the kind of questions that can be handled by the system. Our assumption is that, once the QA system becomes more intelligent and able to carry on conversation, different types of questions (i.e., other than rephrase or new) will be observed. This hypothesis certainly needs to be validated in a conversational setting.

	Problematic	**Error-free**	**Total**
New	*Switch*	*Continue*	
unstruct.	29	90	119
struct.	29	133	162
entire	58	223	281
Rephrase	*Re-try*	*Negotiate*	
unstruct.	19	4	23
struct.	102	6	108
entire	121	10	131
Total-unst	48	94	142
Total-st	131	139	270
Total-ent	179	233	412

Table 1: Categorization of user intent with the corresponding number of occurrences from the unstructured scenarios, the structured scenarios, and the entire dataset.

The third observation is that the rephrased questions seem to strongly correlate with problematic situations, although not always. New questions cannot distinguish a problematic situation from an error-free situation. Table 1 shows the statistics from our data about different combinations of new/rephrase questions and performance situations[2]. What is interesting is that these different combinations can reflect different types of user intent behind the questions. More specifically, given a question, four types of user intent can be captured with respect to the context (e.g., the previous question and answer)

Continue indicates that the user is satisfied with the previous answer and now moves on to this new question.

Switch indicates that the user has given up on the previous question and now moves on to this

[2]The last question from each interaction session is not included in these statistics because there is no follow-up question after that.

new question.

Re-try indicates that the user is not satisfied with the previous answer and now tries to get a better answer.

Negotiate indicates that the user is not satisfied with the previous answer (although it appears to be correct from the system's point of view) and now tries to get a better answer for his/her own needs.

Table 1 summarizes these different types of intent together with the number of corresponding occurrences from both structured and unstructured scenarios. Since in the unstructured scenarios it was hard to anticipate user's questions and therefore take a correct action to respond to a problematic/error-free situation, the distribution of these two situations is much more skewed than the distribution for the structured scenarios. Also as mentioned earlier, in unstructured scenarios, users lacked the motivation to pursue specific information, so the ratio between *switch* and *re-try* is much larger than that observed in the structured scenarios. Nevertheless, we did observe different user behavior in response to different situations. As discussed later in Section 5, identifying these fine-grained intents will allow QA systems to be more proactive in helping users find satisfying answers.

4 Automatic Identification of Problematic Situations and User Intent

Given the discussion above, the next question is how to automatically identify problematic situations and user intent. We formulate this as a classification problem. Given a question Q_i, its answer A_i, and the follow-up question Q_{i+1}:
(1) Automatic identification of problematic situations is to decide whether A_i is problematic (i.e., correct or incorrect) based on the follow-up question Q_{i+1} and the interaction context. This is a binary classification problem.
(2) Automatic identification of user intent is to identify the intent of Q_{i+1} given the interaction context. Because we only have very limited instances of *Negotiate* (see Table 1), we currently merge *Negotiate* with *Re-try* since both of them represent a situation where a better answer is requested. Thus, this problem becomes a trinary classification problem.

To build these classifiers, we identified a set of features, which are illustrated next.

4.1 Features

Given a question Q_i, its answer A_i, and the follow-up question Q_{i+1}, the following set of features are used:

Target matching(TM): a binary feature indicating whether the target type of Q_{i+1} is the same as the target type of Q_i. Our data shows that the repetition of the target type may indicate a rephrase, which could signal a problematic situation has just happened.

Named entity matching (NEM): a binary feature indicating whether all the named entities in Q_{i+1} also appear in Q_i. If no new named entity is introduced in Q_{i+1}, it is likely Q_{i+1} is a rephrase of Q_i.

Similarity between questions (SQ): a numeric feature measuring the similarity between Q_{i+1} and Q_i. Our assumption is that the higher the similarity is, the more likely the current question is a rephrase to the previous one.

Similarity between content words of questions (SQC): this feature is similar to the previous feature (i.e., SQ) except that the similarity measurement is based on the content words excluding named entities. This is to prevent the similarity measurement from being dominated by the named entities.

Similarity between Q_i and A_i (SA): this feature measures how close the retrieved passage matches the question. Our assumption is that although a retrieved passage is the most relevant passage compared to others, it still may not contain the answer (e.g., when an answer does not even exist in the data collection).

Similarity between Q_i and A_i based on the content words (SAC): this feature is essentially the same as the previous feature (SA) except that the similarity is calculated after named entities are removed from the questions and answers.

Note that since our data is currently collected from simulation studies, we do not have the confidence score from the retrieval engine associated with every answer. In practice, the confidence score can be used as an additional feature.

Since our focus is not on the similarity measurement but rather the use of the measurement in the classification models, our current similarity measurement is based on a simple approach that measures commonality and difference between two objects as proposed by Lin (1998). More specifically, the following equation is applied to measure

the similarity between two chunks of text T_1 and T_2:

$$sim_1(T_1, T_2) = \frac{-\log P(T_1 \cap T_2)}{-\log P(T_1 \cup T_2)}$$

Assume the occurrence of each word is independent, then:

$$sim_1(T_1, T_2) = \frac{-\sum_{w \in T_1 \cap T_2} \log P(w)}{-\sum_{w \in T_1 \cup T_2} \log P(w)}$$

where $P(w)$ was calculated based on the data used in the previous TREC evaluations.

4.2 Identification of Problematic Situations

To identify problematic situations, we experimented with three different classifiers: Maximum Entropy Model (MEM) from MALLET[3], SVM from SVM-Light[4], and Decision Trees from WEKA[5]. A leave-one-out validation was applied where one interaction session was used for testing and the remaining interaction sessions were used for training.

Table 2 shows the performance of the three models based on different combinations of features in terms of classification accuracy. The baseline result is the performance achieved by simply assigning the most frequently occurred class. For the unstructured scenarios, the performance of the classifiers is rather poor, which indicates that it is quite difficult to make any generalization based on the current feature sets when users are less motivated in finding specific information. For the structured scenarios, the best performance for each model is highlighted in bold in Table 2. The Decision Tree model achieves the best performance of 77.8% in identifying problematic situations, which is more than 25% better than the baseline performance.

4.3 Identification of User Intent

To identify user intent, we formulate the problem as follows: given an observation feature vector $\mathbf{f}$ where each element of the vector corresponds to a feature described earlier, the goal is to identify an intent c^* from a set of intents I ={*Continue, Switch, Re-try/Negotiate*} that satisfies the following equation:

$$c^* = \arg\max_{c \in I} P(c|\mathbf{f})$$

[3] http://mallet.cs.umass.edu/index.php/
[4] http://svmlight.joachims.org/
[5] http://www.cs.waikato.ac.nz/ml/weka/

Our assumption is that user intent for a question can be potentially influenced by the intent from a preceding question. For example, *Switch* is likely to follow *Re-try*. Therefore, we have implemented a Maximum Entropy Markov Model (MEMM) (McCallum et al., 2000) to take the sequence of interactions into account.

Given a sequence of questions Q_1, Q_2, up to Q_t, there is an observation feature vector $\mathbf{f_i}$ associated with each Q_i. In MEMM, the prediction of user intent c_t for Q_t not only depends on the observation $\mathbf{f_t}$, but also the intent c_{t-1} from the preceding question Q_{t-1}. In fact, this approach finds the best sequence of user intent C^* for Q_1 up to Q_t based on a sequence of observations $\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_t$ as follows:

$$C^* = \arg\max_{C \in I^t} P(C|\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_t)$$

where C is a sequence of intent and I^t is the set of all possible sequences of intent with length t.

To find this sequence of intent C^*, MEMM keeps a variable $\alpha_t(i)$ which is defined to be the maximum probability of seeing a particular sequence of intent ending at intent i ($i \in I$) for question Q_t, given the observation sequence for questions Q_1 up to Q_t:

$$\alpha_t(i) = \max_{c_1,...,c_{t-1}} P(c_1, \ldots, c_{t-1}, c_t = i|\mathbf{f}_1, \ldots, \mathbf{f}_t)$$

This variable can be calculated by a dynamic optimization procedure similar to the Viterbi algorithm in the Hidden Markov Model:

$$\alpha_t(i) = \max_j \alpha_{t-1}(j) \times P(c_t = i|c_{t-1} = j, \mathbf{f}_t)$$

where $P(c_t = i|c_{t-1} = j, \mathbf{f}_t)$ is estimated by the Maximum Entropy Model.

Table 3 shows the best results of identifying user intent based on the Maximum Entropy Model and MEMM using the leave-one-out approach.

The results have shown that both models did not work for the data collected from unstructured scenarios (i.e., the baseline accuracy for intent identification is 63.4%). For structured scenarios, in terms of the overall accuracy, both models performed significantly better than the baseline (i.e., 49.3%). The MEMM worked only slightly better than the MEM. Given our limited data, it is not conclusive whether the transitions between questions will help identify user intent in a basic interactive mode. However, we expect to see more influence from the transitions in fully conversational QA.

Features	MEM			SVM			DTree		
	un	s	ent	un	s	ent	un	s	ent
Baseline	66.2	51.5	56.3	66.2	51.5	56.3	66.2	51.5	56.3
TM, SQC	50.0	57.4	54.9	53.5	60.0	57.8	53.5	55.9	55.1
NEM, SQC	37.3	74.4	61.7	37.3	74.4	61.7	37.3	74.4	61.7
TM, SQ	61.3	64.8	63.6	57.0	64.1	61.7	59.9	64.4	62.9
NEM, SQC, SAC	40.8	**76.7**	64.3	38.0	74.4	61.9	49.3	**77.8**	68.0
TM, SQ, SAC	59.2	67.4	64.6	61.3	66.3	64.6	62.7	65.6	64.6
TM, NEM, SQC	54.2	75.2	68.0	54.2	**75.2**	68.0	53.5	74.4	67.2
TM, SQ, SA	63.4	71.9	68.9	58.5	71.5	67.0	67.6	75.6	72.8
TM, NEM, SQC, SAC	54.9	75.6	68.4	54.2	**75.2**	68.0	55.6	74.4	68.0

* un - unstructured, s - structured, ent - entire

Table 2: Performance of automatic identification of problematic situations

		MEM		MEMM	
		un	s	un	s
CONTINUE	P	64.4	69.7	67.3	70.8
	R	96.7	85.8	80.0	88.8
	F	77.3	76.8	73.1	78.7
RE-TRY	P	28.6	76.2	37.1	79.0
/NEGOTIATE	R	8.7	74.1	56.5	73.1
	F	13.3	75.1	44.8	75.9
SWITCH	P	-	-	-	50.0
	R	0	0	0	3.6
	F	-	-	-	6.7
Overall accuracy		62.7	72.2	59.9	73.7

* un - unstructured, s - structured

Table 3: Performance of automatic identification of user intent

5 Implications of Problematic Situations and User Intent

Automated identification of problematic situations and user intent have potential implications in the design of conversational QA systems. Identification of problematic situations can be considered as implicit feedback. The system can use this feedback to improve its answer retrieval performance and proactively adapt its strategy to cope with problematic situations. One might think that an alternative way is to explicitly ask users for feedback. However, this explicit approach will defeat the purpose of intelligent conversational systems. Soliciting feedback after each question not only will frustrate users and lengthen the interaction, but also will interrupt the flow of user thoughts and conversation. Therefore, our focus here is to investigate the more challenging end of implicit feedback. In practice, the explicit feedback and implicit feedback should be intelligently combined. For example, if the confidence for automatically identifying a problematic situation or an error-free situation is low, then perhaps explicit feedback can be solicited.

Automatic identification of user intent also has important implications in building intelligent conversational QA systems. For example, if *Continue* is identified during interaction, then the system can automatically collect the question answer pairs for potential future use. If *Switch* is identified, the system may put aside the question that has not been correctly answered and proactively come back to that question later after more information is gathered. If *Re-try* is identified, the system may avoid repeating the same answer and at the same time may take the initiative to guide users on how to rephrase a question. If *Negotiate* is identified, the system may want to investigate the user's particular needs that may be different from the general needs. Overall, different strategies can be developed to address problematic situations and different intents. We will investigate these strategies in our future work.

This paper reports our initial effort in investigating interactive QA from a conversational point of view. The current investigation has several simplifications. First, our current work has focused on factoid questions where it is relatively easy to judge a problematic or error-free situation. However, as discussed in earlier work (Small et al., 2003), sometimes it is very hard to judge the truthfulness of an answer, especially for analytical questions. Therefore, our future work will examine the new implications of problematic situations and user intent for analytical questions. Sec-

ond, our current investigation is based on a basic interactive mode. As mentioned earlier, once the QA systems become more intelligent and conversational, more varieties of user intent are anticipated. How to characterize and automatically identify more complex user intent under these different situations is another direction of our future work.

6 Conclusion

This paper presents our initial investigation on automatic identification of problematic situations and user intent in interactive QA. Our results have shown that, once users are motivated in finding specific information related to their information goals, user behavior and interaction context can help automatically identify problematic situations and user intent. Although our current investigation is based on the data collected from a controlled study, the same approaches can be applied during online processing as the question answering proceeds. The identified problematic situations and/or user intent will provide immediate feedback for a QA system to adjust its behavior and adapt better strategies to cope with different situations. This is an important step toward intelligent conversational question answering.

References

Marco De Boni and Suresh Manandhar. 2003. An analysis of clarification dialogues for question answering. In *Proceedings of HLT-NAACL 2003*, pages 48–55.

John Burger, Claire Cardie, Vinay Chaudhri, Robert Gaizauskas, Sanda Harabagiu, David Israel, Christian Jacquemin, Chin-Yew Lin, Steve Maiorano, George Miller, Dan Moldovan, Bill Ogden, John Prager, Ellen Riloff, Amit Singhal, Rohini Shrihari, Tomek Strzalkowski, Ellen Voorhees, and Ralph Weishedel. 2001. Issues, tasks and program structures to roadmap research in question & answering. In *NIST Roadmap Document*.

Sanda Harabagiu, Andrew Hickl, John Lehmann, and Dan Moldovan. 2005. Experiments with interactive question-answering. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 205–214, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Julia Hirschberg, Diane J. Litman, and Marc Swerts. 2001. Identifying user corrections automatically in spoken dialogue systems. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL'01)*.

Gina-Anne Levow. 1998. Characterizeing and recognizing spoken corrections in human-computer dialogue. In *Proceedings of the 36th Annual Meeting of the Association of Computational Linguistics (COLING/ACL-98)*, pages 736–742.

Dekang Lin. 1998. An information-theoretic definition of similarity. In *Proceedings of International Conference on Machine Learning*, Madison, Wisconsin, July.

Diane J. Litman and Shimei Pan. 2000. Predicting and adapting to poor speech recognition in a spoken dialogue system. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, pages 722–728.

Diane J. Litman, Marilyn A. Walker, and Michael S. Kearns. 1999. Automatic detection of poor speech recognition at the dialogue level. In *Proceedings of the 37th Annual meeting of the Association of Computational Linguistics (ACL-99)*, pages 309–316.

Andrew McCallum, Dayne Freitag, and Fernando Pereira. 2000. Maximum entropy markov models for information extraction and segmentation. In *Proceedings of Internatioanl Conference on Machine Learning (ICML 2000)*, pages 591–598.

Sharon Small, Ting Liu, Nobuyuki Shimizu, and Tomek Strzalkowski. 2003. HITIQA: An interactive question answering system: A preliminary report. In *Proceedings of the ACL 2003 Workshop on Multilingual Summarization and Question Answering*.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Marie Meteer, and Carol Van Ess-Dykema. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. In *Computational Linguistics*, volume 26.

Ellen Voorhees. 2001. Overview of TREC 2001 question answering track. In *Proceedings of TREC*.

Ellen Voorhees. 2004. Overview of TREC 2004. In *Proceedings of TREC*.

Marilyn Walker, Irene Langkilde-Geary, Helen Wright Hastie, Jerry Wright, and Allen Gorin. 2002. Automatically training a problematic dialogue predictor for the HMIHY spoken dialog system. In *Journal of Artificial Intelligence Research*.

A Pipeline Framework for Dependency Parsing

Ming-Wei Chang **Quang Do** **Dan Roth**
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801
{mchang21, quangdo2, danr}@uiuc.edu

Abstract

Pipeline computation, in which a task is decomposed into several stages that are solved sequentially, is a common computational strategy in natural language processing. The key problem of this model is that it results in error accumulation and suffers from its inability to correct mistakes in previous stages. We develop a framework for decisions made via in pipeline models, which addresses these difficulties, and presents and evaluates it in the context of *bottom up dependency parsing* for English. We show improvements in the accuracy of the inferred trees relative to existing models. Interestingly, the proposed algorithm shines especially when evaluated globally, at a sentence level, where our results are significantly better than those of existing approaches.

1 Introduction

A pipeline process over the decisions of learned classifiers is a common computational strategy in natural language processing. In this model a task is decomposed into several stages that are solved sequentially, where the computation in the ith stage typically depends on the outcome of computations done in previous stages. For example, a semantic role labeling program (Punyakanok et al., 2005) may start by using a part-of-speech tagger, then apply a shallow parser to chunk the sentence into phrases, identify predicates and arguments and then classify them to types. In fact, any left to right processing of an English sentence may be viewed as a pipeline computation as it processes a token and, potentially, makes use of this result when processing the token to the right.

The pipeline model is a standard model of computation in natural language processing for good reasons. It is based on the assumption that some decisions might be easier or more reliable than others, and their outcomes, therefore, can be counted on when making further decisions. Nevertheless, it is clear that it results in error accumulation and suffers from its inability to correct mistakes in previous stages. Researchers have recently started to address some of the disadvantages of this model. E.g., (Roth and Yih, 2004) suggests a model in which global constraints are taken into account in a later stage to fix mistakes due to the pipeline. (Punyakanok et al., 2005; Marciniak and Strube, 2005) also address some aspects of this problem. However, these solutions rely on the fact that all decisions are made with respect to the same input; specifically, all classifiers considered use the same examples as their input. In addition, the pipelines they study are shallow.

This paper develops a general framework for decisions in pipeline models which addresses these difficulties. Specifically, we are interested in *deep pipelines* – a large number of predictions that are being chained.

A pipeline process is one in which decisions made in the ith stage (1) depend on earlier decisions and (2) feed on input that depends on earlier decisions. The latter issue is especially important at evaluation time since, at training time, a gold standard data set might be used to avoid this issue.

We develop and study the framework in the context of a *bottom up approach to dependency parsing*. We suggest that two principles to guide the pipeline algorithm development:
(i) Make local decisions as reliable as possible.
(ii) Reduce the number of decisions made.

Using these as guidelines we devise an algo-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 65–72,
Sydney, July 2006.

rithm for dependency parsing, prove that it satisfies these principles, and show experimentally that this improves the accuracy of the resulting tree.

Specifically, our approach is based on a shift-reduced parsing as in (Yamada and Matsumoto, 2003). Our general framework provides insights that allow us to improve their algorithm, and to principally justify some of the algorithmic decisions. Specifically, the first principle suggests to improve the reliability of the local predictions, which we do by improving the set of actions taken by the parsing algorithm, and by using a look-ahead search. The second principle is used to justify the control policy of the parsing algorithm – which edges to consider at any point of time. We prove that our control policy is optimal in some sense, and that the decisions we made, guided by these, principles lead to a significant improvement in the accuracy of the resulting parse tree.

1.1 Dependency Parsing and Pipeline Models

Dependency trees provide a syntactic reresentation that encodes functional relationships between words; it is relatively independent of the grammar theory and can be used to represent the structure of sentences in different languages. Dependency structures are more efficient to parse (Eisner, 1996) and are believed to be easier to learn, yet they still capture much of the predicate-argument information needed in applications (Haghighi et al., 2005), which is one reason for the recent interest in learning these structures (Eisner, 1996; McDonald et al., 2005; Yamada and Matsumoto, 2003; Nivre and Scholz, 2004).

Eisner's work – $O(n^3)$ parsing time generative algorithm – embarked the interest in this area. His model, however, seems to be limited when dealing with complex and long sentences. (McDonald et al., 2005) build on this work, and use a global discriminative training approach to improve the edges' scores, along with Eisner's algorithm, to yield the expected improvement. A different approach was studied by (Yamada and Matsumoto, 2003), that develop a bottom-up approach and learn the parsing decisions between consecutive words in the sentence. Local actions are used to generate a dependency tree using a shift-reduce parsing approach (Aho et al., 1986). This is a true pipeline approach, as was done in other successful parsers, e.g. (Ratnaparkhi, 1997), in that the classifiers are trained on individual decisions rather than on the overall quality of the parser, and chained to yield the global structure. Clearly, it suffers from the limitations of pipeline processing, such as accumulation of errors, but nevertheless, yields very competitive parsing results. A somewhat similar approach was used in (Nivre and Scholz, 2004) to develop a hybrid bottom-up/top-down approach; there, the edges are also labeled with semantic types, yielding lower accuracy than the works mentioned above.

The overall goal of dependency parsing (DP) learning is to infer a tree structure. A common way to do that is to predict with respect to each potential edge (i, j) in the tree, and then choose a global structure that (1) is a tree and that (2) maximizes some score. In the context of DPs, this "edge based factorization method" was proposed by (Eisner, 1996). In other contexts, this is similar to the approach of (Roth and Yih, 2004) in that scoring each edge depends only on the raw data observed and not on the classifications of other edges, and that global considerations can be used to overwrite the local (edge-based) decisions.

On the other hand, the key in a pipeline model is that making a decision with respect to the edge (i, j) may gain from taking into account decisions already made with respect to neighboring edges. However, given that these decisions are noisy, there is a need to devise policies for reducing the number of predictions in order to make the parser more robust. This is exemplified in (Yamada and Matsumoto, 2003) – a bottom-up approach, that is most related to the work presented here. Their model is a "traditional" pipeline model – a classifier suggests a decision that, once taken, determines the next action to be taken (as well as the input the next action observes).

In the rest of this paper, we propose and justify a framework for improving pipeline processing based on the principles mentioned above: (i) make local decisions as reliably as possible, and (ii) reduce the number of decisions made. We use the proposed principles to examine the (Yamada and Matsumoto, 2003) parsing algorithm and show that this results in modifying some of the decisions made there and, consequently, better overall dependency trees.

2 Efficient Dependency Parsing

This section describes our DP algorithm and justifies its advantages as a pipeline model. We pro-

pose an improved pipeline framework based on the mentioned principles.

For many languages such as English, Chinese and Japanese (with a few exceptions), projective dependency trees (that is, DPs without edge crossings) are sufficient to analyze most sentences. Our work is therefore concerned only with projective trees, which we define below.

For words x, y in the sentence T we introduce the following notations:

$x \rightarrow y$: x is the *direct parent* of y.
$x \rightarrow^* y$: x is an *ancestor* of y;
$x \leftrightarrow y$: $x \rightarrow y$ or $y \rightarrow x$.
$x < y$: x is *to the left* of y in T.

Definition 1 (Projective Language) *(Nivre, 2003)* $\forall a, b, c \in T, a \leftrightarrow b$ *and* $a < c < b$ *imply that* $a \rightarrow^* c$ *or* $b \rightarrow^* c$.

2.1 A Pipeline DP Algorithm

Our parsing algorithm is a modified shift-reduce parser that makes use of the actions described below and applies them in a left to right manner on consecutive pairs of words (a, b) $(a < b)$ in the sentence. This is a bottom-up approach that uses machine learning algorithms to learn the parsing decisions (actions) between consecutive words in the sentences. The basic actions used in this model, as in (Yamada and Matsumoto, 2003), are:

S*hift*: there is no relation between a and b, or the action is deferred because the relationship between a and b cannot be determined at this point.

R*ight*: b is the parent of a,

L*eft*: a is the parent of b.

This is a true pipeline approach in that the classifiers are trained on individual decisions rather than on the overall quality of the parsing, and chained to yield the global structure. And, clearly, decisions make with respect to a pair of words affect what is considered next by the algorithm.

In order to complete the description of the algorithm we need to describe which edge to consider once an action is taken. We describe it via the notion of the *focus point*: when the algorithm considers the pair (a, b), $a < b$, we call the word a the current *focus point*.

Next we describe several policies for determining the focus point of the algorithm following an action. We note that, with a few exceptions, determining the focus point does not affect the *correctness* of the algorithm. It is easy to show that for (almost) any focus point chosen, if the correct action is selected for the corresponding edge, the algorithm will eventually yield the correct tree (but may require multiple cycles through the sentence). In practice, the actions selected are noisy, and a wasteful focus point policy will result in a large number of actions, and thus in error accumulation. To minimize the number of actions taken, we want to find a good focus point placement policy.

After **S**, the focus point always moves one word to the right. After **L** or **R** there are there natural placement policies to consider:

Start Over: Move focus to the first word in T.

Stay: Move focus to the next word to the right. That is, for $T = (a, b, c)$, and focus being a, an **L** action will result is the focus being a, while **R** action results in the focus being b.

Step Back: The focus moves to the previous word (on the left). That is, for $T = (a, b, c)$, and focus being b, in both cases, a will be the focus point.

In practice, different placement policies have a significant effect on the number of pairs considered by the algorithm and, therefore, on the final accuracy[1]. The following analysis justifies the **Step Back** policy. We claim that if **Step Back** is used, the algorithm will not waste any action. Thus, it achieves the goal of minimizing the number of actions in pipeline algorithms. Notice that using this policy, when **L** is taken, the pair (a, b) is reconsidered, but with new information, since now it is known that c is the child of b. Although this seems wasteful, we will show this is a necessary movement to reduce the number of actions.

As mentioned above, each of these policies yields the correct tree. Table 1 compares the three policies in terms of the number of actions required to build a tree.

Policy	$\#Shift$	$\#Left$	$\#Right$
Start over	156545	26351	27918
Stay	117819	26351	27918
Step back	43374	26351	27918

Table 1: The number of actions required to build all the trees for the sentences in section 23 of Penn Treebank (Marcus et al., 1993) as a function of the focus point placement policy. The statistics are taken with the correct (gold-standard) actions.

It is clear from Table 1 that the policies result

[1]Note that (Yamada and Matsumoto, 2003) mention that they move the focus point back after **R**, but do not state what they do after executing **L** actions, and why. (Yamada, 2006) indicates that they also move focus point back after **L**.

Algorithm 2 Pseudo Code of the dependency parsing algorithm. *getFeatures* extracts the features describing the word pair currently considered; *getAction* determines the appropriate action for the pair; *assignParent* assigns a parent for the child word based on the action; and *deleteWord* deletes the child word in T at the *focus* once the action is taken.

```
Let t represents for a word token
For sentence T = {t_1, t_2, ..., t_n}
focus= 1
while focus< |T| do
    v = getFeatures(t_focus, t_focus+1)
    α = getAction(t_focus, t_focus+1, v)
    if α = L or α = R then
        assignParent(t_focus, t_focus+1, α)
        deleteWord(T, focus, α)
        // performing Step Back here
        focus = focus − 1
    else
        focus = focus + 1
    end if
end while
```

in very different number of actions and that **Step Back** is the best choice. Note that, since the actions are the gold-standard actions, the policy affects only the number of **S** actions used, and not the **L** and **R** actions, which are a direct function of the correct tree. The number of required actions in the testing stage shows the same trend and the **Step Back** also gives the best dependency accuracy. Algorithm 2 depicts the parsing algorithm.

2.2 Correctness and Pipeline Properties

We can prove two properties of our algorithm. First we show that the algorithm builds the dependency tree in only one pass over the sentence. Then, we show that the algorithm does not waste actions in the sense that it never considers a word pair twice in the same situation. Consequently, this shows that under the assumption of a perfect action predictor, our algorithm makes the smallest possible number of actions, among all algorithms that build a tree sequentially in one pass.

Note that this may not be true if the action classifier is not perfect, and one can contrive examples in which an algorithm that makes several passes on a sentence can actually make fewer actions than a single pass algorithm. In practice, however, as our experimental data shows, this is unlikely.

Lemma 1 *A dependency parsing algorithm that uses the Step Back policy completes the tree when it reaches the end of the sentence for the first time.*

In order to prove the algorithm we need the following definition. We call a pair of words (a, b) a *free pair* if and only if there is a relation between a and b and the algorithm can perform **L** or **R** actions on that pair when it is considered. Formally,

Definition 2 (*free pair) A pair* (a, b) *considered by the algorithm is a free pair, if it satisfies the following conditions:*

1. $a \leftrightarrow b$
2. *a, b are consecutive in T (not necessary in the original sentence).*
3. *No other word in T is the child of a or b. (a and b are now part of a complete subtree.)*

Proof. : It is easy to see that there is at least one *free pair* in T, with $|T| > 1$. The reason is that if no such pair exists, there must be three words $\{a, b, c\}$ s.t. $a \leftrightarrow b$, $a < c < b$ and $\neg(a \rightarrow c \vee b \rightarrow c)$. However, this violates the properties of a projective language.

Assume $\{a, b, d\}$ are three consecutive words in T. Now, we claim that when using *Step Back*, the focus point is always to the left of all free pairs in T. This is clearly true when the algorithm starts. Assume that (a, b) is the first *free pair* in T and let c be just to the left of a and b. Then, the algorithm will not make a **L** or **R** action before the focus point meets (a, b), and will make one of these actions then. It's possible that $(c, a \vee b)$ becomes a free pair after removing a or b in T so we need to move the focus point back. However, we also know that there is no *free pair* to the left of c. Therefore, during the algorithm, the focus point will always remain to the left of all free pairs. So, when we reach the end of the sentence, every free pair in the sentence has been taken care of, and the sentence has been completely parsed. □

Lemma 2 *All actions made by a dependency parsing algorithm that uses the Step Back policy are necessary.*

Proof. : We will show that a pair (a, b) will never be considered again given the same situation, that is, when there is no additional information about relations a or b participate in. Note that if **R** or

L is taken, either a or b will become a child word and be eliminate from further consideration by the algorithm. Therefore, if the action taken on (a, b) is **R** or **L**, it will never be considered again.

Assume that the action taken is **S**, and, w.l.o.g. that this is the rightmost **S** action taken before a non-**S** action happens. Note that it is possible that there is a relation between a and b, but we cannot perform **R** or **L** now. Therefore, we should consider (a, b) again only if a child of a or b has changed. When *Step Back* is used, we will consider (a, b) again only if the next action is **L**. (If next action is **R**, b will be eliminated.) This is true because the focus point will move back after performing **L**, which implies that b has a new child so we are indeed in a new situation. Since, from Lemma 1, the algorithm only requires one round. we therefore consider (a, b) again only if the situation has changed. □

2.3 Improving the Parsing Action Set

In order to improve the accuracy of the action predictors, we suggest a new (hierarchical) set of actions: *Shift, Left, Right, WaitLeft, WaitRight*. We believe that predicting these is easier due to finer granularity – the **S** action is broken to sub-actions in a natural way.

W*ait***L***eft*: $a < b$. a is the parent of b, but it's possible that b is a parent of other nodes. Action is deferred. If we perform **L***eft* instead, the child of b can not find its parents later.

W*ait***R***ight*: $a < b$. b is the parent of a, but it's possible that a is a parent of other nodes. Similar to **WL**, action is deferred.

Thus, we also change the algorithm to perform **S** only if there is no relationship between a and b[2]. The new set of actions is shown to better support our parsing algorithm, when tested on different placement policies. When **W***ait***L***eft* or **W***ait***R***ight* is performed, the focus will move to the next word. It is very interesting to notice that **W***ait***R***ight* is not needed in projective languages if **Step Back** is used. This give us another strong reason to use **Step Back**, since the classification becomes more accurate – a more natural class of actions, with a smaller number of candidate actions.

Once the parsing algorithm, along with the focus point policy, is determined, we can train the action classifiers. Given an annotated corpus, the parsing algorithm is used to determine the action taken for each consecutive pair; this is used to train a classifier to predict one of the five actions. The details of the classifier and the feature used are given in Section 4.

When the learned model is evaluated on new data, the sentence is processed left to right and the parsing algorithm, along with the action classifier, are used to produce the dependency tree. The evaluation process is somewhat more involved, since the action classifier is not used as is, but rather via a look ahead inference step described next.

3 A Pipeline Model with Look Ahead

The advantage of a pipeline model is that it can use more information, based on the outcomes of previous predictions. As discussed earlier, this may result in accumulating error. The importance of having a reliable action predictor in a pipeline model motivates the following approach. We devise a look ahead algorithm and use it as a look ahead policy, when determining the predicted action.

This approach can be used in any pipeline model but we illustrate it below in the context of our dependency parser.

The following example illustrates a situation in which an early mistake in predicting an action causes a chain reaction and results in further mistakes. This stresses the importance of correct early decisions, and motivates our look ahead policy.

Let (w, x, y, z) be a sentence of four words, and assume that the correct dependency relations are as shown in the top part of Figure 1. If the system mistakenly predicts that x is a child of w before y and z becomes x's children, we can only consider the relationship between w and y in the next stage. Consequently, we will never find the correct parent for y and z. The previous prediction error propagates and impacts future predictions. On the other hand, if the algorithm makes a correct prediction, in the next stage, we do not need to consider w and y. As shown, getting useful rather than misleading information in a pipeline model, requires correct early predictions. Therefore, it is necessary to utilize some inference framework to that may help resolving the error accumulation problem.

In order to improve the accuracy of the action prediction, we might want to examine all possible combinations of action sequences and choose the one that maximizes some score. It is clearly in-

[2]Interestingly, (Yamada and Matsumoto, 2003) mention the possibility of an additional *single* **W***ait* action, but do not add it to the model.

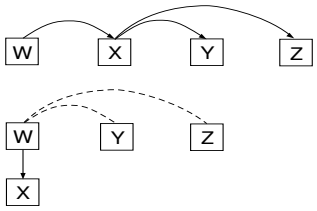


Figure 1: Top figure: the correct dependency relations between w, x, y and z. Bottom figure: if the algorithm mistakenly decides that x is a child of w before deciding that y and z are x's children, we cannot find the correct parent for y and z.

tractable to find the global optimal prediction sequences in a pipeline model of the depth we consider. Therefore, we use a look ahead strategy, implemented via a local search framework, which uses additional information but is still tractable.

The local search algorithm is presented in Algorithm 3. The algorithm accepts three parameters, *model*, *depth* and *State*. We assume a classifier that can give a confidence in its prediction. This is represented here by *model*.

As our learning algorithm we use a regularized variation of the perceptron update rule, as incorporated in SNoW (Roth, 1998; Carlson et al., 1999), a multi-class classifier that is tailored for large scale learning tasks and has been used successfully in a large number of NLP tasks (e.g., (Punyakanok et al., 2005)). SNoW uses softmax over the raw activation values as its confidence measure, which can be shown to produce a reliable approximation of the labels' conditional probabilities.

The parameter *depth* is to determine the depth of the search procedure. *State* encodes the configuration of the environment (in the context of the dependency parsing this includes the sentence, the focus point and the current parent and children for each word). Note that *State* changes when a prediction is made and that the features extracted for the action classifier also depend on *State*.

The search algorithm will perform a search of length *depth*. Additive scoring is used to score the sequence, and the first action in this sequence is selected and performed. Then, the *State* is updated, the new features for the action classifiers are computed and *search* is called again.

One interesting property of this framework is that it allows that use of future information in addition to past information. The pipeline model naturally allows access to all the past information.

Algorithm 3 Pseudo code for the look ahead algorithm. $\mathbf{y}$ represents a action sequence. The function *search* considers all possible action sequences with $|depth|$ actions and returns the sequence with the highest score.

Algo *predictAction*(*model*, *depth*, *State*)
x = getNextFeature(*State*)
$\mathbf{y}$ = *search*(x, *depth*, *model*, *State*)
lab $= \mathbf{y}[1]$
State = *update*(*State*, *lab*)
return lab

Algo *search*(x, *depth*, *model*, *State*)
maxScore $= -\infty$
$F = \{\mathbf{y} \mid \|\mathbf{y}\| = depth\}$
for $\mathbf{y}$ in F **do**
  $s = 0$, *TmpState* = *State*
  for $i = 1 \ldots depth$ **do**
    x = getNextFeature(*TmpState*)
    $s = s+$ score($\mathbf{y}[i]$, x, model)
    TmpState = *update*(*TmpState*, $\mathbf{y}[i]$)
  end for
  if $s >$ *maxScore* **then**
    $\hat{\mathbf{y}} = \mathbf{y}$
    maxScore = s
  end if
end for
return $\hat{\mathbf{y}}$

Since the algorithm uses a look ahead policy, it also uses future predictions. The significance of this becomes clear in Section 4.

There are several parameters, in addition to *depth* that can be used to improve the efficiency of the framework. For example, given that the action predictor is a multi-class classifier, we do not need to consider all future possibilities in order to decide the current action. For example, in our experiments, we only consider two actions with highest score at each level (which was shown to produce almost the same accuracy as considering all four actions).

4 Experiments and Results

We use the standard corpus for this task, the Penn Treebank (Marcus et al., 1993). The training set consists of sections 02 to 21 and the testing set is section 23. The POS tags for the evaluation data sets were provided by the tagger of (Toutanova et al., 2003) (which has an accuracy of 97.2% section

23 of the Penn Treebank).

4.1 Features for Action Classification

For each word pair (w_1, w_2) we use the words, their POS tags and also these features of the children of w_1 and w_2. We also include the lexicon and POS tags of 2 words before w_1 and 4 words after w_2 (as in (Yamada and Matsumoto, 2003)). The key additional feature we use, relative to (Yamada and Matsumoto, 2003), is that we include the previous predicted action as a feature. We also add conjunctions of above features to ensure expressiveness of the model. (Yamada and Matsumoto, 2003) makes use of polynomial kernels of degree 2 which is equivalent to using even more conjunctive features. Overall, the average number of active features in an example is about 50.

4.2 Evaluation

We use the same evaluation metrics as in (McDonald et al., 2005). Dependency accuracy (DA) is the proportion of non-root words that are assigned the correct head. Complete accuracy (CA) indicates the fraction of sentences that have a complete correct analysis. We also measure that root accuracy (RA) and leaf accuracy (LA), as in (Yamada and Matsumoto, 2003). When evaluating the result, we exclude the punctuation marks, as done in (McDonald et al., 2005) and (Yamada and Matsumoto, 2003).

4.3 Results

We present the results of several of the experiments that were intended to help us analyze and understand several of the design decisions in our pipeline algorithm.

To see the effect of the additional action, we present in Table 2 a comparison between a system that does not have the $WaitLeft$ action (similar to the (Yamada and Matsumoto, 2003) approach) with one that does. In both cases, we do not use the look ahead procedure. Note that, as stated above, the action $WaitRight$ is never needed for our parsing algorithm. It is clear that adding $WaitLeft$ increases the accuracy significantly.

Table 3 investigates the effect of the look ahead, and presents results with different *depth* parameters (*depth*$= 1$ means "no search"), showing a consistent trend of improvement.

Table 4 breaks down the results as a function of the sentence length; it is especially noticeable that the system also performs very well for long sentences, another indication for its global performance robustness.

method	DA	RA	CA	LA
w/o $WaitLeft$	90.27	90.73	39.28	93.87
w $WaitLeft$	90.53	90.76	39.74	93.94

Table 2: The significant of the action $WaitLeft$.

method	DA	RA	CA	LA
depth=1	90.53	90.76	39.74	93.94
depth=2	90.67	91.51	40.23	93.96
depth=3	90.69	92.05	40.52	93.94
depth=4	90.79	92.26	40.68	93.95

Table 3: The effect of different *depth* settings.

Table 5 shows the results with three settings of the POS tagger. The best result is, naturally, when we use the gold standard also in testing. However, it is worthwhile noticing that it is better to train with the same POS tagger available in testing, even if its performance is somewhat lower.

Table 6 compares the performances of several of the state of the art dependency parsing systems with ours. When comparing with other dependency parsing systems it is especially worth noticing that our system gives significantly better accuracy on completely parsed sentences.

Interestingly, in the experiments, we allow the parsing algorithm to run many rounds to parse a sentece in the testing stage. However, we found that over 99% sentences can be parsed in a single round. This supports for our justification about the correctness of our model.

5 Further Work and Conclusion

We have addressed the problem of using learned classifiers in a pipeline fashion, where a task is decomposed into several stages and stage classifiers are used sequentially, where each stage may use the outcome of previous stages as its input. This is a common computational strategy in natural language processing and is known to suffer from error accumulation and an inability to correct mistakes in previous stages.

Sent. Len.	DA	RA	CA	LA
<11	93.4	96.7	85.2	94.6
11-20	92.4	93.7	56.1	94.7
21-30	90.4	91.8	32.5	93.4
31-40	90.4	89.8	16.8	94.0
>40	89.7	87.9	8.7	93.3

Table 4: The effect of sentences length. The experiment is done with *depth* $= 4$.

Train-Test	DA	RA	CA	LA
gold−pos	90.7	92.0	40.8	93.8
pos−pos	90.8	92.3	40.7	94.0
gold−gold	92.0	93.9	43.6	95.0

Table 5: Comparing different sources of POS tagging in a pipeline model. We set *depth*$= 4$ in all the experiments of this table.

System	DA	RA	CA	LA
Y&M03	90.3	91.6	38.4	93.5
N&S04	87.3	84.3	30.4	N/A
M&C&P05	90.9	94.2	37.5	N/A
Current Work	90.8	92.3	40.7	94.0

Table 6: The comparison between the current work with other dependency parsing systems.

We abstracted two natural principles, one which calls for making the local classifiers used in the computation more reliable and a second, which suggests to devise the pipeline algorithm in such a way that minimizes the number of decisions (actions) made.

We study this framework in the context of designing a *bottom up dependency parsing*. Not only we manage to use this framework to justify several design decisions, but we also show experimentally that following these results in improving the accuracy of the inferred trees relative to existing models. Interestingly, we can show that the trees produced by our algorithm are relatively good even for long sentences, and that our algorithm is doing especially well when evaluated globally, at a sentence level, where our results are significantly better than those of existing approaches – perhaps showing that the design goals were achieved.

Our future work includes trying to generalize this work to non-projective dependency parsing, as well as attempting to incorporate additional sources of information (e.g., shallow parsing information) into the pipeline process.

6 Acknowledgements

We thank Ryan McDonald for providing the annotated data set and to Vasin Punyakanok for useful comments and suggestions.

This research is supported by the Advanced Research and Development Activity (ARDA)'s Advanced Question Answering for Intelligence (AQUAINT) Program and a DOI grant under the Reflex program.

References

A. V. Aho, R. Sethi, and J. D. Ullman. 1986. Compilers: Principles, techniques, and tools. In *Addison-Wesley Publishing Company, Reading, MA*.

A. Carlson, C. Cumby, J. Rosen, and D. Roth. 1999. The SNoW learning architecture. Technical Report UIUCDCS-R-99-2101, UIUC Computer Science Department, May.

J. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proc. the International Conference on Computational Linguistics (COLING)*, pages 340–345, Copenhagen, August.

A. Haghighi, A. Ng, and C. Manning. 2005. Robust textual inference via graph matching. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 387–394, Vancouver, British Columbia, Canada, October. Association for Computational Linguistics.

T. Marciniak and M. Strube. 2005. Beyond the pipeline: Discrete optimization in NLP. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 136–143, Ann Arbor, Michigan, June. Association for Computational Linguistics.

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

R. McDonald, K. Crammer, and F. Pereira. 2005. Online large-margin training of dependency parsers. In *Proc. of the Annual Meeting of the ACL*, pages 91–98, Ann Arbor, Michigan.

J. Nivre and M. Scholz. 2004. Deterministic dependency parsing of english text. In *COLING2004*, pages 64–70.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *IWPT*, Nancy, France.

V. Punyakanok, D. Roth, and W. Yih. 2005. The necessity of syntactic parsing for semantic role labeling. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1117–1123.

A. Ratnaparkhi. 1997. A linear observed time statistical parser based on maximum entropy models. In *EMNLP-97, The Second Conference on Empirical Methods in Natural Language Processing*, pages 1–10.

D. Roth and W. Yih. 2004. A linear programming formulation for global inference in natural language tasks. In Hwee Tou Ng and Ellen Riloff, editors, *Proc. of the Annual Conference on Computational Natural Language Learning (CoNLL)*, pages 1–8. Association for Computational Linguistics.

D. Roth. 1998. Learning to resolve natural language ambiguities: A unified approach. In *Proc. National Conference on Artificial Intelligence*, pages 806–813.

K. Toutanova, D. Klein, and C. Manning. "2003". Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of HLT-NAACL 03*.

H. Yamada and Y. Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *IWPT2003*.

H. Yamada. 2006. Private communication.

A Hybrid Convolution Tree Kernel for Semantic Role Labeling

Wanxiang Che
Harbin Inst. of Tech.
Harbin, China, 150001
car@ir.hit.edu.cn

Min Zhang
Inst. for Infocomm Research
Singapore, 119613
mzhang@i2r.a-star.edu.sg

Ting Liu, Sheng Li
Harbin Inst. of Tech.
Harbin, China, 150001
{tliu, ls}@ir.hit.edu.cn

Abstract

A hybrid convolution tree kernel is proposed in this paper to effectively model syntactic structures for semantic role labeling (SRL). The hybrid kernel consists of two individual convolution kernels: a Path kernel, which captures predicate-argument link features, and a Constituent Structure kernel, which captures the syntactic structure features of arguments. Evaluation on the datasets of CoNLL-2005 SRL shared task shows that the novel hybrid convolution tree kernel outperforms the previous tree kernels. We also combine our new hybrid tree kernel based method with the standard rich flat feature based method. The experimental results show that the combinational method can get better performance than each of them individually.

1 Introduction

In the last few years there has been increasing interest in Semantic Role Labeling (SRL). It is currently a well defined task with a substantial body of work and comparative evaluation. Given a sentence, the task consists of analyzing the propositions expressed by some target verbs and some constituents of the sentence. In particular, for each target verb (predicate) all the constituents in the sentence which fill a semantic role (argument) of the verb have to be recognized.

Figure 1 shows an example of a semantic role labeling annotation in PropBank (Palmer et al., 2005). The PropBank defines 6 main arguments, Arg0 is the *Agent*, Arg1 is *Patient*, etc. ArgM- may indicate adjunct arguments, such as *Locative*, *Temporal*.

Many researchers (Gildea and Jurafsky, 2002; Pradhan et al., 2005a) use feature-based methods

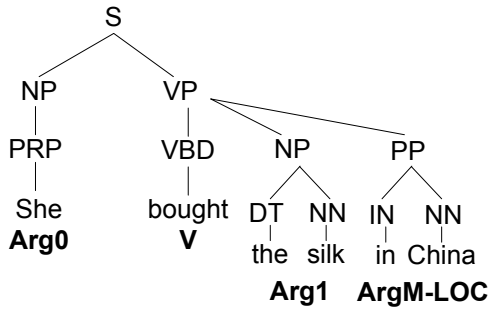


Figure 1: Semantic role labeling in a phrase structure syntactic tree representation

for argument identification and classification in building SRL systems and participating in evaluations, such as Senseval-3 [1], CoNLL-2004 and 2005 shared tasks: SRL (Carreras and Màrquez, 2004; Carreras and Màrquez, 2005), where a flat feature vector is usually used to represent a predicate-argument structure. However, it's hard for this kind of representation method to explicitly describe syntactic structure information by a vector of flat features. As an alternative, convolution tree kernel methods (Collins and Duffy, 2001) provide an elegant kernel-based solution to implicitly explore tree structure features by directly computing the similarity between two trees. In addition, some machine learning algorithms with dual form, such as Perceptron and Support Vector Machines (SVM) (Cristianini and Shawe-Taylor, 2000), which do not need know the exact presentation of objects and only need compute their kernel functions during the process of learning and prediction. They can be well used as learning algorithms in the kernel-based methods. They are named kernel machines.

In this paper, we decompose the Moschitti (2004)'s predicate-argument feature (PAF) kernel into a Path kernel and a Constituent Structure ker-

[1] http://www.cs.unt.edu/∼rada/senseval/senseval3/

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 73–80,
Sydney, July 2006.

nel, and then compose them into a hybrid convolution tree kernel. Our hybrid kernel method using Voted Perceptron kernel machine outperforms the PAF kernel in the development sets of CoNLL-2005 SRL shared task. In addition, the final composing kernel between hybrid convolution tree kernel and standard features' polynomial kernel outperforms each of them individually.

The remainder of the paper is organized as follows: In Section 2 we review the previous work. In Section 3 we illustrate the state of the art feature-based method for SRL. Section 4 discusses our method. Section 5 shows the experimental results. We conclude our work in Section 6.

2 Related Work

Automatic semantic role labeling was first introduced by Gildea and Jurafsky (2002). They used a linear interpolation method and extract features from a parse tree to identify and classify the constituents in the FrameNet (Baker et al., 1998) with syntactic parsing results. Here, the basic features include Phrase Type, Parse Tree Path, Position. Most of the following works focused on feature engineering (Xue and Palmer, 2004; Jiang et al., 2005) and machine learning models (Nielsen and Pradhan, 2004; Pradhan et al., 2005a). Some other works paid much attention to the robust SRL (Pradhan et al., 2005b) and post inference (Punyakanok et al., 2004).

These feature-based methods are considered as the state of the art method for SRL and achieved much success. However, as we know, the standard flat features are less effective to model the syntactic structured information. It is sensitive to small changes of the syntactic structure features. This can give rise to a data sparseness problem and prevent the learning algorithms from generalizing unseen data well.

As an alternative to the standard feature-based methods, kernel-based methods have been proposed to implicitly explore features in a high-dimension space by directly calculating the similarity between two objects using kernel function. In particular, the kernel methods could be effective in reducing the burden of feature engineering for structured objects in NLP problems. This is because a kernel can measure the similarity between two discrete structured objects directly using the original representation of the objects instead of explicitly enumerating their features.

Many kernel functions have been proposed in machine learning community and have been applied to NLP study. In particular, Haussler (1999) and Watkins (1999) proposed the best-known convolution kernels for a discrete structure. In the context of convolution kernels, more and more kernels for restricted syntaxes or specific domains, such as string kernel for text categorization (Lodhi et al., 2002), tree kernel for syntactic parsing (Collins and Duffy, 2001), kernel for relation extraction (Zelenko et al., 2003; Culotta and Sorensen, 2004) are proposed and explored in NLP domain. Of special interest here, Moschitti (2004) proposed Predicate Argument Feature (PAF) kernel under the framework of convolution tree kernel for SRL. In this paper, we follow the same framework and design a novel hybrid convolution kernel for SRL.

3 Feature-based methods for SRL

Usually feature-based methods refer to the methods which use the flat features to represent instances. At present, most of the successful SRL systems use this method. Their features are usually extended from Gildea and Jurafsky (2002)'s work, which uses flat information derived from a parse tree. According to the literature, we select the Constituent, Predicate, and Predicate-Constituent related features shown in Table 1.

Feature	Description
Constituent related features	
Phrase Type	syntactic category of the constituent
Head Word	head word of the constituent
Last Word	last word of the constituent
First Word	first word of the constituent
Named Entity	named entity type of the constituent's head word
POS	part of speech of the constituent
Previous Word	sequence previous word of the constituent
Next Word	sequence next word of the constituent
Predicate related features	
Predicate	predicate lemma
Voice	grammatical voice of the predicate, either active or passive
SubCat	Sub-category of the predicate's parent node
Predicate POS	part of speech of the predicate
Suffix	suffix of the predicate
Predicate-Constituent related features	
Path	parse tree path from the predicate to the constituent
Position	the relative position of the constituent and the predicate, before or after
Path Length	the nodes number on the parse tree path
Partial Path	some part on the parse tree path
Clause Layers	the clause layers from the constituent to the predicate

Table 1: Standard flat features

However, to find relevant features is, as usual, a complex task. In addition, according to the description of the standard features, we can see that the syntactic features, such as Path, Path Length, bulk large among all features. On the other hand, the previous researches (Gildea and Palmer, 2002; Punyakanok et al., 2005) have also recognized the

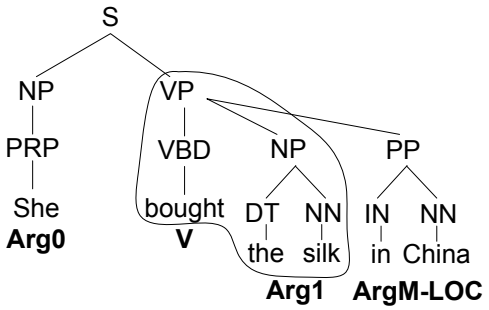


Figure 2: Predicate Argument Feature space

necessity of syntactic parsing for semantic role labeling. However, the standard flat features cannot model the syntactic information well. A predicate-argument pair has two different Path features even if their paths differ only for a node in the parse tree. This data sparseness problem prevents the learning algorithms from generalizing unseen data well. In order to address this problem, one method is to list all sub-structures of the parse tree. However, both space complexity and time complexity are too high for the algorithm to be realized.

4 Hybrid Convolution Tree Kernels for SRL

In this section, we introduce the previous kernel method for SRL in Subsection 4.1, discuss our method in Subsection 4.2 and compare our method with previous work in Subsection 4.3.

4.1 Convolution Tree Kernels for SRL

Moschitti (2004) proposed to apply convolution tree kernels (Collins and Duffy, 2001) to SRL. He selected portions of syntactic parse trees, which include salient sub-structures of predicate-arguments, to define convolution kernels for the task of predicate argument classification. This portions selection method of syntactic parse trees is named as predicate-arguments feature (PAF) kernel. Figure 2 illustrates the PAF kernel feature space of the predicate *buy* and the argument **Arg1** in the circled sub-structure.

The kind of convolution tree kernel is similar to Collins and Duffy (2001)'s tree kernel except the sub-structure selection strategy. Moschitti (2004) only selected the relative portion between a predicate and an argument.

Given a tree portion instance defined above, we design a convolution tree kernel in a way similar to the parse tree kernel (Collins and Duffy, 2001). Firstly, a parse tree T can be represented by a vector of integer counts of each sub-tree type (regardless of its ancestors):

$$\begin{aligned}\Phi(T) \quad &= (\#\text{ of sub-trees of type } 1, \ldots, \\ &\quad \#\text{ of sub-trees of type } i, \ldots, \\ &\quad \#\text{ of sub-trees of type } n)\end{aligned}$$

This results in a very high dimension since the number of different subtrees is exponential to the tree's size. Thus it is computationally infeasible to use the feature vector $\Phi(T)$ directly. To solve this problem, we introduce the tree kernel function which is able to calculate the dot product between the above high-dimension vectors efficiently. The kernel function is defined as following:

$$\begin{aligned}K(T_1, T_2) \quad &= \langle \Phi(T_1), \Phi(T_2) \rangle = \textstyle\sum_i \phi_i(T_1), \phi_i(T_2) \\ &= \textstyle\sum_{n_1 \in N_1} \sum_{n_2 \in N_2} \sum_i I_i(n_1) * I_i(n_2)\end{aligned}$$

where N_1 and N_2 are the sets of all nodes in trees T_1 and T_2, respectively, and $I_i(n)$ is the indicator function whose value is 1 if and only if there is a sub-tree of type i rooted at node n and 0 otherwise. Collins and Duffy (2001) show that $K(T_1, T_2)$ is an instance of convolution kernels over tree structures, which can be computed in $O(|N_1| \times |N_2|)$ by the following recursive definitions (Let $\Delta(n_1, n_2) = \sum_i I_i(n_1) * I_i(n_2)$):

(1) if the children of n_1 and n_2 are different then $\Delta(n_1, n_2) = 0$;

(2) else if their children are the same and they are leaves, then $\Delta(n_1, n_2) = \mu$;

(3) else $\Delta(n_1, n_2) = \mu \prod_{j=1}^{nc(n_1)} (1 + \Delta(ch(n_1, j), ch(n_2, j)))$

where $nc(n_1)$ is the number of the children of n_1, $ch(n, j)$ is the j^{th} child of node n and $\mu (0 < \mu < 1)$ is the decay factor in order to make the kernel value less variable with respect to the tree sizes.

4.2 Hybrid Convolution Tree Kernels

In the PAF kernel, the feature spaces are considered as an integral portion which includes a predicate and one of its arguments. We note that the PAF feature consists of two kinds of features: one is the so-called parse tree *Path* feature and another one is the so-called *Constituent Structure* feature. These two kinds of feature spaces represent different information. The Path feature describes the

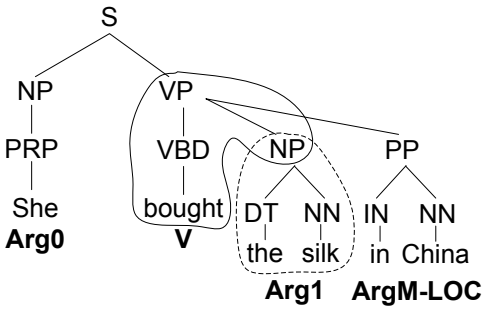
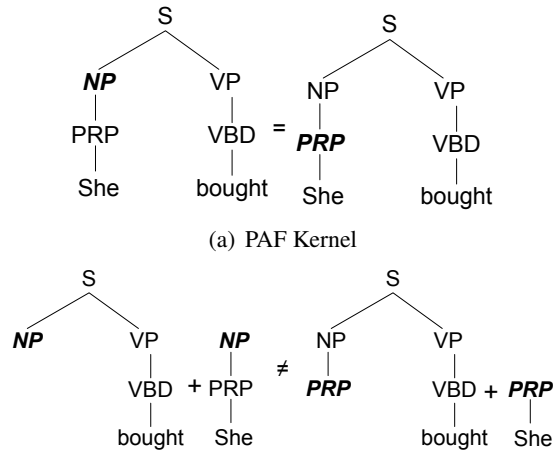


Figure 3: Path and Constituent Structure feature spaces

linking information between a predicate and its arguments while the Constituent Structure feature captures the syntactic structure information of an argument. We believe that it is more reasonable to capture the two different kinds of features separately since they contribute to SRL in different feature spaces and it is better to give different weights to fuse them. Therefore, we propose two convolution kernels to capture the two features, respectively and combine them into one hybrid convolution kernel for SRL. Figure 3 is an example to illustrate the two feature spaces, where the Path feature space is circled by solid curves and the Constituent Structure feature spaces is circled by dotted curves. We name them Path kernel and Constituent Structure kernel respectively.

Figure 4 illustrates an example of the distinction between the PAF kernel and our kernel. In the PAF kernel, the tree structures are equal when considering constitutes ***NP*** and ***PRP***, as shown in Figure 4(a). However, the two constituents play different roles in the sentence and should not be looked as equal. Figure 4(b) shows the computing example with our kernel. During computing the hybrid convolution tree kernel, the NP–PRP substructure is not computed. Therefore, the two trees are distinguished correctly.

On the other hand, the constituent structure feature space reserves the most part in the traditional PAF feature space usually. Then the Constituent Structure kernel plays the main role in PAF kernel computation, as shown in Figure 5. Here, *believes* is a predicate and **A1** is a long sub-sentence. According to our experimental results in Section 5.2, we can see that the Constituent Structure kernel does not perform well. Affected by this, the PAF kernel cannot perform well, either. However, in our hybrid method, we can adjust the compromise



(a) PAF Kernel

(b) Hybrid Convolution Tree Kernel

Figure 4: Comparison between PAF and Hybrid Convolution Tree Kernels

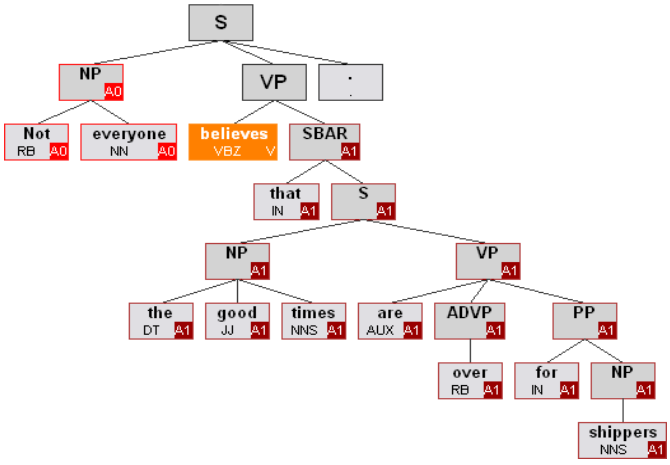


Figure 5: An example of Semantic Role Labeling

of the Path feature and the Constituent Structure feature by tuning their weights to get an optimal result.

Having defined two convolution tree kernels, the Path kernel K_{path} and the Constituent Structure kernel K_{cs}, we can define a new kernel to compose and extend the individual kernels. According to Joachims et al. (2001), the kernel function set is closed under linear combination. It means that the following K_{hybrid} is a valid kernel if K_{path} and K_{cs} are both valid.

$$K_{hybrid} = \lambda K_{path} + (1 - \lambda) K_{cs} \qquad (1)$$

where $0 \leq \lambda \leq 1$.

According to the definitions of the Path and the Constituent Structure kernels, each kernel is explicit. They can be viewed as a matching of fea-

tures. Since the features are enumerable on the given data, the kernels are all valid. Therefore, the new kernel K_{hybrid} is valid. We name the new kernel hybrid convolution tree kernel, K_{hybrid}.

Since the size of a parse tree is not constant, we normalize $K(T_1, T_2)$ by dividing it by $\sqrt{K(T_1, T_1) \cdot K(T_2, T_2)}$

4.3 Comparison with Previous Work

It would be interesting to investigate the differences between our method and the feature-based methods. The basic difference between them lies in the instance representation (parse tree vs. feature vector) and the similarity calculation mechanism (kernel function vs. dot-product). The main difference between them is that they belong to different feature spaces. In the kernel methods, we implicitly represent a parse tree by a vector of integer counts of each sub-tree type. That is to say, we consider all the sub-tree types and their occurring frequencies. In this way, on the one hand, the predicate-argument related features, such as Path, Position, in the flat feature set are embedded in the Path feature space. Additionally, the Predicate, Predicate POS features are embedded in the Path feature space, too. The constituent related features, such as Phrase Type, Head Word, Last Word, and POS, are embedded in the Constituent Structure feature space. On the other hand, the other features in the flat feature set, such as Named Entity, Previous, and Next Word, Voice, SubCat, Suffix, are not contained in our hybrid convolution tree kernel. From the syntactic viewpoint, the tree representation in our feature space is more robust than the Parse Tree Path feature in the flat feature set since the Path feature is sensitive to small changes of the parse trees and it also does not maintain the hierarchical information of a parse tree.

It is also worth comparing our method with the previous kernels. Our method is similar to the Moschitti (2004)'s predicate-argument feature (PAF) kernel. However, we differentiate the Path feature and the Constituent Structure feature in our hybrid kernel in order to more effectively capture the syntactic structure information for SRL. In addition Moschitti (2004) only study the task of argument classification while in our experiment, we report the experimental results on both identification and classification.

5 Experiments and Discussion

The aim of our experiments is to verify the effectiveness of our hybrid convolution tree kernel and and its combination with the standard flat features.

5.1 Experimental Setting

5.1.1 Corpus

We use the benchmark corpus provided by CoNLL-2005 SRL shared task (Carreras and Màrquez, 2005) provided corpus as our training, development, and test sets. The data consist of sections of the Wall Street Journal (WSJ) part of the Penn TreeBank (Marcus et al., 1993), with information on predicate-argument structures extracted from the PropBank corpus (Palmer et al., 2005). We followed the standard partition used in syntactic parsing: sections 02-21 for training, section 24 for development, and section 23 for test. In addition, the test set of the shared task includes three sections of the Brown corpus. Table 2 provides counts of sentences, tokens, annotated propositions, and arguments in the four data sets.

	Train	Devel	tWSJ	tBrown
Sentences	39,832	1,346	2,416	426
Tokens	950,028	32,853	56,684	7,159
Propositions	90,750	3,248	5,267	804
Arguments	239,858	8,346	14,077	2,177

Table 2: Counts on the data set

The preprocessing modules used in CONLL-2005 include an SVM based POS tagger (Giménez and Màrquez, 2003), Charniak (2000)'s full syntactic parser, and Chieu and Ng (2003)'s Named Entity recognizer.

5.1.2 Evaluation

The system is evaluated with respect to $precision$, $recall$, and $F_{\beta=1}$ of the predicted arguments. $Precision$ (p) is the proportion of arguments predicted by a system which are correct. $Recall$ (r) is the proportion of correct arguments which are predicted by a system. $F_{\beta=1}$ computes the harmonic mean of $precision$ and $recall$, which is the final measure to evaluate the performances of systems. It is formulated as: $F_{\beta=1} = 2pr/(p + r)$. *srl-eval.pl*[2] is the official program of the CoNLL-2005 SRL shared task to evaluate a system performance.

[2]http://www.lsi.upc.edu/∼srlconll/srl-eval.pl

5.1.3 SRL Strategies

We use constituents as the labeling units to form the labeled arguments. In order to speed up the learning process, we use a four-stage learning architecture:

Stage 1: To save time, we use a pruning stage (Xue and Palmer, 2004) to filter out the constituents that are clearly not semantic arguments to the predicate.

Stage 2: We then identify the candidates derived from Stage 1 as either arguments or non-arguments.

Stage 3: A multi-category classifier is used to classify the constituents that are labeled as arguments in Stage 2 into one of the argument classes plus NULL.

Stage 4: A rule-based post-processing stage (Liu et al., 2005) is used to handle some unmatched arguments with constituents, such as AM-MOD, AM-NEG.

5.1.4 Classifier

We use the Voted Perceptron (Freund and Schapire, 1998) algorithm as the kernel machine. The performance of the Voted Perceptron is close to, but not as good as, the performance of SVM on the same problem, while saving computation time and programming effort significantly. SVM is too slow to finish our experiments for tuning parameters.

The Voted Perceptron is a binary classifier. In order to handle multi-classification problems, we adopt the *one vs. others* strategy and select the one with the largest margin as the final output. The training parameters are chosen using development data. After 5 iteration numbers, the best performance is achieved. In addition, Moschitti (2004)'s Tree Kernel Tool is used to compute the tree kernel function.

5.2 Experimental Results

In order to speed up the training process, in the following experiments, we **ONLY** use WSJ sections 02-05 as training data. The same as Moschitti (2004), we also set the $\mu = 0.4$ in the computation of convolution tree kernels.

In order to study the impact of λ in hybrid convolution tree kernel in Eq. 1, we only use the hybrid kernel between K_{path} and K_{cs}. The performance curve on development set changing with λ is shown in Figure 6.

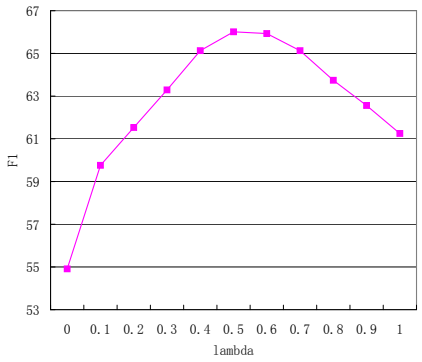


Figure 6: The performance curve changing with λ

The performance curve shows that when $\lambda = 0.5$, the hybrid convolution tree kernel gets the best performance. Either the Path kernel ($\lambda = 1$, $F_{\beta=1} = 61.26$) or the Constituent Structure kernel ($\lambda = 0$, $F_{\beta=1} = 54.91$) cannot perform better than the hybrid one. It suggests that the two individual kernels are complementary to each other. In addition, the Path kernel performs much better than the Constituent Structure kernel. It indicates that the predicate-constituent related features are more effective than the constituent features for SRL.

Table 3 compares the performance comparison among our Hybrid convolution tree kernel, Moschitti (2004)'s PAF kernel, standard flat features with Linear kernels, and Poly kernel ($d = 2$). We can see that our hybrid convolution tree kernel outperforms the PAF kernel. It empirically demonstrates that the weight linear combination in our hybrid kernel is more effective than PAF kernel for SRL.

However, our hybrid kernel still performs worse than the standard feature based system. This is simple because our kernel only use the syntactic structure information while the feature-based method use a large number of hand-craft diverse features, from word, POS, syntax and semantics, NER, etc. The standard features with polynomial kernel gets the best performance. The reason is that the arbitrary binary combination among features implicated by the polynomial kernel is useful to SRL. We believe that combining the two methods can perform better.

In order to make full use of the syntactic information and the standard flat features, we present a composite kernel between hybrid kernel (K_{hybrid}) and standard features with polynomial

	Hybrid	PAF	Linear	Poly
Devel	66.01	64.38	68.71	**70.25**

Table 3: Performance ($F_{\beta=1}$) comparison among various kernels

kernel (K_{poly}):

$$K_{comp} = \gamma K_{hybrid} + (1-\gamma)K_{poly} \quad (2)$$

where $0 \leq \gamma \leq 1$.

The performance curve changing with γ in Eq. 2 on development set is shown in Figure 7.

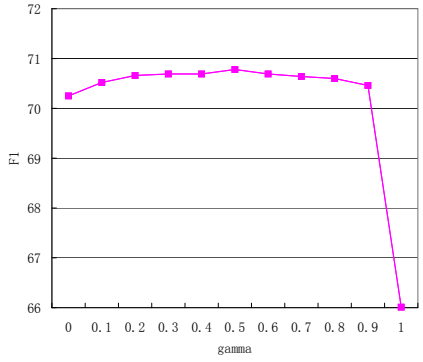


Figure 7: The performance curve changing with γ

We can see that when $\gamma = 0.5$, the system achieves the best performance and $F_{\beta=1} =$ **70.78**. It's statistically significant improvement (χ^2 test with $p = 0.1$) than only using the standard features with the polynomial kernel ($\gamma = 0$, $F_{\beta=1} = 70.25$) and much higher than only using the hybrid convolution tree kernel ($\gamma = 1$, $F_{\beta=1} = 66.01$). The main reason is that the convolution tree kernel can represent more general syntactic features than standard flat features, and the standard flat features include the features that the convolution tree kernel cannot represent, such as Voice, SubCat. The two kind features are complementary to each other.

Finally, we train the composite method using the above setting (Eq. 2 with when γ = 0.5) on the entire training set. The final performance is shown in Table 4.

	Precision	Recall	$F_{\beta=1}$
Development	80.71%	68.49%	74.10
Test WSJ	82.46%	70.65%	76.10
Test Brown	73.39%	57.01%	64.17

Test WSJ	Precision	Recall	$F_{\beta=1}$
Overall	82.46%	70.65%	76.10
A0	87.97%	82.49%	85.14
A1	80.51%	71.69%	75.84
A2	75.79%	52.16%	61.79
A3	80.85%	43.93%	56.93
A4	83.56%	59.80%	69.71
A5	100.00%	20.00%	33.33
AM-ADV	66.27%	43.87%	52.79
AM-CAU	68.89%	42.47%	52.54
AM-DIR	56.82%	29.41%	38.76
AM-DIS	79.02%	75.31%	77.12
AM-EXT	73.68%	43.75%	54.90
AM-LOC	72.83%	50.96%	59.97
AM-MNR	68.54%	42.44%	52.42
AM-MOD	98.52%	96.37%	97.43
AM-NEG	97.79%	96.09%	96.93
AM-PNC	49.32%	31.30%	38.30
AM-TMP	82.15%	68.17%	74.51
R-A0	86.28%	87.05%	86.67
R-A1	80.00%	74.36%	77.08
R-A2	100.00%	31.25%	47.62
R-AM-CAU	100.00%	50.00%	66.67
R-AM-EXT	50.00%	100.00%	66.67
R-AM-LOC	92.31%	57.14%	70.59
R-AM-MNR	20.00%	16.67%	18.18
R-AM-TMP	68.75%	63.46%	66.00
V	98.65%	98.65%	98.65

Table 4: Overall results (top) and detailed results on the WSJ test (bottom).

6 Conclusions and Future Work

In this paper we proposed the hybrid convolution kernel to model syntactic structure information for SRL. Different from the previous convolution tree kernel based methods, our contribution is that we distinguish between the Path and the Constituent Structure feature spaces. Evaluation on the datasets of CoNLL-2005 SRL shared task, shows that our novel hybrid convolution tree kernel outperforms the PAF kernel method. Although the hybrid kernel base method is not as good as the standard rich flat feature based methods, it can improve the state of the art feature-based methods by implicating the more generalizing syntactic information.

Kernel-based methods provide a good framework to use some features which are difficult to model in the standard flat feature based methods. For example the semantic similarity of words can be used in kernels well. We can use general purpose corpus to create clusters of similar words or use available resources like WordNet. We can also use the hybrid kernel method into other tasks, such as relation extraction in the future.

Acknowledgements

The authors would like to thank the reviewers for their helpful comments and Shiqi Zhao, Yanyan Zhao for their suggestions and useful discussions. This work was supported by National Natural Science Foundation of China (NSFC) via grant 60435020, 60575042, and 60503072.

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of the ACL-Coling-1998*, pages 86–90.

Xavier Carreras and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proceedings of CoNLL-2004*, pages 89–97.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of CoNLL-2005*, pages 152–164.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of NAACL-2000*.

Hai Leong Chieu and Hwee Tou Ng. 2003. Named entity recognition with a maximum entropy approach. In *Proceedings of CoNLL-2003*, pages 160–163.

Michael Collins and Nigel Duffy. 2001. Convolution kernels for natural language. In *Proceedings of NIPS-2001*.

Nello Cristianini and John Shawe-Taylor. 2000. *An Introduction to Support Vector Machines*. Cambridge University Press, Cambirdge University.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of ACL-2004*, pages 423–429.

Yoav Freund and Robert E. Schapire. 1998. Large margin classification using the perceptron algorithm. In *Computational Learning Theory*, pages 209–217.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Daniel Gildea and Martha Palmer. 2002. The necessity of parsing for predicate argument recognition. In *Proceedings of ACL-2002*, pages 239–246.

Jesús Giménez and Lluís Màrquez. 2003. Fast and accurate part-of-speech tagging: The svm approach revisited. In *Proceedings of RANLP-2003*.

David Haussler. 1999. Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, July.

Zheng Ping Jiang, Jia Li, and Hwee Tou Ng. 2005. Semantic argument classification exploiting argument interdependence. In *Proceedings of IJCAI-2005*.

Thorsten Joachims, Nello Cristianini, and John Shawe-Taylor. 2001. Composite kernels for hypertext categorisation. In *Proceedings of ICML-2001*, pages 250–257.

Ting Liu, Wanxiang Che, Sheng Li, Yuxuan Hu, and Huaijun Liu. 2005. Semantic role labeling system using maximum entropy classifier. In *Proceedings of CoNLL-2005*, pages 189–192.

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. 2002. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2):313–330.

Alessandro Moschitti. 2004. A study on convolution kernels for shallow statistic parsing. In *Proceedings of ACL-2004*, pages 335–342.

Rodney D. Nielsen and Sameer Pradhan. 2004. Mixing weak learners in semantic parsing. In *Proceedings of EMNLP-2004*.

Martha Palmer, Dan Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1).

Sameer Pradhan, Kadri Hacioglu, Valeri Krugler, Wayne Ward, James H. Martin, and Daniel Jurafsky. 2005a. Support vector learning for semantic argument classification. *Machine Learning Journal*.

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James Martin, and Daniel Jurafsky. 2005b. Semantic role labeling using different syntactic views. In *Proceedings of ACL-2005*, pages 581–588.

Vasin Punyakanok, Dan Roth, Wen-tau Yih, and Dav Zimak. 2004. Semantic role labeling via integer linear programming inference. In *Proceedings of Coling-2004*, pages 1346–1352.

Vasin Punyakanok, Dan Roth, and Wen tau Yih. 2005. The necessity of syntactic parsing for semantic role labeling. In *Proceedings of IJCAI-2005*, pages 1117–1123.

Chris Watkins. 1999. Dynamic alignment kernels. Technical Report CSD-TR-98-11, Jan.

Nianwen Xue and Martha Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of EMNLP 2004*.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106.

A High-Accurate Chinese-English NE Backward Translation System Combining Both Lexical Information and Web Statistics

Conrad Chen **Hsin-Hsi Chen**

Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan

drchen@nlg.csie.ntu.edu.tw hhchen@csie.ntu.edu.tw

Abstract

Named entity translation is indispensable in cross language information retrieval nowadays. We propose an approach of combining lexical information, web statistics, and inverse search based on *Google* to backward translate a Chinese named entity (NE) into English. Our system achieves a high Top-1 accuracy of 87.6%, which is a relatively good performance reported in this area until present.

1 Introduction

Translation of named entities (NE) attracts much attention due to its practical applications in World Wide Web. The most challenging issue behind is: the genres of NEs are various, NEs are open vocabulary and their translations are very flexible.

Some previous approaches use phonetic similarity to identify corresponding transliterations, i.e., translation by phonetic values (Lin and Chen, 2002; Lee and Chang, 2003). Some approaches combine lexical (phonetic and meaning) and semantic information to find corresponding translation of NEs in bilingual corpora (Feng et al., 2004; Huang et al., 2004; Lam et al., 2004). These studies focus on the alignment of NEs in parallel or comparable corpora. That is called "close-ended" NE translation.

In "open-ended" NE translation, an arbitrary NE is given, and we want to find its corresponding translations. Most previous approaches exploit web search engine to help find translating candidates on the Internet. Al-Onaizan and Knight (2003) adopt language models to generate possible candidates first, and then verify these candidates by web statistics. They achieve a Top-1 accuracy of about 72.6% with Arabic-to-English translation. Lu et al. (2004) use statistics of anchor texts in web search result to identify translation and obtain a Top-1 accuracy of about 63.6% in translating English out-of-vocabulary (OOV) words into Traditional Chinese. Zhang et al. (2005) use query expansion to retrieve candidates and then use lexical information, frequencies, and distances to find the correct translation. They achieve a Top-1 accuracy of 81.0% and claim that they outperform state-of-the-art OOV translation techniques then.

In this paper, we propose a three-step approach based on *Google* to deal with open-ended Chinese-to-English translation. Our system integrates various features which have been used by previous approaches in a novel way. We observe that most foreign Chinese NEs would have their corresponding English translations appearing in their returned snippets by *Google*. Therefore we combine lexical information and web statistics to find corresponding translations of given Chinese foreign NEs in returned snippets. A highly effective verification process, *inverse search*, is then adopted and raises the performance in a significant degree. Our approach achieves an overall Top-1 accuracy of 87.6% and a relatively high Top-4 accurracy of 94.7%.

2 Background

Translating NEs, which is different from translating common words, is an "asymmetric" translation. Translations of an NE in various languages can be organized as a tree according to the relations of translation language pairs, as shown in Figure 1. The root of the translating tree is the NE in its original language, i.e., initially de-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 81–88,
Sydney, July 2006.

nominated. We call the translation of an NE along the tree downward as a "*forward translation*". On the contrary, "*backward translation*" is to translate an NE along the tree upward.

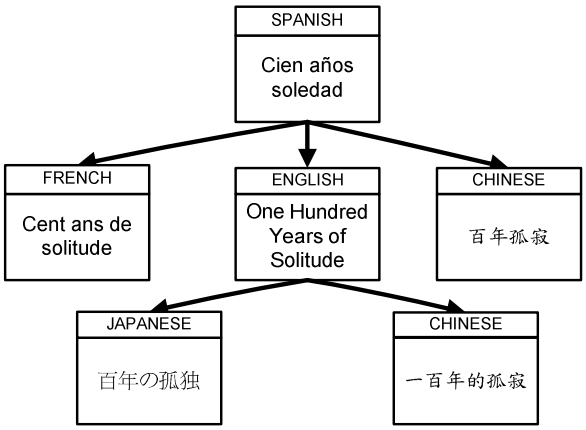


Figure 1. Translating tree of "Cien años soledad".

Generally speaking, forward translation is easier than backward translation. On the one hand, there is no unique answer to forward translation. Many alternative ways can be adopted to forward translate an NE from one language to another. For example, "Jordan" can be translated into "喬丹 (Qiao-Dan)", "喬登 (Qiao-Deng)", "約旦 (Yue-Dan)", and so on. On the other hand, there is generally one unique corresponding term in backward translation, especially when the target language is the root of the translating tree.

In addition, when the original NE appears in documents in the target language in forward translation, it often comes together with a corresponding translation in the target language (Cheng et al., 2004). That makes forward translation less challenging. In this paper, we focus our study on Chinese-English backward translation, i.e., the original language of NE and the target language in translation is English, and the source language to be translated is Chinese.

There are two important issues shown below to deal with backward translation of NEs or OOV words.

- Where to find the corresponding translation?
- How to identify the correct translation?

NEs seldom appear in multi-lingual or even mono-lingual dictionaries, i.e., they are OOV or unknown words. For unknown words, where can we find its corresponding translation? A bilingual corpus might be a possible solution. However, NEs appear in a vast context and bilingual corpora available can only cover a small proportion. Most text resources are monolingual. Can we find translations of NEs in monolingual corpora? While mentioning a translated name during writing, sometimes we would annotate it with its original name in the original foreign language, especially when the name is less commonly known. But how often would it happen? With our testing data, which would be introduced in Section 4, over 97% of translated NEs would have its original NE appearing in the first 100 returned snippets by *Google*. Figure 2 shows several snippets returned by *Google* which contains the original NE of the given foreign NE.

CEPS 思博網-- 文章書目;-1
篇名, 《老人與海》的象徵手法及作者的人生哲學. 並列篇名, Symbolic Means of the Author "The Old Man and the Sea" ... 摘要, 以象徵分析的方法對《老人與海》中老人、海、大魚等元素的象徵涵義進行了探索和解讀，分析了海明威在小說中闡述的主題："...
www.ceps.com.tw/ec/ecjnlarticleView.aspx?jnlcattype=1&jnlptype=4&jnltype=29&jnliid=1370&i... - 26k - 頁庫存檔 - 類似網頁

.:JSDVD Mall:. 世界名著-老人與海
世界名著-老人與海 · 太陽馬戲團-夢幻人生(DTS) · 紐約放電俏姐妹 · 懷舊電影系列 16-秋決 · 艾瑪 · 奪命訓練班 · 新好男孩-電視演唱會 · 神鬼認證-特別版 ... 世界名著-老人與海. The Old Man and The Sea. 4715320115018, 我們提供的付款方式 ...
mall.jsdvd.com/product_info.php?products_id=3198 - 48k - 補充資料 - 頁庫存檔 - 類似網頁

Figure 2. Several Traditional Chinese snippets of "老人與海" returned by *Google* which contains the translation "The Old Man and the Sea".

When translations can be found in snippets, the next work would be identifying which name is the correct translation of NEs. First we should know how NEs would be translated. The commonest case is translating by phonetic values, or so-called transliteration. Most personal names and location names are transliterated. NEs may also be translated by meaning. It is the way in which most titles and nicknames and some organization names would be translated. Another common case is translating by phonetic values for some parts and by meaning for the others. For example, "Sears Tower" is translated into "西爾斯 (Xi-Er-Si) 大廈 (tower)" in Chinese. NEs would sometimes be translated by semantics or contents of the entity it indicates, especially with movies. Table 1 summarizes the possible translating ways of NEs. From the above discussion, we may use similarities in phonetic values, meanings of constituent words, semantics, and so

on to identify corresponding translations. Besides these linguistic features, non-linguistic features such as statistical information may also help use well. We would discuss how to combine these features to identify corresponding translation in detail in the next section.

Translating Way	Description	Examples
Translating by Phonetic Values	The translation would have a similar pronunciation to its original NE.	"New York" and "紐約(pronounced as Niu-Yue)"
Translating by Meaning	The translation would have a similar or a related meaning to its original NE.	"紅(red)樓(chamber)夢(dream)" and "The Dream of the Red Chamber"
Translating by Phonetic Values for Some Parts and by Meaning for the Others	The entire NE is supposed to be translated by its meaning and the name parts are transliterated.	"Uncle Tom's Cabin" and "湯姆(pronounced as Tang-Mu)叔叔的(uncle's)小屋(cabin)"
Translating by Both Phonetic Values and Meaning	The translation would have both a similar pronunciation and a similar meaning to its original NE.	"New Yorker" and "紐約(pronounced as Niu-Yue)客(people, pronounced as Ke)"
Translating NEs by Heterography	The NE is translated by these heterographic words in neighboring languages.	"橫濱" and "Yokohama", "鈴木一朗" and "Ichiro Suzuki"
Translating by Semantic or Content	The NE is translated by its semantic or the content of the entity it refers to.	"The Mask" and "摩登(modern)大(great)聖(saint)"
Parallel Names	NE is initially denominated as more than one name or in more than one language.	"孫中山(Sun Zhong-Shan)" and "Sun Yat-Sen"

Table 1. Possible translating ways of NEs.

3 Chinese-to-English NE Translation

As we have mentioned in the last section, we could find most English translations in Chinese web page snippets. We thus base our system on web search engine: retrieving candidates from returned snippets, combining both linguistic and statistical information to find the correct translation. Our system can be split into three steps: candidate retrieving, candidate evaluating, and candidate verifying. An overview of our system is given in Figure 3.

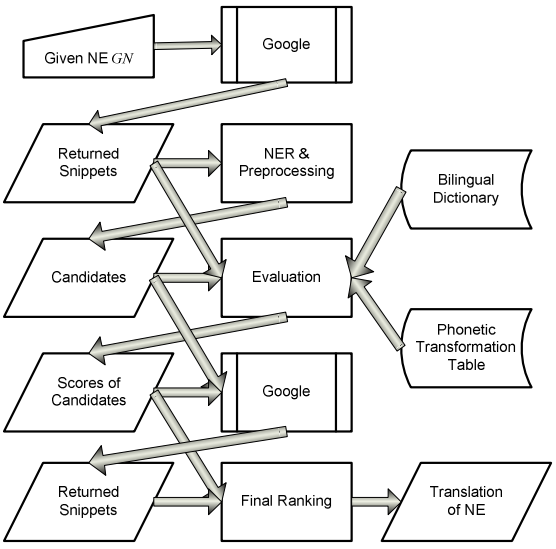


Figure 3. An Overview of the System.

In the first step, the NE to be translated, *GN*, is sent to *Google* to retrieve traditional Chinese web pages, and a simple English NE recognition method and several preprocessing procedures are applied to obtain possible candidates from returned snippets. In the second step, four features (i.e., phonetic values, word senses, recurrences, and relative positions) are exploited to give these candidates a score. In the last step, the candidates with higher scores are sent to *Google* again. Recurrence information and relative positions concerning with the candidate to be verified of *GN* in returned snippets are counted along with the scores to decide the final ranking of candidates. These three steps will be detailed in the following subsections.

3.1 Retrieving Candidates

Before we can identify possible candidates, we must retrieve them first. In the returned traditional Chinese snippets by *Google*, there are still many English fragments. Therefore, the first task our system would do is to separate these English fragments into NEs and non-NEs. We propose a simple method to recognize possible NEs. All fragments conforming to the following properties would be recognized as NEs:

- The first and the last word of the fragment are numerals or capitalized.
- There are no three or more consequent lowercase words in the fragment.
- The whole fragment is within one sentence.

After retrieving possible NEs in returned snippets, there are still some works to do to make a

finer candidate list for verification. First, there might be many different forms for a same NE. For example, "Mr. & Mrs. Smith" may also appear in the form of "Mr. and Mrs. Smith", "Mr. And Mrs. Smith", and so on. To deal with these aliasing forms, we transform all different forms into a standard form for the later ranking and identification. The standard form follows the following rules:

- All letters are transformed into upper cases.
- Words consist "'"s are split.
- Symbols are rewritten into words.

For example, all forms of "Mr. & Mrs. Smith" would be transformed into "MR. AND MRS. SMITH".

The second work we should complete before ranking is filtering useless substrings. An NE may comprise many single words. These component words may all be capitalized and thus all substrings of this NE would be fetched as candidates of our translation work. Therefore, substrings which always appear with a same preceding and following word are discarded here, since they would have a zero recurrence score in the next step, which would be detailed in the next subsection.

3.2 Evaluating Candidates

After candidate retrieving, we would obtain a sequence of m candidates, $C_1, C_2, \ldots, C_m$. An integrated evaluating model is introduced to exploit four features (phonetic values, word senses, recurrences, and relative positions) to score these m candidates, as the following equation suggests:

$$Score(C_i, GN) = SScore(C_i, GN) \cdot LScore(C_i, GN)$$

$LScore(C_i,GN)$ combines phonetic values and word senses to evaluate the lexical similarity between C_i and GN. $SScore(C_i,GN)$ concerns both recurrences information and relative positions to evaluate the statistical relationship between C_i and GN. These two scores are then combined to obtain $Score(C_i,GN)$. How to estimate $LScore(C_n, GN)$ and $SScore(C_n, GN)$ would be discussed in detail in the following subsections.

3.2.1 Lexical Similarity

The lexical similarity concerns both phonetic values and word senses. An NE may consist of many single words. These component words may be translated either by phonetic values or by word senses. Given a translation pair, we could split them into fragments which could be bipartite matched according to their translation relationships, as Figure 4 shows.

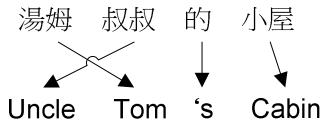


Figure 4. The translation relationships of "湯姆叔叔的小屋".

To identify the lexical similarity between two NEs, we could estimate the similarity scores between the matched fragment pairs first, and then sum them up as a total score. We postulate that the matching with the highest score is the correct matching. Therefore the problem becomes a weighted bipartite matching problem, i.e., given the similarity scores between any fragment pairs, to find the bipartite matching with the highest score. In this way, our next problem is how to estimate the similarity scores between fragments.

We treat an English single word as a fragment unit, i.e., each English single word corresponds to one fragment. An English candidate C_i consisting of n single words would be split into n fragment units, $C_{i1}, C_{i2}, \ldots, C_{in}$. We define a Chinese fragment unit that it could comprise one to four characters and may overlap each other. A fragment unit of GN can be written as GN_{ab}, which denotes the ath to bth characters of GN, and $b - a < 4$. The linguistic similarity score between two fragments is:

$$LSim(GN_{ab}, C_{ij}) = Max\{PVSim(GN_{ab}, C_{ij}), WSSim(GN_{ab}, C_{ij})\}$$

Where $PVSim()$ estimates the similarity in phonetic values while $WSSim()$ estimate it in word senses.

■ Phonetic Value

In this paper, we adopt a simple but novel method to estimate the similarity in phonetic values. Unlike many approaches, we don't introduce an intermediate phonetic alphabet system for comparison. We first transform the Chinese fragments into possible English strings, and then estimate the similarity between transformed strings and English candidates in surface strings, as Figure 5 shows. However, similar pronunciations does not equal to similar surface strings. Two quite dissimilar strings may have very similar pronunciations. Therefore, we take this strat-

egy: generate all possible transformations, and regard the one with the highest similarity as the English candidate.

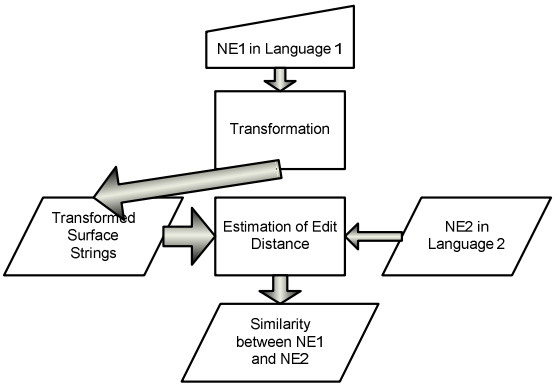


Figure 5. Phonetic similarity estimation of our system.

Edit distances are usually used to estimate the surface similarity between strings. However, the typical edit distance does not completely satisfy the requirement in the context of translation identification. In translation, vowels are an unreliable feature. There are many variations in pronunciation of vowels, and the combinations of vowels are numerous. Different combinations of vowels may have a same phonetic value, however, same combinations may pronounce totally differently. The worst of all, human often arbitrarily determine the pronunciation of unfamiliar vowel combinations in translation. For these reasons, we adopt the strategy that vowels can be ignored in transformation. That is to say when it is hard to determine which vowel combination should be generated from given Chinese fragments, we can only transform the more certain part of consonants. Thus during the calculation of edit distances, the insertion of vowels would not be calculated into edit distances. Finally, the modified edit distance between two strings A and B is defined as follow:

$$ED_{A\to B}(0,t) = t$$

$$ED_{A\to B}(s,0) = s$$

$$ED_{A\to B}(s,t) = \min\begin{Bmatrix} ED_{A\to B}(s,t-1) + Ins(t), \\ ED_{A\to B}(s-1,t) + 1, \\ ED_{A\to B}(s-1,t-1) + Rep(s,t) \end{Bmatrix}$$

$$Ins(t) = \begin{cases} 0, \text{if } B_t \text{ is a vowl} \\ 1, \text{if } B_t \text{ is a consonant} \end{cases}$$

$$Rep(s,t) = \begin{cases} 0, \text{if } A_s = B_t \\ 1, \text{else} \end{cases}$$

The modified edit distances are then transformed to similarity scores:

$$PVSim(A,B) = 1 - \frac{ED_{A\to B}(Len(A), Len(B))}{\max\{Len(A), Len(B)\}}$$

Len() denotes the length of the string. In the above equation, the similarity scores are ranged from 0 to 1.

We build the fixed transformation table manually. All possible transformations from Chinese transliterating characters to corresponding English strings are built. If we cannot precisely indicate which vowel combination should be transformed, or there are too many possible combinations, we ignores vowels. Then we use a training set of 3,000 transliteration names to examine possible omissions due to human ignorance.

■ Word Senses

More or less similar to the estimation of phonetic similarity, we do not use an intermediate representation of meanings to estimate word sense similarity. We treat the English translations in the C-E bilingual dictionary (*reference removed for blind review*) directly as the word senses of their corresponding Chinese word entries. We adopt a simple 0-or-1 estimation of word sense similarity between two strings A and B, as the following equation suggests:

$$WSSim(A,B) = \begin{cases} 0, \text{if } B \text{ is not a translation of } A \\ \quad \text{in the dictionary} \\ 1, \text{if } B \text{ is a translation of } A \\ \quad \text{in the dictionary} \end{cases}$$

All the Chinese foreign names appearing in test data is removed from the dictionary.

From the above equations we could derive that *LSim*() of fragment pairs is also ranged from 0 to 1. Candidates to be evaluated may comprise different number of component words, and this would result the different scoring base of the weighted bipartite matching. We should normalize the result scores of bipartite matching. As a result, the following equation is applied:

$$LScore(C_i, GN) = \min\left\{ \frac{\sum_{\text{all matched pairs } GN_{ab} \text{ and } C_{ij}} LSim(GN_{ab}, C_{ij})}{\text{Total \# of words in } C_i}, \frac{\sum_{\text{all matched pairs } GN_{ab} \text{ and } C_{ij}} LSim(GN_{ab}, C_{ij}) \cdot (b-a+1)}{\text{Total \# of characters in } GN} \right\}$$

3.2.2 Statistical Similarity

Two pieces of information are concerned together to estimate the statistical similarity: recurrences and relative positions. A candidate C_i might appear l times in the returned snippets, as $C_{i,1}$, $C_{i,2}$, …, $C_{i,l}$. For each $C_{i,k}$, we find the dis-

tance between it and the nearest *GN* in the returned snippets, and then compute the relative position scores as the following equation:

$$RP(C_{i,k}, GN) = \frac{1}{\lfloor Distance(GN, C_{i,k})/4 \rfloor + 1}$$

In other words, if the candidate is adjacent to the given NE, it would have a relative position score of 1. Relative position scores of all $C_{i,k}$ would be summed up to obtain the primitive statistical score:

$$PSS(C_i, GN) = \sum_k RP(C_{n,k}, GN)$$

As we mentioned before, since the imprecision of NE recognition, most substrings of NEs would also be recognized as candidates. This would result a problem. There are often typos in the information provided on the Internet. If some component word of an NE is misspelled, the substrings constituted by the rest words would have a higher statistical score than the correct NE. To prevent such kind of situations, we introduce entropy of the context of the candidate. If a candidate has a more varied context, it is more possible to be an independent term instead of a substring of other terms. Entropy provides such a property: if the possible cases are more varied, there is higher entropy, and vice versa. Entropy function here concerns the possible cases of the most adjacent word at both ends of the candidate, as the following equation suggests:

$$Entropy(\text{Context of } C_i) = \begin{cases} 1 & , \text{while \# of possible context} = 1 \\ -\sum_{CT_i} NCT_r / NC_i \cdot \log_{NPT_i} NCT_r / NC_i & , \text{else} \end{cases}$$

Where NCT_r and NC_i denote the appearing times of the *r*th context CT_r and the candidate C_i in the returned snippets respectively, and NPT_i denotes the total number of different cases of the context of C_i. Since we want to normalize the entropy to 0~1, we take NPT_i as the base of the logarithm function.

While concerning context combinations, only capitalized English word is discriminated. All other words would be viewed as one sort "OTHER". For example, assuming the context of "David" comprises three times of (Craig, OTHER), three times of (OTHER, Stern), and six times of (OTHER, OTHER), then:

$$Entropy(\text{Context of "David"}) = -(\frac{3}{12}\log_3\frac{3}{12} + \frac{3}{12}\cdot\log_3\frac{3}{12} + \frac{6}{12}\cdot\log_3\frac{6}{12}) = 0.946$$

Next we use *Entropy*(Context of C_i) to weight the primitive score *PSS*(C_i, *GN*) to obtain the final statistical score.:

$$SScore(C_i, GN) = Entropy(\text{Context of } C_i) \cdot PSS(C_i, GN)$$

3.3 Verifying Candidates

In evaluating candidate, we concern only the appearing frequencies of candidates when the NE to be translated is presented. In the other direction, we should also concern the appearing frequencies of the NE to be translated when the candidate is presented to prevent common words getting an improper high score in evaluation. We perform the *inverse search* approach for this sake. Like the evaluation of statistical scores in the last step, candidates are sent to *Google* to retrieve Traditional Chinese snippets, and the same equation of *SScore*() is computed concerning the candidate. However, since there are too many candidates, we cannot perform this process on all candidates. Therefore, an elimination mechanism is adopted to select candidates for verification. The elimination mechanism works as follows:

1. Send the Top-3 candidates into *Google* for verification.
2. Count *SScore*(*GN*, C_i). (Notice that the order of the parameter is reversed.) Re-weight *Score*(C_i, *GN*) by multiplying *SScore*(*GN*, C_i)
3. Re-rank candidates
4. After re-ranking, if new candidates become the Top-3 ones, redo the first step. Otherwise end this process.

The candidates have been verified would be recorded to prevent duplicate re-weighting and unnecessary verification.

There is one problem in verification we should concern. Since we only consider recurrence information in both directions, but not co-occurrence information, this would result some problem when dealing rarely used translations. For example, "Peter Pan" can be translated into "彼得潘" or "彼德潘" (both pronounced as Bi-De-Pan) in Chinese, but most people would use the former translation. Thus if we send "Peter Pan" to verification when translating "彼德潘", we would get a very low score.

To deal with this situation, we adopt the strategy of disbelieving verification in some situa-

tions. If all candidates have scores lower than the threshold, we presume that the given NE is a rarely used translation. In this situation, we use only *Score*(C_n, *GN*) estimated by the evaluation step to rank its candidates, without multiplying *SScore*(*GN*, C_i) of the inverse search. The threshold is set to 1.5 by heuristic, since we consider that a commonly used translation is supposed to have their *SScore*() larger than 1 in both directions.

4 Experiments

To evaluate the performance of our system, 15 common users are invited to provide 100 foreign NEs per user. These users are asked to simulate a scenario of using web search machine to perform cross-lingual information retrieval. The proportion of different types of NEs is roughly conformed to the real distribution, except for creation titles. We gathers a larger proportion of creation titles than other types of NEs, since the ways of translating creation titles is less regular and we may use them to test how much help could the web statistics provide.

After removing duplicate entries provided by users, finally we obtain 1,119 nouns. Among them 7 are not NEs, 65 are originated from Oriental languages (Chinese, Japanese, and Korean), and the rest 1,047 foreign NEs are our main experimental subjects. Among these 1,047 names there are 455 personal names, 264 location names, 117 organization names, 196 creation titles, and 15 other types of NEs.

Table 2 and Figure 5 show the performance of the system with different types of NEs. We could observe that the translating performance is best with location names. It is within our expectation, since location names are one of the most limited NE types. Human usually provide location names in a very limited range, and thus there are less location names having ambiguous translations and less rare location names in the test data. Besides, because most location names are purely transliterated, it can give us some clues about the performance of our phonetic model.

Our system performs worst with creation titles. One reason is that the naming and translating style of creation titles are less formulated. Many titles are not translated by lexical information, but by semantic information or else. For example, "Mr. & Mrs. Smith" is translated into "史密斯任務(Smiths' Mission)" by the content of the creation it denotes. Another reason is that many titles are not originated from English, such as "le Nozze di Figaro". It results the C-E bilingual dictionary cannot be used in recognizing word sense similarity. A more serious problem with titles is that titles generally consist of more single words than other types of NEs. Therefore, in the returned snippets by *Google*, the correct translation is often cut off. It would results a great bias in estimating statistical scores.

Table 3 compares the result of different feature combinations. It considers only foreign NEs in the test data. From the result we could conclude that both statistical and lexical features are helpful for translation finding, while the inverse search are the key of our system to achieve a good performance.

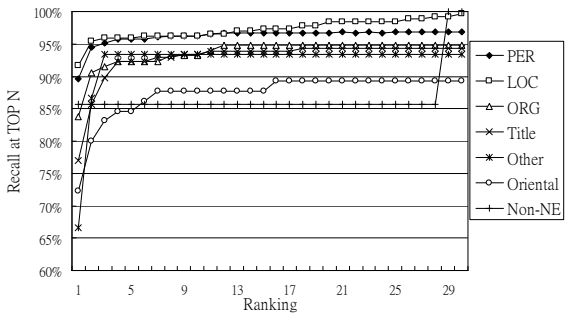


Figure 5. Curve of recall versus ranking.

	Total	Top-1		Top-2		Top-4		Top-M	
		Num	Recall	Num	Recall	Num	Recall	Num	Recall
PER	455	408	89.7%	430	94.5%	436	95.8%	443	97.3%
LOC	264	242	91.7%	252	95.5%	253	95.8%	264	100.0%
ORG	117	98	83.8%	106	90.6%	108	92.3%	114	97.4%
TITLE	196	151	77.0%	168	85.7%	181	92.3%	189	96.4%
Other	15	10	66.7%	13	86.7%	14	93.3%	15	100.0%
All NE	1047	909	87.6%	969	92.6%	992	94.7%	1025	97.9%
Oriental	65	47	72.3%	52	80.0%	55	84.6%	60	92.3%
Non-NE	7	6	85.7%	6	85.7%	6	85.7%	7	100.0%
Overall	1119	962	86.0%	1027	91.8%	1053	94.1%	1092	97.6%

Table 2. Experiment results of our system with different NE types.

	Top-1		Top-2		Top-4	
	Num	Recall	Num	Recall	Num	Recall
SScore	540	51.6%	745	71.2%	887	84.7%
LScore	721	68.9%	789	75.4%	844	80.6%
SScore + LScore	837	79.9%	916	87.5%	953	91.0%
+ Inverse Search	909	87.6%	969	92.6%	992	94.7%

Table 3. Experiment results of our system with different feature combinations.

From the result we could also find that our system has a high recall of 94.7% while considering top 4 candidates. If we only count in the given NEs with their correct translation appearing in the returned snippets, the recall would go to 96.8%. This achievement may be not yet good enough for computer-driven applications, but it is certainly a good performance for user querying.

5 Conclusion

In this study we combine several relatively simple implementations of approaches that have been proposed in the previous studies and obtain a very good performance. We find that the Internet is a quite good source for discovering NE translations. Using snippets returned by *Google* we can efficiently reduce the number of the possible candidates and acquire much useful information to verify these candidates. Since the number of candidates is generally less than processing with unaligned corpus, simple models can performs filtering quite well and the over-fitting problem is thus prevented.

From the failure cases of our system, (see Appendix A) we could observe that the performance of this integrated approach could still be boosted by more sophisticated models, more extensive dictionaries, and more delicate training mechanisms. For example, performing stemming or adopting a more extensive dictionary might enhance the accuracy of estimating word sense similarity; the statistic formula can be replaced by more formal measures such as co-occurrences or mutual information to make a more precise assessment of statistical relationship. These tasks would be our future works in developing a more accurate and efficient NE translation system.

Reference

Al-Onaizan, Yaser and Kevin Knight. 2002. Translating Named Entities Using Monolingual and Bilingual Resources. ACL 2002: 400-408.

Cheng, Pu-Jen, J.W. Teng, R.C. Chen, J.H. Wang, W.H. Lu, and L.F. Chien. Translating unknown queries with web corpora for cross-language information retrieval. SIGIR 2004: 146-153.

Feng, Donghui, Lv Y., and Zhou M. 2004. A New Approach for English-Chinese Named Entity Alignment. EMNLP 2004: 372-379.

Huang, Fei, Stephan Vogel, and Alex Waibel. 2003. Improving Named Entity Translation Combining Phonetic and Semantic Similarities. HLT-NAACL 2004: 281-288.

Lam, Wai, Ruizhang Huang, and Pik-Shan Cheung. 2004. Learning phonetic similarity for matching named entity translations and mining new translations. SIGIR 2004: 289-296.

Lee, Chun-Jen and Jason S. Chang. 2003. Acquisition of. English-Chinese Transliterated Word Pairs from Parallel-Aligned Texts. HLT-NAACL 2003. Workshop on Data Driven MT: 96-103.

Lin, Wei-Hao and Hsin-Hsi Chen. 2002. Backward Machine Transliteration by Learning Phonetic Similarity. *Proceedings of CoNLL-2002*: 139-145.

Lu, Wen-Hsiang, Lee-Feng Chien, and Hsi-Jian Lee. 2004. Anchor Text Mining for Translation of Web Queries: A Transitive Translation Approach. *ACM Transactions on Information Systems* 22(2): 242-269.

Zhang, Ying, Fei Huang, and Stephan Vogel. 2005. Mining translations of OOV terms from the web through cross-lingual query expansion. SIGIR 2005: 669-670.

Zhang, Ying and Phil Vines. 2004. Using the web for automated translation extraction in cross-language information retrieval. SIGIR 2004: 162-169.

Appendix A. Some Failure Cases of Our System

GN	Top 1	Correct Translation	Rank
海珊	CBS	SADDAM HUSSEIN	2
紐澤西	JERSEY	NEW JERSEY	2
天方夜譚	ONLINE	ARABIAN NIGHTS	2
勞斯萊斯	ROYCE	ROLLS ROYCE	2
朱利斯厄文	NBA	JULIUS ERVING	2
艾薇兒	LAVIGNE	AVRIL LAVIGNE	2
羅琳	JK	JK. ROWLING	2
塞爾蒂克	RICKY DAVIS	CELTICS	8
印象日出	MONET	IMPRESSION SUNRISE	9
蘇聯	TUPOLEV TU	USSR	33
梅德維登科	NBA	MEDVENDENKO	N/A
命運交響曲	TOS	SYMPHONY NO. 5	N/A
愛的教育	AROUND03	CUORE	N/A
民主黨	JACK LAYTON	DEMOCRATIC PARTY	N/A

Unsupervised Relation Disambiguation Using Spectral Clustering

Jinxiu Chen[1] **Donghong Ji**[1] **Chew Lim Tan**[2] **Zhengyu Niu**[1]
[1]Institute for Infocomm Research
21 Heng Mui Keng Terrace
119613 Singapore
{jinxiu,dhji,zniu}@i2r.a-star.edu.sg
[2]Department of Computer Science
National University of Singapore
117543 Singapore
tancl@comp.nus.edu.sg

Abstract

This paper presents an unsupervised learning approach to disambiguate various relations between name entities by use of various lexical and syntactic features from the contexts. It works by calculating eigenvectors of an adjacency graph's Laplacian to recover a submanifold of data from a high dimensionality space and then performing cluster number estimation on the eigenvectors. Experiment results on ACE corpora show that this spectral clustering based approach outperforms the other clustering methods.

1 Introduction

In this paper, we address the task of relation extraction, which is to find relationships between name entities in a given context. Many methods have been proposed to deal with this task, including supervised learning algorithms (Miller et al., 2000; Zelenko et al., 2002; Culotta and Soresen, 2004; Kambhatla, 2004; Zhou et al., 2005), semi-supervised learning algorithms (Brin, 1998; Agichtein and Gravano, 2000; Zhang, 2004), and unsupervised learning algorithm (Hasegawa et al., 2004).

Among these methods, supervised learning is usually more preferred when a large amount of labeled training data is available. However, it is time-consuming and labor-intensive to manually tag a large amount of training data. Semi-supervised learning methods have been put forward to minimize the corpus annotation requirement. Most of semi-supervised methods employ the bootstrapping framework, which only need to pre-define some initial seeds for any particular relation, and then bootstrap from the seeds to acquire the relation. However, it is often quite difficult to enumerate all class labels in the initial seeds and decide an "optimal" number of them.

Compared with supervised and semi-supervised methods, Hasegawa et al. (2004)'s unsupervised approach for relation extraction can overcome the difficulties on requirement of a large amount of labeled data and enumeration of all class labels. Hasegawa et al. (2004)'s method is to use a hierarchical clustering method to cluster pairs of named entities according to the similarity of context words intervening between the named entities. However, the drawback of hierarchical clustering is that it required providing cluster number by users. Furthermore, clustering is performed in original high dimensional space, which may induce non-convex clusters hard to identified.

This paper presents a novel application of spectral clustering technique to unsupervised relation extraction problem. It works by calculating eigenvectors of an adjacency graph's Laplacian to recover a submanifold of data from a high dimensional space, and then performing cluster number estimation on a transformed space defined by the first few eigenvectors. This method may help us find non-convex clusters. It also does not need to pre-define the number of the context clusters or pre-specify the similarity threshold for the clusters as Hasegawa et al. (2004)'s method.

The rest of this paper is organized as follows. Section 2 formulates unsupervised relation extraction and presents how to apply the spectral clustering

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 89–96,
Sydney, July 2006.

technique to resolve the task. Then section 3 reports experiments and results. Finally we will give a conclusion about our work in section 4.

2 Unsupervised Relation Extraction Problem

Assume that two occurrences of entity pairs with similar contexts, are tend to hold the same relation type. Thus unsupervised relation extraction problem can be formulated as partitioning collections of entity pairs into clusters according to the similarity of contexts, with each cluster containing only entity pairs labeled by the same relation type. And then, in each cluster, the most representative words are identified from the contexts of entity pairs to induce the label of relation type. Here, we only focus on the clustering subtask and do not address the relation type labeling subtask.

In the next subsections we will describe our proposed method for unsupervised relation extraction, which includes: 1) Collect the context vectors in which the entity mention pairs co-occur; 2) Cluster these Context vectors.

2.1 Context Vector and Feature Design

Let $X = \{x_i\}_{i=1}^{n}$ be the set of context vectors of occurrences of all entity mention pairs, where x_i represents the context vector of the i-th occurrence, and n is the total number of occurrences of all entity mention pairs.

Each occurrence of entity mention pairs can be denoted as follows:

$$R \rightarrow (C_{pre}, e_1, C_{mid}, e_2, C_{post}) \qquad (1)$$

where e_1 and e_2 represents the entity mentions, and C_{pre},C_{mid},and C_{post} are the contexts before, between and after the entity mention pairs respectively.

We extracted features from e_1, e_2, C_{pre}, C_{mid}, C_{post} to construct context vectors, which are computed from the parse trees derived from Charniak Parser (Charniak, 1999) and the Chunklink script [1] written by Sabine Buchholz from Tilburg University.

Words: Words in the two entities and three context windows.

Entity Type: the entity type of both entities, which can be PERSON, ORGANIZATION, FACILITY, LOCATION and GPE.

POS features: Part-Of-Speech tags corresponding to all words in the two entities and three context windows.

Chunking features: This category of features are extracted from the chunklink representation, which includes:

- **Chunk tag information** of the two entities and three context windows. The "0" tag means that the word is outside of any chunk. The "I-XP" tag means that this word is inside an XP chunk. The "B-XP" by default means that the word is at the beginning of an XP chunk.
- **Grammatical function** of the two entities and three context windows. The last word in each chunk is its head, and the function of the head is the function of the whole chunk. "NP-SBJ" means a NP chunk as the subject of the sentence. The other words in a chunk that are not the head have "NOFUNC" as their function.
- **IOB-chains** of the heads of the two entities. So-called IOB-chain, noting the syntactic categories of all the constituents on the path from the root node to this leaf node of tree.

We combine the above lexical and syntactic features with their position information in the context to form the context vector. Before that, we filter out low frequency features which appeared only once in the entire set.

2.2 Context Clustering

Once the context vectors of entity pairs are prepared, we come to the second stage of our method: cluster these context vectors automatically.

In recent years, spectral clustering technique has received more and more attention as a powerful approach to a range of clustering problems. Among the efforts on spectral clustering techniques (Weiss, 1999; Kannan et al., 2000; Shi et al., 2000; Ng et al., 2001; Zha et al., 2001), we adopt a modified version

[1] Software available at http://ilk.uvt.nl/ sabine/chunklink/

Table 1: Context Clustering with Spectral-based Clustering technique.

Input: A set of context vectors $X = \{x_1, x_2, ..., x_n\}$, $X \in \Re^{n \times d}$;
Output: Clustered data and number of clusters;

1. Construct an affinity matrix by $A_{ij} = \exp(-\frac{s_{ij}^2}{\sigma^2})$ if $i \neq j$, 0 if $i = j$. Here, s_{ij} is the similarity between x_i and x_j calculated by Cosine similarity measure. and the free distance parameter σ^2 is used to scale the weights;
2. Normalize the affinity matrix A to create the matrix $L = D^{-1/2}AD^{-1/2}$, where D is a diagonal matrix whose (*i*,*i*) element is the sum of A's ith row;
3. Set $q = 2$;
4. Compute q eigenvectors of L with greatest eigenvalues. Arrange them in a matrix Y.
5. Perform elongated K-means with $q + 1$ centers on Y, initializing the $(q + 1)$-th mean in the origin;
6. If the $q + 1$-th cluster contains any data points, then there must be at least an extra cluster; set $q = q + 1$ and go back to step 4. Otherwise, algorithm stops and outputs clustered data and number of clusters.

(Sanguinetti et al., 2005) of the algorithm by Ng et al. (2001) because it can provide us model order selection capability.

Since we do not know how many relation types in advance and do not have any labeled relation training examples at hand, the problem of model order selection arises, i.e. estimating the "optimal" number of clusters. Formally, let k be the model order, we need to find k in Equation: $k = argmax_k\{criterion(k)\}$. Here, the criterion is defined on the result of spectral clustering.

Table 1 shows the details of the whole algorithm for context clustering, which contains two main stages: 1) Transformation of Clustering Space (Step 1-4); 2) Clustering in the transformed space using Elongated K-means algorithm (Step 5-6).

2.3 Transformation of Clustering Space

We represent each context vector of entity pair as a node in an undirected graph. Each edge (*i*,*j*) in the graph is assigned a weight that reflects the similarity between two context vectors *i* and *j*. Hence, the relation extraction task for entity pairs can be defined as a partition of the graph so that entity pairs that are more similar to each other, e.g. labeled by the same relation type, belong to the same cluster. As a relaxation of such NP-hard discrete graph partitioning problem, spectral clustering technique computes eigenvalues and eigenvectors of a Laplacian matrix related to the given graph, and construct data clusters based on such spectral information.

Thus the starting point of context clustering is to construct an *affinity matrix A* from the data, which is an $n \times n$ matrix encoding the distances between the various points. The affinity matrix is then normalized to form a matrix L by conjugating with the the diagonal matrix $D^{-1/2}$ which has as entries the square roots of the sum of the rows of A. This is to take into account the different spread of the various clusters (points belonging to more rarified clusters will have lower sums of the corresponding row of A). It is straightforward to prove that L is positive definite and has eigenvalues smaller or equal to 1, with equality holding in at least one case.

Let K be the true number of clusters present in the dataset. If K is known beforehand, the first K eigenvectors of L will be computed and arranged as columns in a matrix Y. Each row of Y corresponds to a context vector of entity pair, and the above process can be considered as transforming the original context vectors in a d-dimensional space to new context vectors in the K-dimensional space. Therefore, the rows of Y will cluster upon mutually orthogonal points on the K dimensional sphere,rather than on the coordinate axes.

2.4 The Elongated K-means algorithm

As the step 5 of Table 1 shows, the result of elongated K-means algorithm is used to detect whether the number of clusters selected q is less than the true number K, and allows one to iteratively obtain the number of clusters.

Consider the case when the number of clusters q is less than the true cluster number K present in the dataset. In such situation, taking the first $q < K$ eigenvectors, we will be selecting a q-dimensional subspace in the clustering space. As the rows of the K eigenvectors clustered along mutually orthogonal vectors, their projections in a lower dimensional space will cluster along radial directions. Therefore, the general picture will be of q clusters elongated in the radial direction, with possibly some clusters very near the origin (when the subspace is orthogonal to some of the discarded eigenvectors).

Hence, the K-means algorithm is modified as the elongated K-means algorithm to downweight distances along radial directions and penalize dis-

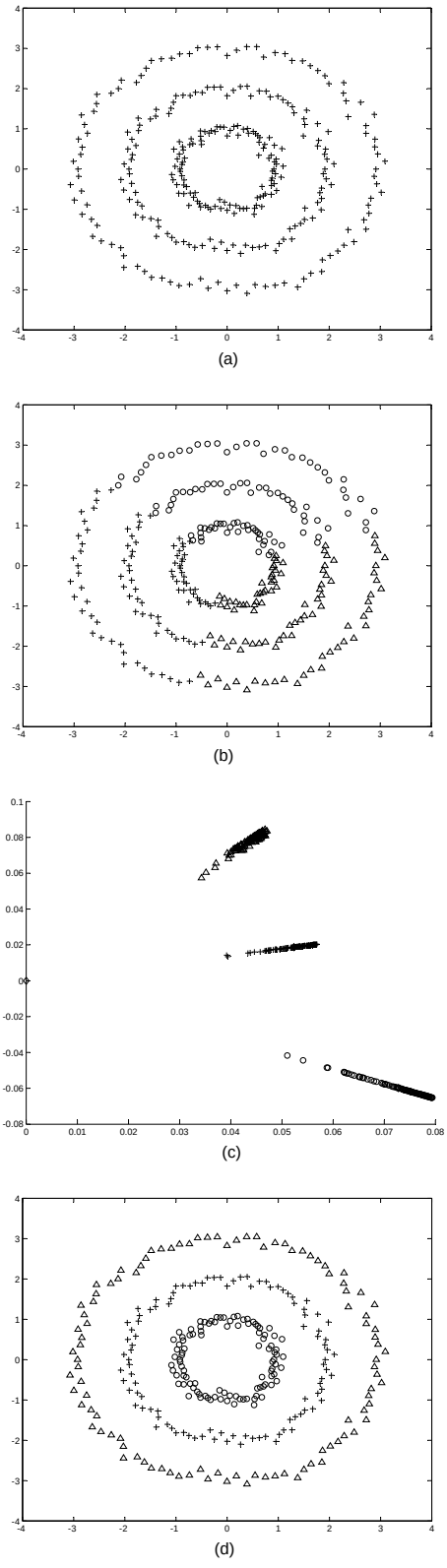


Figure 1: An Example:(a) The Three Circle Dataset. (b) The clustering result using K-means; (c) Three elongated clusters in the 2D clustering space using Spectral clustering: two dominant eigenvectors; (d) The clustering result using Spectral-based clustering (σ^2=0.05). ($\triangle$,$\circ$ and $+$ denote examples in different clusters)

tances along transversal directions. The elongated K-means algorithm computes the distance of point x from the center c_i as follows:

- If the center is not very near the origin, $c_i^T c_i > \epsilon$ (ϵ is a parameter to be fixed by the user), the distances are calculated as: $edist(x, c_i) = (x - c_i)^T M(x - c_i)$, where $M = \frac{1}{\lambda}(I_q - \frac{c_i c_i^T}{c_i^T c_i}) + \lambda \frac{c_i c_i^T}{c_i^T c_i}$, λ is the *sharpness* parameter that controls the elongation (the smaller, the more elongated the clusters) [2].

- If the center is very near the origin,$c_i^T c_i < \epsilon$, the distances are measured using the Euclidean distance.

In each iteration of procedure in Table 1, elongated K-means is initialized with q centers corresponding to data points in different clusters and one center in the origin. The algorithm then will drag the center in the origin towards one of the clusters not accounted for. Compute another eigenvector (thus increasing the dimension of the clustering space to $q + 1$) and repeat the procedure. Eventually, when one reach as many eigenvectors as the number of clusters present in the data, no points will be assigned to the center at the origin, leaving the cluster empty. This is the signal to terminate the algorithm.

2.5 An example

Figure 1 visualized the clustering result of three circle dataset using K-means and Spectral-based clustering. From Figure 1(b), we can see that K-means can not separate the non-convex clusters in three circle dataset successfully since it is prone to local minimal. For spectral-based clustering, as the algorithm described, initially, we took the two eigenvectors of L with largest eigenvalues, which gave us a two-dimensional clustering space. Then to ensure that the two centers are initialized in different clusters, one center is set as the point that is the farthest from the origin, while the other is set as the point that simultaneously farthest the first center and the origin. Figure 1(c) shows the three elongated clusters in the 2D clustering space and the corresponding clustering result of dataset is visualized in Figure 1(d), which exploits manifold structure (cluster structure) in data.

[2] In this paper, the *sharpness* parameter λ is set to 0.2

Table 2: Frequency of Major Relation SubTypes in the ACE training and devtest corpus.

Type	SubType	Training	Devtest
ROLE	General-Staff	550	149
	Management	677	122
	Citizen-Of	127	24
	Founder	11	5
	Owner	146	15
	Affiliate-Partner	111	15
	Member	460	145
	Client	67	13
	Other	15	7
PART	Part-Of	490	103
	Subsidiary	85	19
	Other	2	1
AT	Located	975	192
	Based-In	187	64
	Residence	154	54
SOC	Other-Professional	195	25
	Other-Personal	60	10
	Parent	68	24
	Spouse	21	4
	Associate	49	7
	Other-Relative	23	10
	Sibling	7	4
	GrandParent	6	1
NEAR	Relative-Location	88	32

3 Experiments and Results

3.1 Data Setting

Our proposed unsupervised relation extraction is evaluated on ACE 2003 corpus, which contains 519 files from sources including broadcast, newswire, and newspaper. We only deal with intra-sentence explicit relations and assumed that all entities have been detected beforehand in the EDT sub-task of ACE. To verify our proposed method, we only collect those pairs of entity mentions which have been tagged relation types in the given corpus. Then the relation type tags were removed to test the unsupervised relation disambiguation. During the evaluation procedure, the relation type tags were used as ground truth classes. A break-down of the data by 24 relation subtypes is given in Table 2.

3.2 Evaluation method for clustering result

When assessing the agreement between clustering result and manually annotated relation types (ground truth classes), we would encounter the problem that there was no relation type tags for each cluster in our clustering results.

To resolve the problem, we construct a contingency table T, where each entry $t_{i,j}$ gives the number of the instances that belong to both the i-th estimated cluster and j-th ground truth class. Moreover, to ensure that any two clusters do not share the same labels of relation types, we adopt a permutation procedure to find an one-to-one mapping function Ω from the ground truth classes (relation types) TC to the estimated clustering result EC. There are at most $|TC|$ clusters which are assigned relation type tags. And if the number of the estimated clusters is less than the number of the ground truth clusters, empty clusters should be added so that $|EC| = |TC|$ and the one-to-one mapping can be performed, which can be formulated as the function: $\hat{\Omega} = \arg\max_{\Omega} \sum_{j=1}^{|TC|} t_{\Omega(j),j}$, where $\Omega(j)$ is the index of the estimated cluster associated with the j-th class.

Given the result of one-to-one mapping, we adopt *Precision*, *Recall* and *F-measure* to evaluate the clustering result.

3.3 Experimental Design

We perform our unsupervised relation extraction on the devtest set of ACE corpus and evaluate the algorithm on relation subtype level. Firstly, we observe the influence of various variables, including Distance Parameter σ^2, Different Features, Context Window Size. Secondly, to verify the effectiveness of our method, we further compare it with other two unsupervised methods.

3.3.1 Choice of Distance Parameter σ^2

We simply search over σ^2 and pick the value that finds the best aligned set of clusters on the transformed space. Here, the scattering criterion $trace(P_W^{-1}P_B)$ is used to compare the cluster quality for different value of σ^2 [3], which measures the ratio of between-cluster to within-cluster scatter. The higher the $trace(P_W^{-1}P_B)$, the higher the cluster quality.

In Table 3 and Table 4, with different settings of feature set and context window size, we find out the

[3] $trace(P_W^{-1}P_B)$ is $trace$ of a matrix which is the sum of its diagonal elements. P_W is the within-cluster scatter matrix as: $P_W = \sum_{j=1}^{c} \sum_{X_i \in X_j} (X_i - m_j)(X_i - m_j)^t$ and P_B is the between-cluster scatter matrix as: $P_B = \sum_{j=1}^{c} (m_j - m)(m_j - m)^t$, where m is the total mean vector and m_j is the mean vector for j^{th} cluster and $(X_j - m_j)^t$ is the matrix transpose of the column vector $(X_j - m_j)$.

Table 3: Contribution of Different Features

Features	σ^2	cluster number	$trace$ value	Precison	Recall	F-measure
Words	0.021	15	2.369	41.6%	30.2%	34.9%
+Entity Type	0.016	18	3.198	40.3%	42.5%	41.5%
+POS	0.017	18	3.206	37.8%	46.9%	41.8%
+Chunking Infomation	0.015	19	3.900	43.5%	49.4%	46.3%

Table 4: Different Context Window Size Setting

Context Window Size	σ^2	cluster number	$trace$ value	Precision	Recall	F-measure
0	0.016	18	3.576	37.6%	48.1%	42.2%
2	0.015	19	3.900	43.5%	49.4%	46.3%
5	0.020	21	2.225	29.3%	34.7%	31.7%

corresponding value of σ^2 and cluster number which maximize the $trace$ value in searching for a range of value σ^2.

3.3.2 Contribution of Different Features

As the previous section presented, we incorporate various lexical and syntactic features to extract relation. To measure the contribution of different features, we report the performance by gradually increasing the feature set, as Table 3 shows.

Table 3 shows that all of the four categories of features contribute to the improvement of performance more or less. Firstly,the addition of entity type feature is very useful, which improves *F-measure* by 6.6%. Secondly, adding POS features can increase *F-measure* score but do not improve very much. Thirdly, chunking features also show their great usefulness with increasing *Precision/Recall/F-measure* by 5.7%/2.5%/4.5%.

We combine all these features to do all other evaluations in our experiments.

3.3.3 Setting of Context Window Size

We have mentioned in Section 2 that the context vectors of entity pairs are derived from the contexts before, between and after the entity mention pairs. Hence, we have to specify the three context window size first. In this paper, we set the mid-context window as everything between the two entity mentions. For the pre- and post- context windows, we could have different choices. For example, if we specify the outer context window size as 2, then it means that the pre-context (post-context)) includes two words before (after) the first (second) entity.

For comparison of the effect of the outer context of entity mention pairs, we conducted three different settings of context window size (0, 2, 5) as Table 4 shows. From this table we can find that with the context window size setting, 2, the algorithm achieves the best performance of 43.5%/49.4%/46.3% in *Precision/Recall/F-measure*. With the context window size setting, 5, the performance becomes worse because extending the context too much may include more features, but at the same time, the noise also increases.

Table 5: Performance of our proposed method (Spectral-based clustering) compared with other unsupervised methods: ((Hasegawa et al., 2004))'s clustering method and K-means clustering.

	Precision	Recall	F-measure
Hasegawa's Method1	38.7%	29.8%	33.7%
Hasegawa's Method2	37.9%	36.0%	36.9%
Kmeans	34.3%	40.2%	36.8%
Our Proposed Method	43.5%	49.4%	46.3%

3.3.4 Comparison with other Unsupervised methods

In (Hasegawa et al., 2004), they preformed unsupervised relation extraction based on hierarchical clustering and they only used word features between entity mention pairs to construct context vectors. We reported the clustering results using the same clustering strategy as Hasegawa et al. (2004) proposed. In Table 5, Hasegawa's Method1 means the test used the word feature as Hasegawa et al. (2004) while Hasegawa's Method2 means the test used the same feature set as our method. In both tests, we specified the cluster number as the number of ground truth classes.

We also approached the relation extraction problem using the standard clustering technique, K-

means, where we adopted the same feature set defined in our proposed method to cluster the context vectors of entity mention pairs and pre-specified the cluster number as the number of ground truth classes.

Table 5 reports the performance of our proposed method comparing with the other two unsupervised methods. Table 5 shows our proposed spectral based method clearly outperforms the other two unsupervised methods by 12.5% and 9.5% in *F-measure* respectively. Moreover, the incorporation of various lexical and syntactic features into Hasegawa et al. (2004)'s method2 makes it outperform Hasegawa et al. (2004)'s method1 which only uses word feature.

3.4 Discussion

In this paper, we have shown that the modified spectral clustering technique, with various lexical and syntactic features derived from the context of entity pairs, performed well on the unsupervised relation extraction problem. Our experiments show that by the choice of the distance parameter σ^2, we can estimate the cluster number which provides the tightest clusters. We notice that the estimated cluster number is less than the number of ground truth classes in most cases. The reason for this phenomenon may be that some relation types can not be easily distinguished using the context information only. For example, the relation subtypes "Located", "Based-In" and "Residence" are difficult to disambiguate even for human experts to differentiate.

The results also show that various lexical and syntactic features contain useful information for the task. Especially, although we did not concern the dependency tree and full parse tree information as other supervised methods (Miller et al., 2000; Culotta and Soresen, 2004; Kambhatla, 2004; Zhou et al., 2005), the incorporation of simple features, such as words and chunking information, still can provide complement information for capturing the characteristics of entity pairs. This perhaps dues to the fact that two entity mentions are close to each other in most of relations defined in ACE. Another observation from the result is that extending the outer context window of entity mention pairs too much may not improve the performance since the process may incorporate more noise information and affect the clustering result.

As regards the clustering technique, the spectral-based clustering performs better than direct clustering, K-means. Since the spectral-based algorithm works in a transformed space of low dimensionality, data can be easily clustered so that the algorithm can be implemented with better efficiency and speed. And the performance using spectral-based clustering can be improved due to the reason that spectral-based clustering overcomes the drawback of K-means (prone to local minima) and may find non-convex clusters consistent with human intuition.

Generally, from the point of view of unsupervised resolution for relation extraction, our approach already achieves best performance of 43.5%/49.4%/46.3% in *Precision/Recall/F-measure* compared with other clustering methods.

4 Conclusion and Future work

In this paper, we approach unsupervised relation extraction problem by using spectral-based clustering technique with diverse lexical and syntactic features derived from context. The advantage of our method is that it doesn't need any manually labeled relation instances, and pre-definition the number of the context clusters. Experiment results on the ACE corpus show that our method achieves better performance than other unsupervised methods, i.e.Hasegawa et al. (2004)'s method and Kmeans-based method.

Currently we combine various lexical and syntactic features to construct context vectors for clustering. In the future we will further explore other semantic information to assist the relation extraction problem. Moreover, instead of cosine similarity measure to calculate the distance between context vectors, we will try other distributional similarity measures to see whether the performance of relation extraction can be improved. In addition, if we can find an effective unsupervised way to filter out unrelated entity pairs in advance, it would make our proposed method more practical.

References

Agichtein E. and Gravano L.. 2000. *Snowball: Extracting Relations from large Plain-Text Collections, In Proc. of the* 5^{th} *ACM International Conference on Digital Libraries (ACMDL'00).*

Brin Sergey. 1998. *Extracting patterns and relations from world wide web. In Proc. of WebDB Workshop at 6th International Conference on Extending Database Technology (WebDB'98)*. pages 172-183.

Charniak E.. 1999. *A Maximum-entropy-inspired parser. Technical Report CS-99-12.*. Computer Science Department, Brown University.

Culotta A. and Soresen J. 2004. *Dependency tree kernels for relation extraction, In proceedings of 42th Annual Meeting of the Association for Computational Linguistics*. 21-26 July 2004. Barcelona, Spain.

Defense Advanced Research Projects Agency. 1995. *Proceedings of the Sixth Message Understanding Conference (MUC-6)* Morgan Kaufmann Publishers, Inc.

Hasegawa Takaaki, Sekine Satoshi and Grishman Ralph. 2004. *Discovering Relations among Named Entities from Large Corpora, Proceeding of Conference ACL2004*. Barcelona, Spain.

Kambhatla N. 2004. *Combining lexical, syntactic and semantic features with Maximum Entropy Models for extracting relations, In proceedings of 42th Annual Meeting of the Association for Computational Linguistics*. 21-26 July 2004. Barcelona, Spain.

Kannan R., Vempala S., and Vetta A.. 2000. *On clustering: Good,bad and spectral. In Proceedings of the 41st Foundations of Computer Science*. pages 367-380.

Miller S.,Fox H.,Ramshaw L. and Weischedel R. 2000. *A novel use of statistical parsing to extract information from text. In proceedings of 6th Applied Natural Language Processing Conference.* 29 April-4 may 2000, Seattle USA.

Ng Andrew.Y, Jordan M., and Weiss Y.. 2001. *On spectral clustering: Analysis and an algorithm. In Proceedings of Advances in Neural Information Processing Systems*. pages 849-856.

Sanguinetti G., Laidler J. and Lawrence N.. 2005. *Automatic determination of the number of clusters using spectral algorithms.In: IEEE Machine Learning for Signal Processing*. 28-30 Sept 2005, Mystic, Connecticut, USA.

Shi J. and Malik.J. 2000. *Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence*. 22(8):888-905.

Weiss Yair. 1999. *Segmentation using eigenvectors: A unifying view. ICCV(2)*. pp.975-982.

Zelenko D., Aone C. and Richardella A.. 2002. *Kernel Methods for Relation Extraction, Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Philadelphia.

Zha H.,Ding C.,Gu.M,He X.,and Simon H.. 2001. *Spectral Relaxation for k-means clustering. In Neural Information Processing Systems (NIPS2001)*. pages 1057-1064, 2001.

Zhang Zhu. 2004. *Weakly-supervised relation classification for Information Extraction, In proceedings of ACM 13th conference on Information and Knowledge Management (CIKM'2004)*. 8-13 Nov 2004. Washington D.C.,USA.

Zhou GuoDong, Su Jian, Zhang Jie and Zhang min. 2005. *Exploring Various Knowledge in Relation Extraction, In proceedings of 43th Annual Meeting of the Association for Computational Linguistics*. USA.

An Empirical Study of Chinese Chunking

Wenliang Chen, Yujie Zhang, Hitoshi Isahara
Computational Linguistics Group
National Institute of Information and Communications Technology
3-5 Hikari-dai, Seika-cho, Soraku-gun, Kyoto, Japan, 619-0289
{chenwl, yujie, isahara}@nict.go.jp

Abstract

In this paper, we describe an empirical study of Chinese chunking on a corpus, which is extracted from UPENN Chinese Treebank-4 (CTB4). First, we compare the performance of the state-of-the-art machine learning models. Then we propose two approaches in order to improve the performance of Chinese chunking. 1) We propose an approach to resolve the special problems of Chinese chunking. This approach extends the chunk tags for every problem by a tag-extension function. 2) We propose two novel voting methods based on the characteristics of chunking task. Compared with traditional voting methods, the proposed voting methods consider long distance information. The experimental results show that the SVMs model outperforms the other models and that our proposed approaches can improve performance significantly.

1 Introduction

Chunking identifies the non-recursive cores of various types of phrases in text, possibly as a precursor to full parsing or information extraction. Steven P. Abney was the first person to introduce chunks for parsing(Abney, 1991). Ramshaw and Marcus(Ramshaw and Marcus, 1995) first represented base noun phrase recognition as a machine learning problem. In 2000, CoNLL-2000 introduced a shared task to tag many kinds of phrases besides noun phrases in English(Sang and Buchholz, 2000). Additionally, many machine learning approaches, such as Support Vector Machines (SVMs)(Vapnik, 1995), Conditional Random Fields (CRFs)(Lafferty et al., 2001), Memory-based Learning (MBL)(Park and Zhang, 2003), Transformation-based Learning (TBL)(Brill, 1995), and Hidden Markov Models (HMMs)(Zhou et al., 2000), have been applied to text chunking(Sang and Buchholz, 2000; Hammerton et al., 2002).

Chinese chunking is a difficult task, and much work has been done on this topic(Li et al., 2003a; Tan et al., 2005; Wu et al., 2005; Zhao et al., 2000). However, there are many different Chinese chunk definitions, which are derived from different data sets(Li et al., 2004; Zhang and Zhou, 2002). Therefore, comparing the performance of previous studies in Chinese chunking is very difficult. Furthermore, compared with the other languages, there are some special problems for Chinese chunking(Li et al., 2004).

In this paper, we extracted the chunking corpus from UPENN Chinese Treebank-4(CTB4). We presented an empirical study of Chinese chunking on this corpus. First, we made an evaluation on the corpus to clarify the performance of state-of-the-art models in Chinese chunking. Then we proposed two approaches in order to improve the performance of Chinese chunking. 1) We proposed an approach to resolve the special problems of Chinese chunking. This approach extended the chunk tags for every problem by a tag-extension function. 2) We proposed two novel voting methods based on the characteristics of chunking task. Compared with traditional voting methods, the proposed voting methods considered long distance information. The experimental results showed the proposed approaches can improve the performance of Chinese chunking significantly.

The rest of this paper is as follows: Section 2 describes the definitions of Chinese chunks. Sec-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 97–104,
Sydney, July 2006.

tion 3 simply introduces the models and features for Chinese chunking. Section 4 proposes a tag-extension method. Section 5 proposes two new voting approaches. Section 6 explains the experimental results. Finally, in section 7 we draw the conclusions.

2 Definitions of Chinese Chunks

We defined the Chinese chunks based on the CTB4 dataset[1]. Many researchers have extracted the chunks from different versions of CTB(Tan et al., 2005; Li et al., 2003b). However, these studies did not provide sufficient detail. We developed a tool[2] to extract the corpus from CTB4 by modifying the tool Chunklink[3].

2.1 Chunk Types

Here we define 12 types of chunks[4]: ADJP, ADVP, CLP, DNP, DP, DVP, LCP, LST, NP, PP, QP, VP(Xue et al., 2000). Table 1 provides definitions of these chunks.

Type	Definition
ADJP	Adjective Phrase
ADVP	Adverbial Phrase
CLP	Classifier Phrase
DNP	DEG Phrase
DP	Determiner Phrase
DVP	DEV phrase
LCP	Localizer Phrase
LST	List Marker
NP	Noun Phrase
PP	Prepositional Phrase
QP	Quantifier Phrase
VP	Verb Phrase

Table 1: Definition of Chunks

2.2 Data Representation

To represent the chunks clearly, we represent the data with an IOB-based model as the CoNLL00 shared task did, in which every word is to be tagged with a chunk type label extended with I (inside a chunk), O (outside a chunk), and B (inside a chunk, but also the first word of the chunk). Each chunk type could be extended with I or B tags. For instance, NP could be represented as two types of tags, B-NP or I-NP. Therefore, we have 25 types of chunk tags based on the IOB-based model. Every word in a sentence will be tagged with one of these chunk tags. For instance, the sentence (word segmented and Part-of-Speech tagged) "他-NR(He) /到达-VV(reached) /北京-NR(Beijing) /机场-NN(airport) /。/" will be tagged as follows:

Example 1:

S1: [NP 他][VP 到达][NP 北京/机场][O 。]

S2: 他B-NP /到达B-VP /北京B-NP /机场I-NP /。O /

Here S1 denotes that the sentence is tagged with chunk types, and S2 denotes that the sentence is tagged with chunk tags based on the IOB-based model.

With data representation, the problem of Chinese chunking can be regarded as a sequence tagging task. That is to say, given a sequence of tokens (words pairing with Part-of-Speech tags), $x = x_1, x_2, ..., x_n$, we need to generate a sequence of chunk tags, $y = y_1, y_2, ..., y_n$.

2.3 Data Set

CTB4 dataset consists of 838 files. In the experiments, we used the first 728 files (FID from chtb_001.fid to chtb_899.fid) as training data, and the other 110 files (FID from chtb_900.fid to chtb_1078.fid) as testing data. In the following sections, we use the CTB4 Corpus to refer to the extracted data set. Table 2 lists details on the CTB4 Corpus data used in this study.

	Training	Test
Num of Files	728	110
Num of Sentences	9,878	5,290
Num of Words	238,906	165,862
Num of Phrases	141,426	101,449

Table 2: Information of the CTB4 Corpus

3 Chinese Chunking

3.1 Models for Chinese Chunking

In this paper, we applied four models, including SVMs, CRFs, TBL, and MBL, which have achieved good performance in other languages. We only describe these models briefly since full details are presented elsewhere(Kudo and Matsumoto, 2001; Sha and Pereira, 2003; Ramshaw and Marcus, 1995; Sang, 2002).

[1]More detailed information at http://www.cis.upenn.edu/ chinese/.

[2]Tool is available at http://www.nlplab.cn/chenwl/tools/chunklinkctb.txt.

[3]Tool is available at http://ilk.uvt.nl/software.html#chunklink.

[4]There are 15 types in the Upenn Chinese TreeBank. The other chunk types are FRAG, PRN, and UCP.

3.1.1 SVMs

Support Vector Machines (SVMs) is a powerful supervised learning paradigm based on the Structured Risk Minimization principle from computational learning theory(Vapnik, 1995). Kudo and Matsumoto(Kudo and Matsumoto, 2000) applied SVMs to English chunking and achieved the best performance in the CoNLL00 shared task(Sang and Buchholz, 2000). They created 231 SVMs classifiers to predict the unique pairs of chunk tags.The final decision was given by their weighted voting. Then the label sequence was chosen using a dynamic programming algorithm. Tan et al. (Tan et al., 2004) applied SVMs to Chinese chunking. They used sigmoid functions to extract probabilities from SVMs outputs as the post-processing of classification. In this paper, we used Yamcha (V0.33)[5] in our experiments.

3.1.2 CRFs

Conditional Random Fields is a powerful sequence labeling model(Lafferty et al., 2001) that combine the advantages of both the generative model and the classification model. Sha and Pereira(Sha and Pereira, 2003) showed that state-of-the-art results can be achieved using CRFs in English chunking. CRFs allow us to utilize a large number of observation features as well as different state sequence based features and other features we want to add. Tan et al. (Tan et al., 2005) applied CRFs to Chinese chunking and their experimental results showed that the CRFs approach provided better performance than HMM. In this paper, we used MALLET (V0.3.2)[6](McCallum, 2002) to implement the CRF model.

3.1.3 TBL

Transformation based learning(TBL), first introduced by Eric Brill(Brill, 1995), is mainly based on the idea of successively transforming the data in order to correct the error. The transformation rules obtained are usually few , yet powerful. TBL was applied to Chinese chunking by Li et al.(Li et al., 2004) and TBL provided good performance on their corpus. In this paper, we used fnTBL (V1.0)[7] to implement the TBL model.

[5]Yamcha is available at http://chasen.org/ taku/software/yamcha/

[6]MALLET is available at http://mallet.cs.umass.edu/index.php/Main_Page

[7]fnTBL is available at http://nlp.cs.jhu.edu/ rflorian/fntbl/index.html

3.1.4 MBL

Memory-based Learning (also called instance based learning) is a non-parametric inductive learning paradigm that stores training instances in a memory structure on which predictions of new instances are based(Walter et al., 1999). The similarity between the new instance X and example Y in memory is computed using a distance metric. Tjong Kim Sang(Sang, 2002) applied memory-based learning(MBL) to English chunking. MBL performs well for a variety of shallow parsing tasks, often yielding good results. In this paper, we used TiMBL[8](Daelemans et al., 2004) to implement the MBL model.

3.2 Features

The observations are based on features that are able to represent the difference between the two events. We utilize both lexical and Part-Of-Speech(POS) information as the features.

We use the lexical and POS information within a fixed window. We also consider different combinations of them. The features are listed as follows:

- WORD: uni-gram and bi-grams of words in an n window.
- POS: uni-gram and bi-grams of POS in an n window.
- WORD+POS: Both the features of WORD and POS.

where n is a predefined number to denote window size.

For instance, the WORD features at the 3rd position (北京-NR) in Example 1 (set n as 2): ”他_L2 到达_L1 北京_0 机场_R1 。_R2”(uni-gram) and ”他_到达_LB1 到达_北京_B0 北京_机场_RB1 机场_。_RB2”(bi-gram). Thus features of WORD have 9 items(5 from uni-gram and 4 from bi-grams). In the similar way, features of POS also have 9 items and features of WORD+POS have 18 items(9+9).

4 Tag-Extension

In Chinese chunking, there are some difficult problems, which are related to Special Terms, Noun-Noun Compounds, Named Entities Tagging and Coordination. In this section, we propose an approach to resolve these problems by extending the chunk tags.

[8]TiMBL is available at http://ilk.uvt.nl/timbl/

In the current data representation, the chunk tags are too generic to construct accurate models. Therefore, we define a tag-extension function f_s in order to extend the chunk tags as follows:

$$T_e = f_s(T, Q) = T \cdot Q \quad (1)$$

where, T denotes the original tag set, Q denotes the problem set, and T_e denotes the extended tag set. For instance, we have an q problem($q \in Q$). Then we extend the chunk tags with q. For NP Recognition, we have two new tags: B-NP-q and I-NP-q. Here we name this approach as Tag-Extension.

In the following three cases study, we demonstrate that how to use Tag-Extension to resolve the difficult problems in NP Recognition.

1) Special Terms: this kind of noun phrases is special terms such as "『/ 生命(Life)/ 禁区(Forbidden Zone)/ 』/", which are bracketed with the punctuation "『, 』, 「, 」, 《, 》". They are divided into two types: chunks with these punctuation and chunks without these punctuation. For instance, "『/ 生命/ 禁区/ 』/" is an NP chunk (『B-NP/ 生命I-NP/ 禁区I-NP/ 』I-NP/) while "『/永远(forever)/ 盛开(full-blown)/ 的(DE)/ 紫荆花(Chinese Redbud)/ 』/" is tagged as (『O/ 永远O /盛开O/ 的O/ 紫荆花B-NP/ 』O/). We extend the tags with SPE for Special Terms: B-NP-SPE and I-NP-SPE.

2) Coordination: These problems are related to the conjunctions "和(and), 与(and), 或(or), 暨(and)". They can be divided into two types: chunks with conjunctions and chunks without conjunctions. For instance, "香港(HongKong)/ 和(and)/ 澳门(Macau)/" is an NP chunk (香港B-NP/ 和I-NP/ 澳门I-NP/), while in "最低(least)/ 工资(salary)/ 和(and)/ 生活费(living maintenance)/" it is difficult to tell whether "最低" is a shared modifier or not, even for people. We extend the tags with COO for Coordination: B-NP-COO and I-NP-COO.

3) Named Entities Tagging: Named Entities(NE)(Sang and Meulder, 2003) are not distinguished in CTB4, and they are all tagged as "NR". However, they play different roles in chunks, especial in noun phrases. For instance, "澳门-NR(Macau)/ 机场-NN(Airport)" and "香港-NR(Hong Kong)/ 机场-NN(Airport)" vs "邓小平-NR(Deng Xiaoping)/ 先生-NN(Mr.)" and "宋卫平-NR(Song Weiping) 主席-NN(President)". Here "澳门" and "香港" are LOCATION, while "邓小平" and "宋卫平" are PERSON. To investigate the effect of Named Entities, we use a LOCATION dictionary, which is generated from the PFR corpus[9] of ICL, Peking University, to tag location words in the CTB4 Corpus. Then we extend the tags with LOC for this problem: B-NP-LOC and I-NP-LOC.

From the above cases study, we know the steps of Tag-Extension. Firstly, identifying a special problem of chunking. Secondly, extending the chunk tags via Equation (1). Finally, replacing the tags of related tokens with new chunk tags. After Tag-Extension, we use new added chunk tags to describe some special problems.

5 Voting Methods

Kudo and Matsumoto(Kudo and Matsumoto, 2001) reported that they achieved higher accuracy by applying voting of systems that were trained using different data representations. Tjong Kim Sang et al.(Sang and Buchholz, 2000) reported similar results by combining different systems.

In order to provide better results, we also apply the voting of basic systems, including SVMs, CRFs, MBL and TBL. Depending on the characteristics in the chunking task, we propose two new voting methods. In these two voting methods, we consider long distance information.

In the weighted voting method, we can assign different weights to the results of the individual system(van Halteren et al., 1998). However, it requires a larger amount of computational capacity as the training data is divided and is repeatedly used to obtain the voting weights. In this paper, we give the same weight to all basic systems in our voting methods. Suppose, we have K basic systems, the input sentence is $x = x_1, x_2, ..., x_n$, and the results of K basic systems are $t_j = t_{1j}, t_{2j}, ..., t_{nj}, 1 \leq j \leq K$. Then our goal is to gain a new result $y = y_1, y_2, ..., y_n$ by voting.

5.1 Basic Voting

This is traditional voting method, which is the same as Uniform Weight in (Kudo and Matsumoto, 2001). Here we name it as Basic Voting. For each position, we have K candidates from K basic systems. After voting, we choose the candidate with the most votes as the final result for each position.

[9]More information at http://www.icl.pku.edu

5.2 Sent-based Voting

In this paper, we treat chunking as a sequence labeling task. Here we apply this idea in computing the votes of one sentence instead of one word. We name it as Sent-based Voting. For one sentence, we have K candidates, which are the tagged sequences produced by K basic systems. First, we vote on each position, as done in Basic Voting. Then we compute the votes of every candidate by accumulating the votes of each position. Finally, we choose the candidate with the most votes as the final result for the sentence. That is to say, we make a decision based on the votes of the whole sentence instead of each position.

5.3 Phrase-based Voting

In chunking, one phrase includes one or more words, and the word tags in one phrase depend on each other. Therefore, we propose a novel voting method based on phrases, and we compute the votes of one phrase instead of one word or one sentence. Here we name it as Phrase-based Voting.

There are two steps in the Phrase-based Voting procedure. First, we segment one sentence into pieces. Then we calculate the votes of the pieces. Table 3 is the algorithm of Phrase-based Voting, where $F(t_{ij}, t_{ik})$ is a binary function:

$$F(t_{ij}, t_{ik}) = \begin{cases} 1 & : \quad t_{ij} = t_{ik} \\ 0 & : \quad t_{ij} \neq t_{ik} \end{cases} \qquad (2)$$

In the segmenting step, we seek the "O" or "B-XP" (XP can be replaced by any type of phrase) tags, in the results of basic systems. Then we get a new piece if all K results have the "O" or "B-XP" tags at the same position.

In the voting step, the goal is to choose a result for each piece. For each piece, we have K candidates. First, we vote on each position within the piece, as done in Basic Voting. Then we accumulate the votes of each position for every candidate. Finally, we pick the one, which has the most votes, as the final result for the piece.

The difference in these three voting methods is that we make the decisions in different ranges: Basic Voting is at one word; Phrase-based Voting is in one piece; and Sent-based Voting is in one sentence.

Input:
Sequence: $x = x_1, ..., x_n$;
K results: $t_j = t_{1j}, ..., t_{nj}, 1 \leq j \leq K$.
Output:
Voted results: $y = y_1, y_2, ..., y_n$

Segmenting: Segment the sentence into pieces.

Pieces[]=null; begin = 1
For each i in (2, n){
For each j in (1,K)
if(t_{ij} is not "O" and "B-XP") break;
if($j > K$){
add new piece: $p = x_{begin}, ..., x_{i-1}$ into Pieces;
begin = i; }}

Voting: Choose the result with the most votes for each piece: $p = x_{begin}, ..., x_{end}$.

Votes[K] = 0;
For each k in (1,K)

$$Votes[k] = \sum_{begin \leq i \leq end, 1 \leq j \leq K} F(t_{ij}, t_{ik}) \qquad (3)$$

$k_{max} = argmax_{1 \leq k \leq K}(Votes[k])$;
Choose $t_{begin,k_{max}}, ..., t_{end,k_{max}}$ as the result for piece p.

Table 3: Algorithm of Phrase-based Voting

6 Experiments

In this section, we investigated the performance of Chinese chunking on the CTB4 Corpus.

6.1 Experimental Setting

To investigate the chunker sensitivity to the size of the training set, we generated different sizes of training sets, including 1%, 2%, 5%, 10%, 20%, 50%, and 100% of the total training data.

In our experiments, we used all the default parameter settings of the packages. Our SVMs and CRFs chunkers have a first-order Markov dependency between chunk tags.

We evaluated the results as CONLL2000 share-task did. The performance of the algorithm was measured with two scores: precision P and recall R. Precision measures how many chunks found by the algorithm are correct and the recall rate contains the percentage of chunks defined in the corpus that were found by the chunking program. The two rates can be combined in one measure:

$$F_1 = \frac{2 \times P \times R}{R + P} \qquad (4)$$

In this paper, we report the results with F_1 score.

6.2 Experimental Results

6.2.1 POS vs. WORD+POS

In this experiment, we compared the performance of different feature representations, in-

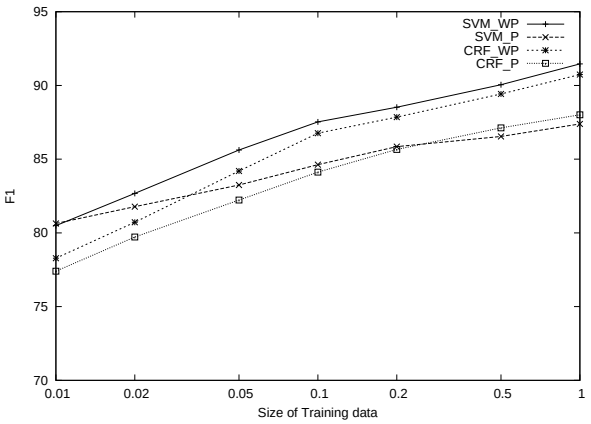


Figure 1: Results of different features

cluding POS and WORD+ POS(See section 3.2), and set the window size as 2. We also investigated the effects of different sizes of training data. The SVMs and CRFs approaches were used in the experiments because they provided good performance in chunking(Kudo and Matsumoto, 2001)(Sha and Pereira, 2003).

Figure 1 shows the experimental results, where xtics denotes the size of the training data, "WP" refers to WORD+POS, "P" refers to POS. We can see from the figure that WORD+POS yielded better performance than POS in the most cases. However, when the size of training data was small, the performance was similar. With WORD+POS, SVMs provided higher accuracy than CRFs in all training sizes. However, with POS, CRFs yielded better performance than SVMs in large scale training sizes. Furthermore, we found SVMs with WORD+POS provided 4.07% higher accuracy than with POS, while CRFs provided 2.73% higher accuracy.

6.2.2 Comparison of Models

In this experiment, we compared the performance of the models, including SVMs, CRFs, MBL, and TBL, in Chinese chunking. In the experiments, we used the feature WORD+POS and set the window size as 2 for the first two models. For MBL, WORD features were within a one-window size, and POS features were within a two-window size. We used the original data for TBL without any reformatting.

Table 4 shows the comparative results of the models. We found that the SVMs approach was superior to the other ones. It yielded results that were 0.72%, 1.51%, and 3.58% higher accuracy than respective CRFs, TBL, and MBL approaches.

	SVMs	CRFs	TBL	MBL
ADJP	84.45	84.55	85.95	80.48
ADVP	83.12	82.74	81.98	77.95
CLP	5.26	0.00	0.00	3.70
DNP	99.65	99.64	99.65	99.61
DP	99.70	99.40	99.70	99.46
DVP	96.77	92.89	99.61	99.41
LCP	99.85	99.85	99.74	99.82
LST	68.75	68.25	56.72	64.75
NP	90.54	89.79	89.82	87.90
PP	99.67	99.66	99.67	99.59
QP	96.73	96.53	96.60	96.40
VP	89.74	88.50	85.75	82.51
+	91.46	90.74	89.95	87.88

Table 4: Comparative Results of Models

Method	Precision	Recall	F_1
CRFs	91.47	90.01	90.74
SVMs	92.03	90.91	91.46
V1	91.97	90.66	91.31
V2	92.32	90.93	91.62
V3	92.40	90.97	91.68

Table 5: Voting Results

Giving more details for each category, the SVMs approach provided the best results in ten categories, the CRFs in one category, and the TBL in five categories.

6.2.3 Comparison of Voting Methods

In this section, we compared the performance of the voting methods of four basic systems, which were used in Section 6.2.2. Table 5 shows the results of the voting systems, where V1 refers to Basic Voting, V2 refers to Sent-based Voting, and V3 refers to Phrase-based Voting. We found that Basic Voting provided slightly worse results than SVMs. However, by applying the Sent-based Voting method, we achieved higher accuracy than any single system. Furthermore, we were able to achieve more higher accuracy by applying Phrase-based Voting. Phrase-based Voting provided 0.22% and 0.94% higher accuracy than respective SVMs, CRFs approaches, the best two single systems.

The results suggested that the Phrase-based Voting method is quite suitable for chunking task. The Phrase-based Voting method considers one chunk as a voting unit instead of one word or one sentence.

	SVMs	CRFs	TBL	MBL	V3
NPR	90.62	89.72	89.89	87.77	90.92
COO	90.61	**89.78**	**90.05**	**87.80**	**91.03**
SPE	**90.65**	**90.14**	**90.31**	87.77	**91.00**
LOC	90.53	**89.83**	89.69	**87.78**	90.86
NPR*	-	-	-	-	**91.13**

Table 6: Results of Tag-Extension in NP Recognition

6.2.4 Tag-Extension

NP is the most important phrase in Chinese chunking and about 47% phrases in the CTB4 Corpus are NPs. In this experiment, we presented the results of Tag-Extension in NP Recognition.

Table 6 shows the experimental results of Tag-Extension, where "NPR" refers to chunking without any extension, "SPE" refers to chunking with Special Terms Tag-Extension, "COO" refers to chunking with Coordination Tag-Extension, "LOC" refers to chunking with LOCATION Tag-Extension, "NPR*" refers to voting of eight systems(four of SPE and four of COO), and "V3" refers to Phrase-based Voting method.

For NP Recognition, SVMs also yielded the best results. But it was surprised that TBL provided 0.17% higher accuracy than CRFs. By applying Phrase-based Voting, we achieved better results, 0.30% higher accuracy than SVMs.

From the table, we can see that the Tag-Extension approach can provide better results. In COO, TBL got the most improvement with 0.16%. And in SPE, TBL and CRFs got the same improvement with 0.42%. We also found that Phrase-based Voting can improve the performance significantly. NPR* provided 0.51% higher than SVMs, the best single system.

For LOC, the voting method helped to improve the performance, provided at least 0.33% higher accuracy than any single system. But we also found that CRFs and MBL provided better results while SVMs and TBL yielded worse results. The reason was that our NE tagging method was very simple. We believe NE tagging can be effective in Chinese chunking, if we use a highly accurate Named Entity Recognition system.

7 Conclusions

In this paper, we conducted an empirical study of Chinese chunking. We compared the performance of four models, SVMs, CRFs, MBL, and TBL. We also investigated the effects of using different sizes of training data. In order to provide higher accuracy, we proposed two new voting methods according to the characteristics of the chunking task. We proposed the Tag-Extension approach to resolve the special problems of Chinese chunking by extending the chunk tags.

The experimental results showed that the SVMs model was superior to the other three models. We also found that part-of-speech tags played an important role in Chinese chunking because the gap of the performance between WORD+POS and POS was very small.

We found that the proposed voting approaches can provide higher accuracy than any single system can. In particular, the Phrase-based Voting approach is more suitable for chunking task than the other two voting approaches. Our experimental results also indicated that the Tag-Extension approach can improve the performance significantly.

References

Steven P. Abney. 1991. Parsing by chunks. In Robert C. Berwick, Steven P. Abney, and Carol Tenny, editors, *Principle-Based Parsing: Computation and Psycholinguistics*, pages 257–278. Kluwer, Dordrecht.

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. *Computational Linguistics*, 21(4):543–565.

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2004. Timbl: Tilburg memory-based learner v5.1.

James Hammerton, Miles Osborne, Susan Armstrong, and Walter Daelemans. 2002. Introduction to special issue on machine learning approaches to shallow parsing. *JMLR*, 2(3):551–558.

Taku Kudo and Yuji Matsumoto. 2000. Use of support vector learning for chunk identification. In *In Proceedings of CoNLL-2000 and LLL-2000*, pages 142–144.

Taku Kudo and Yuji Matsumoto. 2001. Chunking with support vector machines. In *In Proceedings of NAACL01*.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning (ICML01)*.

Heng Li, Jonathan J. Webster, Chunyu Kit, and Tianshun Yao. 2003a. Transductive hmm based chinese text chunking. In *Proceedings of IEEE NLP-KE2003*, pages 257–262, Beijing, China.

Sujian Li, Qun Liu, and Zhifeng Yang. 2003b. Chunking parsing with maximum entropy principle (in chinese). *Chinese Journal of Computers*, 26(12):1722–1727.

Hongqiao Li, Changning Huang, Jianfeng Gao, and Xiaozhong Fan. 2004. Chinese chunking with another type of spec. In *The Third SIGHAN Workshop on Chinese Language Processing*.

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu.

Seong-Bae Park and Byoung-Tak Zhang. 2003. Text chunking by combining hand-crafted rules and memory-based learning. In *ACL*, pages 497–504.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In David Yarovsky and Kenneth Church, editors, *Proceedings of the Third Workshop on Very Large Corpora*, pages 82–94, Somerset, New Jersey. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the conll-2000 shared task: Chunking. In *Proceedings of CoNLL-2000 and LLL2000*, pages 127–132, Lisbin, Portugal.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2003*.

Erik F. Tjong Kim Sang. 2002. Memory-based shallow parsing. *JMLR*, 2(3):559–594.

Fei Sha and Fernando Pereira. 2003. Shallow parsing with conditional random fields. In *Proceedings of HLT-NAACL03*.

Yongmei Tan, Tianshun Yao, Qing Chen, and Jingbo Zhu. 2004. Chinese chunk identification using svms plus sigmoid. In *IJCNLP*, pages 527–536.

Yongmei Tan, Tianshun Yao, Qing Chen, and Jingbo Zhu. 2005. Applying conditional random fields to chinese shallow parsing. In *Proceedings of CICLing-2005*, pages 167–176, Mexico City, Mexico. Springer.

Hans van Halteren, Jakub Zavrel, and Walter Daelemans. 1998. Improving data driven wordclass tagging by system combination. In *COLING-ACL*, pages 491–497.

V. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.

Daelemans Walter, Sabine Buchholz, and Jorn Veenstra. 1999. Memory-based shallow parsing.

Shih-Hung Wu, Cheng-Wei Shih, Chia-Wei Wu, Tzong-Han Tsai, and Wen-Lian Hsu. 2005. Applying maximum entropy to robust chinese shallow parsing. In *Proceedings of ROCLING2005*.

Nianwen Xue, Fei Xia, Shizhe Huang, and Anthony Kroch. 2000. The bracketing guidelines for the penn chinese treebank. Technical report, University of Pennsylvania.

Yuqi Zhang and Qiang Zhou. 2002. Chinese base-phrases chunking. In *Proceedings of The First SIGHAN Workshop on Chinese Language Processing*.

Tiejun Zhao, Muyun Yang, Fang Liu, Jianmin Yao, and Hao Yu. 2000. Statistics based hybrid approach to chinese base phrase identification. In *Proceedings of Second Chinese Language Processing Workshop*.

GuoDong Zhou, Jian Su, and TongGuan Tey. 2000. Hybrid text chunking. In Claire Cardie, Walter Daelemans, Claire Nédellec, and Erik Tjong Kim Sang, editors, *Proceedings of the CoNLL00, Lisbon, 2000*, pages 163–165. Association for Computational Linguistics, Somerset, New Jersey.

Soft Syntactic Constraints for Word Alignment through Discriminative Training

Colin Cherry
Department of Computing Science
University of Alberta
Edmonton, AB, Canada, T6G 2E8
colinc@cs.ualberta.ca

Dekang Lin
Google Inc.
1600 Amphitheatre Parkway
Mountain View, CA, USA, 94043
lindek@google.com

Abstract

Word alignment methods can gain valuable guidance by ensuring that their alignments maintain cohesion with respect to the phrases specified by a monolingual dependency tree. However, this hard constraint can also rule out correct alignments, and its utility decreases as alignment models become more complex. We use a publicly available structured output SVM to create a max-margin syntactic aligner with a soft cohesion constraint. The resulting aligner is the first, to our knowledge, to use a discriminative learning method to train an ITG bitext parser.

1 Introduction

Given a parallel sentence pair, or bitext, bilingual word alignment finds word-to-word connections across languages. Originally introduced as a byproduct of training statistical translation models in (Brown et al., 1993), word alignment has become the first step in training most statistical translation systems, and alignments are useful to a host of other tasks. The dominant IBM alignment models (Och and Ney, 2003) use minimal linguistic intuitions: sentences are treated as flat strings. These carefully designed generative models are difficult to extend, and have resisted the incorporation of intuitively useful features, such as morphology.

There have been many attempts to incorporate syntax into alignment; we will not present a complete list here. Some methods parse two flat strings at once using a bitext grammar (Wu, 1997). Others parse one of the two strings before alignment begins, and align the resulting tree to the remaining string (Yamada and Knight, 2001). The statistical models associated with syntactic aligners tend to be very different from their IBM counterparts. They model operations that are meaningful at a syntax level, like re-ordering children, but ignore features that have proven useful in IBM models, such as the preference to align words with similar positions, and the HMM preference for links to appear near one another (Vogel et al., 1996).

Recently, discriminative learning technology for structured output spaces has enabled several discriminative word alignment solutions (Liu et al., 2005; Moore, 2005; Taskar et al., 2005). Discriminative learning allows easy incorporation of any feature one might have access to during the alignment search. Because the features are handled so easily, discriminative methods use features that are not tied directly to the search: the search and the model become decoupled.

In this work, we view synchronous parsing only as a vehicle to expose syntactic features to a discriminative model. This allows us to include the constraints that would usually be imposed by a tree-to-string alignment method as a feature in our model, creating a powerful soft constraint. We add our syntactic features to an already strong flat-string discriminative solution, and we show that they provide new information resulting in improved alignments.

2 Constrained Alignment

Let an **alignment** be the complete structure that connects two parallel sentences, and a **link** be one of the word-to-word connections that make up an alignment. All word alignment methods benefit from some set of constraints. These limit the alignment search space and encourage competition between potential links. The IBM models (Brown et al., 1993) benefit from a one-to-many constraint, where each target word has ex-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 105–112,
Sydney, July 2006.

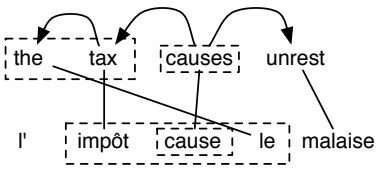


Figure 1: A cohesion constraint violation.

actly one generator in the source. Methods like competitive linking (Melamed, 2000) and maximum matching (Taskar et al., 2005) use a one-to-one constraint, where words in either sentence can participate in at most one link. Throughout this paper we assume a one-to-one constraint in addition to any syntax constraints.

2.1 Cohesion Constraint

Suppose we are given a parse tree for one of the two sentences in our sentence pair. We will refer to the parsed language as English, and the unparsed language as Foreign. Given this information, a reasonable expectation is that English phrases will move together when projected onto Foreign. When this occurs, the alignment is said to maintain **phrasal cohesion**.

Fox (2002) measured phrasal cohesion in gold standard alignments by counting crossings. Crossings occur when the projections of two disjoint phrases overlap. For example, Figure 1 shows a head-modifier crossing: the projection of the *the tax* subtree, *impôt . . . le*, is interrupted by the projection of its head, *cause*. Alignments with no crossings maintain phrasal cohesion. Fox's experiments show that cohesion is generally maintained for French-English, and that dependency trees produce the highest degree of cohesion among the tested structures.

Cherry and Lin (2003) use the phrasal cohesion of a dependency tree as a constraint on a beam search aligner. This constraint produces a significant reduction in alignment error rate. However, as Fox (2002) showed, even in a language pair as close as French-English, there are situations where phrasal cohesion should not be maintained. These include incorrect parses, systematic violations such as *not* $\rightarrow$ *ne . . . pas*, paraphrases, and linguistic exceptions.

We aim to create an alignment system that obeys cohesion constraints most of the time, but can violate them when necessary. Unfortunately, Cherry and Lin's beam search solution does not lend itself to a soft cohesion constraint. The imperfect beam search may not be able to find the optimal alignment under a soft constraint. Furthermore, it is not clear what penalty to assign to crossings, or how to learn such a penalty from an iterative training process. The remainder of this paper will develop a complete alignment search that is aware of cohesion violations, and use discriminative learning technology to assign a meaningful penalty to those violations.

3 Syntax-aware Alignment Search

We require an alignment search that can find the globally best alignment under its current objective function, and can account for phrasal cohesion in this objective. IBM Models 1 and 2, HMM (Vogel et al., 1996), and weighted maximum matching alignment all conduct complete searches, but they would not be amenable to monitoring the syntactic interactions of links. The tree-to-string models of (Yamada and Knight, 2001) naturally consider syntax, but special modeling considerations are needed to allow any deviations from the provided tree (Gildea, 2003). The Inversion Transduction Grammar or ITG formalism, described in (Wu, 1997), is well suited for our purposes. ITGs perform string-to-string alignment, but do so through a parsing algorithm that will allow us to inform the objective function of our dependency tree.

3.1 Inversion Transduction Grammar

An ITG aligns bitext through synchronous parsing. Both sentences are decomposed into constituent phrases simultaneously, producing a word alignment as a byproduct. Viewed generatively, an ITG writes to two streams at once. Terminal productions produce a token in each stream, or a token in one stream with the null symbol $\emptyset$ in the other. We will use standard ITG notation: $A \rightarrow e/f$ indicates that the token e is produced on the English stream, while f is produced on the Foreign stream. To allow for some degree of movement during translation, non-terminal productions are allowed to be either straight or inverted. Straight productions, with their non-terminals inside square brackets $[\ldots]$, produce their symbols in the same order on both streams. Inverted productions, indicated by angled brackets $\langle\ldots\rangle$, have their non-terminals produced in the given order on the English stream, but this order is reversed in the Foreign stream.

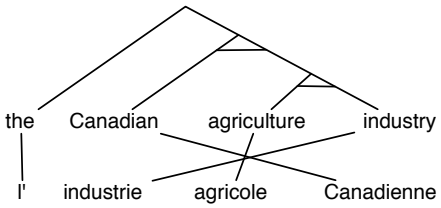


Figure 2: An example of an ITG alignment. A horizontal bar across an arc indicates an inversion.

An ITG chart parser provides a polynomial-time algorithm to conduct a complete enumeration of all alignments that are possible according to its grammar. We will use a binary bracketing ITG, the simplest interesting grammar in this formalism:

$$A \rightarrow [AA] \mid \langle AA \rangle \mid e/f$$

This grammar enforces its own weak cohesion constraint: for every possible alignment, a corresponding binary constituency tree must exist for which the alignment maintains phrasal cohesion. Figure 2 shows a word alignment and the corresponding tree found by an ITG parser. Wu (1997) provides anecdotal evidence that only incorrect alignments are eliminated by ITG constraints. In our French-English data set, an ITG rules out only 0.3% of necessary links beyond those already eliminated by the one-to-one constraint (Cherry and Lin, 2006).

3.2 Dependency-augmented ITG

An ITG will search all alignments that conform to a possible binary constituency tree. We wish to confine that search to a specific n-array dependency tree. Fortunately, Wu (1997) provides a method to have an ITG respect a known partial structure. One can seed the ITG parse chart so that spans that do not agree with the provided structure are assigned a value of $-\infty$ before parsing begins. The result is that no constituent is ever constructed with any of these **invalid spans**.

In the case of phrasal cohesion, the invalid spans correspond to spans of the English sentence that interrupt the phrases established by the provided dependency tree. To put this notion formally, we first define some terms: given a subtree $T_{[i,k]}$, where i is the left index of the leftmost leaf in $T_{[i,k]}$ and k is the right index of its rightmost leaf, we say any index $j \in (i, k)$ is **internal** to $T_{[i,k]}$. Similarly, any index $x \notin [i, k]$ is **external** to $T_{[i,k]}$. An invalid span is any span for which our provided tree has a subtree $T_{[i,k]}$ such that one endpoint of the span is internal to $T_{[i,k]}$ while the other is external to it. Figure 3 illustrates this definition, while Figure 4 shows the invalid spans induced by a simple dependency tree.

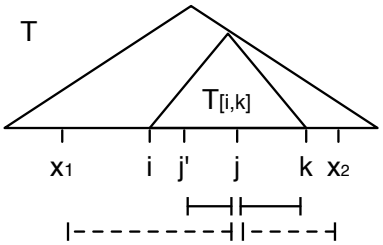


Figure 3: Illustration of invalid spans. $[j', j]$ and $[j, k]$ are legal, while $[x_1, j]$ and $[j, x_2]$ are not.

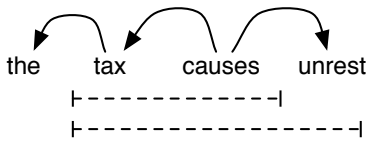


Figure 4: The invalid spans induced by a dependency tree.

With these invalid spans in place, the ITG can no longer merge part of a dependency subtree with anything other than another part of the same subtree. Since all ITG movement can be explained by inversions, this constrained ITG cannot interrupt one dependency phrase with part of another. Therefore, the phrasal cohesion of the input dependency tree is maintained. Note that this will not search the exact same alignment space as a cohesion-constrained beam search; instead it uses the union of the cohesion constraint and the weaker ITG constraints (Cherry and Lin, 2006).

Transforming this form of the cohesion constraint into a soft constraint is straight-forward. Instead of overriding the parser so it cannot use invalid English spans, we will note the invalid spans and assign the parser a penalty should it use them. The value of this penalty will be determined through discriminative training, as described in Section 4. Since the penalty is available within the dynamic programming algorithm, the parser will be able to incorporate it to find a globally optimal alignment.

4 Discriminative Training

To discriminatively train our alignment systems, we adopt the Support Vector Machine (SVM) for

Structured Output (Tsochantaridis et al., 2004). We have selected this system for its high degree of modularity, and because it has an API freely available[1]. We will summarize the learning mechanism briefly in this section, but readers should refer to (Tsochantaridis et al., 2004) for more details.

SVM learning is most easily expressed as a constrained numerical optimization problem. All constraints mentioned in this section are constraints on this optimizer, and have nothing to do with the cohesion constraint from Section 2.

4.1 SVM for Structured Output

Traditional SVMs attempt to find a linear separator that creates the largest possible **margin** between two classes of vectors. Structured output SVMs attempt to separate the correct structure from all incorrect structures by the largest possible margin, for all training instances. This may sound like a much more difficult problem, but with a few assumptions in place, the task begins to look very similar to a traditional SVM.

As in most discriminative training methods, we begin by assuming that a candidate structure y, built for an input instance x, can be adequately described using a feature vector $\Psi(x, y)$. We also assume that our $\Psi(x, y)$ decomposes in such a way that the features can guide a search to recover the structure y from x. That is:

$$\text{struct}(x; \vec{w}) = \text{argmax}_{y \in \mathcal{Y}} \langle \vec{w}, \Psi(x, y) \rangle \quad (1)$$

is computable, where $\mathcal{Y}$ is the set of all possible structures, and $\vec{w}$ is a vector that assigns weights to each component of $\Psi(x, y)$. $\vec{w}$ is the parameter vector we will learn using our SVM.

Now the learning task begins to look straightforward: we are working with vectors, and the task of building a structure y has been recast as an argmax operator. Our learning goal is to find a $\vec{w}$ so that the correct structure is found:

$$\forall i, \forall y \in \mathcal{Y} \setminus y_i : \langle \vec{w}, \Psi_i(y_i) \rangle > \langle \vec{w}, \Psi_i(y) \rangle \quad (2)$$

where x_i is the i^{th} training example, y_i is its correct structure, and $\Psi_i(y)$ is short-hand for $\Psi(x_i, y)$. As several $\vec{w}$ will fulfill (2) in a linearly separable training set, the unique max-margin objective is defined to be the $\vec{w}$ that maximizes the minimum distance between y_i and the incorrect structures in $\mathcal{Y}$.

[1] At http://svmlight.joachims.org/svm_struct.html

This learning framework also incorporates a notion of structured loss. In a standard vector classification problem, there is 0-1 loss: a vector is either classified correctly or it is not. In the structured case, some incorrect structures can be better than others. For example, having the argmax select an alignment missing only one link is better than selecting one with no correct links and a dozen wrong ones. A loss function $\Delta(y_i, y)$ quantifies just how incorrect a particular structure y is. Though Tsochantaridis et al. (2004) provide several ways to incorporate loss into the SVM objective, we will use margin re-scaling, as it corresponds to loss usage in another max-margin alignment approach (Taskar et al., 2005). In margin re-scaling, high loss structures must be separated from the correct structure by a larger margin than low loss structures.

To allow some misclassifications during training, a soft-margin requirement replaces our max-margin objective. A slack variable ξ_i is introduced for each training example x_i, to allow the learner to violate the margin at a penalty. The magnitude of this penalty to determined by a hand-tuned parameter C. After a few transformations (Tsochantaridis et al., 2004), the soft-margin learning objective can be formulated as a quadratic program:

$$\min_{\vec{w},\xi} \quad \frac{1}{2}||\vec{w}||^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i, \text{ s.t. } \forall i \xi_i \geq 0 \quad (3)$$

$$\forall i, \quad \forall y \in \mathcal{Y} \setminus y_i : \quad (4)$$
$$\langle \vec{w}, \Psi_i(y_i) - \Psi_i(y) \rangle \geq \Delta(y_i, y) - \xi_i$$

Note how the slack variables ξ_i allow some incorrect structures to be built. Also note that the loss $\Delta(y_i, y)$ determines the size of the margin between structures.

Unfortunately, (4) provides one constraint for every possible structure for every training example. Enumerating these constraints explicitly is infeasible, but in reality, only a subset of these constraints are necessary to achieve the same objective. Re-organizing (4) produces:

$$\forall i, \forall y \in \mathcal{Y} \setminus y_i : \\ \xi_i \geq \Delta(y_i, y) - \langle \vec{w}, \Psi_i(y_i) - \Psi_i(y) \rangle \quad (5)$$

which is equivalent to:

$$\forall i : \xi_i \geq \max_{y \in \mathcal{Y} \setminus y_i} \text{cost}_i(y; \vec{w}) \quad (6)$$

where cost_i is defined as:

$$\text{cost}_i(y; \vec{w}) = \Delta(y_i, y) - \langle \vec{w}, \Psi_i(y_i) - \Psi_i(y) \rangle$$

Provided that the max cost structure can be found in polynomial time, we have all the components needed for a constraint generation approach to this optimization problem.

Constraint generation places an outer loop around an optimizer that minimizes (3) repeatedly for a growing set of constraints. It begins by minimizing (3) with an empty constraint set in place of (4). This provides values for $\vec{w}$ and $\vec{\xi}$. The max cost structure

$$\bar{y} = \text{argmax}_{y \in \mathcal{Y} \setminus y_i} \text{cost}_i(y; \vec{w})$$

is found for $i = 1$ with the current $\vec{w}$. If the resulting $\text{cost}_i(\bar{y}; \vec{w})$ is greater than the current value of ξ_i, then this represents a violated constraint[2] in our complete objective, and a new constraint of the form $\xi_i \geq \text{cost}_i(\bar{y}; \vec{w})$ is added to the constraint set. The algorithm then iterates: the optimizer minimizes (3) again with the new constraint set, and solves the max cost problem for $i = i + 1$ with the new $\vec{w}$, growing the constraint set if necessary. Note that the constraints on ξ change with $\vec{w}$, as cost is a function of $\vec{w}$. Once the end of the training set is reached, the learner loops back to the beginning. Learning ends when the entire training set can be processed without needing to add any constraints. It can be shown that this will occur within a polynomial number of iterations (Tsochantaridis et al., 2004).

With this framework in place, one need only fill in the details to create an SVM for a new structured output space:

1. A $\Psi(x, y)$ function to transform instance-structure pairs into feature vectors
2. A search to find the best structure given a weight vector: $\text{argmax}_y \langle \vec{w}, \Psi(x, y) \rangle$. This has no role in training, but it is necessary to use the learned weights.
3. A structured loss function $\Delta(y, \bar{y})$
4. A search to find the max cost structure: $\text{argmax}_y \text{cost}_i(y; w)$

4.2 SVMs for Alignment

Using the Structured SVM API, we have created two SVM word aligners: a baseline that uses weighted maximum matching for its argmax operator, and a dependency-augmented ITG that will satisfy our requirements for an aligner with a soft cohesion constraint. Our x becomes a bilingual sentence-pair, while our y becomes an alignment, represented by a set of links.

4.2.1 Weighed Maximum Matching

Given a bipartite graph with edge values, the weighted maximum matching algorithm (West, 2001) will find the matching with maximum summed edge values. To create a matching alignment solution, we reproduce the approach of (Taskar et al., 2005) within the framework described in Section 4.1:

1. We define a feature vector ψ for each potential link l in x, and Ψ in terms of y's component links: $\Psi(x, y) = \sum_{l \in y} \psi(l)$.
2. Our structure search is the matching algorithm. The input bipartite graph has an edge for each l. Each edge is given the value $v(l) \leftarrow \langle \vec{w}, \psi(l) \rangle$.
3. We adopt the weighted Hamming loss in described (Taskar et al., 2005):

   $$\Delta(y, \bar{y}) = c_o |y - \bar{y}| + c_c |\bar{y} - y|$$

   where c_o is an omission penalty and c_c is a commission penalty.
4. Our max cost search corresponds to their loss-augmented matching problem. The input graph is modified to prefer costly links:

   $$\forall l \notin y : v(l) \leftarrow \langle \vec{w}, \psi(l) \rangle + c_c$$
   $$\forall l \in y : v(l) \leftarrow \langle \vec{w}, \psi(l) \rangle - c_o$$

Note that our max cost search could not have been implemented as loss-augmented matching had we selected one of the other loss objectives presented in (Tsochantaridis et al., 2004) in place of margin rescaling.

We use the same feature representation $\psi(l)$ as (Taskar et al., 2005), with some small exceptions. Let $l = (E_j, F_k)$ be a potential link between the j^{th} word of English sentence E and the k^{th} word of Foreign sentence F. To measure correlation between E_j and F_k we use conditional link probability (Cherry and Lin, 2003) in place of the Dice coefficient:

$$\text{cor}(E_j, F_k) = \frac{\#\text{links}(E_j, F_k) - d}{\#\text{cooccurrences}(E_j, F_k)}$$

where the link counts are determined by word-aligning 50K sentence pairs with another matching SVM that uses the ϕ^2 measure (Gale and

[2] Generally the test to see if $\xi_i > \text{cost}_i(\bar{y}; \vec{w})$ is approximated as $\xi_i > \text{cost}_i(\bar{y}; \vec{w}) + \epsilon$ for a small constant ϵ.

Church, 1991) in place of Dice. The ϕ^2 measure requires only co-occurrence counts. d is an absolute discount parameter as in (Moore, 2005). Also, we omit the IBM Model 4 Prediction features, as we wish to know how well we can do without resorting to traditional word alignment techniques.

Otherwise, the features remain the same, including distance features that measure $\text{abs}\left(\frac{j}{|E|} - \frac{k}{|F|}\right)$; orthographic features; word frequencies; common-word features; a bias term set always to 1; and an HMM approximation $\text{cor}(E_{j+1}, F_{k+1})$.

4.2.2 Soft Dependency-augmented ITG

Because of the modularity of the structured output SVM, our SVM ITG re-uses a large amount infrastructure from the matching solution. We essentially plug an ITG parser in the place of the matching algorithm, and add features to take advantage of information made available by the parser. x remains a sentence pair, and y becomes an ITG parse tree that decomposes x and specifies an alignment. Our required components are as follows:

1. We define a feature vector ψ_T on instances of production rules, r. Ψ is a function of the decomposition specified by y: $\Psi(x, y) = \sum_{r \in y} \psi_T(r)$.
2. The structure search is a weighted ITG parser that maximizes summed production scores. Each instance of a production rule r is assigned a score of $\langle \vec{w}, \psi_T(r) \rangle$
3. Loss is unchanged, defined in terms of the alignment induced by y.
4. A loss-augmented ITG is used to find the max cost. Productions of the form $A \rightarrow e/f$ that correspond to links have their scores augmented as in the matching system.

The ψ_T vector has two new features in addition to those present in the matching system's ψ. These features can be active only for non-terminal productions, which have the form $A \rightarrow [AA] \mid \langle AA \rangle$. One feature indicates an inverted production $A \rightarrow \langle AA \rangle$, while the other indicates the use of an invalid span according to a provided English dependency tree, as described in Section 3.2. These are the only features that can be active for non-terminal productions.

A terminal production r_l that corresponds to a link l is given that link's features from the matching system: $\psi_T(r_l) = \psi(l)$. Terminal productions $r_\emptyset$ corresponding to unaligned tokens are given blank feature vectors: $\psi_T(r_\emptyset) = \vec{0}$.

The SVM requires complete Ψ vectors for the correct training structures. Unfortunately, our training set contains gold standard alignments, not ITG parse trees. The gold standard is divided into sure and possible link sets S and P (Och and Ney, 2003). Links in S must be included in a correct alignment, while P links are optional. We create ITG trees from the gold standard using the following sorted priorities during tree construction:

- maximize the number of links from S
- minimize the number of English dependency span violations
- maximize the number of links from P
- minimize the number of inversions

This creates trees that represent high scoring alignments, using a minimal number of invalid spans. Only the span and inversion counts of these trees will be used in training, so we need not achieve a perfect tree structure. We still evaluate all methods with the original alignment gold standard.

5 Experiments and Results

We conduct two experiments. The first tests the dependency-augmented ITG described in Section 3.2 as an aligner with hard cohesion constraints. The second tests our discriminative ITG with soft cohesion constraints against two strong baselines.

5.1 Experimental setup

We conduct our experiments using French-English Hansard data. Our ϕ^2 scores, link probabilities and word frequency counts are determined using a sentence-aligned bitext consisting of 50K sentence pairs. Our training set for the discriminative aligners is the first 100 sentence pairs from the French-English gold standard provided for the 2003 WPT workshop (Mihalcea and Pedersen, 2003). For evaluation we compare to the remaining 347 gold standard pairs using the alignment evaluation metrics: precision, recall and alignment error rate or AER (Och and Ney, 2003). SVM learning parameters are tuned using the 37-pair development set provided with this data. English dependency trees are provided by Minipar (Lin, 1994).

Table 1: The effect of hard cohesion constraints on a simple unsupervised link score.

Search	Prec	Rec	AER
Matching	0.723	0.845	0.231
ITG	0.764	0.860	0.200
D-ITG	0.830	0.873	0.153

Table 2: The performance of SVM-trained aligners with various degrees of cohesion constraint.

Method	Prec	Rec	AER
Matching	0.916	0.860	0.110
D-ITG	0.940	0.854	0.100
SD-ITG	0.944	0.878	0.086

5.2 Hard Constraint Performance

The goal of this experiment is to empirically confirm that the English spans marked invalid by Section 3.2's dependency-augmented ITG provide useful guidance to an aligner. To do so, we compare an ITG with hard cohesion constraints, an unconstrained ITG, and a weighted maximum matching aligner. All aligners use the same simple objective function. They maximize summed link values $v(l)$, where $v(l)$ is defined as follows for an $l = (E_j, F_k)$:

$$v(l) = \phi^2(E_j, F_k) - 10^{-5}\text{abs}\left(\frac{j}{|E|} - \frac{k}{|F|}\right)$$

All three aligners link based on ϕ^2 correlation scores, breaking ties in favor of closer pairs. This allows us to evaluate the hard constraints outside the context of supervised learning.

Table 1 shows the results of this experiment. We can see that switching the search method from weighted maximum matching to a cohesion-constrained ITG (D-ITG) has produced a 34% relative reduction in alignment error rate. The bulk of this improvement results from a substantial increase in precision, though recall has also gone up. This indicates that these cohesion constraints are a strong alignment feature. The ITG row shows that the weaker ITG constraints are also valuable, but the cohesion constraint still improves on them.

5.3 Soft Constraint Performance

We now test the performance of our SVM ITG with soft cohesion constraint, or **SD-ITG**, which is described in Section 4.2.2. We will test against two strong baselines. The first baseline, **matching** is the matching SVM described in Section 4.2.1, which is a re-implementation of the state-of-the-art work in (Taskar et al., 2005)[3]. The second baseline, **D-ITG** is an ITG aligner with hard cohesion constraints, but which uses the weights trained by the matching SVM to assign link values. This is the most straight-forward way to combine discriminative training with the hard syntactic constraints.

The results are shown in Table 2. The first thing to note is that our Matching baseline is achieving scores in line with (Taskar et al., 2005), which reports an AER of 0.107 using similar features and the same training and test sets.

The effect of the hard cohesion constraint has been greatly diminished after discriminative training. Matching and D-ITG correspond to the the entries of the same name in Table 1, only with a much stronger, learned value function $v(l)$. However, in place of a 34% relative error reduction, the hard constraints in the D-ITG produce only a 9% reduction from 0.110 to 0.100. Also note that this time the hard constraints result in a reduction in recall. This indicates that the hard cohesion constraint is providing little guidance not provided by other features, and that it is actually eliminating more sure links than it is helping to find.

The soft-constrained SD-ITG, which has access to the D-ITG's invalid spans as a feature during SVM training, is fairing substantially better. Its AER of 0.086 represents a 22% relative error reduction compared to the matching system. The improved error rate is caused by gains in both precision and recall. This indicates that the invalid span feature is doing more than just ruling out links; perhaps it is de-emphasizing another, less accurate feature's role. The SD-ITG overrides the cohesion constraint in only 41 of the 347 test sentences, so we can see that it is indeed a soft constraint: it is obeyed nearly all the time, but it can be broken when necessary. The SD-ITG achieves by far the strongest ITG alignment result reported on this French-English set; surpassing the 0.16 AER reported in (Zhang and Gildea, 2004).

Training times for this system are quite low; unsupervised statistics can be collected quickly over a large set, while only the 100-sentence training

[3]Though it is arguably lacking one of its strongest features: the output of GIZA++ (Och and Ney, 2003)

set needs to be iteratively aligned. Our matching SVM trains in minutes on a single-processor machine, while the SD-ITG trains in roughly one hour. The ITG is the bottleneck, so training time could be improved by optimizing the parser.

6 Related Work

Several other aligners have used discriminative training. Our work borrows heavily from (Taskar et al., 2005), which uses a max-margin approach with a weighted maximum matching aligner. (Moore, 2005) uses an averaged perceptron for training with a customized beam search. (Liu et al., 2005) uses a log-linear model with a greedy search. To our knowledge, ours is the first alignment approach to use this highly modular structured SVM, and the first discriminative method to use an ITG for the base aligner.

(Gildea, 2003) presents another aligner with a soft syntactic constraint. This work adds a cloning operation to the tree-to-string generative model in (Yamada and Knight, 2001). This allows subtrees to move during translation. As the model is generative, it is much more difficult to incorporate a wide variety of features as we do here. In (Zhang and Gildea, 2004), this model was tested on the same annotated French-English sentence pairs that we divided into training and test sets for our experiments; it achieved an AER of 0.15.

7 Conclusion

We have presented a discriminative, syntactic word alignment method. Discriminative training is conducted using a highly modular SVM for structured output, which allows code reuse between the syntactic aligner and a maximum matching baseline. An ITG parser is used for the alignment search, exposing two syntactic features: the use of inverted productions, and the use of spans that would not be available in a tree-to-string system. This second feature creates a soft phrasal cohesion constraint. Discriminative training allows us to maintain all of the features that are useful to the maximum matching baseline in addition to the new syntactic features. We have shown that these features produce a 22% relative reduction in error rate with respect to a strong flat-string model.

References

P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–312.

C. Cherry and D. Lin. 2003. A probability model to improve word alignment. In *Meeting of the Association for Computational Linguistics*, pages 88–95, Sapporo, Japan, July.

C. Cherry and D. Lin. 2006. A comparison of syntactically motivated word alignment spaces. In *Proceedings of EACL*, pages 145–152, Trento, Italy, April.

H. J. Fox. 2002. Phrasal cohesion and statistical machine translation. In *Proceedings of EMNLP*, pages 304–311.

W. A. Gale and K. W. Church. 1991. Identifying word correspondences in parallel texts. In *4th Speech and Natural Language Workshop*, pages 152–157. DARPA.

D. Gildea. 2003. Loosely tree-based alignment for machine translation. In *Meeting of the Association for Computational Linguistics*, pages 80–87, Sapporo, Japan.

D. Lin. 1994. Principar - an efficient, broad-coverage, principle-based parser. In *Proceedings of COLING*, pages 42–48, Kyoto, Japan.

Y. Liu, Q. Liu, and S. Lin. 2005. Log-linear models for word alignment. In *Meeting of the Association for Computational Linguistics*, pages 459–466, Ann Arbor, USA.

I. D. Melamed. 2000. Models of translational equivalence among words. *Computational Linguistics*, 26(2):221–249.

R. Mihalcea and T. Pedersen. 2003. An evaluation exercise for word alignment. In *HLT-NAACL Workshop on Building and Using Parallel Texts*, pages 1–10, Edmonton, Canada.

R. Moore. 2005. A discriminative framework for bilingual word alignment. In *Proceedings of HLT-EMNLP*, pages 81–88, Vancouver, Canada, October.

F. J. Och and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–52, March.

B. Taskar, S. Lacoste-Julien, and D. Klein. 2005. A discriminative matching approach to word alignment. In *Proceedings of HLT-EMNLP*, pages 73–80, Vancouver, Canada.

I. Tsochantaridis, T. Hofman, T. Joachims, and Y. Altun. 2004. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of ICML*, pages 823–830.

S. Vogel, H. Ney, and C. Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proceedings of COLING*, pages 836–841, Copenhagen, Denmark.

D. West. 2001. *Introduction to Graph Theory*. Prentice Hall, 2nd edition.

D. Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.

K. Yamada and K. Knight. 2001. A syntax-based statistical translation model. In *Meeting of the Association for Computational Linguistics*, pages 523–530.

H. Zhang and D. Gildea. 2004. Syntax-based alignment: Supervised or unsupervised? In *Proceedings of COLING*, Geneva, Switzerland, August.

An Account for Compound Prepositions in Farsi

Zahra Abolhassani Chime
Research Center of Samt, Tehran, 14636
Ph.D in Linguistics
zabolhassani@hotmail.com

Abstract

There are some sorts of 'Preposition + Noun' combinations in Farsi that apparently a Prepositional Phrase almost behaves as Compound Prepositions. As they are not completely behaving as compounds, it is doubtful that the process of word formation is a morphological one.

The analysis put forward by this paper proposes "incorporation" by which an N^o is incorporated to a P^o constructing a compound preposition. In this way tagging prepositions and parsing texts in Natural Language Processing is defined in a proper manner.

1 Introduction

Prepositions have very versatile functions in Farsi and at the same time very important roles in linguistics especially in computational linguistics. Most of the linguists consider them as members of a closed set in which nothing can be added and behavior of which is completely static. However this paper tries to touch some aspects of the fact that not only this set is not a closed one but also the behaviors of its members are so dynamic that we can call the set a productive one. Having considered this fact about very frequent Farsi prepositions, we can come up with a useful model for language recognition.

There is a large discrepancy among linguists in classifying Farsi prepositions that whether or not there are compound prepositions and if there are how the process of their word formation should be accounted for as their characteristics are not as straight forward as it is expected from other compound categories.

Some Iranian Linguists have ignored this class altogether (Khānlari (1351), Shafāii (1363), Bāteni (1356), Seyed vafāii (1353)). Some believe they are not compound without putting forward any explanation but some sort of description. (Homā`yanfarox (1337), Sādeghi (1357), Kalbāsi (1371)). Some believe they are compounds without analyzing them (Mashkur (1346), Khatib Rahbar (1367), Gharib (1371), Meshkatodini (1366)) and still some have defined them as prepositional phrases in one way or another (Gholam Alizade (1371), Samiian (1983)). However we can not find a comprehensive account for this class of prepositions. This paper tries to tackle the problem from a different generative view as well as a familiar way in LA-morph (Hausser: 2001) in parsing through which we can account for the diversity of their behavior and present them in tree configuration.

For reasons of computational efficiency and linguistic concreteness (surface compositionality) the morphological component of the SLIM theory of language take great care to assign no more than one category (syntactic reading) per word form surface whenever possible (Hausser, 2001: 244). As Farsi does not enjoy the benefit of "space" in word recognition we have to resort to other clues to find out exact way of parsing and tagging. This paper helps to make sure about the category of one construction of prepositions.

2 Constructions of 'Preposition + Noun' in Farsi

From among all constructions in Farsi in which a preposition and a complement -generally NPs - occurs, there are 4 classes which seem to have different behaviors of that usual PPs (prepositional phrases) although they have exactly similar structure to that of PPs; These classes are as follows from which we just turn our attention to the first one:

1. preposition + noun

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 113–119,
Sydney, July 2006.

e.g. /bar/ + /asās-e/
on + basis
/e/ an obligatory genitive ending,
2. noun + preposition
e.g. /banā/ + /bar/
based + on
3. preposition + time / location item
e.g. /az/ + /pase/
from + behind
4. time / location item + preposition
e.g. /poŝt/ + /be/
back + to

From the form point of view, we can simply consider preposition such as /bar/ 'on', /az/ 'from/of', /dar/ 'in', /bā/ 'with', /be/ 'to' as (real) prepositions and what comes immediately after, as complement.

However, a close observation reveals that not in all constructions consisting of a proposition and a noun the immediate noun can be considered as the noun head of the NP complement. That is in some phrases the head preposition is the compound preposition (a preposition and a noun) and then the noun after this construction is the complement:

5. /bar/ + /asās-e/ + /motāle'āt/
p complement (n)
"on + bases" (of) researches

The first question we try to answer is: Does the immediate noun after the preposition in (5), behave like other nouns as complements in PPs?

To answer this question we should make sure whether the noun (complement) is as independent as the other nouns in 'preposition + nouns' making prepositional phrases, or it is somehow merged with the preposition producing compound preposition.

There are some structural tests to reveal this. If the noun here expands as other nouns in other prepositional phrases we can conclude that the related structure is a phrase, otherwise it is better to think about them as compound prepositions.

3 Extending the structure under discussion

3.1 Premodifiers

The noun in prepositional phrases, can be extended in different ways while as the examples below show, the related structures cannot:

3.1.1 Demonstratives

6. bar (*in) asās-e motale'āte dānešmandān
on (this) bases-of researches-of scientists
havā-ye zamin garmtaršode 'ast
climate-of earth increased has

"Based of scientists' researches the climate of earth has increased".

6′) bar (in) bām-e xāne kasi rāh miraft.
on (this) roof-of house someone (was) walking

3.1.2 Superlatives

7) bar (*jadid-tarin) asās-e motāle'at-e …
on the newest basis-of researches-of

7′) bar (zibā-tarin) bām-e xāne …
on the most beautiful roof-of house

3.1.3 Exclamatories

8) bar (*che!) asās-e motāle'āt-e …
on what! a basis-of researches-of

8′) bar (che!) bām-e xāne …
on (what!) a roof of house

3.1.4 Quantifiers

9) bar (*har) asās-e motāle'āt-e …
on (every) basis-of researches-of

9′) bar (har) bām-e xāne …
on (every) roof-of house

3.1.5 Question words

10) bar (* che) asās-e motāle'āt-e …?
on what basis-of researches

10′) bar (che) bām-e xāne-i …?
on what roof-of house

3.1.6 Indefinite /yek/ 'one'

11) bar (*yek) asās-e motāle'āt-e …
on one basis-of researches

11′) bar (yek) bām-e xāne …
on (one) roof-of house

3.2 Post Modifiers

Nouns in prepositional phrases can expand with post modifiers while nouns in our structure cannot.

3.2.1 Plural Markers

12) az Jāneb (*haye) dowlat va mardom
from side (s)-of government and nation
masā'eli matrah šod.
affairs raised was

"Some affairs were raised by government and nation."

12′) as ketāb (ha-ye) Ali estefāde kardam.
from book (s)-of Ali used I did.

"I used Ali's books."

3.2.2 Adjectives

13) be elate (*puš-e) bārandegi madāres ta'til šod.
to cause-of (vain-of) raining schools closed were.

"schools were closed because of the vain reason of raining."

13′) bar bām-e (ziba-ye) xāne qadam bogzar.
on roof-of (beautiful-of) house step put.

"step on the beautiful roof of the house."

3.2.3 Appositives

14) bar asās-e (*pāye-ye) motāle'āt-e dānešmandān
on basis-of (base-of) researches-of scientists

14′) Ali az xāne (mahale zendegi)-ash dur šode ast.
Ali from house (place-of living)-his far made is.

"Ali has left his house-his place of living."

3.3 Conclusion

The conclusion we extract out of these observations imposes some hypotheses:
1) The noun in these kinds of structures has lost its independent status and the whole structure has turned into a morphological compound preposition.
2) The intended construction, is a special kind of "compound" probably a syntactic compound, in which not all characteristics of morphological compounds can be observed.

To evaluate the first hypothesis, we should first identify the criteria of compound words in these apparent phrases.

4 Compound Words in Farsi

Farshid vard (1351) believes it's very difficult to identify and define the compound words in Farsi, because to gain the criteria of compound words, we should recognize compound forms from some other related and close structures, such as derived words and phrases.

In a phrase, grammatical roles of the parts are devoted as one to the head and the whole group rather than the parts contributes to the role of the phrase. Different ways of argumentation that can be established for distinction between phrases and compound words can be classified into 4 classes: phonological, morphological, syntactic and semantic

4.1 Phonological Argumentation

It is assumed that prepositions in Farsi do not bear any accent. This assumption comes from the fact that accent pattern in Farsi is in a any that the last or the farthest member of the group (phrase) takes the accent, except in marked structures; and as prepositions do not occur at the end of the phrase (PPs are head-first, as the other phrases in Farsi), they never take the accent. Eslami (1379: 28) states this fact as the "Head-escape Principle":
"In all cases, with expanding the head of a syntactic phrase, the accent of the phrase falls on the farthest member."

15. [[az] [′xāne]]
"from the house"

16. [[az] [xāne-ye] [′rezā]]
"from the house-of Reza"

The above observations, i.e.: 1. Accent on the last modifier and 2. Accent on the last syllabus of the word we conclude that the pattern of accent of the compound prepositions and prepositional phrases are absolutely the same.

In fact phonological reasons and criteria do not help of any kind.

4.2 Morphological Argumentation

All what was mentioned in previous section as expanding possibility of PPs can also be considered as morphological criteria.

4.3 Syntactic Argumentation

4.3.1 Topicalization

In topicalization "one word" can be topicalized out of a phrase but not out of a compound word.

17. Tamiz kardan-e ketāb-xāne bā Ali-st.
cleaning-of book-case with Ali is.

"cleaning book-case is with Ali"

17′. *ketāb tamiz kardan-e xāne-ash bā Ali-st.
book cleaning-of case-its with Ali is.

"book, cleaning of its case is with Ali."

In (17) (ketāb) is a part of a compound word from which no part can be topicalized.

Now let's see what happens if we topicalize a word in our construction.

18. bā Ali dar mored-e dānešgāh sohbat kardam.
with Ali in case-of university talk I made.

"I talked with Ali about the university."

18′. *mored-e dānešgāh, bā Ali daresh sohbat kardam.
case-of university, with Ali in-it talk I made.

"About university, I talk about it with Ali."

4.3.2 Coordination

Two similar constituents can be coordinated but not parts of compound words:
Noun out of PPs:
19. Hasan bā [dust va došman] modārā mikonad.
Hassan with [friend and enemy] bears

"Hassan bears every one."

Parts of prepositions:
19′. *be [dalil-e va ellat-e] sarmā madrese-ha ta'til šod.
to [reason-of and cause-of] cold schools closed became.
"Because of cold schools were closed."

4.4 Semantic Argumentation

Close semantic observation of these constructions reveal that the nouns in the above mentioned combinations are special kind of nouns with particular semantic features.

All the nouns are "noun-referential" and "abstract".

/dar mored-e/,	/dar zamine-ye/,	/bar asās-e/
in case-of	in field-of	on basis-of
"about"	"about"	"on"

/bar hasb-e/,	/az heis-e/,	/az lahāz-e/
on according	from aspect	from aspect
"according"	"according"	"point of view"

/bar asar-e/
on cause-of
"because of"

Another point to be mentioned is a delicate semantic difference between the meaning of these nouns in other constructions and in combination with prepositions. For example "dalil" in following two sentences does not bear the same semantic features.
20. man dalil-e harf-haye šomā rā nemifahmam.
I reason-of talks your don't understand.

"I do not understand the reason of your talks".

20′. man be dalilt-e harf-haye šomā jalase rā tark kardam.
I to cause-of talks your meeting left.

"I left the meeting because of your talks".

"dalil" in (20) has the semantic components of "argumentation, base, reason", but in (20′) "because, for".

Still another point worth mentioning is that most of the class members are synonymous in one way or another:
– dar mored-e, dar zamine-ye, dar xosus-e, dar bāre-ye, dar bāb-e, dar atrāfe,
"about"
– bar asās, bar paye-ye, bar hasb-e
"on, on the basis"
– az nazar-e, az heis-e, az lahāz-e, az jahat-e
"according to"
– be mojarad-e, be mahze
"once"
– be mojeb-e, be ellat-e, be dalil-e
"because of"

5 Concluding the Discussion

Through same constituency tests, we showed that these constituents do not obey the phrasal characteristics. On the other hand, criteria of distinguishing compound words from syntactic phrases demonstrate that these forms are not so merged into each other in a way that they can be called fixed morphological compounds. It seems that they are in a transition phase from PPs to compound Ps. So although they are compounds we should look for the process of word formation

to take place in some other places rather than the morphology, i.e. in syntax.

The argumentation proposed by the author is "incorporation", which can account for the behavior of such constructions in Farsi.

6 Incorporation

Incorporation brings out two changes in sentence representation: 1. It produces a compound category of word level (X^o). 2. It establishes a syntactic relationship between two places: the original position of the moved category (situ) and the target position. The former is a morphological and the latter is a syntactic change.

Baker (1988) considers X^o movements similar to those of XP, with all constraints and conditions applicable to both. He also proposes "Government Transparency Corollary" to account for the grammatical changes. Movement automatically changes the governance features of a structure and the reason is that it creates a grammatical dependency between two distinct phrases.

Leiber (1992: 14) says that there are some facts that show to some extend there should be same interaction between syntax and morphology. Thus X parameters and related systems are not merely applicable to syntax, but morphology too.

However incorporation of this kind in Farsi is abstract, i.e. there is no overt movement.

During incorporation process head X^o (here N^o) moves from its place towards P node and attaches to the P (dar) as it is shown in figure 1 and 2.

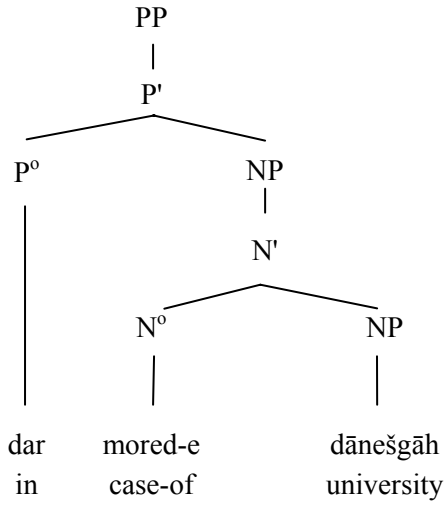


Figure 1

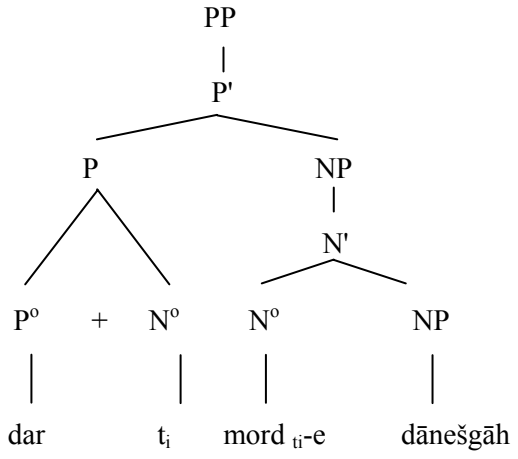


Figure 2

"dar+mored-e" dominated by a P node has the features of preposition and in this way θ-role change of "mored" is realized as preposition in combination with an original preposition. This syntactic process gives the following results:

1. A noun head (N^o) dominated by NP as a complement of a pp, α-moves and incorporates to the preposition head (P^o).
2. Moved N^o is governed and dominated by a preposition node.
3. The output of the combination of the N^o and P^o is a compound P^o.
4. The preposition (dar) "in" which before incorporation assigned θ-r to NP, after incorporation together with the noun (mored-e) assigns the θ-r to the NP (dānešgāh).
5. The resulted compound is a "syntactic compound".

The needed conditions for incorporation of N^o to P^o can be summarized as follows:

1. P^o should be morphologically simple and among the members of this group: dar "in", be "to", bā "with", az "of, from", bar "on". They do not take genitive ending /-e/ (kasre-ezāfe) and having the [-V, -N] features are considered as "true" prepositions (Samiian, 1992)
2. N^o should be morphologically simple and having all the features of [non-referential, abstract, complement-taking, indefinite].

Hereby it becomes clear why not every combination of "preposition + noun" lead to "compound prepositions" through incorporation, even if their occurrence bears a high frequency. The algorithm-like of this process is shown in figure 3.

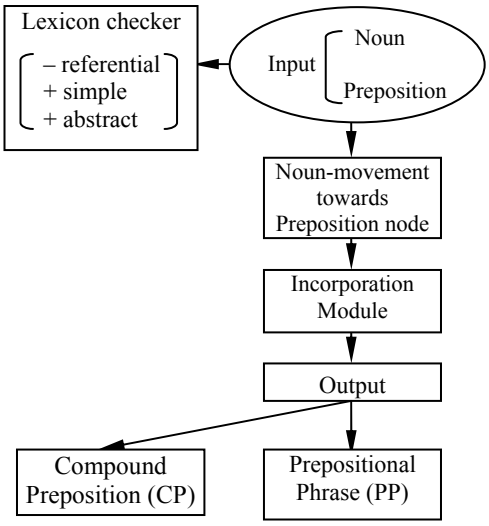


Figure 3

Prepositions are functional and so syntactic categories rather than lexical ones. I believe word formation of this category is motivated by syntax, in different ways one of which was argued here. This account contributes to the discipline of computational linguistics in labeling prepositions in Farsi, as this area of preposition labeling has been very challenging.

Although Voutilainen (2003) believes that data-driven taggers seem to be better suited for the analysis of fixed-word-order poor-morphology languages like English, but the finding of this paper is applicable to Farsi parts of speech recognition at least in the area of compound prepositions.

Prepositions are one sort of parts of speech, the recognition of which can be helpful in stemming for information retrieval (IR), since knowing a word's POS can help tell us which morphological affixes it can take. It can also help an IR application by helping select out nouns or other important words from a document. Automatic POS taggers can help in building automatic word-sense disambiguating algorithms, and POS taggers are also used in advanced ASR language models such as class-based *n*-grams (Jurafsky and Martin, 2000: 288)

Acknowledgement

My special thanks go to Masood Ghayoomi at the Institute for Humanities and Cultural Studies for his supports and encouragements in my research.

References

Baker, M. C. (1988) *Incorporation, A Theory of grammatical function changing*. The University of Chicago Press, Chicago.

Bateni, Mohammadreza (1356) *Tosife Sāxtemane Dasturie Zabāne Farsi*, Tehran, Amirkabir Publication.

Eslami, Moharam (1379) *Šenaxte Navāye Goftāre Zabāne Farsi va Karborde ān dar Bāzsazi va Bāzšenāsie Rayaneie Goftar*, Ph.D diss., Tehran University, Linguistic department.

Farshidvard, Khosrow (1351) "Kalameye morakab va meyāre tašxise ān", *Proceedings of 2nd Iranian Researches Seminar*, Vol. 1, Mašhad University.

Gharib, Abdolazim et al (1371) *Dastare Panj Ostaād*, Ašrafi Publication, 10th ed.

Gholām Ali Zade, Khosrow (1374) *Sāxte Zabāne Farsi*, Ehyāye Ketāb Publication.

Hausser, Roland (2001) *Foundations of Computational Linguistics*, Springer.

Homayoun Farokh, Abdorahim (1337) *Dasture Jāme Zabāne Fārsi,* Tehran, Elmi Publication.

Jurafski, D. and J. H. Martin (2000) *Speech and Language Processing: An Introduction to Natural Language Processing, Computational linguistics and Speech Recognition*. Prentice Hall, Pearson Higher Education.

Kalbasi, Iran (1371) *Sāxte Ešteqāqie Vāje dar Fārsie Emruz*. The Institute of Studies and Cultural Researches.

Khanlari, Parviz (1351) *Dasture Zabāne Fārsi*, Tehran Bonyad Farhangy Iran.

Khatibrahbar, Khalil (1367) *Dasture Zabāne Farsi: Ketabe Harfe ezāfe va Rabt*. Sadi Publication.

Lieber, R. (1992) *Deconstructing Morphology*, The University of Chicago Press.

Mashkur, M. Javad (1346) *Dasturnāme dar Sarf va Nahve Zabāne Fārsi*, Shargh Publication Institute.

Meshkatodini, Mehdi (1366) *Dasture Zabāne Fārsi bar Payeye Nazariye Gaštāri*, Ferdowsi University

Sadegi, Aliashraf (1349) "Horufe ezafe dar Farsie moaser", *Journal of literature and Humanities*, Tehran University, pp (441-470).

Samiian, Vida (1983) *Structure of Phrasal Categories in Persian: An X-bar Analysis.* Ph.D diss. University of California, Los Angeles.

Samiian, V. (1991) *Prepositions in Persian and the Neutralization Hypothesis*. California State University, Fresno.

Seyed Vafai (1353) "Horufe ezāfe dar zabāne Farsie moaser", *Journal of Literture and Humanities*, Tehran University, pp (49-86).

Shafaii, Ahmad (1363) *Mabanie Elmie Dasture Zabāne Farsi*, Novin Publication.

Voutilainen, Atro (2003) in Mitkov, Ruslan(ed), *The Oxford Handbook of Computational Linguistics*, Oxford University Press.

Techniques to incorporate the benefits of a Hierarchy in a modified hidden Markov model

Lin-Yi Chou
University of Waikato
Hamilton
New Zealand
lc55@cs.waikato.ac.nz

Abstract

This paper explores techniques to take advantage of the fundamental difference in structure between hidden Markov models (HMM) and hierarchical hidden Markov models (HHMM). The HHMM structure allows repeated parts of the model to be merged together. A merged model takes advantage of the recurring patterns within the hierarchy, and the clusters that exist in some sequences of observations, in order to increase the extraction accuracy. This paper also presents a new technique for reconstructing grammar rules automatically. This work builds on the idea of combining a phrase extraction method with HHMM to expose patterns within English text. The reconstruction is then used to simplify the complex structure of an HHMM

The models discussed here are evaluated by applying them to natural language tasks based on *CoNLL-2004*[1] and a sub-corpus of the Lancaster Treebank[2].

Keywords: information extraction, natural language, hidden Markov models.

1 Introduction

Hidden Markov models (HMMs) were introduced in the late 1960s, and are widely used as a probabilistic tool for modeling sequences of observations (Rabiner and Juang, 1986). They have proven to be capable of assigning semantic labels to tokens over a wide variety of input types. This is useful for text-related tasks that involve some uncertainty, including part-of-speech tagging (Brill, 1995), text segmentation (Borkar et al., 2001), named entity recognition (Bikel et al., 1999) and information extraction tasks (McCallum et al., 1999). However, most natural language processing tasks are dependent on discovering a hierarchical structure hidden within the source information. An example would be predicting semantic roles from English sentences. HMMs are less capable of reliably modeling these tasks. In contrast *hierarchical hidden Markov models* (HHMMs) are better at capturing the underlying hierarchy structure. While there are several difficulties inherent in extracting information from the patterns hidden within natural language information, by discovering the hierarchical structure more accurate models can be built.

HHMMs were first proposed by Fine (1998) to resolve the complex multi-scale structures that pervade natural language, such as speech (Rabiner and Juang, 1986), handwriting (Nag et al., 1986), and text. Skounakis (2003) described the HHMM as multiple "levels" of HMM states, where lower levels represents each individual output symbol, and upper levels represents the combinations of lower level sequences.

Any HHMM can be converted to a HMM by creating a state for every possible observation, a process called "flattening". Flattening is performed to simplify the model to a linear sequence of Markov states, thus decreasing processing time. But as a result of this process the model no longer contains any hierarchical structure. To reduce the models complexity while maintaining some hierarchical structure, our algorithm uses a "partial flattening" process.

In recent years, artificial intelligence re-

[1] The 2004 Conference on Computational Natural Language Learning, http://cnts.uia.ac.be/conll2004

[2] Lancaster/IBM Treebank, http://www.ilc.cnr.it/EAGLES96/synlex/node23.html

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 120–127,
Sydney, July 2006.

searchers have made strenuous efforts to reproduce the human interpretation of language, whereby patterns in grammar can be recognised and simplified automatically. Brill (1995) describes a simple rule-based approach for learning by rewriting the bracketing rule—a method for presenting the structure of natural language text—for linguistic knowledge. Similarly, Krotov (1999) puts forward a method for eliminating redundant grammar rules by applying a compaction algorithm. This work draws upon the lessons learned from these sources by automatically detecting situations in which the grammar structure can be reconstructed. This is done by applying the phrase extraction method introduced by Pantel (2001) to rewrite the bracketing rule by calculating the dependency of each possible phrase. The outcome of this restructuring is to reduce the complexity of the hierarchical structure and reduce the number of levels in the hierarchy.

This paper considers the tasks of identifying the syntactic structure of text chunking and grammar parsing with previously annotated text documents. It analyses the use of HHMMs—both before and after the application of improvement techniques—for these tasks, then compares the results with HMMs. This paper is organised as follows: Section 2 describes the method for training HHMMs. Section 3 describes the flattening process for reducing the depth of hierarchical structure for HHMMs. Section 4 discusses the use of HHMMs for the text chunking task and the grammar parser. The evaluation results of the HMM, the plain HHMM and the merged and partially flattened HHMM are presented in Section 5. Finally, Section 6 discusses the results.

2 Hierarchical Hidden Markov Model

A HHMM is a structured multi-level stochastic process, and can be visualised as a tree structured HMM (see Figure 1(b)). There are two types of states:

- **Production state**: a leaf node of the tree structure, which contains only observations (represented in Figure 1(b) as the empty circle $\bigcirc$).

- **Internal state**: contains several production states or other internal states (represented in Figure 1(b) as a circle with a cross inside $\bigoplus$).

The output of a HHMM is generated by a process of traversing some sequence of states within the model. At each internal state, the automation traverses down the tree, possibly through further internal states, until it encounters a production state where an observation is contained. Thus, as it continues through the tree, the process generates a sequence of observations. The process ends when a final state is entered. The difference between a standard HMM and a hierarchical HMM is that individual states in the hierarchical model can traverse to a sequence of production states, whereas each state in the standard model corresponds is a production state that contains a single observation.

2.1 Merging

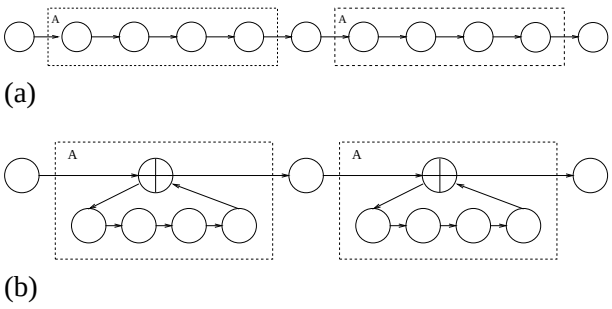


Figure 1: Example of a HHMM

Figure 1(a) and Figure 1(b) illustrate the process of reconstructing a HMM as a HHMM. Figure 1(a) shows a HMM with 11 states. The two dashed boxes (A) indicate regions of the model that have a repeated structure. These regions are furthermore independent of the other states in the model. Figure 1(b) models the same structure as a hierarchical HMM, where each repeated structure is now grouped under an internal state. This HHMM uses a two level hierarchical structure to expose more information about the transitions and probabilities within the internal states. These states, as discussed earlier, produce no observation of their own. Instead, that is left to the child production states within them. Figure 1(b) shows that each internal state contains four production states.

In some cases, different internal states of a HHMM correspond to exactly the same structure in the output sequence. This is modelled by making them share the same sub-models. Using a HHMM allows for the merging of repeated parts of the structure, which results in fewer states needing to be identified—one of the three fundamental problems of HMM construction (Rabiner and

Juang, 1986).

2.2 Sub-model Calculation

Estimating the parameters for multi-level HHMMs is a complicated process. This section describes a probability estimation method for internal states, which transforms each internal state into three production states. Each internal state S_i in the HHMM is transformed by resolving each child production state $S_{i,j}$, into one of three transformed states, $S_i \Rightarrow \{s_{in}^{(i)}, s_{stay}^{(i)}, s_{out}^{(i)}\}$. The transformation requires re-calculating the new observational and transition probabilities for each of these transformed states. Figure 2 shows the internal states of S_2 have been transformed into $s_{in}^{(2)}$, $s_{stay}^{(2)}$, $s_{stay}^{(2)}$ and $s_{out}^{(2)}$.

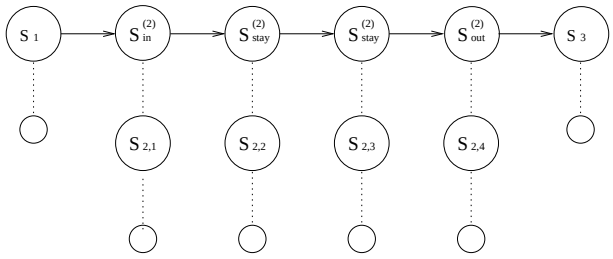


Figure 2: Example of a transformed HHMM with the internal state S_2.

The procedure to transform internal states is: I) calculate the transformed observation ($\bar{O}$) for each internal state; II) apply the forward algorithm to estimate the state probabilities ($\bar{b}$) for the three transformed states; III) reform the transition matrix by including estimated values for additional transformed internal states ($\bar{A}$).

I. **Calculate the observation probabilities $\bar{O}$:** Every observation in each internal state S_i is re-calculated by summing up all the observation probabilities in each production state S_j as:

$$\bar{O}_{i,t} = \sum_{j=1}^{N_i} O_{j,t}, \qquad (1)$$

where time t corresponds to a position in the sequence, O is an observation sequence over t, $O_{j,t}$ is the observation probability for state S_j at time t, and N_i represents the number of production states for internal state S_i.

II. **Apply forward algorithm to estimate the transform observation value $\bar{b}$:** The transformed observation values are simplified to $\{\bar{b}_{in,t}^{(i)}, \bar{b}_{stay,t}^{(i)}, \bar{b}_{out,t}^{(i)}\}$, which are then given as the observation values for the three productions states ($s_{in}^{(i)}$, $s_{stay}^{(i)}$, $s_{out}^{(i)}$). The observational probability of entering state S_i at time t, i.e. production state $s_{in}^{(i)}$, is given by:

$$\bar{b}_{in,t}^{(i)} = \max_{j=1..N_i} \left[\pi_j \times \bar{O}_{j,t}\right], \qquad (2)$$

where π_j represents the transition probabilities of entering child state S_j. The second probability of staying in state S_i at time t, i.e. production state, $s_{stay}^{(i)}$, is given by:

$$\begin{aligned} \bar{b}_{stay,t}^{(i)} &= \max_{j=1..N_i} \left[A_{\hat{j}^*,j} \times \bar{O}_{j,t}\right], \qquad (3) \\ \hat{j} &= \arg\max_{j=1..N_i} \left[A_{\hat{j}^*,j} \times \bar{O}_{j,t}\right], \end{aligned}$$

where $\hat{j}^*$ is the state corresponding to $\hat{j}$ calculated at previous time $t-1$, and $A_{\hat{j}^*,j}$ represents the transition probability from state $S_{\hat{j}^*}$ to state to S_j. The third probability of exiting state S_i at time t, i.e. production state, $s_{out}^{(i)}$, is given by:

$$\bar{b}_{out,t}^{(i)} = \max_{j=1..N_i} \left[A_{\hat{j}^*,j} \times \bar{O}_{j,t} \times \tau_j\right], \qquad (4)$$

where τ_j is the transition probabilities for leaving the state S_j.

III. **Reform transition probability $\bar{A}^{(i)}$:** Each internal state S_i reforms a new 3×3 transition probability matrix $\bar{A}$, which records the transition status for the transform matrix. The formula for the estimated cells in $\bar{A}$ are:

$$\begin{aligned} \bar{A}_{in,stay}^{(i)} &= \sum_{j=1}^{N_i} \pi_j \qquad (5) \\ \bar{A}_{in,out}^{(i)} &= \sum_{j=1}^{N_i} \frac{\pi_j}{2} \qquad (6) \\ \bar{A}_{stay,stay}^{(i)} &= \sum_{k=1,j=1}^{N_i,N_i} A_{k,j} \qquad (7) \\ \bar{A}_{stay,out}^{(i)} &= \sum_{j=1}^{N_i} \tau_j \qquad (8) \end{aligned}$$

where N_i is the number of child states for state S_i, $\bar{A}_{in,stay}^{(i)}$ is estimated by summing

up all entry state probabilities for state S_i, $\bar{A}^{(i)}_{in,out}$ is estimated from the observation that 50% of sequences transit from state $s^{(i)}_{in}$ directly to state $s^{(i)}_{out}$, $\bar{A}^{(i)}_{stay,stay}$ is the sum of all the internal transition probabilities within state S_i, and $\bar{A}^{(i)}_{stay,out}$ is the sum of all exit state probabilities. The rest of the probabilities for transition matrix $\bar{A}$ are set to zero to prevent illegal transitions.

Each internal state is implemented by a bottom-up algorithm using the values from equations (1)-(8), where lower levels of the hierarchy tree are calculated first to provide information for upper level states. Once all the internal states have been calculated, the system then need only to use the top-level of the hierarchy tree to estimate the probability sequences. This means the model will now become a linear HMM for the final Viterbi search process (Viterbi, 1967).

3 Partial flattening

Partial flattening is a process for reducing the depth of hierarchical structure trees. The process involves moving sub-trees from one node to another. This section presents an interesting automatic partial flattening process that makes use of the term extractor method (Pantel and Lin, 2001). The method discovers ways of more tightly coupling observation sequences within sub-models thus eliminating rules within the HHMM. This results in more accurate model. This process involves calculating dependency values to measure the dependency between the elements in the state sequence (or observation sequence).

This method uses *mutual information* and *log-likelihood*, which Dunning (1993) used to calculate the dependency value between words. Where there is a higher dependency value between words they are more likely to be treat as a term. The process involves collecting bigram frequencies from a large dataset, and identifying the possible two word candidates as terms. The first measurement used is *mutual information*, which is calculated using the formula:

$$mi(x,y) = \frac{P(x,y)}{P(x)P(y)} \tag{9}$$

where x and y are words adjacent to each other in the training corpus, $C(x,y)$ to be the frequency of the two words, and $*$ represents all the words in entire training corpus. The log-likelihood ratio of x and y is defined as:

$$\begin{aligned} logL(x,y) &= ll(\frac{k_1}{n_1}, k_1, n_1) + ll(\frac{k_2}{n_2}, k_2, n_2) \\ &\quad -ll(\frac{k_1+k_2}{n_1+n_2}, k_1, n_1) \\ &\quad -ll(\frac{k_1+k_2}{n_1+n_2}, k_2, n_2) \end{aligned} \tag{10}$$

where $k_1 = C(x,y)$, $n_1 = C(x,*)$, $k_2 = C(\neg x, y)$, $n_2 = C(\neg x, *)$ and

$$ll(p,k,n) = k\log(p) + (n-k)\log(1-p) \tag{11}$$

The system computes dependency values between states (tree nodes) or observations (tree leaves) in the tree in the same way. The mutual information and log-likelihood values are highest when the words are adjacent to each other throughout the entire corpus. By using these two values, the method is more robust against low frequency events.

Figure 3 is a tree representation of the HHMM, the figure illustrates the flattening process for the sentence:

> (S (N^* A_AT1 graphical_JJ zoo_NN1 (P^* of_IO (N (strange_JJ and_CC peculiar_JJ) attractors_NN2)))).

where only the part-of-speech tags and grammar information are considered. The left hand side of the figure shows the original structure of the sentence, and the right hand side shows the transformed structure. The model's hierarchy is reduced by one level, where the state P^* has become a sub-state of state S instead of N^*. The process is likely to be useful when state P^* is highly dependent on state N^*.

The flattening process can be applied to the model based on two types of sequence dependancy; observation dependancy and state dependancy.

- **Observation dependency :** The observation dependency value is based upon the observation sequence, which in Figure 3 would be the sequence of part-of-speech tags {*AT1 JJ NN1 IO JJ CC JJ NN2*}. Given observations *NN1* and *IO*'s as terms with a high dependency value, the model then re-construct the sub-tree at *IO* parent state P^* moving it to the same level as state N^*, where the states of P^* and N^* now share the same parent, state *S*.

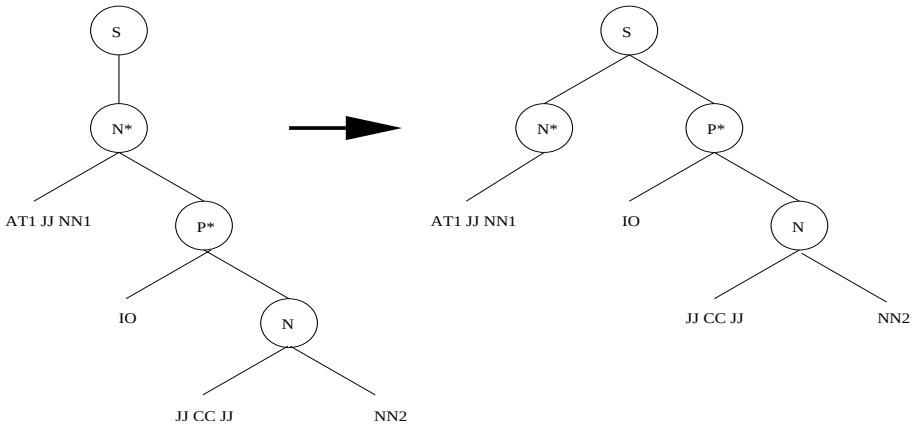


Figure 3: Partial flattening process for state N^* and P^*.

- **State dependency :** The state dependency value is based upon the state sequence, which in Figure 3 would be $\{N^*, P^*, N\}$. The flattening process occurs when the current state has a high dependency value with the previous state, say N^* and P^*.

term	dependency value
NN1 IO	570.55
IO JJ	570.55
JJ CC	570.55
CC JJ	570.55
JJ NN2	295.24
AT1 JJ	294.25
JJ NN1	294.25

Table 1: Observation dependency values of part-of-speech tags

This paper determines the high dependency values by selecting the top n values from a list of all possible terms ranked by either observation or state dependency values, where n is a parameter that can be configured by the user for better performance. Table 1 shows the observation dependency values of terms for part-of-speech tags for Figure 3. The term *NN1 IO* has a higher dependency value than *JJ NN1*, therefore state P^* is joined as a sub-tree of state *S*. States P^* and *N* remain unchanged since state P^* has already been moved up a level of the tree. After the flattening process, the state P^* no longer belongs to the child state of state N^*, and is instead joined as the sub-tree to state *S* as shown in Figure 3.

4 Application

4.1 Text Chunking

Text chunking involves producing non-overlapping segments of low-level noun groups. The system uses the clause information to construct the hierarchical structure of text chunks, where the clauses represent the phrases within the sentence. The clauses can be embedded in other clauses but cannot overlap one another. Furthermore each clause contains one or more text chunks.

Consider a sentence from a CoNLL-2004 corpus:

> (S (NP He_PRP) (VP reckons_VBZ) (S (NP the_DT current_JJ account_NN deficit_NN) (VP will_MD narrow_VB) (PP to_TO) (NP only_RB #_# 1.8_CD billion_D) (PP in_IN) (NP September_NNP)) (O ._.))

where the part-of-speech tag associated with each word is attached with an underscore, the clause information is identified by the *S* symbol and the chunk information is identified by the rest of the symbols *NP* (noun phrase), *VP* (verb phrase), *PP* (prepositional phrase) and *O* (null complementizer). The brackets are in Penn Treebank II style[3]. The sentence can be re-expressed just as its part-of-speech tags thusly: {PRP VBZ DT JJ NN NN MD VB TO RB # CD D IN NNP}, where only the part-of-speech tags and grammar information are to be considered for the extraction tasks. This is done so the system can minimise the computation cost inherent in learning a large number of unrequired observation symbols. Such an approach

[3]The Penn Treebank Project, http://www.cis.upenn.edu/~ treebank/home.html

also maximises the efficiency of trained data by learning the pattern that is hidden within words (syntax) rather than the words themselves (semantics).

Figure 4 represents an example of the tree representation of an HHMM for the text chunking task. This example involves a hierarchy with a depth of three. Note that state *NP* appears in two different levels of the hierarchy. In order to build an HHMM, the sentence shown above must be restructured as:

> (S (NP PRP) (VP VBZ) (S (NP DT JJ NN NN) (VP MD VB) (PP TO) (NP RB # CD D) (PP IN) (NP NNP)) (O .))

where the model makes no use of the word information contained in the sentence.

4.2 Grammar Parsing

Creation of a parse tree involves describing language grammar in a tree representation, where each path of the tree represents a grammar rule. Consider a sentence from the Lancaster Treebank[4]:

> (S (N A_AT1 graphical_JJ zoo_NN1 (P of_IO (N (strange_JJ and_CC peculiar_JJ) attractors_NN2))))

where the part-of-speech tag associated with each word is attached with an underscore, and the syntactic tag for each phrase occurs immediately after the opening square-bracket. In order to build the models from the parse tree, the system takes the part-of-speech tags as the observation sequences, and learns the structure of the model using the information expressed by the syntactic tags. During construction, phrases, such as the noun phrase "(strange_JJ and_CC peculiar_JJ)", are grouped under a dummy state (N_d). Figure 5 illustrates the model in the tree representation with the structure of the model based on the previous sentence from Lancaster Treebank.

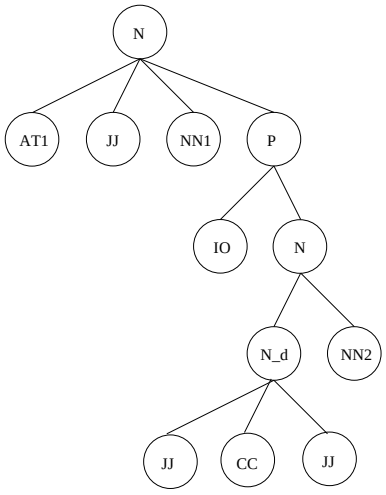


Figure 5: Parse tree for the HHMM

[4]Lancaster/IBM Treebank, http://www.ilc.cnr.it/EAGLES96/synlex/node23.html

5 Evaluation

The first evaluation presents preliminary evidence that the merged hierarchical hidden Markov Model (MHHMM) is able to produce more accurate results either a plain HHMM or a HMM during the text chunking task. The results suggest that the partial flattening process is capable of improving model accuracy when the input data contains complex hierarchical structures. The evaluation involves analysing the results over two sets of data. The first is a selection of data from CoNLL-2004 and contains 8936 sentences. The second dataset is part of the Lancaster Treebank corpus and contains 1473 sentences. Each sentence contains hand-labeled syntactic roles for natural language text.

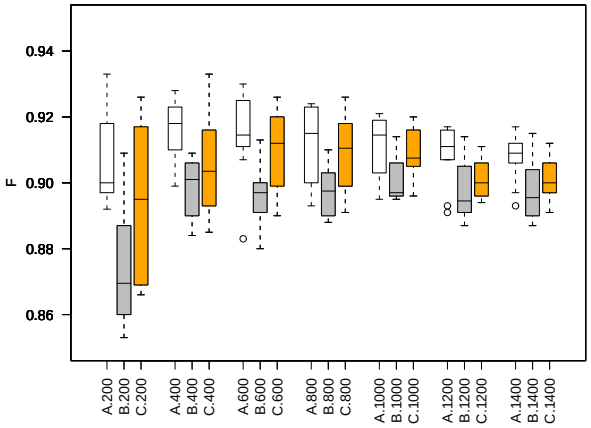


Figure 6: The graph of micro-average F-measure against the number of training sentences during text chunking (A: MHHMM, B: HHMM and C: HMM)

The first finding is that the size of training data dramatically affects the prediction accuracy. A model with an insufficient number of observations

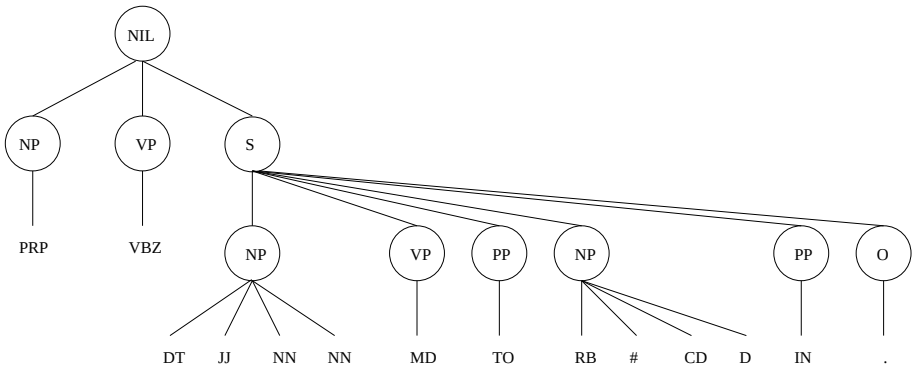


Figure 4: HHMM for syntax roles

typically has poor accuracy. In the text chunking task the number of observation symbol relies on the number of part-of-speech tags contained in training data. Figure 6 plots the relationship of micro-average F-measure for three types of models (A: MHHMM, B: HHMM and C: HMM) on 10-fold cross validation with the number of training sentences ranging from 200 to 1400. The result shows that the MHHMM has the better performance in accuracy over both the HHMM and HMM, although the difference is less marked for the latter.

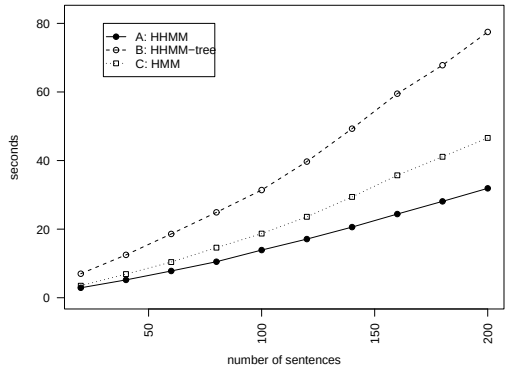


Figure 7: The average processing time for text chunking

Figure 7 represents the average processing time for testing (in seconds) for the 10-fold cross validation. The test were carried out on a dual P4-D computer running at 3GHz and with 1Gb RAM. The results indicate that the MHHMM gains efficiency, in terms of computation cost, by merging repeated sub-models, resulting in fewer states in the model. In contrast the HMM has lower efficiency as it is required to identify every single path, leading to more states within the model and higher computation cost. The extra costs of constructing a HHMM, which will have the same number of production states as the HMM, make it the least efficient.

The second evaluation presents preliminary evidence that the partially flattened hierarchical hidden Markov model (PFHHMM) can assign propositions to language texts (grammar parsing) at least as accurately as the HMM. This is assignment is a task that HHMMs are generally not well suited to. Table 2 shows the F_1-measures of identified semantic roles for each different model on the Lancaster Treebank data set. The models used in this evaluation were trained with observation data from the Lancaster Treebank training set. The training set and testing set are sub-divided from the corpus in proportions of $\frac{2}{3}$ and $\frac{1}{3}$. The PFHHMMs had extra training conditions as follows: *PFHHMM obs 2000* made use of the partial flattening process, with the high dependency parameter determined by considering the highest 2000 dependency values from observation sequences from the corpus. *PFHHMM state 150* again uses partial flattening, however this time the highest 150 dependency values from state sequences were utilized in discovering the high dependency threshold. The n values of 2000 and 150 were determined to be the optimal values when applied to the training set.

The results show that applying the partial flattening process to a model using observation sequences to determine high dependency values reduces the complexity of the model's hierarchy and consequently improves the model's accuracy. The state dependency method is shown to be less favorable for this particular task, but the micro-average result is still comparable with the HMM's performance. The results also show no significant re-

State	Count	HMM	HHMM	PFHHMM obs 2000	PFHHMM state 150
N	16387	0.874	0.811	0.882	0.874
NULL	4670	0.794	0.035	0.744	0.743
V	4134	0.768	0.755	0.804	0.791
P	2099	0.944	0.936	0.928	0.926
Fa	180	0.525	0.814	0.687	0.457
Micro-Average		0.793	0.701	0.809	0.792

Table 2: F1-measure of top 5 states during grammar parsing set.

lationship between the occurance count of a state against the various models prediction accuracy.

6 Discussion and Future Work

Due to the hierarchical structure of a HHMM, the model has the advantage of being able to reuse information for repeated sub-models. Thus the HHMM can perform more accurately and requires less computational time than the HMM in certain situations.

The merging and flattening techniques have been shown to be effective and could be applied to many kinds of data with hierarchical structures. The methods are especially appealing where the model involves complex structure or there is a shortage of training data. Furthermore, they address an important issue when dealing with small datasets: by using the hierarchical model to uncover less obvious structures, the model is able to increase model performance even over more limited source materials. The experimental results have shown the potential of the merging and partial flattening techniques in building hierarchical models and providing better handling of states with less observation counts. Further research in both experimental and theoretical aspects of this work is planned, specifically in the area of reconstructing hierarchies where recursive formations are present and formal analysis and testing of techniques.

References

D. M. Bikel, R. Schwartz and R. M. Weischedel. 1999. An Algorithm that Learns What's in a Name. *Machine Learning*, 34:211–231.

V. R. Borkar, K. Deshmukh and S. Sarawagi. 2001. Automatic Segmentation of Text into Structured Records. *Proceedings of SIGMOD.*

E. Brill. 1995. Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565.

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

S. Fine , Y. Singer and N. Tishby. 1998. The Hierarchical Hidden Markov Model: Analysis and Applications. *Machine Learning*, 32:41–62.

A. Krotov, M Heple, R. Gaizauskas and Y. Wilks. 1999. Compacting the Penn Treebank Grammar. *Proceedings of COLING-98* (Montreal), pages 699-703.

A. McCallum, K. Nigam, J. Rennie and K. Seymore. 1999. Building Domain-Specific Search Engines with Machine Learning Techniques. *In AAAI-99 Spring Symposium on Intelligent Agents in Cyberspace*.

R. Nag, K. H. Wong, and F. Fallside. 1986. Script Recognition Using Hidden Markov Models. *Proc. of ICASSP 86*, pp. 2071-1074, Toyko.

P. Pantel and D. Lin. 2001. A Statistical Corpus-Based Term Extractor. In Stroulia, E. and Matwin, S. (Eds.) AI 2001. *Lecture Notes in Artificial Intelligence*, pp. 36-46. Springer-Verlag.

L. R. Rabiner and B. H. Juang. 1986. An Introduction to Hidden Markov Models. *IEEE Acoustics Speech and Signal Processing ASSP Magazine*, ASSP-3(1): 4–16, January.

M. Skounakis, M. Craven and S. Ray. 2003. Hierarchical Hidden Markov Models for Information Extraction. *In Proceedings of the 18th International Joint Conference on Artificial Intelligence*, Acapulco, Mexico, Morgan Kaufmann.

A. J. Viterbi. 1967. Error bounds for convolutional codes and an asymtotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, IT-13:260–267.

Analysis and Synthesis of the Distribution of Consonants over Languages: A Complex Network Approach

Monojit Choudhury and **Animesh Mukherjee** and **Anupam Basu** and **Niloy Ganguly**
Department of Computer Science and Engineering,
Indian Institute of Technology Kharagpur
{monojit,animeshm,anupam,niloy}@cse.iitkgp.ernet.in

Abstract

Cross-linguistic similarities are reflected by the speech sound systems of languages all over the world. In this work we try to model such similarities observed in the consonant inventories, through a complex bipartite network. We present a systematic study of some of the appealing features of these inventories with the help of the bipartite network. An important observation is that the occurrence of consonants follows a two regime power law distribution. We find that the consonant inventory size distribution together with the principle of preferential attachment are the main reasons behind the emergence of such a two regime behavior. In order to further support our explanation we present a synthesis model for this network based on the general theory of preferential attachment.

1 Introduction

Sound systems of the world's languages show remarkable regularities. Any arbitrary set of consonants and vowels does not make up the sound system of a particular language. Several lines of research suggest that cross-linguistic similarities get reflected in the consonant and vowel inventories of the languages all over the world (Greenberg, 1966; Pinker, 1994; Ladefoged and Maddieson, 1996). Previously it has been argued that these similarities are the results of certain general principles like *maximal perceptual contrast* (Lindblom and Maddieson, 1988), *feature economy* (Martinet, 1968; Boersma, 1998; Clements, 2004) and *robustness* (Jakobson and Halle, 1956; Chomsky and Halle, 1968). Maximal perceptual contrast between the phonemes of a language is desirable for proper perception in a noisy environment. In fact the organization of the vowel inventories across languages has been satisfactorily explained in terms of the single principle of maximal perceptual contrast (Jakobson, 1941; Wang, 1968).

There have been several attempts to reason the observed patterns in consonant inventories since 1930s (Trubetzkoy, 1969/1939; Lindblom and Maddieson, 1988; Boersma, 1998; Flemming, 2002; Clements, 2004), but unlike the case of vowels, the structure of consonant inventories lacks a complete and holistic explanation (de Boer, 2000). Most of the works are confined to certain individual principles (Abry, 2003; Hinskens and Weijer, 2003) rather than formulating a general theory describing the structural patterns and/or their stability. Thus, the structure of the consonant inventories continues to be a *complex* jigsaw puzzle, though the parts and pieces are known.

In this work we attempt to represent the cross-linguistic similarities that exist in the consonant inventories of the world's languages through a *bipartite network* named **PlaNet** (the **P**honeme **L**anguage **N**etwork). PlaNet has two different sets of nodes, one labeled by the languages while the other labeled by the consonants. Edges run between these two sets depending on whether or not a particular consonant occurs in a particular language. This representation is motivated by similar modeling of certain complex phenomena observed in nature and society, such as,

- Movie-actor network, where movies and actors constitute the two partitions and an edge between them signifies that a particular actor acted in a particular movie (Ramasco et al., 2004).

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 128–135,
Sydney, July 2006.

- Article-author network, where the edges denote which person has authored which articles (Newman, 2001b).

- Metabolic network of organisms, where the corresponding partitions are chemical compounds and metabolic reactions. Edges run between partitions depending on whether a particular compound is a substrate or result of a reaction (Jeong et al., 2000).

Modeling of complex systems as networks has proved to be a comprehensive and emerging way of capturing the underlying generating mechanism of such systems (for a review on complex networks and their generation see (Albert and Barabási, 2002; Newman, 2003)). There have been some attempts as well to model the intricacies of human languages through complex networks. Word networks based on synonymy (Yook et al., 2001b), co-occurrence (Cancho et al., 2001), and phonemic edit-distance (Vitevitch, 2005) are examples of such attempts. The present work also uses the concept of complex networks to develop a platform for a holistic analysis as well as synthesis of the distribution of the consonants across the languages.

In the current work, with the help of PlaNet we provide a systematic study of certain interesting features of the consonant inventories. An important property that we observe is the two regime power law degree distribution[1] of the nodes labeled by the consonants. We try to explain this property in the light of the size of the consonant inventories coupled with the principle of *preferential attachment* (Barabási and Albert, 1999). Next we present a simplified mathematical model explaining the emergence of the two regimes. In order to support our analytical explanations, we also provide a synthesis model for PlaNet.

The rest of the paper is organized into five sections. In section 2 we formally define PlaNet, outline its construction procedure and present some studies on its degree distribution. We dedicate section 3 to state and explain the inferences that can be drawn from the degree distribution studies of PlaNet. In section 4 we provide a simplified theoretical explanation of the analytical results obtained. In section 5 we present a synthesis model for PlaNet to hold up the inferences that we draw in section 3. Finally we conclude in section 6 by summarizing our contributions, pointing out some of the implications of the current work and indicating the possible future directions.

[1]Two regime power law distributions have also been observed in syntactic networks of words (Cancho et al., 2001), network of mathematics collaborators (Grossman et al., 1995), and language diversity over countries (Gomes et al., 1999).

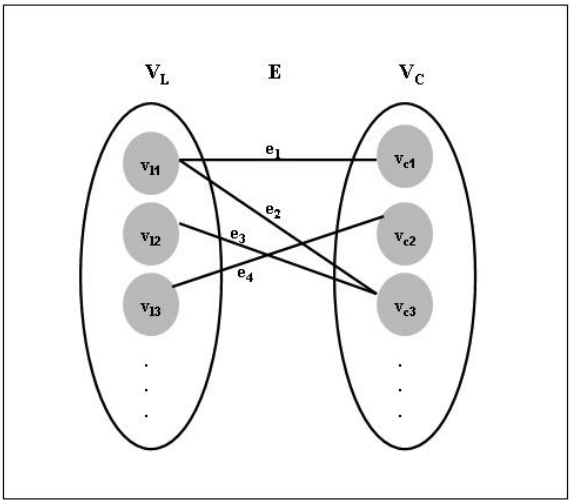


Figure 1: Illustration of the nodes and edges of PlaNet

2 PlaNet: The Phoneme-Language Network

We define the network of consonants and languages, PlaNet, as a *bipartite graph* represented as G = $\langle$V$_L$, V$_C$, E$\rangle$ where V$_L$ is the set of *nodes* labeled by the languages and V$_C$ is the set of nodes labeled by the consonants. E is the set of edges that run between V$_L$ and V$_C$. There is an *edge* $e \in$ E between two nodes $v_l \in$ V$_L$ and $v_c \in$ V$_C$ if and only if the consonant *c* occurs in the language *l*. Figure 1 illustrates the nodes and edges of PlaNet.

2.1 Construction of PlaNet

Many typological studies (Lindblom and Maddieson, 1988; Ladefoged and Maddieson, 1996; Hinskens and Weijer, 2003) of segmental inventories have been carried out in past on the UCLA Phonological Segment Inventory Database (UPSID) (Maddieson, 1984). UPSID initially had 317 languages and was later extended to include 451 languages covering all the major language families of the world. In this work we have used the older version of UPSID comprising of 317 languages and 541 consonants (henceforth UPSID$_{317}$), for constructing PlaNet. Consequently, there are 317 elements (nodes) in the set V$_L$ and 541 elements

(nodes) in the set V_C. The number of elements (edges) in the set E as computed from PlaNet is 7022. At this point it is important to mention that in order to avoid any confusion in the construction of PlaNet we have appropriately filtered out the anomalous and the ambiguous segments (Maddieson, 1984) from it. We have completely ignored the anomalous segments from the data set (since the existence of such segments is doubtful), and included the ambiguous ones as separate segments because there are no descriptive sources explaining how such ambiguities might be resolved. A similar approach has also been described in Pericliev and Valdés-Pérez (2002).

2.2 Degree Distribution of PlaNet

The *degree* of a node u, denoted by k_u is defined as the number of edges connected to u. The term *degree distribution* is used to denote the way degrees (k_u) are distributed over the nodes (u). The degree distribution studies find a lot of importance in understanding the complex topology of any large network, which is very difficult to visualize otherwise. Since PlaNet is bipartite in nature it has two degree distribution curves one corresponding to the nodes in the set V_L and the other corresponding to the nodes in the set V_C.

Degree distribution of the nodes in V_L: Figure 2 shows the degree distribution of the nodes in V_L where the x-axis denotes the degree of each node expressed as a fraction of the maximum degree and the y-axis denotes the number of nodes having a given degree expressed as a fraction of the total number of nodes in V_L .

It is evident from Figure 2 that the number of consonants appearing in different languages follow a β-distribution [2] (see (Bulmer, 1979) for reference). The figure shows an asymmetric right skewed distribution with the values of α and β equal to 7.06 and 47.64 (obtained using maximum likelihood estimation method) respectively. The asymmetry points to the fact that languages usually tend to have smaller consonant inventory size,

[2] A random variable is said to have a β-distribution with parameters $\alpha > 0$ and $\beta > 0$ if and only if its probability mass function is given by

$$f(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}$$

for 0 < x < 1 and $f(x)$ = 0 otherwise. $\Gamma(\cdot)$ is the Euler's gamma function.

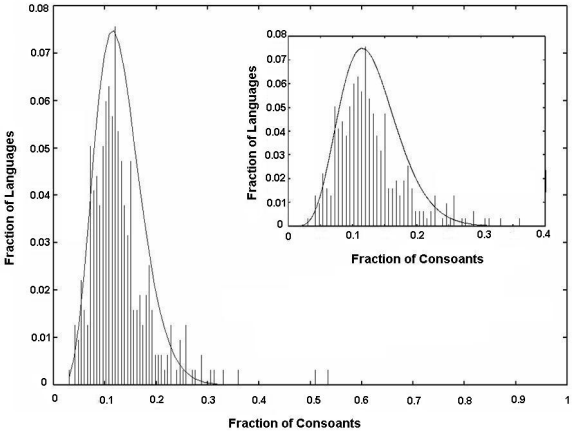


Figure 2: Degree distribution of PlaNet for the set V_L. The figure in the inner box is a magnified version of a portion of the original figure.

the best value being somewhere between 10 and 30. The distribution peaks roughly at 21 indicating that majority of the languages in $UPSID_{317}$ have a consonant inventory size of around 21 consonants.

Degree distribution of the nodes in V_C: Figure 3 illustrates two different types of degree distribution plots for the nodes in V_C; Figure 3(a) corresponding to the rank, i.e., the sorted order of degrees, (x-axis) versus degree (y-axis) and Figure 3(b) corresponding to the degree (k) (x-axis) versus P_k (y-axis) where P_k is the fraction of nodes having degree greater than or equal to k.

Figure 3 clearly shows that both the curves have two distinct regimes and the distribution is scale-free. Regime 1 in Figure 3(a) consists of 21 consonants which have a very high frequency (i.e., the degree k) of occurrence. Regime 2 of Figure 3(b) also correspond to these 21 consonants. On the other hand Regime 2 of Figure 3(a) as well as Regime 1 of Figure 3(b) comprises of the rest of the consonants. The point marked as **x** in both the figures indicates the breakpoint. Each of the regime in both Figure 3(a) and (b) exhibit a power law of the form

$$y = Ax^{-\alpha}$$

In Figure 3(a) y represents the degree k of a node corresponding to its rank x whereas in Figure 3(b) y corresponds to P_k and x, the degree k. The values of the parameters A and α, for Regime 1 and Regime 2 in both the figures, as computed by the least square error method, are shown in Table 1.

Regime	Figure 3(a)	Figure 3(b)
Regime 1	A = 368.70 α = 0.4	A = 1.040 α = 0.71
Regime 2	A = 12456.5 α = 1.54	A = 2326.2 α = 2.36

Table 1: The values of the parameters A and α

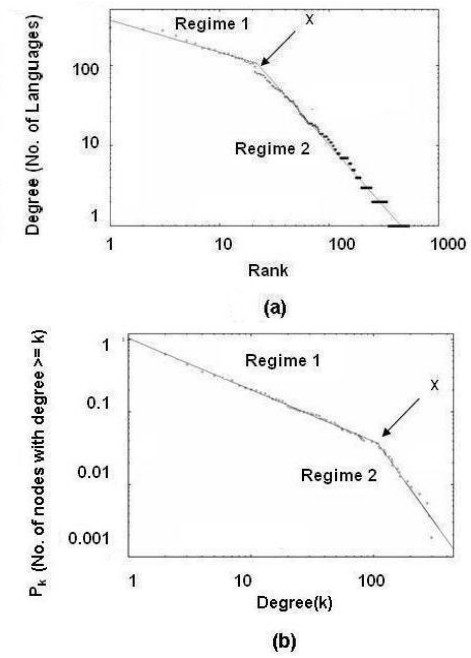


Figure 3: Degree distribution of PlaNet for the set V_C in a log-log scale

It becomes necessary to mention here that such power law distributions, known variously as Zipf's law (Zipf, 1949), are also observed in an extraordinarily diverse range of phenomena including the frequency of the use of words in human language (Zipf, 1949), the number of papers scientists write (Lotka, 1926), the number of hits on web pages (Adamic and Huberman, 2000) and so on. Thus our inferences, detailed out in the next section, mainly centers around this power law behavior.

3 Inferences Drawn from the Analysis of PlaNet

In most of the networked systems like the society, the Internet, the World Wide Web, and many others, power law degree distribution emerges for the phenomenon of preferential attachment, i.e., when "the rich get richer" (Simon, 1955). With reference to PlaNet this preferential attachment can be interpreted as the tendency of a language to choose a consonant that has been already chosen by a large number of other languages. We posit that it is this preferential property of languages that results in the power law degree distributions observed in Figure 3(a) and (b).

Nevertheless there is one question that still remains unanswered. Whereas the power law distribution is well understood, the reason for the two distinct regimes (with a sharp break) still remains unexplored. We hypothesize that,

Hypothesis *The typical distribution of the consonant inventory size over languages coupled with the principle of preferential attachment enforces the two distinct regimes to appear in the power law curves.*

As the average consonant inventory size in UPSID_{317} is 21, so following the principle of preferential attachment, on an average, the first 21 most frequent consonants are much more preferred than the rest. Consequently, the nature of the frequency distribution for the highly frequent consonants is different from the less frequent ones, and hence there is a transition from Regime 1 to Regime 2 in the Figure 3(a) and (b).

Support Experiment: In order to establish that the consonant inventory size plays an important role in giving rise to the two regimes discussed above we present a support experiment in which we try to observe whether the breakpoint **x** shifts as we shift the average consonant inventory size.

Experiment: In order to shift the average consonant inventory size from 21 to 25, 30 and 38 we neglected the contribution of the languages with consonant inventory size less than n where n is 15, 20 and 25 respectively and subsequently recorded the degree distributions obtained each time. We did not carry out our experiments for average consonant inventory size more than 38 because the number of such languages are very rare in UPSID_{317}.

Observations: Figure 4 shows the effect of this shifting of the average consonant inventory size on the rank versus degree distribution curves. Table 2 presents the results observed from these curves with the left column indicating the average inventory size and the right column the breakpoint **x**.

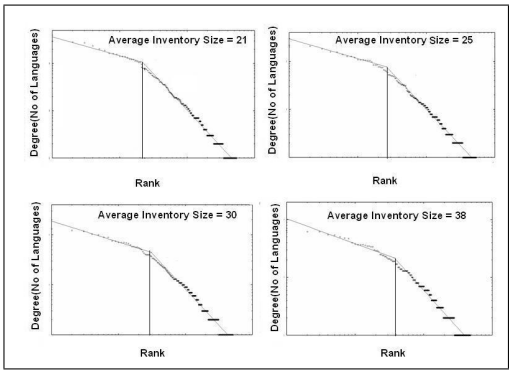


Figure 4: Degree distributions at different average consonant inventory sizes

Avg. consonant inv. size	Transition
25	25
30	30
38	37

Table 2: The transition points for different average consonant inventory size

The table clearly indicates that the transition occurs at values corresponding to the average consonant inventory size in each of the three cases.

Inferences: It is quite evident from our observations that the breakpoint **x** has a strong correlation with the average consonant inventory size, which therefore plays a key role in the emergence of the two regime degree distribution curves.

In the next section we provide a simplistic mathematical model for explaining the two regime power law with a breakpoint corresponding to the average consonant inventory size.

4 Theoretical Explanation for the Two Regimes

Let us assume that the inventory of all the languages comprises of 21 consonants. We further assume that the consonants are arranged in their hierarchy of preference. A language traverses the hierarchy of consonants and at every step decides with a probability p to choose the current consonant. It stops as soon as it has chosen all the 21 consonants. Since languages must traverse through the first 21 consonants regardless of whether the previous consonants are chosen or not, the probability of choosing any one of these 21 consonants must be p. But the case is different for the 22^{nd} consonant, which is chosen by a language if it has previously chosen zero, one, two, or at most 20, but not all of the first 21 consonants. Therefore, the probability of the 22^{nd} consonant being chosen is,

$$P(22) = p\sum_{i=0}^{20} \left(\begin{array}{c} 21 \\ i \end{array} \right) p^i(1-p)^{21-i}$$

where

$$\left(\begin{array}{c} 21 \\ i \end{array} \right) p^i(1-p)^{21-i}$$

denotes the probability of choosing i consonants from the first 21. In general the probability of choosing the n+1^{th} consonant from the hierarchy is given by,

$$P(n+1) = p\sum_{i=0}^{20} \left(\begin{array}{c} n \\ i \end{array} \right) p^i(1-p)^{n-i}$$

Figure 5 shows the plot of the function $P(n)$ for various values of p which are 0.99, 0.95, 0.9, 0.85, 0.75 and 0.7 respectively in log-log scale. All the curves, for different values of p, have a nature similar to that of the degree distribution plot we obtained for PlaNet. This is indicative of the fact that languages choose consonants from the hierarchy with a probability function comparable to $P(n)$.

Owing to the simplified assumption that all the languages have only 21 consonants, the first regime is a straight line; however we believe a more rigorous mathematical model can be built taking into consideration the β-distribution rather than just the mean value of the inventory size that can explain the negative slope of the first regime. We look forward to do the same as a part of our future work. Rather, here we try to investigate the effect of the exact distribution of the language inventory size on the nature of the degree distribution of the consonants through a synthetic approach based on the principle of preferential attachment, which is described in the subsequent section.

5 The Synthesis Model based on Preferential Attachment

Albert and Barabási (1999) observed that a common property of many large networks is that the vertex connectivities follow a scale-free power law distribution. They remarked that two generic mechanisms can be considered to be the cause of this observation: (i) networks expand continuously by the addition of new vertices, and (ii) new vertices attach preferentially to sites (vertices) that are already well connected. They found that

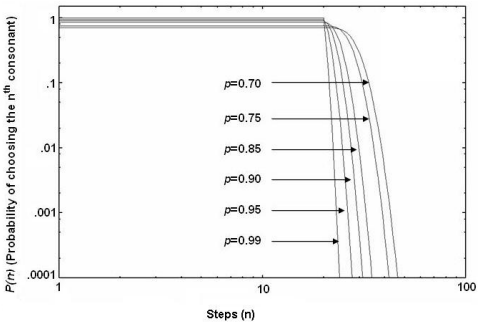


Figure 5: Plot of the function $P(n)$ in log-log scale

a model based on these two ingredients reproduces the observed stationary scale-free distributions, which in turn indicates that the development of large networks is governed by robust self-organizing phenomena that go beyond the particulars of the individual systems.

Inspired by their work and the empirical as well as the mathematical analysis presented above, we propose a preferential attachment model for synthesizing PlaNet (PlaNet$_{syn}$ henceforth) in which the degree distribution of the nodes in V_L is known. Hence V_L=$\{\mathrm{L}_1, \mathrm{L}_2, \ldots, \mathrm{L}_{317}\}$ have degrees (consonant inventory size) $\{k_1, k_2, \ldots, k_{317}\}$ respectively. We assume that the nodes in the set V_C are *unlabeled*. At each time step, a node L_j (j = 1 to 317) from V_L tries to attach itself with a new node $i \in \mathrm{V}_C$ to which it is not already connected. The probability $Pr(i)$ with which the node L_j gets attached to i depends on the current degree of i and is given by

$$Pr(i) = \frac{k_i + \epsilon}{\sum_{i' \in V_j}(k_{i'} + \epsilon)}$$

where k_i is the current degree of the node i, V_j is the set of nodes in V_C to which L_j is not already connected and ϵ is the smoothing parameter which is used to reduce bias and favor at least a few attachments with nodes in V_j that do not have a high $Pr(i)$. The above process is repeated until all $\mathrm{L}_j \in \mathrm{V}_L$ get connected to exactly k_j nodes in V_C. The entire idea is summarized in Algorithm 1. Figure 6 shows a partial step of the synthesis process illustrated in Algorithm 1.

Simulation Results: Simulations reveal that for PlaNet$_{syn}$ the degree distribution of the nodes belonging to V_C fit well with the analytical results we obtained earlier in section 2. Good fits emerge

repeat
  for *j = 1 to 317* **do**
    if *there is a node* $L_j \in V_L$ *with at least one or more consonants to be chosen from* V_C **then**
      Compute V_j = V_C-$V(L_j)$, where $V(L_j)$ is the set of nodes in V_C to which L_j is already connected;
    end
    for *each node i* $\in V_j$ **do**
      $$Pr(i) = \frac{k_i + \epsilon}{\sum_{i' \in V_j}(k_{i'} + \epsilon)}$$
      where k_i is the current degree of the node i and ϵ is the model parameter. $Pr(i)$ is the probability of connecting L_j to i.
    end
    Connect L_j to a node $i \in V_j$ following the distribution $Pr(i)$;
  end
until *all languages complete their inventory quota* ;

Algorithm 1: Algorithm for synthesis of PlaNet based on preferential attachment

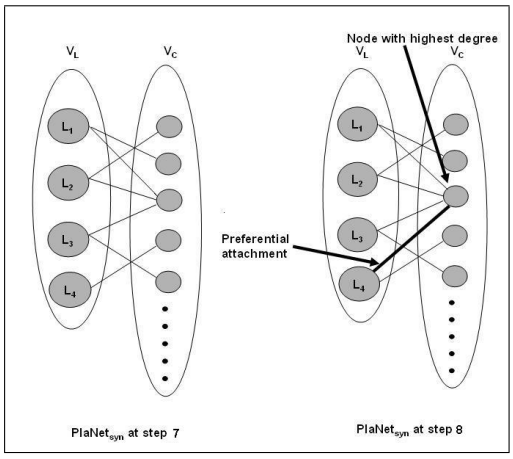


Figure 6: A partial step of the synthesis process. When the language L_4 has to connect itself with one of the nodes in the set V_C it does so with the one having the highest degree (=3) rather than with others in order to achieve preferential attachment which is the working principle of our algorithm

for the range $0.06 \leq \epsilon \leq 0.08$ with the best being at ϵ = 0.0701. Figure 7 shows the degree k versus

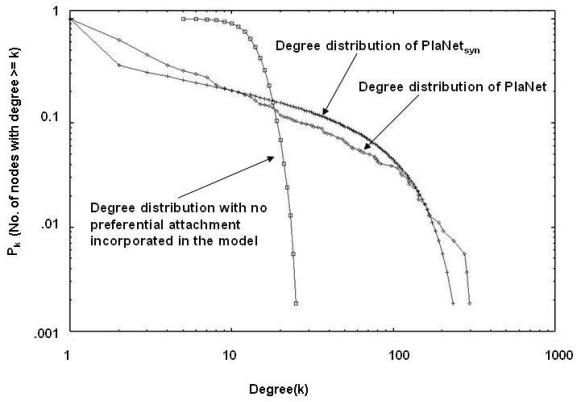


Figure 7: Degree distribution of the nodes in V_C for both PlaNet$_{syn}$, PlaNet, and when the model incorporates no preferential attachment; for PlaNet$_{syn}$, ϵ = 0.0701 and the results are averaged over 100 simulation runs

P_k plots for ϵ = 0.0701 averaged over 100 simulation runs.

The mean error[3] between the degree distribution plots of PlaNet and PlaNet$_{syn}$ is 0.03 which intuitively signifies that on an average the variation in the two curves is 3%. On the contrary, if there were no preferential attachment incorporated in the model (i.e., all connections were equiprobable) then the mean error would have been 0.35 (35% variation on an average).

6 Conclusions, Discussion and Future Work

In this paper, we have analyzed and synthesized the consonant inventories of the world's languages in terms of a complex network. We dedicated the preceding sections essentially to,

- Represent the consonant inventories through a bipartite network called PlaNet,
- Provide a systematic study of certain important properties of the consonant inventories with the help of PlaNet,
- Propose analytical explanations for the two regime power law curves (obtained from PlaNet) on the basis of the distribution of the consonant inventory size over languages together with the principle of preferential attachment,
- Provide a simplified mathematical model to support our analytical explanations, and
- Develop a synthesis model for PlaNet based on preferential attachment where the consonant inventory size distribution is known *a priori*.

We believe that the general explanation provided here for the two regime power law is a fundamental result, and can have a far reaching impact, because two regime behavior is observed in many other networked systems.

Until now we have been mainly dealing with the computational aspects of the distribution of consonants over the languages rather than exploring the real world dynamics that gives rise to such a distribution. An issue that draws immediate attention is that how preferential attachment, which is a general phenomenon associated with network evolution, can play a prime role in shaping the consonant inventories of the world's languages. The answer perhaps is hidden in the fact that language is an evolving system and its present structure is determined by its past evolutionary history. Indeed an explanation based on this evolutionary model, with an initial disparity in the distribution of consonants over languages, can be intuitively verified as follows – let there be a language community of N speakers communicating among themselves by means of only two consonants say /k/ and /g/. If we assume that every speaker has l descendants and language inventories are transmitted with high fidelity, then after i generations it is expected that the community will consist of ml^i /k/ speakers and nl^i /g/ speakers. Now if $m > n$ and $l > 1$, then for sufficiently large i, $ml^i \gg nl^i$. Stated differently, the /k/ speakers by far outnumbers the /g/ speakers even if initially the number of /k/ speakers is only slightly higher than that of the /g/ speakers. This phenomenon is similar to that of preferential attachment where language communities get attached to, i.e., select, consonants that are already highly preferred. Nevertheless, it remains to be seen where from such an initial disparity in the distribution of the consonants over languages might have originated.

In this paper, we mainly dealt with the occurrence principles of the consonants in the inventories of the world's languages. The work can be further extended to identify the co-occurrence likelihood of the consonants in the language inventories

[3]Mean error is defined as the average difference between the ordinate pairs where the abscissas are equal.

and subsequently identify the groups or communities within them. Information about such communities can then help in providing an improved insight about the organizing principles of the consonant inventories.

References

C. Abry. 2003. [b]-[d]-[g] as a universal triangle as acoustically optimal as [i]-[a]-[u]. *15th Int. Congr. Phonetics Sciences ICPhS*, 727–730.

L. A. Adamic and B. A. Huberman. 2000. The nature of markets in the World Wide Web. *Quarterly Journal of Electronic Commerce 1*, 512.

R. Albert and A.-L. Barabási. 2002. Statistical mechanics of complex networks. *Reviews of Modern Physics 74*, 47–97.

A.-L. Barabási and R. Albert. 1999. Emergence of scaling in random networks. *Science 286*, 509-512.

Bart de Boer. 2000. Self-Organisation in Vowel Systems. *Journal of Phonetics*, Elsevier.

P. Boersma. 1998. *Functional Phonology. (Doctoral thesis, University of Amsterdam)*, The Hague: Holland Academic Graphics.

M. G. Bulmer. 1979. *Principles of Statistics*, Mathematics.

Ferrer i Cancho and R. V. Solé. 2001. Santa Fe working paper 01-03-016.

N. Chomsky and M. Halle. 1968. *The Sound Pattern of English*, New York: Harper and Row.

N. Clements. 2004. Features and Sound Inventories. *Symposium on Phonological Theory: Representations and Architecture*, CUNY.

E. Flemming. 2002. *Auditory Representations in Phonology*, New York and London: Routledge.

M. A. F. Gomes, G. L. Vasconcelos, I. J. Tsang, and I. R. Tsang. 1999. Scaling relations for diversity of languages. Physica A, 271, 489.

J. H. Greenberg. 1966. *Language Universals with Special Reference to Feature Hierarchies*, The Hague Mouton.

J. W. Grossman and P. D. F. Ion. 1995. On a portion of the well-known collaboration graph. *Congressus Numerantium*, 108, 129-131.

F. Hinskens and J. Weijer. 2003. Patterns of segmental modification in consonant inventories: a cross-linguistic study. *Linguistics.*

R. Jakobson. 1941. *Kindersprache, Aphasie und allgemeine Lautgesetze*, Uppsala, Reprinted in *Selected Writings I. Mouton*, The Hague, 1962, pages 328-401.

H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A. L. Barabási. 2000. The large-scale organization of metabolic networks. Nature, 406:651-654.

R. Jakobson and M. Halle. 1956. *Fundamentals of Language*, The Hague: Mouton and Co.

P. Ladefoged and I. Maddieson. 1996. *Sounds of the Worlds Languages*, Oxford: Blackwell.

B. Lindblom and I. Maddieson. 1988. Phonetic Universals in Consonant Systems. In L.M. Hyman and C.N. Li, eds., *Language, Speech, and Mind*, Routledge, London, 62–78.

A. J. Lotka. 1926. The frequency distribution of scientific production. *J. Wash. Acad. Sci. 16*, 317-323.

I. Maddieson. 1984. *Patterns of Sounds*, Cambridge University Press, Cambridge.

A. Martinet. 1968. Phonetics and linguistic evolution. In Bertil Malmberg (ed.), *Manual of phonetics, revised and extended edition*, Amsterdam: North-Holland Publishing Co. 464–487.

M. E. J. Newman. 2001b. Scientific collaboration networks. *I and II. Phys. Rev.*, E 64.

M. E. J. Newman. 2003. The structure and function of complex networks. *SIAM Review* 45, 167–256.

V. Pericliev, R. E. Valdés-Pérez. 2002. Differentiating 451 languages in terms of their segment inventories. *Studia Linguistica*, Blackwell Publishing.

S. Pinker. 1994. *The Language Instinct*, New York: Morrowo.

José J. Ramasco, S. N. Dorogovtsev, and Romualdo Pastor-Satorras. 2004. Self-organization of collaboration networks. *Physical Review E*, 70, 036106.

H. A. Simon. 1955. On a class of skew distribution functions. *Biometrika 42*, 425-440.

N. Trubetzkoy. 1969. *Principles of phonology. (English translation of Grundzüge der Phonologie, 1939)*, Berkeley: University of California Press.

M. S. Vitevitch. 2005. Phonological neighbors in a small world: What can graph theory tell us about word learning? *Spring 2005 Talk Series on Networks and Complex Systems*, Indiana University, Bloomington.

William S.-Y. Wang. 1968. The basis of speech, Project on Linguistic Analysis Reports, University of California at Berkeley. Reprinted in *The Learning of Language*, ed. by C. E. Reed, 1971.

S. Yook, H. Jeong and A.-L. Barabási. 2001b. preprint.

G. K. Zipf. 1949. *Human Behaviour and the Principle of Least Effort*, Addison-Wesley, Reading, MA.

Using Machine-Learning to Assign Function Labels to Parser Output for Spanish

Grzegorz Chrupała[1] and **Josef van Genabith**[1,2]
[1]National Center for Language Technology
Dublin City University
Glasnevin, Dublin 9, Ireland
[2]IBM Dublin Center for Advanced Studies
grzegorz.chrupala@computing.dcu.ie
josef@computing.dcu.ie

Abstract

Data-driven grammatical function tag assignment has been studied for English using the Penn-II Treebank data. In this paper we address the question of whether such methods can be applied successfully to other languages and treebank resources. In addition to tag assignment accuracy and f-scores we also present results of a task-based evaluation. We use three machine-learning methods to assign Cast3LB function tags to sentences parsed with Bikel's parser trained on the Cast3LB treebank. The best performing method, SVM, achieves an f-score of 86.87% on gold-standard trees and 66.67% on parser output - a statistically significant improvement of 6.74% over the baseline. In a task-based evaluation we generate LFG functional-structures from the function-tag-enriched trees. On this task we achive an f-score of 75.67%, a statistically significant 3.4% improvement over the baseline.

1 Introduction

The research presented in this paper forms part of an ongoing effort to develop methods to induce wide-coverage multilingual Lexical-Functional Grammar (LFG) (Bresnan, 2001) resources from treebanks by means of automatically associating LFG f-structure information with constituency trees produced by probabilistic parsers (Cahill et al., 2004). Inducing deep syntactic analyses from treebank data avoids the cost and time involved in manually creating wide-coverage resources.

Lexical Functional Grammar f-structures provide a level of syntactic representation based on the notion of grammatical functions (e.g. Subject, Object, Oblique, Adjunct etc.). This level is more abstract and cross-linguistically more uniform than constituency trees. F-structures also include explicit encodings of phenomena such as control and raising, pro-drop or long distance dependencies. Those characteristics make this level a suitable representation for many NLP applications such as transfer-based Machine Translation or Question Answering.

The f-structure annotation algorithm used for inducing LFG resources from the Penn-II treebank for English (Cahill et al., 2004) uses configurational, categorial, function tag and trace information. In contrast to English, in many other languages configurational information is not a good predictor for LFG grammatical function assignment. For such languages the function tags included in many treebanks are a much more important source of information for the LFG annotation algorithm than Penn-II tags are for English.

Cast3LB (Civit and Martí, 2004), the Spanish treebank used in the current research, contains comprehensive grammatical function annotation. In the present paper we use a machine-learning approach in order to add Cast3LB function tags to nodes of basic constituent trees output by a probabilistic parser trained on Cast3LB. To our knowledge, this paper is the first to describe applying a data-driven approach to function-tag assignment to a language other than English.

Our method statistically significantly outperforms the previously used approach which relied exclusively on the parser to produce trees with Cast3LB tags (O'Donovan et al., 2005). Additionally, we perform a task-driven evaluation of our Cast3LB tag assignment method by using the tag-enriched trees as input to the Spanish LFG f-structure annotation algorithm and evaluating the quality of the resulting f-structures.

Section 2 describes the Spanish Cast3LB treebank. In Section 3 we describe previous research in LFG induction for English and Spanish as well

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 136–143,
Sydney, July 2006.

as research on data-driven function tag assignment to parsed text in English. Section 4 provides the details of our approach to the Cast3LB function tag assignment task. In Sections 5 and 6 we present evaluation results for our method. In Section 7 we present the error analysis of the results. Finally, in Section 8 we conclude and discuss ideas for further research.

2 The Spanish Treebank

As input to our LFG annotation algorithm we use the output of Bikel's parser (Bikel, 2002) trained on the Cast3LB treebank (Civit and Martí, 2004). Cast3LB contains around 3,500 constituency trees (100,000 words) taken from different genres of European and Latin American Spanish. The POS tags used in Cast3LB encode morphological information in addition to Part-of-Speech information.

Due to the relatively flexible order of main sentence constituents in Spanish, Cast3LB uses a flat, multiply-branching structure for the S node. There is no VP node, but rather all complements and adjuncts depending on a verb are sisters to the *gv* (Verb Group) node containing this verb. An example sentence (with the corresponding f-structure) is shown in Figure 1.

Tree nodes are additionally labelled with grammatical function tags. Table 1 provides a list of function tags with short explanations. Civit (2004) provides Cast3LB function tag guidelines.

Functional tags carry some of the information that would be encoded in terms of tree configurations in languages with stricter constituent order constraints than Spanish.

Tag	Meaning
ATR	Attribute of copular verb
CAG	Agent of passive verb
CC	Compl. of circumstance
CD	Direct object
CD.Q	Direct object of quantity
CI	Indirect object
CPRED	Predicative complement
CPRED.CD	Predicative of Direct Object
CPRED.SUJ	Predicative of Subject
CREG	Prepositional object
ET	Textual element
IMPERS	Impersonal marker
MOD	Verbal modifier
NEG	Negation
PASS	Passive marker
SUJ	Subject
VOC	Vocative

Table 1: List of function tags in Cast3LB.

3 Previous Work

3.1 LFG Annotation

A methodology for automatically obtaining LFG f-structures from trees output by probabilistic parsers trained on the Penn-II treebank has been described by Cahill et al. (2004). It has been shown that the methods can be ported to other languages and treebanks (Burke et al., 2004; Cahill et al., 2003), including Cast3LB (O'Donovan et al., 2005).

Some properties of Spanish and the encoding of syntactic information in the Cast3LB treebank make it non-trivial to apply the method of automatically mapping c-structures to f-structures used by Cahill et al. (2004), which assigns grammatical functions to tree nodes based on their phrasal category, the category of the mother node and their position relative to the local head.

In Spanish, the order of sentence constituents is flexible and their position relative to the head is an imperfect predictor of grammatical function. Also, much of the information that the Penn-II Treebank encodes in terms of tree configurations is encoded in Cast3LB in the form of function tags. As Cast3LB trees lack a VP node, the configurational information normally used in English to distinguish Subjects (NP which is left sister to VP) from Direct Objects (NP which is right sister to V) is not available in Cast3LB-style trees. This means that assigning correct LFG functional annotations to nodes in Cast3LB trees is rather difficult without use of Cast3LB function tags, and those tags are typically absent in output generated by probabilistic parsers.

In order to solve this difficulty, O'Donovan et al. (2005) train Bikel's parser to output complex category-function labels. A complex label such as *sn-SUJ* (an NP node tagged with the Subject grammatical function) is treated as an atomic category in the training data, and is output in the trees produced by the parser. This baseline process is represented in Figure 2.

This approach can be problematic for two main reasons. Firstly, by treating complex labels as atomic categories the number of unique labels increases and parse quality can deteriorate due to sparse data problems. Secondly, this approach, by relying on the parser to assign function tags, offers

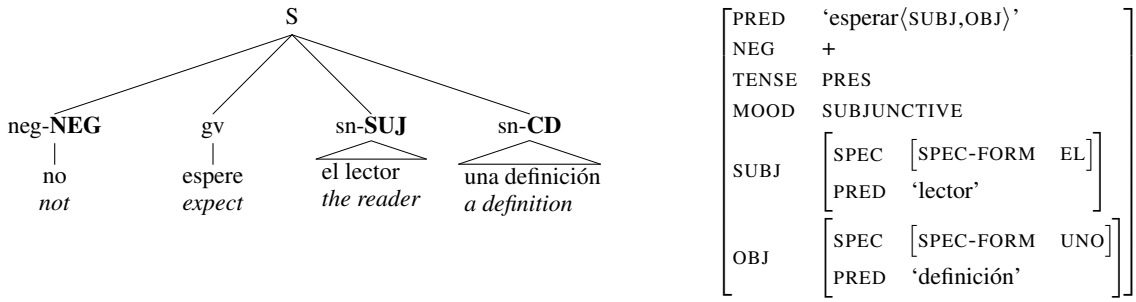


Figure 1: On the left flat structure of S. Cast3LB function tags are shown in bold. On the right the corresponding (simplified) LFG f-structure. Translation: *Let the reader not expect a definition*.

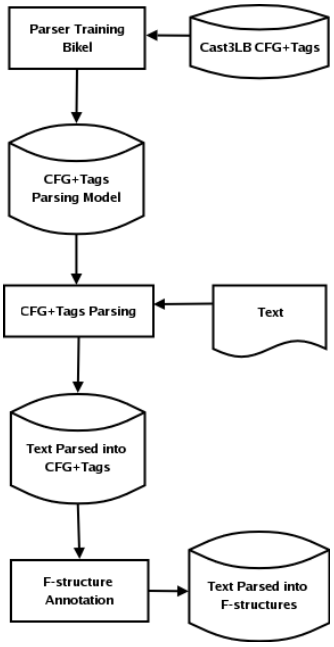


Figure 2: Processing architecture for the baseline.

limited control over, or room for improvement in, this task.

3.2 Adding Function Tags to Parser Output

The solution we adopt instead is to add Cast3LB functional tags to simple constituent trees output by the parser, as a postprocessing step. For English, such approaches have been shown to give good results for the output of parsers trained on the Penn-II Treebank.

Blaheta and Charniak (2000) use a probabilistic model with feature dependencies encoded by means of feature trees to add Penn-II Treebank function tags to Charniak's parser output. They report an f-score 88.472% on original treebank trees and 87.277% on the correctly parsed subset of tree nodes.

Jijkoun and de Rijke (2004) describe a method of enriching output of a parser with information that is included in the original Penn-II trees, such as function tags, empty nodes and coindexations. They first transform Penn trees to a dependency format and then use memory-based learning to perform various graph transformations. One of the transformations is node relabelling, which adds function tags to parser output. They report an f-score of 88.5% for the task of function tagging on correctly parsed constituents.

4 Assigning Cast3LB Function Tags to Parsed Spanish Text

The complete processing architecture of our approach is depicted in Figure 3. We describe it in detail in this and the following sections.

We divided the Spanish treebank into a training set of 80%, a development set of 10%, and a test set of 10% of all trees. We randomly assigned treebank files to these sets to ensure that different textual genres are about equally represented among the training, development and test trees.

4.1 Constituency Parsing

For constituency parsing we use Bikel's (2002) parser for which we developed a Spanish language package adapted to the Cast3LB data. Prior to parsing, we perform one of the tree transformations described by Cowan and Collins (2005), i.e. we add a CP and SBAR nodes to subordinate and relative clauses. This is undone in parser output.

The category labels in the Spanish treebank are rather fine grained and often contain redundant information.[1] We preprocess the treebank and re-

[1]For example there are several labels for Nominal Group,

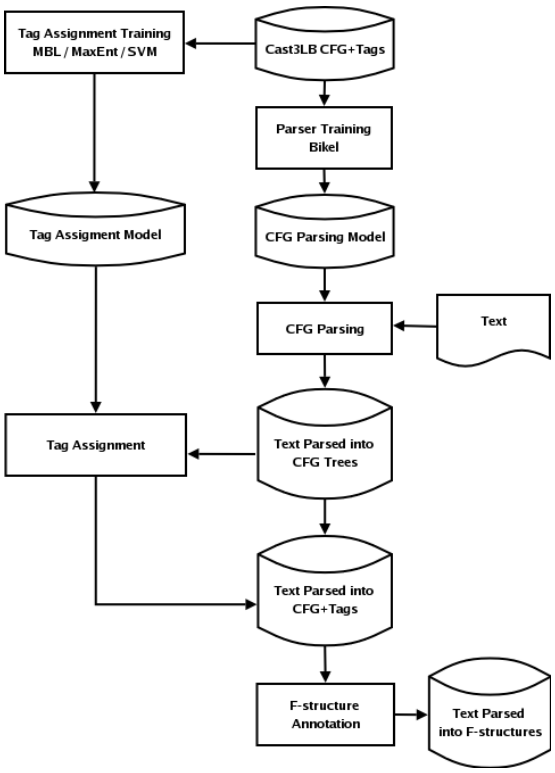


Figure 3: Processing architecture for the machine-learning-based method.

duce the number of category labels, only retaining distinctions that we deem useful for our purposes.[2]

For constituency parsing we also reduce the number of POS tags by including only selected morphological features. Table 2 provides the list of features included for the different parts of speech. In our experiments we use gold standard POS tagged development and test-set sentences as input rather than tagging text automatically.

The results of the evaluation of parsing performance on the test set are shown in Table 3. Labelled bracketing f-score for all sentences is just below 84% for all sentences, and 84.58% for sentences of length ≤ 70. In comparison, Cowan and Collins (2005) report an f-score of 85.1% (≤ 70) using a version of Collins' parser adapted for Cast3LB, and using reranking to boost performance. They use a different, more reduced category label set as well as a different training-test split. Both Cowan and Collins and the present paper report scores which ignore punctuation.

such as *grup.nom.ms* (masculine singular), *grup.nom.fs* (feminine singular), *grup.nom.mp* (masculine plural) etc. This number and gender information is already encoded in the POS tags of nouns heading these constituents.

[2]The labels we retain are the following: *INC, S, S.NF, S.NF.R, S.NF, S.R, conj.subord, coord, data, espec, gerundi, grup.nom, gv, infinitiu, interjeccio, morf, neg, numero, prep, relatiu, s.a, sa, sadv, sn, sp*, and versions of those suffixed with *.co* to indicate coordination).

Part of Speech	Features included
Determiner	type, number
Noun	type, number
Adjective	type, number
Pronoun	type, number, person
Verb	type, number, mood
Adverb	type
Conjunction	type

Table 2: Features included in POS tags. Type refers to subcategories of parts of speech such as e.g. *common* and *proper* for nouns, or *main*, *auxiliary* and *semiauxiliary* for verbs. For details see (Civit, 2000).

	LB Precision	LB Recall	F-score
All	84.18	83.74	83.96
≤ 70	84.82	84.35	84.58

Table 3: Parser performance.

4.2 Cast3LB Function Tagging

For the task of Cast3LB function tag assignment we experimented with three generic machine learning algorithms: a memory-based learner (Daelemans and van den Bosch, 2005), a maximum entropy classifier (Berger et al., 1996) and a Support Vector Machine classifier (Vapnik, 1998). For each algorithm we use the same set of features to represent nodes that are to be assigned one of the Cast3LB function tags. We use a special null tag for nodes where no Cast3LB tag is present.

In Cast3LB only nodes in certain contexts are eligible for function tags. For this reason we only consider a subset of all nodes as candidates for function tag assignment, namely those which are sisters of nodes with the category labels *gv* (Verb Group), *infinitiu* (Infinitive) and *gerundi* (Gerund). For these candidates we extract the following three types of features encoding configurational, morphological and lexical information for the target node and neighboring context nodes:

- Node features: position relative to head, head lemma, alternative head lemma (i.e. the head of NP in PP), head POS, category, definiteness, agreement with head verb, yield, human/nonhuman

- Local features: head verb, verb person, verb number, parent category
- Context features: node features (except position) of the two previous and two following sister nodes (if present).

We used cross-validation for refining the set of features and for tuning the parameters of the machine-learning algorithms. We did not use any additional automated feature-selection procedure. We made use of the following implementations: TiMBL (Daelemans et al., 2004) for Memory-Based Learning, the MaxEnt Toolkit (Le, 2004) for Maximum Entropy and LIBSVM (Chang and Lin, 2001) for Support Vector Machines. For TiMBL we used k nearest neighbors $= 7$ and the gain ratio metric for feature weighting. For MaxEnt, we used the L-BFGS parameter estimation and 110 iterations, and we regularize the model using a Gaussian prior with $\sigma^2 = 1$. For SVM we used the RBF kernel with $\gamma = 2^{-7}$ and the cost parameter $C = 32$.

5 Cast3LB Tag Assignment Evaluation

We present evaluation results on the original gold-standard trees of the test set as well as on the test-set sentences parsed by Bikel's parser. For the evaluation of Cast3LB function tagging performance on gold trees the most straightforward metric is the accuracy, or the proportion of all candidate nodes that were assigned the correct label.

However we cannot use this metric for evaluating results on the parser output. The trees output by the parser are not identical to gold standard trees due to parsing errors, and the set of candidate nodes extracted from parsed trees will not be the same as for gold trees. For this reason we use an alternative metric which is independent of tree configuration and uses only the Cast3LB function labels and positional indices of tokens in a sentence. For each function-tagged tree we first remove the punctuation tokens. Then we extract a set of tuples of the form $\langle GF, i, j \rangle$, where GF is the Cast3LB function tag and $i - j$ is the range of tokens spanned by the node annotated with this function. We use the standard measures of precision, recall and f-score to evaluate the results.

Results for the three algorithms are shown in Table 4. MBL and MaxEnt show a very similar performance, while SVM outperforms both, scoring 89.34% on accuracy and 86.87% on f-score. The learning curves for the three algorithms, shown in Figure 4, are also informative, with SVM outperforming the other two methods for all training set sizes. In particular, the last section of the plot shows SVM performing almost as well as MBL with half as much learning material.

Neither of the three curves shows signs of having reached a maximum, which indicates that in-

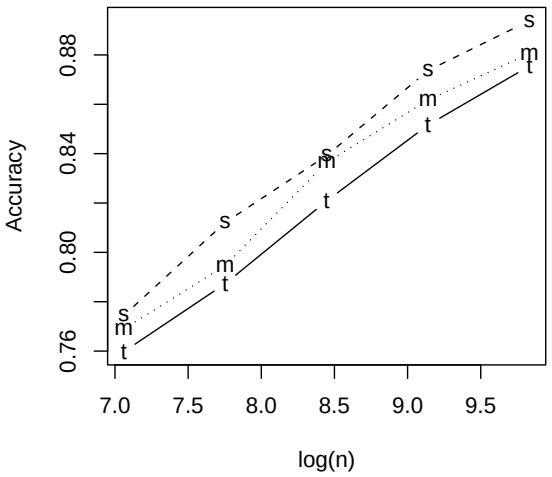


Figure 4: Learning curves for TiMBL (t), MaxEnt (m) and SVM (s).

	Acc.	Prec.	Recall	F-score
MBL	87.55	87.00	82.98	84.94
MaxEnt	88.06	87.66	86.87	85.52
SVM	**89.34**	88.93	84.90	**86.87**

Table 4: Cast3LB function tagging performance for gold-standard trees

	Precision		Recall		F-score	
	all	corr.	all	corr.	all	corr.
Baseline	59.26	72.63	60.61	75.35	59.93	73.96
MBL	64.74	78.09	64.18	78.75	64.46	78.42
MaxEnt	65.48	78.90	64.55	79.44	65.01	79.17
SVM	66.96	80.58	66.38	81.27	**66.67**	**80.92**

Table 5: Cast3LB function tagging performance for parser output, for all constituents, and for correctly parsed constituents only

Methods	p-value
Baseline vs SVM	1.169×10^{-9}
Baseline vs MBL	2.117×10^{-6}
MBL vs MaxEnt	0.0799
MaxEnt vs SVM	0.0005

Table 6: Statistical significance testing results on for the Cast3LB tag assignment on parser output.

	Precision	Recall	F-score
Baseline	73.95	70.67	72.27
SVM	76.90	74.48	75.67

Table 7: LFG F-structure evaluation results for parser output

creasing the size of the training data should result in further improvements in performance.

Table 5 shows the performance of the three methods on parser output. The baseline contains the results achieved by treating compound category-function labels as atomic during parser training so that they are included in parser output. For this task we present two sets of results: (i) for all constituents, and (ii) for correctly parsed constituents only. Again the best algorithm turns out to be SVM. It outperforms the baseline by a large margin (6.74% for all constituents).

The difference in performance for gold standard trees, and the correctly parsed constituents in parser output is rather larger than what Blaheta and Charniak report. Further analysis is needed to identify the source of this difference but we suspect that one contributing factor is the use of greater number of context features combined with a higher parse error rate in comparison to their experiments on the Penn II Treebank. Since any misanalysis of constituency structure in the vicinity of target node can have negative impact, greater reliance on context means greater susceptibility to parse errors. Another factor to consider is the fact that we trained and adjusted parameters on gold-standard trees, and the model learned may rely on features of those trees that the parser is unable to reproduce.

For the experiments on parser output (all constituents) we performed a series of sign tests in order to determine to what extent the differences in performance between the different methods are statistically significant. For each pair of methods we calculate the f-score for each sentence in the test set. For those sentences on which the scores differ (i.e. the number of trials) we calculate in how many cases the second method is better than the first (i.e. the number of successes). We then perform the test with the null hypothesis that the probability of success is chance ($= 0.5$) and the alternative hypothesis that the probability of success is greater than chance (> 0.5). The results are summarized in Table 6. Given that we perform 4 pairwise comparisons, we apply the Bonferroni correction and adjust our target $\alpha_\beta = \frac{\alpha}{4}$. For the confidence level 95% ($\alpha_\beta = 0.0125$) all pairs give statistically significant results, except for MBL vs MaxEnt.

6 Task-Based LFG Annotation Evaluation

Finally, we also evaluated the actual f-structures obtained by running the LFG-annotation algorithm on trees produced by the parser and enriched with Cast3LB function tags assigned using SVM. For this task-based evaluation we produced a gold standard consisting of f-structures corresponding to all sentences in the test set. The LFG-annotation algorithm was run on the test set trees (which contained original Cast3LB treebank function tags), and the resulting f-structures were manually corrected.

Following Crouch et al. (2002), we convert the f-structures to triples of the form $\langle GF, P_i, P_j \rangle$, where P_i is the value of the PRED attribute of the f-structure, GF is an LFG grammatical function attribute, and P_j is the value of the PRED attribute of the f-structure which is the value of the GF attribute. This is done recursively for each level of embedding in the f-structure. Attributes with atomic values are ignored for the purposes of this evaluation. The results obtained are shown in Table 7. We also performed a statistical significance test for these results, using the same method as for the Cast3LB tag assigment task. The p-value given by the sign test was 2.118×10^{-5}, comfortably below $\alpha = 1\%$.

The higher scores achieved in the LFG f-structure evaluation in comparison with the preceding Cast3LB tag assignment evaluation (Table 5) can be attributed to two main factors. Firstly, the mapping from Cast3LB tags to LFG grammatical functions is not one-to-one. For example three Cast3LB tags (CC, MOD and ET) are all mapped to LFG ADJUNCT. Thus mistagging a MOD as

	ATR	CC	CD	CI	CREG	MOD	SUJ
ATR	136	2	0	0	0	0	5
CC	6	552	12	4	25	18	6
CD	1	19	418	5	3	0	26
CI	0	6	1	50	1	0	0
CREG	0	6	0	2	43	0	0
MOD	0	0	0	0	0	19	0
SUJ	0	8	24	2	0	0	465

Table 8: Simplified confusion matrix for SVM on test-set gold-standard trees. The gold-standard Cast3LB function tags are shown in the first row, the predicted tags in the first column. So e.g. SUJ was mistagged as CD in 26 cases. Low frequency function tags as well as those rarely mispredicted have been omitted for clarity.

CC does not affect the f-structure score. On the other hand the Cast3LB CD tag can be mapped to OBJ, COMP, or XCOMP, and it can be easily decided which one is appropriate depending on the category label of the target node. Additionally many nodes which receive no function tag in Cast3LB, such as noun modifiers, are straightforwardly mapped to LFG ADJUNCT. Similarly, objects of prepositions receive the LFG OBJ function.

Secondly, the f-structure evaluation metric is less sensitive to small constituency misconfigurations: it is not necessary to correctly identify the token range spanned by a target node as long as the head (which provides the PRED attribute) is correct.

7 Error Analysis

In order to understand sources of error and determine how much room for further improvement there is, we examined the most common cases of Cast3LB function mistagging. A simplified confusion matrix with the most common Cast3LB tags is shown in Table 8. The most common mistakes occur between SUJ and CD, in both directions, and many also CREGs are erroneously tagged as CC.

7.1 Subject vs Direct Object

We noticed that in over 50% of cases when a Direct Object (CD) was misidentified as Subject (SUJ), the target node's mother was a relative clause. It turns out that in Spanish relative clauses genuine syntactic ambiguity is not uncommon. Consider the following Spanish phrase:

(1) *Sistemas que usan el 95% de*
Systems which use DET 95% of
los ordenadores.
DET computers

Its translation into English is either *Systems that use 95% of computers* or alternatively *Systems that 95% of computers use*. In Spanish, unlike in English, preverbal / postverbal position of a constituent is not a good guide to its grammatical function in this and similar contexts. Human annotators can use their world knowledge to decide on the correct semantic role of a target constituent and use it in assigning a correct grammatical function, but such information is obviously not used in our machine learning methods. Thus such mistakes seem likely to remain unresolvable in our current approach.

7.2 Prepositional Object vs Adjunct

The frequent misidentification of Prepositional Objects (CREG) as Adjuncts (CC) seen in Table 8 can be accounted for by several factors. Firstly, Prepositional Objects are strongly dependent on specific verbs and the comparatively small size of our training data means that there is limited opportunity for a machine-learning algorithm to learn low-frequency lexical dependencies. Here the obvious solution is to use a more adequate amount of training material when it becomes available.

A further problem with the Prepositional Object - Adjunct distinction is its inherent fuzziness. Because of this, treebank designers may fail to provide easy-to-follow, clearcut guidelines and human annotators necessarily exercise a certain degree of arbitrariness in assigning one or the other function.

8 Conclusions and Future Research

Our research has shown that machine-learning-based Cast3LB tag assignment as a post-processing step to raw tree parser output statistically significantly outperforms a baseline where the parser itself is trained to learn category / Cast3LB-function pairs. In contrast to the parser-based method, the machine-learning-based method avoids some sparse data problems and allows for more control over Cast3LB tag assignment. We have found that the SVM algorithm outperforms the other two machine learning methods used.

In addition, we evaluated Cast3LB tag assignment in a task-based setting in the context of automatically acquiring LFG resources for Spanish from Cast3LB. Machine-learning-based Cast3LB tag assignment yields statistically-significantly improved LFG f-structures compared to parser-based assignment.

One limitation of our method is the fact that it treats the classification task separately for each target node. It thus fails to observe constraints on the possible sequences of grammatical function tags in the same local context. Some functions are unique, such as the Subject, whereas others (Direct and Indirect Object) can only be realized by a full NP once, although they can be doubled by a clitic pronoun. Capturing such global constraints will need further work.

Acknowledgements

We gratefully acknowledge support from Science Foundation Ireland grant 04/IN/I527 for the research reported in this paper.

References

A. L. Berger, V. J. Della Pietra, and S. A. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, March.

D. Bikel. 2002. Design of a multi-lingual, parallel-processing statistical parsing engine. In *Human Language Technology Conference (HLT)*, San Diego, CA, USA. Software available at `http://www.cis.upenn.edu/~dbikel/software.html#stat-parser`.

D. Blaheta and E. Charniak. 2000. Assigning function tags to parsed text. In *Proceedings of the 1st Conference of the North American Chapter of the ACL*, pages 234–240, Rochester, NY, USA.

J. Bresnan. 2001. *Lexical-Functional Syntax*. Blackwell Publishers, Oxford.

M. Burke, O. Lam, A. Cahill, R. Chan, R. O'Donovan, A. Bodomo, J. van Genabith, and A. Way. 2004. Treebank-based acquisition of a Chinese Lexical-Functional Grammar. In *Proceedings of the 18th Pacific Asia Conference on Language, Information and Computation (PACLIC-18)*.

A. Cahill, M. Forst, M. McCarthy, R. O'Donovan, and C. Roher. 2003. Treebank-based multilingual unification-grammar development. In *Proceedings of the 15th Workshop on Ideas and Strategies for Multilingual Grammar Development, ESSLLI 15*, Vienna, Austria.

A. Cahill, M. Burke, R. O'Donovan, J. van Genabith, and A. Way. 2004. Long-distance dependency resolution in automatically acquired wide-coverage PCFG-based LFG approximations. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 319–326, Barcelona, Spain.

Chih-Chung Chang and Chih-Jen Lin, 2001. *LIBSVM: a library for support vector machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

M. Civit and M. A. Martí. 2004. Building Cast3LB: A Spanish treebank. *Research on Language and Computation*, 2(4):549–574, December.

M. Civit. 2000. Guía para la anotación morfosintáctica del corpus CLiC-TALP, X-TRACT Working Paper. Technical report. Available at `http://clic.fil.ub.es/personal/civit/PUBLICA/guia_morfol.ps`.

M. Civit. 2004. Guía para la anotación de las funciones sintácticas de Cast3LB. Technical report. Available at `http://clic.fil.ub.es/personal/civit/PUBLICA/funcions.pdf`.

B. Cowan and M. Collins. 2005. Morphology and reranking for the statistical parsing of Spanish. In *Conference on Empirical Methods in Natural Language Processing*, Vancouver, B.C., Canada.

R. Crouch, R. M. Kaplan, T. H. King, and S. Riezler. 2002. A comparison of evaluation metrics for a broad-coverage stochastic parser. In *Conference on Language Resources and Evaluation (LREC 02)*.

W. Daelemans and A. van den Bosch. 2005. *Memory-Based Language Processing*. Cambridge University Press, September.

W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch. 2004. TiMBL: Tilburg Memory Based Learner, version 5.1, Reference Guide. Technical report. Available from `http://ilk.uvt.nl/downloads/pub/papers/ilk0402.pdf`.

V. Jijkoun and M. de Rijke. 2004. Enriching the output of a parser using memory-based learning. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Barcelona, Spain.

Zh. Le, 2004. *Maximum Entropy Modeling Toolkit for Python and C++*. Available at `http://homepages.inf.ed.ac.uk/s0450736/software/maxent/manual.pdf`.

R. O'Donovan, A. Cahill, J. van Genabith, and A. Way. 2005. Automatic acquisition of Spanish LFG resources from the CAST3LB treebank. In *Proceedings of the Tenth International Conference on LFG*, Bergen, Norway.

V. N. Vapnik. 1998. *Statistical Learning Theory*. Wiley-Interscience, September.

Constraint-based Sentence Compression
An Integer Programming Approach

James Clarke and **Mirella Lapata**
School of Informatics, University of Edinburgh
2 Bucclecuch Place, Edinburgh EH8 9LW, UK
jclarke@ed.ac.uk, mlap@inf.ed.ac.uk

Abstract

The ability to compress sentences while preserving their grammaticality and most of their meaning has recently received much attention. Our work views sentence compression as an optimisation problem. We develop an integer programming formulation and infer globally optimal compressions in the face of linguistically motivated constraints. We show that such a formulation allows for relatively simple and knowledge-lean compression models that do not require parallel corpora or large-scale resources. The proposed approach yields results comparable and in some cases superior to state-of-the-art.

1 Introduction

A mechanism for automatically compressing sentences while preserving their grammaticality and most important information would greatly benefit a wide range of applications. Examples include text summarisation (Jing 2000), subtitle generation from spoken transcripts (Vandeghinste and Pan 2004) and information retrieval (Olivers and Dolan 1999). Sentence compression is a complex paraphrasing task with information loss involving substitution, deletion, insertion, and reordering operations. Recent years have witnessed increased interest on a simpler instantiation of the compression problem, namely word deletion (Knight and Marcu 2002; Riezler et al. 2003; Turner and Charniak 2005). More formally, given an input sentence of words $W = w_1, w_2, \ldots, w_n$, a compression is formed by removing any subset of these words.

Sentence compression has received both generative and discriminative formulations in the literature. Generative approaches (Knight and Marcu 2002; Turner and Charniak 2005) are instantiations of the noisy-channel model: given a long sentence l, the aim is to find the corresponding short sentence s which maximises the conditional probability $P(s|l)$. In a discriminative setting (Knight and Marcu 2002; Riezler et al. 2003; McDonald 2006), sentences are represented by a rich feature space (typically induced from parse trees) and the goal is to learn rewrite rules indicating which words should be deleted in a given context. Both modelling paradigms assume access to a training corpus consisting of original sentences and their compressions.

Unsupervised approaches to the compression problem are few and far between (see Hori and Furui 2004 and Turner and Charniak 2005 for exceptions). This is surprising considering that parallel corpora of original-compressed sentences are not naturally available in the way multilingual corpora are. The scarcity of such data is demonstrated by the fact that most work to date has focused on a single parallel corpus, namely the Ziff-Davis corpus (Knight and Marcu 2002). And some effort into developing appropriate training data would be necessary when porting existing algorithms to new languages or domains.

In this paper we present an unsupervised model of sentence compression that does not rely on a parallel corpus – all that is required is a corpus of uncompressed sentences and a parser. Given a long sentence, our task is to form a compression by preserving the words that maximise a scoring function. In our case, the scoring function is an n-gram language model, "with a few strings attached". While straightforward to estimate, a language model is a fairly primitive scoring function: it has no notion of the overall sentence structure, grammaticality or underlying meaning. We thus couple our language model with a small number of structural and semantic constraints capturing global properties of the compression process.

We encode the language model and linguistic constraints as linear inequalities and use Integer Programming (IP) to infer compressions that are consistent with both. The IP formulation allows us to capture global sentence properties and can be easily manipulated to provide compressions tailored for specific applications. For example, we

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 144–151,
Sydney, July 2006.

could prevent overly long or overly short compressions or generally avoid compressions that lack a main verb or consist of repetitions of the same word.

In the following section we provide an overview of previous approaches to sentence compression. In Section 3 we motivate the treatment of sentence compression as an optimisation problem and formulate our language model and constraints in the IP framework. Section 4 discusses our experimental set-up and Section 5 presents our results. Discussion of future work concludes the paper.

2 Previous Work

Jing (2000) was perhaps the first to tackle the sentence compression problem. Her approach uses multiple knowledge sources to determine which phrases in a sentence to remove. Central to her system is a grammar checking module that specifies which sentential constituents are grammatically obligatory and should therefore be present in the compression. This is achieved using simple rules and a large-scale lexicon. Other knowledge sources include WordNet and corpus evidence gathered from a parallel corpus of original-compressed sentence pairs. A phrase is removed only if it is not grammatically obligatory, not the focus of the local context and has a reasonable deletion probability (estimated from the parallel corpus).

In contrast to Jing (2000), the bulk of the research on sentence compression relies exclusively on corpus data for modelling the compression process without recourse to extensive knowledge sources (e.g., WordNet). Approaches based on the noisy-channel model (Knight and Marcu 2002; Turner and Charniak 2005) consist of a source model $P(s)$ (whose role is to guarantee that the generated compression is grammatical), a channel model $P(l|s)$ (capturing the probability that the long sentence l is an expansion of the compressed sentence s), and a decoder (which searches for the compression s that maximises $P(s)P(l|s)$). The channel model is typically estimated using a parallel corpus, although Turner and Charniak (2005) also present semi-supervised and unsupervised variants of the channel model that estimate $P(l|s)$ without parallel data.

Discriminative formulations of the compression task include decision-tree learning (Knight and Marcu 2002), maximum entropy (Riezler et al. 2003), support vector machines (Nguyen et al. 2004), and large-margin learning (McDonald 2006). We describe here the decision-tree model in more detail since we will use it as a basis for comparison when evaluating our own models (see Section 4). According to this model, compression is performed through a tree rewriting process inspired by the shift-reduce parsing paradigm. A sequence of shift-reduce-drop actions are performed on a long parse tree, l, to create a smaller tree, s.

The compression process begins with an input list generated from the leaves of the original sentence's parse tree and an empty stack. 'Shift' operations move leaves from the input list to the stack while 'drop' operations delete from the input list. Reduce operations are used to build trees from the leaves on the stack. A decision-tree is trained on a set of automatically generated learning cases from a parallel corpus. Each learning case has a target action associated with it and is decomposed into a set of indicative features. The decision-tree learns which action to perform given this set of features. The final model is applied in a deterministic fashion in which the features for the current state are extracted and the decision-tree is queried. This is repeated until the input list is empty and the final compression is recovered by traversing the leaves of resulting tree on the stack.

While most compression models operate over constituents, Hori and Furui (2004) propose a model which generates compressions through word deletion. The model does not utilise parallel data or syntactic information in any form. Given a prespecified compression rate, it searches for the compression with the highest score according to a function measuring the importance of each word and the linguistic likelihood of the resulting compressions (language model probability). The score is maximised through a dynamic programming algorithm.

Although sentence compression has not been explicitly formulated as an optimisation problem, previous approaches have treated it in these terms. The decoding process in the noisy-channel model searches for the best compression given the source and channel models. However, the compression found is usually sub-optimal as heuristics are used to reduce the search space or is only locally optimal due to the search method employed. The decoding process used in Turner and Charniak's (2005) model first searches for the best combination of rules to apply. As they traverse their list of compression rules they remove sentences outside the 100 best compressions (according to their channel model). This list is eventually truncated to 25 compressions.

In other models (Hori and Furui 2004; McDonald 2006) the compression score is maximised

using dynamic programming. The latter guarantees we will find the global optimum provided the principle of optimality holds. This principle states that given the current state, the optimal decision for each of the remaining stages does not depend on previously reached stages or previously made decisions (Winston and Venkataramanan 2003). However, we know this to be false in the case of sentence compression. For example, if we have included modifiers to the left of a head noun in the compression then it makes sense that we must include the head also. With a dynamic programming approach we cannot easily guarantee such constraints hold.

3 Problem Formulation

Our work models sentence compression explicitly as an optimisation problem. There are 2^n possible compressions for each sentence and while many of these will be unreasonable (Knight and Marcu 2002), it is unlikely that only one compression will be satisfactory. Ideally, we require a function that captures the operations (or rules) that can be performed on a sentence to create a compression while at the same time factoring how desirable each operation makes the resulting compression. We can then perform a search over *all* possible compressions and select the best one, as determined by how desirable it is.

Our formulation consists of two basic components: a language model (scoring function) and a small number of constraints ensuring that the resulting compressions are structurally and semantically valid. Our task is to find a globally optimal compression in the presence of these constraints. We solve this inference problem using Integer Programming without resorting to heuristics or approximations during the decoding process. Integer programming has been recently applied to several classification tasks, including relation extraction (Roth and Yih 2004), semantic role labelling (Punyakanok et al. 2004), and the generation of route directions (Marciniak and Strube 2005).

Before describing our model in detail, we introduce some of the concepts and terms used in Linear Programming and Integer Programming (see Winston and Venkataramanan 2003 for an introduction). Linear Programming (LP) is a tool for solving optimisation problems in which the aim is to maximise (or minimise) a given function with respect to a set of *constraints*. The function to be maximised (or minimised) is referred to as the *objective function*. Both the objective function and constraints must be linear. A number of *decision variables* are under our control which exert influence on the objective function. Specifically, they have to be optimised in order to maximise (or minimise) the objective function. Finally, a set of constraints restrict the values that the decision variables can take. Integer Programming is an extension of linear programming where all decision variables must take integer values.

3.1 Language Model

Assume we have a sentence $W = w_1, w_2, \ldots, w_n$ for which we wish to generate a compression. We introduce a decision variable for each word in the original sentence and constrain it to be binary; a value of 0 represents a word being dropped, whereas a value of 1 includes the word in the compression. Let:

$$y_i = \begin{cases} 1 & \text{if } w_i \text{ is in the compression} \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in [1 \ldots n]$$

If we were using a unigram language model, our objective function would maximise the overall sum of the decision variables (i.e., words) multiplied by their unigram probabilities (all probabilities throughout this paper are log-transformed):

$$\max z = \sum_{i=1}^{n} y_i \cdot P(w_i)$$

Thus if a word is selected, its corresponding y_i is given a value of 1, and its probability $P(w_i)$ according to the language model will be counted in our total score, z.

A unigram language model will probably generate many ungrammatical compressions. We therefore use a more context-aware model in our objective function, namely a trigram model. Formulating a trigram model in terms of an integer program becomes a more involved task since we now must make decisions based on word sequences rather than isolated words. We first create some extra decision variables:

$$p_i = \begin{cases} 1 \text{ if } w_i \text{ starts the compression} \\ 0 \text{ otherwise} \end{cases} \forall i \in [1 \ldots n]$$

$$q_{ij} = \begin{cases} 1 \text{ if sequence } w_i, w_j \text{ ends} \\ \quad \text{the compression} & \forall i \in [1 \ldots n-1] \\ 0 \text{ otherwise} & \forall j \in [i+1 \ldots n] \end{cases}$$

$$x_{ijk} = \begin{cases} 1 \text{ if sequence } w_i, w_j, w_k & \forall i \in [1 \ldots n-2] \\ \quad \text{is in the compression} & \forall j \in [i+1 \ldots n-1] \\ 0 \text{ otherwise} & \forall k \in [j+1 \ldots n] \end{cases}$$

Our objective function is given in Equation (1). This is the sum of all possible trigrams that can occur in all compressions of the original sentence where w_0 represents the 'start' token and w_i is the ith word in sentence W. Equation (2) constrains

the decision variables to be binary.

$$\begin{aligned}\max z \;=\; & \sum_{i=1}^{n} p_i \cdot P(w_i|\text{start}) \\ & + \sum_{i=1}^{n-2}\sum_{j=i+1}^{n-1}\sum_{k=j+1}^{n} x_{ijk} \cdot P(w_k|w_i,w_j) \\ & + \sum_{i=0}^{n-1}\sum_{j=i+1}^{n} q_{ij} \cdot P(\text{end}|w_i,w_j) \quad (1)\end{aligned}$$

subject to:

$$y_i, p_i, q_{ij}, x_{ijk} = 0 \text{ or } 1 \quad (2)$$

The objective function in (1) allows any combination of trigrams to be selected. This means that invalid trigram sequences (e.g., two or more trigrams containing the symbol 'end') could appear in the output compression. We avoid this situation by introducing *sequential constraints* (on the decision variables y_i, x_{ijk}, p_i, and q_{ij}) that restrict the set of allowable trigram combinations.

Constraint 1 Exactly one word can begin a sentence.

$$\sum_{i=1}^{n} p_i = 1 \quad (3)$$

Constraint 2 If a word is included in the sentence it must either start the sentence or be preceded by two other words or one other word and the 'start' token w_0.

$$y_k - p_k - \sum_{i=0}^{k-2}\sum_{j=1}^{k-1} x_{ijk} = 0 \quad (4)$$
$$\forall k : k \in [1 \ldots n]$$

Constraint 3 If a word is included in the sentence it must either be preceded by one word and followed by another or it must be preceded by one word and end the sentence.

$$y_j - \sum_{i=0}^{j-1}\sum_{k=j+1}^{n} x_{ijk} - \sum_{i=0}^{j-1} q_{ij} = 0 \quad (5)$$
$$\forall j : j \in [1 \ldots n]$$

Constraint 4 If a word is in the sentence it must be followed by two words or followed by one word and then the end of the sentence or it must be preceded by one word and end the sentence.

$$y_i - \sum_{j=i+1}^{n-1}\sum_{k=j+1}^{n} x_{ijk} - \sum_{j=i+1}^{n} q_{ij} - \sum_{h=0}^{i-1} q_{hi} = 0 \quad (6)$$
$$\forall i : i \in [1 \ldots n]$$

Constraint 5 Exactly one word pair can end the sentence.

$$\sum_{i=0}^{n-1}\sum_{j=i+1}^{n} q_{ij} = 1 \quad (7)$$

Example compressions using the trigram model just described are given in Table 1. The model in its current state does a reasonable job of modelling local word dependencies, but is unable to capture syntactic dependencies that could potentially allow more meaningful compressions. For example, it does not know that *Pasok Party* is the object of *founded* or that *Appleshare* modifies *Printer Server*.

O:	He became a power player in Greek Politics in 1974, when he founded the socialist Pasok Party.
LM:	He became a player in the Pasok.
Mod:	He became a player in the Pasok Party.
Sen:	He became a player in politics.
Sig:	He became a player in politics when he founded the Pasok Party.
O:	Finally, AppleShare Printer Server, formerly a separate package, is now bundled with AppleShare File Server.
LM:	Finally, AppleShare, a separate, AppleShare.
Mod:	Finally, AppleShare Server, is bundled.
Sen:	Finally, AppleShare Server, is bundled with Server.
Sig:	AppleShare Printer Server package is now bundled with AppleShare File Server.

Table 1: Compression examples (O: original sentence, LM: compression with the trigram model, Mod: compression with LM and modifier constraints, Sen: compression with LM, Mod and sentential constraints, Sig: compression with LM, Mod, Sen, and significance score)

3.2 Linguistic Constraints

In this section we propose a set of global constraints that extend the basic language model presented in Equations (1)–(7). Our aim is to bring some syntactic knowledge into the compression model and to preserve the meaning of the original sentence as much as possible. Our constraints are linguistically and semantically motivated in a similar fashion to the grammar checking component of Jing (2000). Importantly, we do not require any additional knowledge sources (such as a lexicon) beyond the parse and grammatical relations of the original sentence. This is provided in our experiments by the Robust Accurate Statistical Parsing (RASP) toolkit (Briscoe and Carroll 2002). However, there is nothing inherent in our formulation that restricts us to RASP; any other parser with similar output could serve our purposes.

Modifier Constraints Modifier constraints ensure that relationships between head words and their modifiers remain grammatical in the compression:

$$y_i - y_j \geq 0 \quad (8)$$
$$\forall i,j : w_j \in w_i\text{'s } \texttt{ncmods}$$
$$y_i - y_j \geq 0 \quad (9)$$
$$\forall i,j : w_j \in w_i\text{'s } \texttt{detmods}$$

Equation (8) guarantees that if we include a non-clausal modifier (ncmod) in the compression then the head of the modifier must also be included; this is repeated for determiners (detmod) in (9).

We also want to ensure that the meaning of the original sentence is preserved in the compression, particularly in the face of negation. Equation (10) implements this by forcing *not* in the compression when the head is included. A similar constraint is added for possessive modifiers (e.g., *his*, *our*), as shown in Equation (11). Genitives (e.g., *John's gift*) are treated separately, mainly because they are encoded as different relations in the parser (see Equation (12)).

$$y_i - y_j = 0 \quad (10)$$
$$\forall i, j : w_j \in w_i\text{'s ncmods} \wedge w_j = \text{not}$$
$$y_i - y_j = 0 \quad (11)$$
$$\forall i, j : w_j \in w_i\text{'s possessive detmods}$$
$$y_i - y_j = 0 \quad (12)$$
$$\forall i, j : w_i \in \text{possessive ncmods}$$
$$\wedge w_j = \text{possessive}$$

Compression examples with the addition of the modifier constraints are shown in Table 1. Although the compressions are grammatical (see the inclusion of *Party* due to the modifier *Pasok* and *Server* due to *AppleShare*), they are not entirely meaning preserving.

Sentential Constraints We also define a few intuitive constraints that take the overall sentence structure into account. The first constraint (Equation (13)) ensures that if a verb is present in the compression then so are its arguments, and if any of the arguments are included in the compression then the verb must also be included. We thus force the program to make the same decision on the verb, its subject, and object.

$$y_i - y_j = 0 \quad (13)$$
$$\forall i, j : w_j \in \text{subject/object of verb } w_i$$

Our second constraint forces the compression to contain at least one verb provided the original sentence contains one as well:

$$\sum_{i \in verbs} y_i \geq 1 \quad (14)$$

Other sentential constraints include Equations (15) and (16) which apply to prepositional phrases, wh-phrases and complements. These constraints force the introducing term (i.e., the preposition, complement or wh-word) to be included in the compression if any word from within the syntactic constituent is also included. The reverse is also true, i.e., if the introducing term is included at least one other word from the syntactic constituent should also be included.

$$y_i - y_j \geq 0 \quad (15)$$
$$\forall i, j : w_j \in \text{PP/COMP/WH-P}$$
$$\wedge w_i \text{ starts PP/COMP/WH-P}$$
$$\sum_{i \in \text{PP/COMP/WH-P}} y_i - y_j \geq 0 \quad (16)$$
$$\forall j : w_j \text{ starts PP/COMP/WH-P}$$

We also wish to handle coordination. If two head words are conjoined in the original sentence, then if they are included in the compression the coordinating conjunction must also be included:

$$(1 - y_i) + y_j \geq 1 \quad (17)$$
$$(1 - y_i) + y_k \geq 1 \quad (18)$$
$$y_i + (1 - y_j) + (1 - y_k) \geq 1 \quad (19)$$
$$\forall i, j, k : w_j \wedge w_k \text{ conjoined by } w_i$$

Table 1 illustrates the compression output when sentential constraints are added to the model. We see that *politics* is forced into the compression due to the presence of *in*; furthermore, since *bundled* is in the compression, its object *with Server* is included too.

Compression-related Constraints Finally, we impose some hard constraints on the compression output. First, Equation (20) disallows anything within brackets in the original sentence from being included in the compression. This is a somewhat superficial attempt at excluding parenthetical and potentially unimportant material from the compression. Second, Equation (21) forces personal pronouns to be included in the compression. The constraint is important for generating coherent document as opposed to sentence compressions.

$$y_i = 0 \quad (20)$$
$$\forall i : w_i \in \text{brackets}$$
$$y_i = 1 \quad (21)$$
$$\forall i : w_i \in \text{personal pronouns}$$

It is also possible to influence the length of the compressed sentence. For example, Equation (22) forces the compression to contain at least b tokens. Alternatively, we could force the compression to be exactly b tokens (by substituting $\geq$ with $=$ in (22)) or to be less than b tokens (by replacing $\geq$ with $\leq$).[1]

$$\sum_{i=1}^{n} y_i \geq b \quad (22)$$

3.3 Significance Score

While the constraint-based language model produces more grammatical output than a regular lan-

[1] Compression rate can be also limited to a range by including two inequality constraints.

guage model, the sentences are typically not great compressions. The language model has no notion of which content words to include in the compression and thus prefers words it has seen before. But words or constituents will be of different relative importance in different documents or even sentences.

Inspired by Hori and Furui (2004), we add to our objective function (see Equation (1)) a significance score designed to highlight important content words. Specifically, we modify Hori and Furui's significance score to give more weight to content words that appear in the deepest level of embedding in the syntactic tree. The latter usually contains the gist of the original sentence:

$$I(w_i) = \frac{l}{N} \cdot f_i \log \frac{F_a}{F_i} \quad (23)$$

The significance score above is computed using a large corpus where w_i is a topic word (i.e., a noun or verb), f_i and F_i are the frequency of w_i in the document and corpus respectively, and F_a is the sum of all topic words in the corpus. l is the number of clause constituents above w_i, and N is the deepest level of embedding. The modified objective function is given below:

$$\begin{aligned} \max z \quad = \quad & \sum_{i=1}^{n} y_i \cdot I(w_i) + \sum_{i=1}^{n} p_i \cdot P(w_i|\text{start}) \\ & + \sum_{i=1}^{n-2} \sum_{j=i+1}^{n-1} \sum_{k=j+1}^{n} x_{ijk} \cdot P(w_k|w_i, w_j) \\ & + \sum_{i=0}^{n-1} \sum_{j=i+1}^{n} q_{ij} \cdot P(\text{end}|w_i, w_j) \quad (24) \end{aligned}$$

A weighting factor could be also added to the objective function, to counterbalance the importance of the language model and the significance score.

4 Evaluation Set-up

We evaluated the approach presented in the previous sections against Knight and Marcu's (2002) decision-tree model. This model is a good basis for comparison as it operates on parse trees and therefore is aware of syntactic structure (as our models are) but requires a large parallel corpus for training whereas our models do not; and it yields comparable performance to the noisy-channel model.[2] The decision-tree model was compared against two variants of our IP model. Both variants employed the constraints described in Section 3.2 but differed in that one variant included the significance score in its objective function (see (24)), whereas the other one did not (see (1)). In both cases the sequential constraints from Section 3.1 were applied to ensure that the language model was well-formed. We give details below on the corpora we used and explain how the different model parameters were estimated. We also discuss how evaluation was carried out using human judgements.

Corpora We evaluate our systems on two different corpora. The first is the compression corpus of Knight and Marcu (2002) derived automatically from document-abstract pairs of the Ziff-Davis corpus. This corpus has been used in most previous compression work. We also created a compression corpus from the HUB-4 1996 English Broadcast News corpus (provided by the LDC). We asked annotators to produce compressions for 50 broadcast news stories (1,370 sentences).[3]

The Ziff-Davis corpus is partitioned into training (1,035 sentences) and test set (32 sentences). We held out 50 sentences from the training for development purposes. We also split the Broadcast News corpus into a training and test set (1,237/133 sentences). Forty sentences were randomly selected for evaluation purposes, 20 from the test portion of the Ziff-Davis corpus and 20 from the Broadcast News corpus test set.

Parameter Estimation The decision-tree model was trained, using the same feature set as Knight and Marcu (2002) on the Ziff-Davis corpus and used to obtain compressions for both test corpora.[4] For our IP models, we used a language model trained on 25 million tokens from the North American News corpus using the CMU-Cambridge Language Modeling Toolkit (Clarkson and Rosenfeld 1997) with a vocabulary size of 50,000 tokens and Good-Turing discounting. The significance score used in our second model was calculated using 25 million tokens from the Broadcast News Corpus (for the spoken data) and 25 million tokens from the American News Text Corpus (for the written data). Finally, the model that includes the significance score was optimised against a loss function similar to McDonald (2006) to bring the language model and the score into harmony. We used Powell's method (Press et al. 1992) and 50 sentences (randomly selected from the training set).

[2] Turner and Charniak (2005) argue that the noisy-channel model is not an appropriate compression model since it uses a source model trained on uncompressed sentences and as a result tends to consider compressed sentences less likely than uncompressed ones.

[3] The corpus is available from `http://homepages.inf.ed.ac.uk/s0460084/data/`.

[4] We found that the decision-tree was unable to produce meaningful compressions when trained on the Broadcast News corpus (in most cases it recreated the original sentence). Thus we used the decision model trained on Ziff-Davis to generate Broadcast News compressions.

We also set a minimum compression length (using the constraint in Equation (22)) in both our models to avoid overly short compressions. The length was set at 40% of the original sentence length or five tokens, whichever was larger. Sentences under five tokens were not compressed.

In our modeling framework, we generate and solve an IP for every sentence we wish to compress. We employed *lp_solve* for this purpose, an efficient Mixed Integer Programming solver.[5] Sentences typically take less than a few seconds to compress on a 2 GHz Pentium IV machine.

Human Evaluation As mentioned earlier, the output of our models is evaluated on 40 examples. Although the size of our test set is comparable to previous studies (which are typically assessed on 32 sentences from the Ziff-Davis corpus), the sample is too small to conduct significance testing. To counteract this, human judgements are often collected on compression output; however the evaluations are limited to small subject pools (often four judges; Knight and Marcu 2002; Turner and Charniak 2005; McDonald 2006) which makes difficult to apply inferential statistics on the data. We overcome this problem by conducting our evaluation using a larger sample of subjects.

Specifically, we elicited human judgements from 56 unpaid volunteers, all self reported native English speakers. The elicitation study was conducted over the Internet. Participants were presented with a set of instructions that explained the sentence compression task with examples. They were asked to judge 160 compressions in total. These included the output of the three automatic systems on the 40 test sentences paired with their gold standard compressions. Participants were asked to read the original sentence and then reveal its compression by pressing a button. They were told that all compressions were generated automatically. A Latin square design ensured that subjects did not see two different compressions of the same sentence. The order of the sentences was randomised. Participants rated each compression on a five point scale based on the information retained and its grammaticality. Examples of our experimental items are given in Table 2.

O:	Apparently Fergie very much wants to have a career in television.
G:	Fergie wants a career in television.
D:	A career in television.
LM:	Fergie wants to have a career.
Sig:	Fergie wants to have a career in television.
O:	The SCAMP module, designed and built by Unisys and based on an Intel process, contains the entire 48-bit A-series processor.
G:	The SCAMP module contains the entire 48-bit A-series processor.
D:	The SCAMP module designed Unisys and based on an Intel process.
LM:	The SCAMP module, contains the 48-bit A-series processor.
Sig:	The SCAMP module, designed and built by Unisys and based on process, contains the A-series processor.

Table 2: Compression examples (O: original sentence, G: Gold standard, D: Decision-tree, LM: IP language model, Sig: IP language model with significance score)

Model	CompR	Rating
Decision-tree	56.1%	$2.22^{*\dagger}$
LangModel	49.0%	$2.23^{*\dagger}$
LangModel+Significance	73.6%	2.83^{*}
Gold Standard	62.3%	$3.68^{\dagger}$

Table 3: Compression results; compression rate (CompR) and average human judgements (Rating); *: sig. diff. from gold standard; †: sig. diff. from LangModel+Significance

5 Results

Our results are summarised in Table 3 which details the compression rates[6] and average human ratings (Rating) for the three systems and the gold standard. As can be seen, the IP language model (LangModel) is most aggressive in terms of compression rate as it reduces the original sentences on average by half (49%). Recall that we enforce a minimum compression rate of 40% (see (22)). The fact that the resulting compressions are longer, indicates that our constraints instill some linguistic knowledge into the language model, thus enabling it to prefer longer sentences over extremely short ones. The decision-tree model compresses slightly less than our IP language model at 56.1% but still below the gold standard rate. We see a large compression rate increase from 49% to 73.6% when we introduce the significance score into the objective function. This is around 10% higher than the gold standard compression rate.

We now turn to the results of our elicitation study. We performed an Analysis of Variance (ANOVA) to examine the effect of different system compressions. Statistical tests were carried out on the mean of the ratings shown in Table 3. We observe a reliable effect of compression type by sub-

[5]The software is available from `http://www.geocities.com/lpsolve/`.

[6]We follow previous work (see references) in using the term "compression rate" to refer to the percentage of words *retained* in the compression.

jects ($F_1(3,165) = 132.74,\ p < 0.01$) and items ($F_2(3,117) = 18.94,\ p < 0.01$). Post-hoc Tukey tests revealed that gold standard compressions are perceived as significantly better than those generated by all automatic systems ($\alpha < 0.05$). There is no significant difference between the IP language model and decision-tree systems. However, the IP model with the significance score delivers a significant increase in performance over the language model and the decision tree ($\alpha < 0.05$).

These results indicate that reasonable compressions can be obtained with very little supervision. Our constraint-based language model does not make use of a parallel corpus, whereas our second variant uses only 50 parallel sentences for tuning the weights of the objective function. The models described in this paper could be easily adapted to other domains or languages provided that syntactic analysis tools are to some extent available.

6 Conclusions and Future Work

In this paper we have presented a novel method for automatic sentence compression. A key aspect of our approach is the use of integer programming for inferring globally optimal compressions in the presence of linguistically motivated constraints. We have shown that such a formulation allows for a relatively simple and knowledge-lean compression model that does not require parallel corpora or access to large-scale knowledge bases. Our results demonstrate that the IP model yields performance comparable to state-of-the-art without any supervision. We also observe significant performance gains when a small amount of training data is employed (50 parallel sentences). Beyond the systems discussed in this paper, the approach holds promise for other models using decoding algorithms for searching the space of possible compressions. The search process could be framed as an integer program in a similar fashion to our work here.

We obtain our best results using a model whose objective function includes a significance score. The significance score relies mainly on syntactic and lexical information for determining whether a word is important or not. An appealing future direction is the incorporation of discourse-based constraints into our models. The latter would highlight topical words at the document-level instead of considering each sentence in isolation. Another important issue concerns the portability of the models presented here to other languages and domains. We plan to apply our method to languages with more flexible word order than English (e.g., German) and more challenging spoken domains (e.g., meeting data) where parsing technology may be less reliable.

Acknowledgements

Thanks to Jean Carletta, Amit Dubey, Frank Keller, Steve Renals, and Sebastian Riedel for helpful comments and suggestions. Lapata acknowledges the support of EPSRC (grant GR/T04540/01).

References

Briscoe, E. J. and J. Carroll. 2002. Robust accurate statistical annotation of general text. In *Proceedings of the 3rd LREC*. Las Palmas, Gran Canaria, pages 1499–1504.

Clarkson, Philip and Ronald Rosenfeld. 1997. Statistical language modeling using the CMU–cambridge toolkit. In *Proceedings of Eurospeech*. Rhodes, Greece, pages 2707–2710.

Hori, Chiori and Sadaoki Furui. 2004. Speech summarization: an approach through word extraction and a method for evaluation. *IEICE Transactions on Information and Systems* E87-D(1):15–25.

Jing, Hongyan. 2000. Sentence reduction for automatic text summarization. In *Proceedings of the 6th ANLP*. Seattle, WA, pages 310–315.

Knight, Kevin and Daniel Marcu. 2002. Summarization beyond sentence extraction: a probabilistic approach to sentence compression. *Artificial Intelligence* 139(1):91–107.

Marciniak, Tomasz and Michael Strube. 2005. Beyond the pipeline: Discrete optimization in NLP. In *Proceedings of the 9th CoNLL*. Ann Arbor, MI, pages 136–143.

McDonald, Ryan. 2006. Discriminative sentence compression with soft syntactic constraints. In *Proceedings of the 11th EACL*. Trento, Italy, pages 297–304.

Nguyen, Minh Le, Akira Shimazu, Susumu Horiguchi, Tu Bao Ho, and Masaru Fukushi. 2004. Probabilistic sentence reduction using support vector machines. In *Proceedings of the 20th COLING*. Geneva, Switzerland, pages 743–749.

Olivers, S. H. and W. B. Dolan. 1999. Less is more; eliminating index terms from subordinate clauses. In *Proceedings of the 37th ACL*. College Park, MD, pages 349–356.

Press, William H., Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. 1992. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press.

Punyakanok, Vasin, Dan Roth, Wen-tau Yih, and Dav Zimak. 2004. Semantic role labeling via integer linear programming inference. In *Proceedings of the 20th COLING*. Geneva, Switzerland, pages 1346–1352.

Riezler, Stefan, Tracy H. King, Richard Crouch, and Annie Zaenen. 2003. Statistical sentence condensation using ambiguity packing and stochastic disambiguation methods for lexical-functional grammar. In *Proceedings of the HLT/NAACL*. Edmonton, Canada, pages 118–125.

Roth, Dan and Wen-tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *Proceedings of the 8th CoNLL*. Boston, MA, pages 1–8.

Turner, Jenine and Eugene Charniak. 2005. Supervised and unsupervised learning for sentence compression. In *Proceedings of the 43rd ACL*. Ann Arbor, MI, pages 290–297.

Vandeghinste, Vincent and Yi Pan. 2004. Sentence compression for automated subtitling: A hybrid approach. In *Proceedings of the ACL Workshop on Text Summarization*. Barcelona, Spain, pages 89–95.

Winston, Wayne L. and Munirpallam Venkataramanan. 2003. *Introduction to Mathematical Programming*. Brooks/Cole.

Topic-Focused Multi-document Summarization Using an Approximate Oracle Score

John M. Conroy, Judith D. Schlesinger
IDA Center for Computing Sciences
Bowie, Maryland, USA
conroy@super.org, judith@super.org

Dianne P. O'Leary
University of Maryland
College Park, Maryland, USA
oleary@cs.umd.edu

Abstract

We consider the problem of producing a multi-document summary given a collection of documents. Since most successful methods of multi-document summarization are still largely extractive, in this paper, we explore just how well an extractive method can perform. We introduce an "oracle" score, based on the probability distribution of unigrams in human summaries. We then demonstrate that with the oracle score, we can generate extracts which score, on average, better than the human summaries, when evaluated with ROUGE. In addition, we introduce an approximation to the oracle score which produces a system with the best known performance for the 2005 Document Understanding Conference (DUC) evaluation.

1 Introduction

We consider the problem of producing a multi-document summary given a collection of documents. Most automatic methods of multi-document summarization are largely extractive. This mimics the behavior of humans for single document summarization; (Kupiec, Pendersen, and Chen 1995) reported that 79% of the sentences in a human-generated abstract were a "direct match" to a sentence in a document. In contrast, for multi-document summarization, (Copeck and Szpakowicz 2004) report that no more than 55% of the vocabulary contained in human-generated abstracts can be found in the given documents. Furthermore, multiple human summaries on the same collection of documents often have little agreement. For example, (Hovy and Lin 2002) report that unigram overlap is around 40%. (Teufel and van Halteren 2004) used a "factoid" agreement analysis of human summaries for a single document and concluded that a resulting consensus summary is stable only if 30–40 summaries are collected.

In light of the strong evidence that nearly half of the terms in human-generated multi-document abstracts are not from the original documents, and that agreement of vocabulary among human abstracts is only about 40%, we pose two coupled questions about the quality of summaries that can be attained by document extraction:

1. Given the sets of unigrams used by four human summarizers, can we produce an extract summary that is statistically indistinguishable from the human abstracts when measured by current automatic evaluation methods such as ROUGE?

2. If such unigram information can produce good summaries, can we replace this information by a statistical model and still produce good summaries?

We will show that the answer to the first question is, indeed, yes and, in fact, the unigram set information gives rise to extract summaries that usually score better than the 4 human abstractors! Secondly, we give a method to statistically approximate the set of unigrams and find it produces extracts of the DUC 05 data which outperform all known evaluated machine entries. We conclude with experiments on the extent that redundancy removal improves extracts, as well as a method of moving beyond simple extracting by employing shallow parsing techniques to shorten the sentences prior to selection.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 152–159,
Sydney, July 2006.

2 The Data

The 2005 Document Understanding Conference (DUC 2005) data used in our experiments is partitioned into 50 topic sets, each containing 25–50 documents. A topic for each set was intended to mimic a real-world complex questioning-answering task for which the answer could not be given in a short "nugget." For each topic, four human summarizers were asked to provide a 250-word summary of the topic. Topics were labeled as either "general" or "specific". We present an example of one of each category.

`Set d408c`

Granularity: Specific

Title: Human Toll of Tropical Storms

Narrative: What has been the human toll in death or injury of tropical storms in recent years? Where and when have each of the storms caused human casualties? What are the approximate total number of casualties attributed to each of the storms?

`Set d436j`

Granularity: General

Title: Reasons for Train Wrecks

Narrative: What causes train wrecks and what can be done to prevent them? Train wrecks are those events that result in actual damage to the trains themselves not just accidents where people are killed or injured.

For each topic, the goal is to produce a 250-word summary. The basic unit we extract from a document is a sentence.

To prepare the data for processing, we segment each document into sentences using a POS (part-of-speech) tagger, NLProcessor (http://www.infogistics.com/posdemo.htm). The newswire documents in the DUC 05 data have markers indicating the regions of the document, including titles, bylines, and text portions. All of the extracted sentences in this study are taken from the text portions of the documents only.

We define a "term" to be any "non-stop word." Our stop list contains the 400 most frequently occurring English words.

3 The Oracle Score

Recently, a crisp analysis of the frequency of content words used by humans relative to the high frequency content words that occur in the relevant documents has yielded a simple and powerful summarization method called SumBasic (Nenkova and Vanderwende, 2005). SumBasic produced extract summaries which performed nearly as well as the best machine systems for generic 100 word summaries, as evaluated in DUC 2003 and 2004, as well as the Multi-lingual Summarization Evaluation (MSE 2005).

Instead of using term frequencies of the corpus to infer highly likely terms in human summaries, we propose to directly model the *set* of terms (vocabulary) that is likely to occur in a sample of human summaries. We seek to estimate the probability that a term will be used by a human summarizer to first get an estimate of the best possible extract and later to produce a statistical model for an extractive summary system. While the primary focus of this work is "task oriented" summaries, we will also address a comparison with SumBasic and other systems on generic multi-document summaries for the DUC 2004 dataset in Section 8.

Our extractive summarization system is given a topic, τ, specified by a text description. It then evaluates each sentence in each document in the set to determine its appropriateness to be included in the summary for the topic τ.

We seek a statistic which can score an individual sentence to determine if it should be included as a candidate. We desire that this statistic take into account the great variability that occurs in the space of human summaries on a given topic τ. One possibility is to simply judge a sentence based upon the expected fraction of the "human summary"-terms that it contains. We posit an oracle, which answers the question "Does human summary i contain the term t?"

By invoking this oracle over the set of terms and a sample of human summaries, we can readily compute the expected fraction of human summary-terms the sentence contains. To model the variation in human summaries, we use the oracle to build a probabilistic model of the space of human abstracts. Our "oracle score" will then compute the expected number of summary terms a sentence contains, where the expectation is taken from the space of all human summaries on the topic τ.

We model human variation in summary generation with a unigram bag-of-words model on the terms. In particular, consider $P(t|\tau)$ to be the probability that a human will select term t in a summary given a topic τ. The oracle score for a sentence x, $\omega(x)$, can then be defined in terms of

P :

$$\omega(x) = \frac{1}{|x|} \sum_{t \in T} x(t) P(t|\tau)$$

where $|x|$ is the number of distinct terms sentence x contains, T is the universal set of all terms used in the topic τ and $x(t) = 1$ if the sentence x contains the term t and 0 otherwise. (We affectionally refer to this score as the "Average Jo" score, as it is derived the average uni-gram distribution of terms in human summaries.)

While we will consider several approximations to $P(t|\tau)$ (and, correspondingly, ω), we first explore the maximum-likelihood estimate of $P(t|\tau)$ given by a sample of human summaries. Suppose we are given h sample summaries generated independently. Let $c_{it}(\tau) = 1$ if the i-th summary contains the term t and 0 otherwise. Then the maximum-likelihood estimate of $P(t\tau)$ is given by

$$\hat{P}(t|\tau) = \frac{1}{h} \sum_{i=1}^{h} c_{it}(\tau).$$

We define $\hat{\omega}$ by replacing P with $\hat{P}$ in the definition of ω. Thus, $\hat{\omega}$ is the maximum-likelihood estimate for ω, given a set of h human summaries.

Given the score $\hat{\omega}$, we can compute an extract summary of a desired length by choosing the top scoring sentences from the collection of documents until the desired length (250 words) is obtained. We limit our selection to sentences which have 8 or more distinct terms to avoid selecting incomplete sentences which may have been tagged by the sentence splitter.

Before turning to how well our idealized score, $\hat{\omega}$, performs on extract summaries, we first define the scoring mechanism used to evaluate these summaries.

4 ROUGE

The state-of-the-art automatic summarization evaluation method is ROUGE (Recall Oriented Understudy for Gisting Evaluation, (Hovy and Lin 2002)), an n-gram based comparison that was motivated by the machine translation evaluation metric, Bleu (Papineni et. al. 2001). This system uses a variety of n-gram matching approaches, some of which allow gaps within the matches as well as more sophistcated analyses. Surprisingly, simple unigram and bigram matching works extremely well. For example, at DUC 05, ROUGE-2 (bigram match) had a Spearman correlation of 0.95 and a Pearson correlation of 0.97 when compared with human evaluation of the summaries for responsiveness (Dang 2005). ROUGE-n for matching $n-$grams of a summary X against h model human summaries is given by:

$$R_n(X) = \frac{\sum_{j=1}^{h} \sum_{i \in N_n} \min(X_n(i), M_n(i,j))}{\sum_{j=1}^{h} \sum_{i \in N_n} M_n(i,j),}$$

where $X_n(i)$ is the count of the number of times the n-gram i occurred in the summary and $M_n(i,j)$ is the number of times the n-gram i occurred in the j-th model (human) summary. (Note that for brevity of notation, we assume that lemmatization (stemming) is done *apriori* on the terms.)

When computing ROUGE scores, a jackknife procedure is done to make comparison of machine systems and humans more amenable. In particular, if there are k human summaries available for a topic, then the ROUGE score is computed for a human summary by comparing it to the remaining $k-1$ summaries, while the ROUGE score for a machine summary is computed against all k subsets of size $k-1$ of the human summaries and taking the average of these k scores.

5 The Oracle or *Average Jo* Summary

We now present results on the performance of the oracle method as compared with human summaries. We give the ROUGE-2 (R_2) scores as well as the 95% confidence error bars. In Figure 1, the human summarizers are represented by the letters A–H, and systems 15, 17, 8, and 4 are the top performing machine summaries from DUC 05. The letter "O" represents the ROUGE-2 scores for extract summaries produced by the oracle score, $\hat{\omega}$. Perhaps surprisingly, the oracle produced extracts which performed better than the human summaries! Since each human only summarized 10 document clusters, the human error bars are larger. However, even with the large error bars, we observe that the mean ROUGE-2 scores for the oracle extracts exceeds the 95% confidence error bars for several humans.

While the oracle was, of course, given the unigram term probabilities, its performance is notable on two counts. First, the evaluation metric scored on 2-grams, while the oracle was only given unigram information. In a sense, optimizing for ROUGE-1 is a "sufficient statistic" scoring at

the human level for ROUGE-2. Second, the humans wrote abstracts while the oracle simply did extracting. Consequently, the documents contain sufficient text to produce human-quality extract summaries as measured by ROUGE. The human performance ROUGE scores indicate that this approach is capable of producing automatic extractive summaries that produce vocabulary comparable to that chosen by humans. Human evaluation (which we have not yet performed) is required to determine to what extent this high ROUGE-2 performance is indicative of high quality summaries for human use.

The encouraging results of the oracle score naturally lead to approximations, which, perhaps, will give rise to strong machine system performance. Our goal is to approximate $P(t|\tau)$, the probability that a term will be used in a human abstract. In the next section, we present two approaches which will be used in tandem to make this approximation.

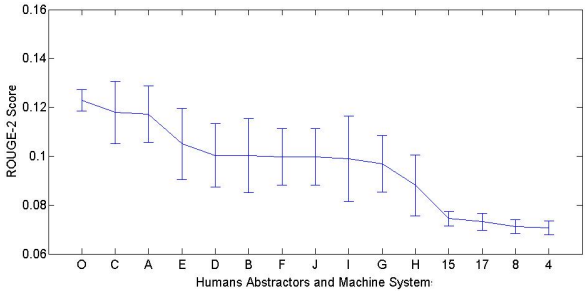


Figure 1: The Oracle (Average Jo score) Score $\hat{\omega}$

6 Approximating $P(t|\tau)$

We seek to approximate $P(t|\tau)$ in an analogous fashion to the maximum-likelihood estimate $\hat{P}(t|\tau)$. To this end, we devise methods to isolate a subset of terms which would likely be included in the human summary. These terms are gleaned from two sources, the topic description and the collection of documents which were judged relevant to the topic. The former will give rise to *query terms* and the latter to *signature terms*.

6.1 Query Term Identification

A set of query terms is automatically extracted from the given topic description. We identified individual words and phrases from both the <topic> (Title) tagged paragraph as well as whichever of the <narr> (Narrative) tagged paragraphs occurred in the topic description. We made no use of the <granularity> paragraph marking. We tagged the topic description using the POS-tagger, NLProcessor (http://www.infogistics.com/posdemo.htm), and any words that were tagged with any NN (noun), VB (verb), JJ (adjective), or RB (adverb) tag were included in a list of words to use as query terms. Table 1 shows a list of query terms for our two illustrative topics.

`Set d408c:` approximate, casualties, death, human, injury, number, recent, storms, toll, total, tropical, years
`Set d436j:` accidents, actual, causes, damage, events, injured, killed, prevent, result, train, train wrecks, trains, wrecks

Table 1: Query Terms for "Tropical Storms" and "Train Wrecks" Topics

The number of query terms extracted in this way ranged from a low of 3 terms for document set d360f to 20 terms for document set d324e.

6.2 Signature Terms

The second collection of terms we use to estimate $P(t|\tau)$ are signature terms. Signature terms are the terms that are more likely to occur in the document set than in the background corpus. They are generally indicative of the content contained in the collection of documents. To identify these terms, we use the log-likelihood statistic suggested by Dunning (Dunning 1993) and first used in summarization by Lin and Hovy (Hovy and Lin 2000). The statistic is equivalent to a mutual information statistic and is based on a 2-by-2 contingency table of counts for each term. Table 2 shows a list of signature terms for our two illustrative topics.

6.3 An estimate of $P(t|\tau)$

To estimate $P(t|\tau)$, we view both the query terms and the signature terms as "samples" from idealized human summaries. They represent the terms that we would most likely see in a human summary. As such, we expect that these sample terms may approximate the underlying set of human summary terms. Given a collection of query terms and signature terms, we can readily estimate our target objective, $P(t|\tau)$ by the following:

$$P_{qs}(t|\tau) = \frac{1}{2}q_t(\tau) + \frac{1}{2}s_t(\tau)$$

`Set d408c:` ahmed, allison, andrew, bahamas, bangladesh, bn, caribbean, carolina, caused, cent, coast, coastal, croix, cyclone, damage, destroyed, devastated, disaster, dollars, drowned, flood, flooded, flooding, floods, florida, gulf, ham, hit, homeless, homes, hugo, hurricane, insurance, insurers, island, islands, lloyd, losses, louisiana, manila, miles, nicaragua, north, port, pounds, rain, rains, rebuild, rebuilding, relief, remnants, residents, roared, salt, st, storm, storms, supplies, tourists, trees, tropical, typhoon, virgin, volunteers, weather, west, winds, yesterday.
`Set d436j:` accident, accidents, ammunition, beach, bernardino, board, boulevard, brake, brakes, braking, cab, car, cargo, cars, caused, collided, collision, conductor, coroner, crash, crew, crossing, curve, derail, derailed, driver, emergency, engineer, engineers, equipment, fe, fire, freight, grade, hit, holland, injured, injuries, investigators, killed, line, locomotives, maintenance, mechanical, miles, morning, nearby, ntsb, occurred, officials, pacific, passenger, passengers, path, rail, railroad, railroads, railway, routes, runaway, safety, san, santa, shells, sheriff, signals, southern, speed, station, train, trains, transportation, truck, weight, wreck

Table 2: Signature Terms for "Tropical Storms" and "Train Wrecks" Topics

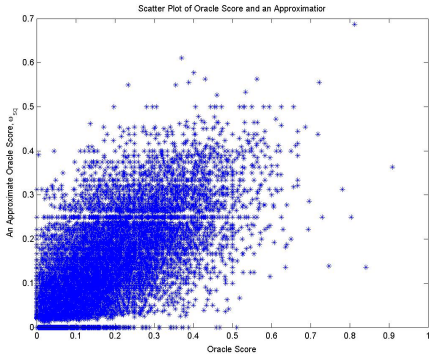


Figure 2: Scatter Plot of $\hat{\omega}$ versus ω_{qs}

where $s_t(\tau)$=1 if t is a signature term for topic τ and 0 otherwise and $q_t(\tau) = 1$ if t is a query term for topic τ and 0 otherwise.

More sophisticated weightings of the query and signature have been considered; however, for this paper we limit our attention to the above elementary scheme. (Note, in particular, a psuedo-relevance feedback method was employed by (Conroy et. al. 2005), which gives improved performance.)

Similarly, we estimate the oracle score of a sentence's expected number of human abstract terms as

$$\omega_{qs}(x) = \frac{1}{|x|} \sum_{t \in T} x(t) P_{qs}(t|\tau)$$

where $|x|$ is the number of distinct terms that sentence x contains, T is the universal set of all terms and $x(t) = 1$ if the sentence x contains the term t and 0 otherwise.

For both the oracle score and the approximation, we form the summary by taking the top scoring sentences among those sentences with at least 8 distinct terms, until the desired length (250 words for the DUC05 data) is achieved or exceeded. (The threshold of 8 was based upon previous analysis of the sentence splitter, which indicated that sentences shorter than 8 terms tended not be be well formed sentences or had minimal, if any, content.) If the length is too long, the last sentence chosen is truncated to reach the target length.

Figure 2 gives a scatter plot of the oracle score ω and its approximation ω_{qs} for all sentences with at least 8 unique terms. The overall Pearson correlation coefficient is approximately 0.70. The correlation varies substantially over the topics. Figure 3 gives a histogram of the Pearson correlation coefficients for the 50 topic sets.

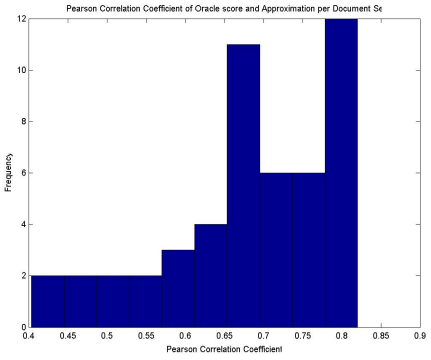


Figure 3: Histogram of Document Set Pearson Coefficients of $\hat{\omega}$ versus ω_{qs}

7 Enhancements

In the this section we explore two approaches to improve the quality of the summary, linguistic preprocessing (sentence trimming) and a redundancy removal method.

7.1 Linguistic Preprocessing

We developed patterns using "shallow parsing" techniques, keying off of lexical cues in the sentences after processing them with the POS-tagger. We initially used some full sentence eliminations along with the phrase eliminations itemized below; analysis of DUC 03 results, however, demonstrated that the full sentence eliminations were not useful.

The following phrase eliminations were made, when appropriate:

- gerund clauses;
- restricted relative-clause appositives;
- intra-sentential attribution;
- lead adverbs.

See (Dunlavy et. al) for the specific rules used for these eliminations. Comparison of two runs in DUC 04 convinced us of the benefit of applying these phrase eliminations on the full documents, prior to summarization, rather than on the selected sentences after scoring and sentence selection had been performed. See (Conroy et. al. 2004) for details on this comparison.

After the trimmed text has been generated, we then compute the signature terms of the document sets and recompute the approximate oracle scores. Note that since the sentences have usually had some extraneous information removed, we expect some improvement in the quality of the signature terms and the resulting scores. Indeed, the median ROUGE-2 score increases from 0.078 to 0.080.

7.2 Redundancy Removal

The greedy sentence selection process we described in Section 6 gives no penalty for sentences which are redundant to information already contained in the partially formed summary. A method for reducing redundancy can be employed. One popular method for reducing redundancy is maximum marginal relevance (MMR) (2). Based on previous studies, we have found that a pivoted QR, a method from numerical linear algebra, has some advantages over MMR and performs somewhat better.

Pivoted QR works on a term-sentence matrix formed from a set of candidate sentences for inclusion in the summary. We start with enough sentences so the total number of terms is approximately twice the desired summary length. Let B be the term-sentence matrix with $B_{ij} = 1$ if sentence j contains term i.

The columns of B are then normalized so their 2-norm (Euclidean norm) is the corresponding approximate oracle score, i.e. $\omega_{qs}(b_j)$, where b_j is the j-th column of B. We call this normalized term sentence matrix A.

Given a normalized term-sentence matrix A, QR factorization attempts to select columns of A in the order of their importance in spanning the subspace spanned by all of the columns. The standard implementation of pivoted QR decomposition is a "Gram-Schmidt" process. The first r sentences (columns) selected by the pivoted QR are used to form the summary. The number r is chosen so that the summary length is close to the target length. A more complete description can be found in (Conroy and O'Leary 2001).

Note, that the selection process of using the pivoted QR on the weighted term sentence matrix will first choose the sentence with the highest ωpq score as was the case with the greedy selection process. Its subsequent choices are affected by previous choices as the weights of the columns are decreased for any sentence which can be approximated by a linear combination of the current set of selected sentences. This is more general than simply demanding that the sentence have small overlap with the set of previous chosen sentences as

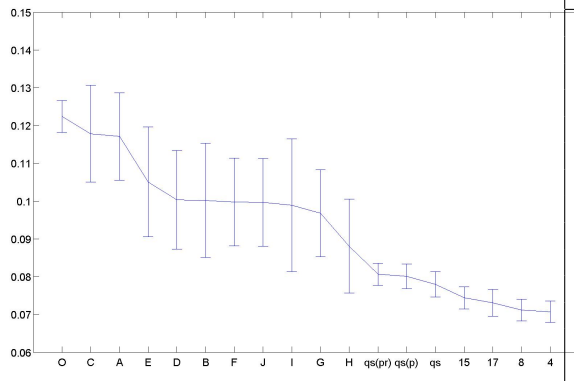


Figure 4: ROUGE-2 Performance of Oracle Score Approximations $\hat{\omega}$ vs. Humans and Peers

would be done using MMR.

8 Results

Figure 4 gives the ROUGE-2 scores with error bars for the approximations of the oracle score as well as the ROUGE-2 scores for the human summarizers and the top performing systems at DUC 2005. In the graph, qs is the approximate oracle, qs(p) is the approximation using linguistic preprocessing, and qs(pr) is the approximation with both linguistic preprocessing and redundancy removal. Note that while there is some improvement using the linguistic preprocessing, the improvement using our redundancy removal technique is quite minor. Regardless, our system using signature terms and query terms as estimates for the oracle score performs comparably to the top scoring system at DUC 05.

Table 3 gives the ROUGE-2 scores for the recent DUC 06 evaluation which was essentially the same task as for DUC 2005. The manner in which the linguistic preprocessing is performed has changed from DUC 2005, although the types of removals have remained the same. In addition, pseudo-relevance feedback was employed for redundancy removal as mentioned earlier. See (Conroy et. al. 2005) for details.

While the main focus of this study is task-oriented multidocument summarization, it is instructive to see how well such an approach would perform for a generic summarization task as with the 2004 DUC Task 2 dataset. Note, the ω score for generic summaries uses only the signature term portion of the score, as no topic description is given. We present ROUGE-1 (rather than

Submission	Mean	95% CI Lower	95% CI Upper
O (ω)	0.13710	0.13124	0.14299
C	0.13260	0.11596	0.15197
D	0.12380	0.10751	0.14003
B	0.11788	0.10501	0.13351
G	0.11324	0.10195	0.12366
F	0.10893	0.09310	0.12780
H	0.10777	0.09833	0.11746
J	0.10717	0.09293	0.12460
I	0.10634	0.09632	0.11628
E	0.10365	0.08935	0.11926
A	0.10361	0.09260	0.11617
24	0.09558	0.09144	0.09977
$\omega_{qs}^{(pr)}$	0.09160	0.08729	0.09570
15	0.09097	0.08671	0.09478
12	0.08987	0.08583	0.09385
8	0.08954	0.08540	0.09338
23	0.08792	0.08371	0.09204
$\omega_{qs}^{(p)}$	0.08738	0.08335	0.09145
ω_{qs}	0.08713	0.08317	0.09110
28	0.08700	0.08332	0.09096

Table 3: Average ROUGE 2 Scores for DUC06: Humans A-I

ROUGE-2) scores with stop words removed for comparison with the published results given in (Nenkova and Vanderwende, 2005).

Table 4 gives these scores for the top performing systems at DUC04 as well as SumBasic and $\omega_{qs}^{(pr)}$, the approximate oracle based on signature terms alone with linguistic preprocess trimming and pivot QR for redundancy removal. As displayed, $\omega_{qs}^{(pr)}$ scored second highest and within the 95% confidence intervals of the top system, peer 65, as well as SumBasic, and peer 34.

Submission	Mean	95% CI Lower	95% CI Upper
F	0.36787	0.34442	0.39467
B	0.36126	0.33387	0.38754
O (ω)	0.35810	0.34263	0.37330
H	0.33871	0.31540	0.36423
A	0.33289	0.30591	0.35759
D	0.33212	0.30805	0.35628
E	0.33277	0.30959	0.35687
C	0.30237	0.27863	0.32496
G	0.30909	0.28847	0.32987
$\omega_{qs}^{(pr)}$	0.308	0.294	0.322
peer 65	0.308	0.293	0.323
SumBasic	0.302	0.285	0.319
peer 34	0.290	0.273	0.307
peer 124	0.286	0.268	0.303
peer 102	0.285	0.267	0.302

Table 4: Average ROUGE 1 Scores with stop words removed for DUC04, Task 2

9 Conclusions

We introduced an oracle score based upon the simple model of the probability that a human will choose to include a term in a summary. The oracle score demonstrated that for task-based summarization, extract summaries score as well as human-generated abstracts using ROUGE. We then demonstrated that an approximation of the oracle score based upon query terms and signature terms gives rise to an automatic method of summarization, which outperforms the systems entered in DUC05. The approximation also performed very well in DUC 06. Further enhancements based upon linguistic trimming and redundancy removal via a pivoted QR algorithm give significantly better results.

References

Jamie Carbonnell and Jade Goldstein "The of MMR, diversity-based reranking for reordering documents and producing summaries." In Proc. ACM SIGIR, pages 335–336.

John M. Conroy and Dianne P. O'Leary. "Text Summarization via Hidden Markov Models and Pivoted QR Matrix Decomposition". Technical report, University of Maryland, College Park, Maryland, March, 2001.

John M. Conroy and Judith D. Schlesinger and Jade Goldstein and Dianne P. O'Leary, Left-Brain Right-Brain Multi-Document Summarization, *Document Understanding Conference 2004* http://duc.nist.gov/ 2004

John M. Conroy and Judith D. Schlesinger and Jade Goldstein, CLASSY Tasked Based Summarization: Back to Basics, *Document Understanding Conference 2005* http://duc.nist.gov/ 2005

John M. Conroy and Judith D. Schlesinger Dianne P. O'Leary, and Jade Goldstein, Back to Basciss: CLASSY 2006, *Document Understanding Conference 2006* http://duc.nist.gov/ 2006

Terry Copeck and Stan Szpakowicz 2004 Vocabulary Agreement Among Model Summaries and Source Documents In *ACL Text Summarization Workshop, ACL 2004*.

Hoa Trang Dang Overview of DUC 2005 *Document Understanding Conference 2005* http://duc.nist.gov

Daniel M. Dunlavy and John M. Conroy and Judith D. Schlesinger and Sarah A. Goodman and Mary Ellen Okurowski and Dianne P. O'Leary and Hans van Halteren, "Performance of a Three-Stage System for Multi-Document Summarization", DUC 03 Conference Proceedings, http://duc.nist.gov/, 2003

Ted Dunning, "Accurate Methods for Statistics of Surprise and Coincidence", *Computational Linguistics*, 19:61-74, 1993.

Julian Kupiec,, Jan Pedersen, and Francine Chen. "A Trainable Document Summarizer". *Proceedings of the 18th Annual International SIGIR Conference on Research and Development in Information Retrieval*, pages 68–73, 1995.

Chin-Yew Lin and Eduard Hovy. The automated acquisition of topic signatures for text summarization. In *Proceedings of the 18th conference on Computational linguistics*, pages 495–501, Morristown, NJ, USA, 2000. Association for Computational Linguistics.

Chin-Yew Lin and Eduard Hovy. Manual and Automatic Evaluation of Summaries In *Document Understanding Conference 2002* http:/duc.nist.gov

Multi-Lingual Summarization Evaluation http://www.isi.edu/ cyl/MTSE2005/MLSummEval.html

NLProcessor http://www.infogistics.com/posdemo.htm

Ani Nenkova and Lucy Vanderwende. 2005. *The Impact of Frequency on Summarization*,MSR-TR-2005-101. Microsoft Research Technical Report.

Kishore Papineni and Salim Roukos and Todd Ward and Wei-Jing Zhu, Bleu: a method for automatic evaluation of machine translation, *Technical Report RC22176 (W0109-022), IBM Research Division, Thomas J. Watson Research Center (2001)*

Simone Teufel and Hans van Halteren. 2004. *4: Evaluating Information Content by Factoid Analysis: Human Annotation and Stability*, EMNLP-04, Barcelona

Using WordNet to Automatically Deduce Relations between Words in Noun-Noun Compounds

Fintan J. Costello,
School of Computer Science,
University College Dublin,
Dublin 6, Ireland.
fintan.costello@ucd.ie

Tony Veale,
Department of Computer Science,
University College Dublin,
Dublin 6, Ireland.
tony.veale@ucd.ie

Simon Dunne,
Department of Computer Science,
University College Dublin, Dublin 6, Ireland.
sdunne@inismor.ucd.ie

Abstract

We present an algorithm for automatically disambiguating noun-noun compounds by deducing the correct semantic relation between their constituent words. This algorithm uses a corpus of 2,500 compounds annotated with WordNet senses and covering 139 different semantic relations (we make this corpus available online for researchers interested in the semantics of noun-noun compounds). The algorithm takes as input the WordNet senses for the nouns in a compound, finds all parent senses (hypernyms) of those senses, and searches the corpus for other compounds containing any pair of those senses. The relation with the highest proportional co-occurrence with any sense pair is returned as the correct relation for the compound. This algorithm was tested using a 'leave-one-out' procedure on the corpus of compounds. The algorithm identified the correct relations for compounds with high precision: in 92% of cases where a relation was found with a proportional co-occurrence of 1.0, it was the correct relation for the compound being disambiguated.

Keywords: Noun-Noun Compounds, Conceptual Combination, Word Relations, WordNet

1 Introduction

Noun-noun compounds are short phrases made up of two or more nouns. These compounds are common in everyday language and are especially frequent, and important, in technical documents (Justeson & Katz, 1995, report that such phrases form the majority of technical content of scientific and technical documents surveyed). Understanding these compounds requires the listener or reader to infer the correct semantic relationship between the words making up the compound, inferring, for example, that the phrase 'flu virus' refers to a virus that *causes* flu, while 'skin virus' describes a virus that *affects* the skin, and *marsh virus* a virus *contracted in* marshes. In this paper we describe a novel algorithm for disambiguating noun-noun compounds by automatically deducing the correct semantic relationship between their constituent words.

Our approach to compound disambiguation combines statistical and ontological information about words and relations in compounds. Ontological information is derived from WordNet (Miller, 1995), a hierarchical machine readable dictionary, which is introduced in Section 1. Section 2 describes the construction of an annotated corpus of 2,500 noun-noun compounds covering 139 different semantic relations, with each noun and each relation annotated with its correct WordNet sense.[1]

Section 3 describes our algorithm for finding the correct relation between nouns in a compound, which makes use of this annotated corpus. Our general approach is that the correct relation between two words in a compound can be deduced by finding other compounds containing words from the same semantic categories as the words in the compound to be disambiguated: if a particular relation occurs frequently in those other compounds, that relation is probably also the correct relation for the compound in question. Our al-

[1] A file containing this corpus is available for download from http://inismor.ucd.ie/∼fintanc/wordnet_compounds

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 160–167,
Sydney, July 2006.

Table 1: Thematic relations proposed by Gagné.

relation	example
head *causes* modifier	flu virus
modifier *causes* head	college headache
head *has* modifier	picture book
modifier *has* head	lemon peel
head *makes* modifier	milk cow
head *made of* modifier	chocolate bird
head *for* modifier	cooking toy
modifier *is* head	dessert food
head *uses* modifier	gas antiques
head *about* modifier	travel magazine
head *located* modifier	mountain cabin
head *used by* modifier	servant language
modifier *located* head	murder town
head *derived from* modifier	oil money

gorithm implements this approach by taking as input the correct WordNet senses for the constituent words in a compound (both base senses and parent or hypernyms of those senses), and searching the corpus for other compounds containing any pair of those base or hypernym senses. Relations are given a score equal to their proportional occurrence with those sense pairs, and the relation with the highest proportional occurrence score across all sense-pairs is returned as the correct relation for the compound. Section 4 describes two different leave-one-out tests of this 'Proportional Relation Occurrence' (PRO) algorithm, in which each compound is consecutively removed from the corpus and the algorithm is used to deduce the correct sense for that compound using the set of compounds left behind. These tests show that the PRO algorithm can identify the correct relations for compounds, and the correct senses of those relations, with high precision. Section 6 compares our algorithm for compound disambiguation with one recently presented alternative, Rosario et al.'s (2002) rule-based system for the disambiguation of noun-noun compounds. The paper concludes with a discussion of future developments of the PRO algorithm.

2 Introduction to WordNet

In both our annotated corpus of 2,500 noun-noun compounds and our proportional relation selection algorithm we use WordNet (Miller, 1995). The basic unit of WordNet is the sense. Each word in WordNet is linked to a set of senses, with each sense identifying one particular meaning of that word. For example, the noun 'skin' has senses representing (i) the cutis or skin of human beings, (ii) the rind or peel of vegetables or fruit, (iii) the hide or pelt of an animal, (iv) a skin or bag used as a container for liquids, and so on. Each sense contains an identifying number and a 'gloss' (explaining what that sense means). Each sense is linked to its parent sense, which subsumes that sense as part of its meaning. For example, sense (i) of the word 'skin' (the cutis or skin of human beings) has a parent sense 'connective tissue' which contains that sense of skin and also contains the relevant sense of 'bone', 'muscle', and so on. Each parent sense has its own parents, which in turn have their own parent senses, and so on up to the (notional) root node of the WordNet hierarchy. This hierarchical structure allows computer programs to analyse the semantics of natural language expressions, by finding the senses of the words in a given expression and traversing the WordNet graph to make generalisations about the meanings of those words.

3 Corpus of Annotated Compounds

In this section we describe the construction of a corpus of noun-noun compounds annotated with the correct WordNet noun senses for constituent words, the correct semantic relation between those words, and the correct WordNet verb sense for that relation. In addition to providing a set of compounds to use as input for our compound disambiguation algorithm, one aim in constructing this corpus was to examine the relations that exist in naturally occurring noun-noun compounds. This follows from existing research on the relations that occur between noun-noun compounds (e.g. Gagné & Shoben, 1997). Gagné and her colleagues provide a set of 'thematic relations' (derived from relations proposed by, for example, Levi, 1978) which, they argue, cover the majority of semantic relations between modifier (first word) and head (second word) in noun-noun compounds. Table 1 shows the set of thematic relations proposed in Gagné & Shoben (1997). A side-effect of the construction of our corpus of noun-noun compounds was an assessment of the coverage and usefulness of this set of relations.

3.1 Procedure

The first step in constructing a corpus of annotated noun-noun compounds involved selection of a set of noun-noun compounds to classify. The source used was the set of noun-noun compounds

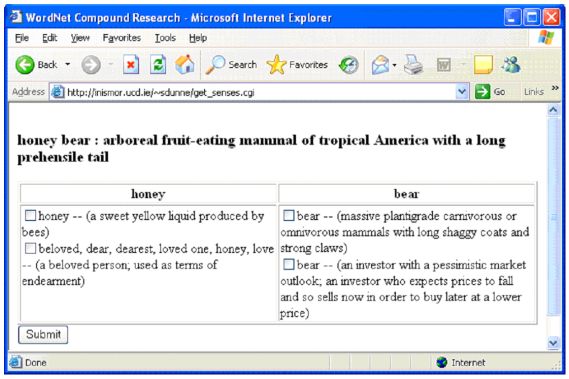


Figure 1: Selecting WordNet senses for nouns.

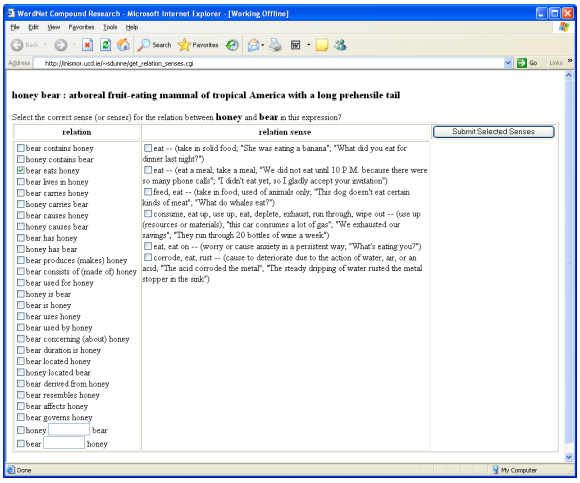


Figure 2: Selecting relation and relation senses.

defined in WordNet. Compounds from WordNet were used for two reasons. First, each compound had an associated gloss or definition written by the lexicographer who entered that compound into the corpus: this explains the relation between the two words in that compound. Sets of compounds from other sources would not have such associated definitions. Second, by using compounds from WordNet, we could guarantee that all constituent words of those compounds would also have entries in WordNet, ensuring their acceptability to our compound disambiguation algorithm. An initial list of over 40,000 two-word noun-noun compounds were extracted from WordNet version 2.0. From this list we selected a random subset of compounds and went through that set excluding all compounds using scientific latin (e.g. ocimum basilicum), idiomatic compounds (e.g. zero hour, ugli fruit), compounds containing proper nouns (e.g. Yangtze river), non-english compounds (e.g. faux pas), and chemical terminology (e.g. carbon dioxide).

The remaining compounds were placed in random order, and the third author annotated each compound with the WordNet noun senses of the constituent words, the semantic relation between those words, and the WordNet verb sense of that relation (again, with senses extracted from WordNet version 2.0). A web page was created for this annotation task, showing the annotator the compound to be annotated and the WordNet gloss (meaning) for that compound (see Figure 1). This page also showed the annotator the list of possible WordNet senses for the modifier noun and head noun in the compound, allowing the annotator to select the correct WordNet sense for each word. After selecting correct senses for the words in the compound, another page was presented (Figure 2) allowing the annotator to identify the correct semantic relation for that compound, and then to select the correct WordNet sense for the verb in that relation.

We began by assuming that Gagné & Shoben's (1997) set of 14 relations was complete and could account for all compounds being annotated. However, a preliminary test revealed some common relations (e.g., *eats*, *lives in*, *contains*, and *resembles*) that were not in Gagné & Shoben's set. These relations were therefore added to the list of relations we used. Various other less commonly-occuring relations were also observed. To allow for these other relations, a function was added to the web page allowing the annotator to enter the appropriate relation appearing in the form "noun (insert relation) modifier" and "modifier (insert relation) noun". They would then be shown the set of verb senses for that relation and asked to select the correct sense.

3.2 Results

Word sense, relation, and relation sense information was gathered for 2,500 compounds. Relation occurrence was well distributed across these compounds: there were 139 different relations used in the corpus. Frequency of these relations ranged widely: there were 86 relations that occured for just one compound in the corpus, and 53 relations that occurred more than once. For the relations that occured more than once in the corpus, the average number of occurrences was 46. Table 2 shows the 5 most frequent relations in the corpus: these 5 relations account for 54% of compounds. Note that 2 of the 5 relations in Table 2 (*head con-*

Table 2: 5 most frequent relations in the corpus.

relation	frequency	number of relation senses
head *used for* modifier	382	3
head *about* modifier	360	1
head *located* modifier	226	3
head *contains* modifier	217	3
head *resembles* modifier	169	1

tains modifier and *head resembles modifier*) are not listed in Gagné's set of taxonomic relations. This suggests that the taxonomy needs to be extended by the addition of further relations.

In addition to identifying the relations used in compounds in our corpus, we also identified the WordNet verb sense of each relation. In total 146 different relation senses occurred in the corpus. Most relations in the corpus were associated with just 1 relation sense. However, a significant minority of relations (29 relations, or 21% of all relations) had more than one relation sense; on average, these relations had three different senses each. Relations with more than one sense in the corpus tended to be the more frequently occurring relations: as Table 2 shows, of the 5 most frequent relations in the corpus, 3 were identified as having more than one relation sense. The two relations with the largest number of different relation senses occurring were *carry* (9 senses) and *makes* (8 senses). Table 3 shows the 3 most frequent senses for both relations. This diversity of relation senses suggests that Gagné's set of thematic relations may be too coarse-grained to capture distinctions between relations.

Table 3: Senses for relations *makes* and *carries*.

relation	relation sense gloss	example
Makes	bring forth or yield;	spice tree
Makes	cause to occur or exist;	smoke bomb
Makes	create or manufacture a man-made product;	cider mill
Carries	contain or hold, have within;	pocket watch
Carries	move while supporting, in a vehicle or one's hands;	passenger van
Carries	transmit or serve as the medium for transmission;	radio wave

4 Compound Disambiguation Algorithm

The previous section described the development of a corpus of associations between word-sense and relation data for a large set of noun-noun compounds. This section presents the 'Proportional Relation Occurrence' (PRO) algorithm which makes use of this information to deduce the correct relation for a given compound.

Our approach to compound disambiguation works by finding other compounds containing words from the same semantic categories as the words in the compound to be disambiguated: if a particular relation occurs frequently in those other compounds, that relation is probably also the correct relation for the compound in question. We take WordNet senses to represent semantic categories. Once the correct WordNet sense for a word has been identified, that word can placed a set of nested semantic categories: the category represented by that WordNet sense, by the parent sense (or hypernym) of that sense, the parent of that parent, and so on up to the (notional) root sense of WordNet (the semantic category which subsumes every other category in WordNet). Our algorithm uses the set of semantic categories for the words in a compound, and searches for other compounds containing words from any pair of those categories.

Figure 3 shows the algorithm in pseudocode. The algorithm uses a corpus of annotated noun-noun compounds and, to disambiguate a given compound, takes as input the correct WordNet sense for the modifier and head words of that compound, plus all hypernyms of those senses. The algorithm pairs each modifier sense with each head sense (lines 1 & 2 in Figure 3). For each sense-pair, the algorithm goes through the corpus of noun-noun compounds and extracts every compound whose modifier sense (or a hypernym of that sense) is equal to the modifier sense in the current sense-pair, and whose head sense (or a hypernym of that sense) is equal to the head sense in that pair (lines 5 to 8). The algorithm counts the number of times each relation occurs in that set of compounds, and assigns each relation a Proportional Relation Occurrence (PRO) score for that sense-pair (lines 10 to 12). The PRO score for a given relation R in a sense-pair S is a tuple with two components, as in Equation 1:

$$PRO(R,S) = \langle \frac{|R \cap S|}{|S|}, \frac{|R \cap S|}{|D|} \rangle. \quad (1)$$

The first term of this tuple is the proportion of times relation R occurs with sense-pair S (in other words, the conditional probability of relation R

Preconditions:
The entry for each compound C in corpus D contains:
$C_{modList}$ = sense + hypernym senses for modifier of C;
$C_{headList}$ = sense + hypernym senses for head of C;
C_{rel} = semantic relation of C;
$C_{relSense}$ = verb sense for semantic relation for C;

Input:
X = compound for which a relation is required;
$modList$ = sense + hypernym senses for modifier of X;
$headList$ = sense + hypernym senses for head of X;
$finalResultList$ = ();

Begin:
1 for each modifier sense $M \in modList$
2 for each head sense $H \in headList$
3 $relCount$ = ();
4 $matchCount$ = 0;
5 for each compound $C \in$ corpus D
6 if (($M \in C_{modList}$) and ($H \in C_{headList}$))
7 $relCount[C_{rel}] = relCount[C_{rel}] + 1$;
8 $matchCount = matchCount + 1$;
9 for each relation $R \in relCount$
10 $condProb = relCount[R]/matchCount$;
11 $jointProb = relCount([R]/|D|$;
12 $scoreTuple = (relProp, jointProb)$;
13 $prevScoreTuple = finalResultList[R]$;
14 if ($scoreTuple[1] > prevScoreTuple[1]$)
15 $finalResultList[R] = relSscoreTuple$;
16 if ($scoreTuple[1] = prevScoreTuple[1]$)
17 if ($scoreTuple[2] > prevScoreTuple[2]$)
18 $finalResultList[R] = scoreTuple$;
19 sort $finalResultList$ by relation score tuples;
20 return $finalResultList$;
End.

Figure 3: Compound disambiguation algorithm.

given sense-pair S); the second term is simply the proportion of times the relation co-occurs with the sense pair in the database of compounds D (in other words, the joint probability of relation R and sense-pair S). The algorithm compares the PRO score obtained for each relation R from the current sense-pair with the score obtained for that relation from any other sense-pair, using the first term of the score tuple as the main key for comparison (lines 14 and 15), and using the second term as a tie-breaker (lines 16 to 18). If the PRO score for relation R in the current sense-pair is greater than the PRO score obtained for that relation with some other sense pair (or if no previous score for the relation has been entered), the current PRO tuple is recorded for relation R. In this way the algorithm finds the maximum PRO score for each relation R across all possible sense-pairs for the compound in question. The algorithm returns a list of candidate relations for the compound, sorted by PRO score (lines 19 and 20). The relations at the front of that list (those with the highest PRO scores) are those most likely to be the correct relation for that compound.

Tests of this algorithm suggest that, in many cases, candidate relations for a given compound will be tied on the first term of their PRO score tuple. The use of the second score-tuple term is therefore an important part of the algorithm. For example, suppose that two competing relations for some compound have a proportional occurence of 1.0 (both relations occur in every occurrence of some sense-pair in the compound corpus). If the first relation occurs 20 times with its selected sense pair (i.e. there are 20 occurrences of the sense-pair in the corpus, and the relation occurs in each of those 20 occurrences), but the second relation only occurs occurs 2 times with its selected sense pair (i.e. there are 2 occurrences of that sense-pair in the corpus, and the relation occurs in each of those 2 occurrences), the first relation will be preferred over the second relation, because there is more evidence for that relation being the correct relation for the compound in question.

The algorithm in Figure 3 returns a list of candidate semantic relations for a given compound (returning relations such as 'head carries modifier' for the compound *vegetable truck* or 'modifier causes head' for the compound *storm damage*, for example). This algorithm can also return a list of relation senses for a given compound (returning the WordNet verb sense 'carries: moves while supporting, in a vehicle or one's hands' for the relation for the compound *vegetable truck* but the verb sense 'carries: transmits or serves as the medium for transmission' for the compound *radio wave*, for example). To return a list of relation senses rather than relations, we replace C_{rel} with $C_{relSense}$ throughout the algorithm in Figure 3. Section 5 describes a test of both versions of the algorithm.

5 Testing the Algorithm

To test the PRO algorithm it was implemented in a Perl program and applied to the corpus of compounds described in Section 3. We applied the program to two tasks: computing the correct relation for a given compound, and computing the correct relation sense for that compound. We used a 'leave-one-out' cross-validation approach, in which we consecutively removed each compound from the corpus (making it the 'query compound'), recorded the correct relation or relation sense for that compound, then passed the correct

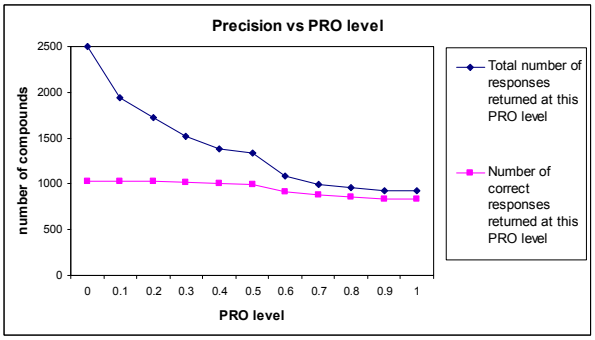


Figure 4: Graph of precision versus PRO value for returned relations

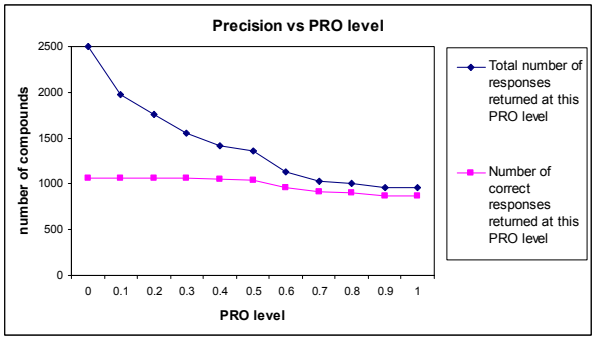


Figure 5: Graph of precision versus PRO value for returned relation senses

head and modifier senses of that query compound (plus their hypernyms), and the corpus of remaining compounds (excluding the query compound), to the Perl program. We carried out this process for each compound in the corpus. The result of this procedure was a list, for each compound, of candidate relations or relation senses sorted by PRO score.

We assessed the performance of the algorithm in two ways. We first considered the rank of the correct relation or relation sense for a given compound in the sorted list of candidate relations/relation senses returned by the algorithm. The algorithm always returned a large list of candidate relations or relation senses for each compound (over 100 different candidates returned for all compounds). In the relation selection task, the correct relation for a compound occurred in the first position in this list for 41% of all compounds (1,026 out of 2,500 compounds), and occured in one of the first 5 positions (in the top 5% of returned relations or relation senses) for 72% of all compounds (1780 compounds). In the relation-sense selection task, the correct relation for a compound occured in the first position in this list for 43% of all compounds, and occured in one of the first 5 positions for 74% of all compounds. This performance suggests that the algorithm is doing well in both tasks, given the large number of possible relations and relation senses available.

Our second assessment considered the *precision* and the *recall* of relation/relation senses returned by the algorithm at different proportional occurrence levels (different levels for the first term in PRO score tuples as described in Equation 1). For each proportional occurrence level between 0 and 1, we assumed that the algorithm would only return a relation or relation sense when the first relation in the list of candidate relations returned had a score at or above that level. We then counted the total number of compounds for which a response was returned at that level, and the total number of compounds for which a correct response was returned. The precision of the algorithm at a given PRO level was equal to the number of correct responses returned by the algorithm at that PRO level, divided by the total number of responses returned by the algorithm at that level. The recall of the algorithm at a given PRO level was equal to the number of correct responses returned by the algorithm at that level, divided by the total number of compounds in the database (the total number of compounds for which the algorithm could have returned a correct response).

Figure 4 shows the total number of responses, and the total number of correct responses, returned at each PRO level for the relation selection task. Figure 5 shows the same data for the relation-sense selection task. As both graphs show, as PRO level increases, the total number of responses returned by the algorithm declines, but the total number of correct responses does not fall significantly. For example, in the relation selection task, at a PRO level of 0 the algorithm return a response (selects a relation) for all 2,500 compounds, and approximately 1,000 of those responses are correct (the algorithm's precision at this level is 0.41). At a PRO level of 1, the algorithm return a response (selects a relation) for just over 900 compounds, and approximately 850 of those responses are correct (the algorithm's precision at this level is 0.92). A similar pattern is seen for the relation sense responses returned by the algorithm. These graphs show that with a PRO level around 1, the algorithm makes a relatively small number of errors when selecting the correct relation or relation sense for a

given compound (an error rate of less than 10%). The PRO algorithm thus has a high degree of precision in selecting relations for compounds.

As Figures 4 and 5 show, the number of correct responses returned by the PRO algorithm did not vary greatly across PRO levels. This means that the recall of the algorithm remained relatively constant across PRO levels: in the relation selection task, for example, recall ranged from 0.41 (at a PRO level of 0) to 0.35 (at a PRO level of 1). A similar pattern occurred in the relation-sense selection task.

6 Related Work

Various approaches to noun-noun compound disambiguation in the literature have used the semantic category membership of the constituent words in a compound to determine the relation between those words. Most of these use hand-crafted lexical hierarchies designed for particular semantic domains. We compare our algorithm for compound disambiguation with one recently presented alternative, Rosario, Hearst, and Fillmore's (2002) rule-based system for the disambiguation of noun-noun compounds in the biomedical domain.

6.1 Rule-based disambiguation algorithm

Rosario et al.'s (2002) general approach to noun-noun compound disambiguation is based, as ours is, on the semantic categories of the nouns making up a compound. Rosario et al. make use of the MeSH (Medical Subject Headings) hierarchy, which provides detailed coverage of the biomedical domain they focus on. Their analysis involves automatically extracting a corpus of noun-noun compounds from a large set of titles and abstracts from the MedLine collection of biomedical journal articles, and identifying the MeSH semantic categories under which the modifier and head words of those compounds fall. This analysis generates a set of *category pairs* for each compound (similar to our sense pairs), with each pair consisting of a MeSH category for the modifier word and a MeSH category for the head.

The aim of Rosario et al.'s analysis was to produce a set of rules which would link the MeSH category pair for a given compound to the correct semantic relation for that compound. Given such a set of rules, their algorithm for disabmiguating noun-noun compounds involves obtaining the MeSH category membership for the constituent words of the compounds to be disambiguated, forming category pairs, and looking up those category pairs in the list of *category-pair→relation* rules. If a rule was found linking the category pair for a given compound to a particular semantic relation, that relation was returned as the correct relation for the compound in question.

To produce a list of *category-pair→relation* rules, Rosario et al. first selected a set of category pairs occurring in their corpus of compounds. For each category pair, they manually examined 20% of the compounds falling under that category pair, paraphrasing the relation between the nouns in that compound by hand, and seeing if that relation was the same across all compounds under that category pair. If that relation was the same across all selected compounds, that category pair was recorded as a rule linked to the relation produced. If, on the other hand, several different relations were produced for a given category pair, analysis decended one level in the MeSH hierarchy, splitting that category pair into several subcategories. This repeated until a rule was produced assigning a relation to every compound examined. The rules produced by this process were then tested using a randomly chosen test set of 20% of compounds falling under each category pair, entirely distinct from the compound set used in rule construction, and applying the rules to those new compounds. An evaluator checked each compound to see whether the relation returned for that compound was an acceptable reflection of that compound's meaning. The results varied between 78.6% correct to 100% correct across the different category pairs.

6.2 Comparing the algorithms

In this section we first compare Rosario et al.'s algorithm for compound disambiguation with our own, and then compare the procedures used to assess those algorithms. While both algorithms are based on the association between category pairs (sense pairs) and semantic relations, they differ in that Rosario et al.'s algorithm uses a static list of manually-defined rules linking category pairs and semantic relations, while our PRO algorithm automatically and dynamically computes links between sense pairs and relations on the basis of proportional co-occurrence in a corpus of compounds. This gives our algorithm an advantage in terms of coverage: where Rosario et al.'s algorithm can

only disambiguate compounds whose constituent words match one of the *category-pair→relation* rules on their list, our algorithm should be able to apply to any compound whose constituent words are defined in WordNet. This also gives our algorithm an advantage in terms of extendability, in that while adding a new compound to the corpus of compounds used by Rosario et al. could potentially require the manual removal or re-definition of a number of *category-pair→relation* rules, adding a new compound to the annotated corpus used by our PRO algorithm requires no such intervention. Of course, the fact that Rosario et al.'s algorithm is based on a static list of rules linking categories and relations, while our algorithm dynamically computes such links, gives Rosario et al.'s algorithm a clear efficiency advantage. Improving the efficiency of the PRO algorithm, perhaps by automatically compiling a tree of associations between word senses and semantic relations and using that tree in compound disambiguation, is an important aim for future research.

Our second point of comparison concerns the procedures used to assess the two algorithms. In Rosario et al.'s assessment of their rule-based algorithm, an evaluator checked the relations returned by the algorithm for a set of compounds, and found that those relations were acceptable in a large proportion of cases (up to 100%). A problem with this procedure is that many compounds can fall equally under a number of different acceptable semantic relations. The compound *storm damage*, for example, is best defined by the relation *causes* ('damage caused by a storm'), but also falls under the relations *makes* ('damage made by a storm') and *derived from* ('damage derived from a storm'): most people would agree that these paraphrases all acceptably describe the meaning of the compound (Devereux & Costello, 2005). This means that, while the relations returned for compounds by Rosario et al.'s algorithm may have been judged acceptable for those compounds by the evaluator, they were not necessarily the most appropriate relations for those compounds: the algorithm could have returned other relations that would have been equally acceptable. In other words, Rosario et al.'s assessment procedure is somewhat weaker than the assessment procedure we used to test the PRO algorithm, in which there was one correct relation identified for each compound and the algorithm was taken to have performed correctly only if it returned that relation. One aim for future work is to apply the assessment procedure used by Rosario et al. to the PRO algorithm's output, asking an evaluator to assess the acceptability of the relations returned rather than simply counting the cases where the best relation was returned. This would provide a clearer basis for comparison between the algorithms.

6.3 Conclusions

In this paper we've described an algorithm for noun-noun compound disambiguation which automatically identifies the semantic relations and relation senses used in such compounds. We've given evidence showing that, coupled with a corpus of noun-noun compounds annotated with WordNet senses and semantic relations, this algorithm can identify the correct semantic relations for compounds with high precision. Unlike other approaches to automatic compound disambiguation which typically apply to particular specific domains, our algorithm is not domain specific and can identify relations for a random sample of noun-noun compounds drawn from the WordNet dictionary. Further, our algorithm is fully automatic: unlike other approaches, our algorithm does not require the manual construction of relation rules to produce successful compound disambiguation. In future work we hope to extend this algorithm to provide a more efficient algorithmic implementation, and also to apply the algorithm in areas such as the machine translation of noun-noun compounds, where the identification of semantic relations in compounds is a crucial step in the translation process.

References

B. Devereux & F. J. Costello. 2005. Investigating the Relations used in Conceptual Combination. *Artificial Intelligence Review*, 24(3–4): 489–515.

C. L. Gagné, & E. J. Shoben, E. 1997. Influence of thematic relations on the comprehension of modifier-noun combinations. *Journal of Experimental Psychology: Learning, Memory and Cognition*, 23: 71–87.

J. Justeson & S. Katz. 1995. Technical Terminology: Some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, 1–1: 9–27.

J. Levi. 1978. *The Syntax and Semantics of Complex Nominals*. New York: Academic Press.

G. Miller. 1995. *WordNet: A lexical database*. Communication of the ACM, 38(11), 39–41.

B. Rosario, M. Hearst, & C. Fillmore. 2002. The Descent of Hierarchy, and Selection in Relational Semantics. *Proceedings of ACL-02*: 247–254.

Automatically Extracting Nominal Mentions of Events with a Bootstrapped Probabilistic Classifier*

Cassandre Creswell[†] and **Matthew J. Beal**[‡] and **John Chen**[†]
Thomas L. Cornell[†] and **Lars Nilsson**[†] and **Rohini K. Srihari**[†‡]

[†]Janya, Inc.
1408 Sweet Home Road, Suite 1
Amherst NY 14228
{ccreswell,jchen,cornell,
lars,rohini}@janyainc.com

[‡]Dept. of Computer Science and Engineering
University at Buffalo
The State University of New York
Amherst NY 14260
mbeal@cse.buffalo.edu

Abstract

Most approaches to event extraction focus on mentions anchored in verbs. However, many mentions of events surface as noun phrases. Detecting them can increase the recall of event extraction and provide the foundation for detecting relations between events. This paper describes a weakly-supervised method for detecting nominal event mentions that combines techniques from word sense disambiguation (WSD) and lexical acquisition to create a classifier that labels noun phrases as denoting events or non-events. The classifier uses bootstrapped probabilistic generative models of the contexts of events and non-events. The contexts are the lexically-anchored semantic dependency relations that the NPs appear in. Our method dramatically improves with bootstrapping, and comfortably outperforms lexical lookup methods which are based on very much larger hand-crafted resources.

1 Introduction

The goal of information extraction is to generate a set of abstract information objects that represent the entities, events, and relations of particular types mentioned in unstructured text. For example, in a judicial domain, relevant event types might be ARREST, CHARGING, TRIAL, etc.

Although event extraction techniques usually focus on extracting mentions textually anchored by verb phrases or clauses, e.g. (Aone and Ramos-Santacruz, 2000), many event mentions, especially subsequent mentions of events that are the primary topic of a document, are referred to with nominals. Because of this, detecting nominal event mentions, like those in (1), can increase the recall of event extraction systems, in particular for the most important events in a document.[1]

(1) The slain journalist was a main organizer of **the massive demonstrations** that forced Syria to withdraw its troops from Lebanon last April, after Assad was widely accused of planning **Hariri's assassination** in **a February car bombing** that was similar to **today's blast**.

Detecting event nominals is also an important step in detecting relations between event mentions, as in the causal relation between the demonstrations and the withdrawal and the similarity relation between the bombing and the blast in (1).

Finally, detecting nominal events can improve detection and coreference of non-named mentions of non-event entities (e.g. persons, locations, and organizations) by removing event nominals from consideration as mentions of entities.

Current extraction techniques for verbally-anchored events rest on the assumption that most verb phrases denote eventualities. A system to extract untyped event mentions can output all constituents headed by a non-auxiliary verb with a filter to remove instances of *to be, to seem*, etc. A statistical or rule-based classifier designed to detect event mentions of specific types can then be applied to filter these remaining instances. Noun phrases, in contrast, can be used to denote anything—eventualities, entities, abstractions, and only some are suitable for event-type filtering.

* This work was supported in part by SBIR grant FA8750-05-C-0187 from the Air Force Research Laboratory (AFRL)/IFED.

[1] For example, in the 2005 Automatic Content Extraction training data, of the 5,349 event mentions, over 35% (1934) were nominals.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 168–175,
Sydney, July 2006.

1.1 Challenges of nominal event detection

Extraction of nominal mentions of events encompasses many of the fundamental challenges of natural language processing. Creating a general purpose lexicon of all potentially event-denoting terms in a language is a labor-intensive task. On top of this, even utilizing an existing lexical resource like WordNet requires sense disambiguation at run-time because event nominals display the full spectrum of sense distinction behaviors (Copestake and Briscoe, 1995), including idiosyncratic polysemy, as in (2); constructional polysemy, as in (3); coactivation, (4); and copredication, as in (5).

(2) a. On May 30 a group of Iranian mountaineers hoisted the Iranian tricolor on **the summit**.

b. EU Leaders are arriving here for **their two-day summit** beginning Thursday.

(3) Things are getting back to normal in the Baywood Golf Club after **a chemical spill**[=event]. Clean-up crews said **the chemical spill**[=result] was 99 percent water and shouldn't cause harm to area residents.

(4) Managing partner Naimoli said he wasn't concerned about recent media **criticism**.

(5) The **construction** lasted 30 years and was inaugurated in the presence of the king in June 1684.

Given the breadth of lexical sense phenomena possible with event nominals, no existing approach can address all aspects. Lexical lookup, whether using a manually- or automatically-constructed resource, does not take context into consideration and so does not allow for vagueness or unknown words. Purely word-cooccurrence-based approaches (e.g. (Schütze, 1998)) are unsuitable for cases like (3) where both senses are possible in a single discourse. Furthermore, most WSD techniques, whether supervised or unsupervised, must be retrained for each individual lexical item, a computationally expensive procedure both at training and run time. To address these limitations, we have developed a technique which combines automatic lexical acquisition and sense disambiguation into a single-pass weakly-supervised algorithm for detecting event nominals.

The remainder of this paper is organized as follows: Section 2 describes our probabilistic classifier. Section 3 presents experimental results of this model, assesses its performance when bootstrapped to increase its coverage, and compares it to a lexical lookup technique. We describe related work in Section 4 and present conclusions and implications for future work in Section 5.

2 Weakly-supervised, simultaneous lexical acquisition and disambiguation

In this section we present a computational method that learns the distribution of context patterns that correlate with event vs. non-event mentions based on unambiguous seeds. Using these seeds we build two Bayesian probabilistic generative models of the data, one for non-event nominals and the other for event nominals. A classifier is then constructed by comparing the probability of a candidate instance under each model, with the winning model determining the classification. In Section 3 we show that this classifier's coverage can be increased beyond the initial labeled seed set by automatically selecting additional seeds from a very large unlabeled, parsed corpus.

The technique proceeds as follows. First, two lexicons of seed terms are created by hand. One lexicon includes nominal terms that are highly likely to unambiguously denote events; the other includes nominal terms that are highly likely to unambiguously denote anything other than events. Then, a very large corpus (>150K documents) is parsed using a broad-coverage dependency parser to extract all instantiations of a core set of semantic dependency relations, including verb-logical subject, verb-logical object, subject-nominal predicate, noun phrase-appositive-modifier, etc.

Format of data: Each instantiation is in the form of a dependency triple, (w_a, R, w_b), where R is the relation type and where each argument is represented just by its syntactic head, w_n. Each partial instantiation of the relation—i.e. either w_a or w_b is treated as a wild card $*$ that can be filled by any term—becomes a feature in the model. For every common noun term in the corpus that appears with at least one feature (including each entry in the seed lexicons), the times it appears with each feature are tabulated and stored in a matrix of counts. Each column of the matrix represents a feature, e.g. (*occur*,Verb-Subj, $*$); each row represents an individual term,[2] e.g. *murder*; and each entry is the number of times a term appeared with the feature in the corpus, i.e. as the instantiation of $*$. For each row, if the corresponding term appears in a lexicon it is given that designation, i.e. EVENT or NONEVENT, or if it does not appear in either lexicon, it is left unlabeled.

[2] A term is any common noun whether it is a single or multiword expression.

Probabilistic model: Here we present the details of the EVENT model—the computations for the NONEVENT model are identical. The probability model is built using a set of seed words labeled as EVENTs and is designed to address two desiderata: **(I)** the EVENT model should assign high probability to an unlabeled vector, $\mathbf{v}$, if its features (as recorded in the count matrix) are similar to the vectors of the EVENT seeds; **(II)** each seed term s should contribute to the model in proportion to its prevalence in the training data.[3] These desiderata can be incorporated naturally into a mixture model formalism, where there are as many components in the mixture model as there are EVENT seed terms. Desideratum **(I)** is addressed by having each component of the mixture model assigning a multinomial probability to the vector, $\mathbf{v}$. For the ith mixture component built around the ith seed, $s^{(i)}$, the probability is

$$p(\mathbf{v}|\mathbf{s}^{(i)}) = \prod_{f=1}^{F} \left(\overline{s}_f^{(i)}\right)^{v_f} ,$$

where $\overline{s}_f^{(i)}$ is defined as the proportion of the times the seed was seen with feature f compared to the number of times the seed was seen with any feature $f' \in F$. Thus $\overline{s}_f^{(i)}$ is simply the (i, f)th entry in a row-sum normalized count matrix,

$$\overline{s}_f^{(i)} = \frac{s_f^{(i)}}{\sum_{f'=1}^{F} s_{f'}^{(i)}} .$$

Desideratum **(II)** is realized using a mixture density by forming a weighted mixture of the above multinomial distributions from all the provided seeds $i \in \mathcal{E}$. The weighting of the ith component is fixed to be the ratio of the number of occurrences of the ith EVENT seed, denoted $|\mathbf{s}^{(i)}|$, to the total number of all occurrences of event seed words. This gives more weight to more prevalent seed words:

$$p(\mathbf{s}^{(i)}) = \frac{|\mathbf{s}^{(i)}|}{\sum_{i' \in \mathcal{E}} |\mathbf{s}^{(i')}|} .$$

The EVENT generative probability is then:

$$p(\mathbf{v}|\text{EVENT}) = \sum_{i \in \mathcal{E}} \left[p(\mathbf{s}^{(i)}) \cdot p(\mathbf{v}|\mathbf{s}^{(i)}) \right] .$$

An example of the calculation for a model with just two event seeds and three features is given in Figure 1. A second model is built from the non-event seeds as well, and a corresponding probability $p(\mathbf{v}|\text{NONEVENT})$ is computed. The following difference (*log odds-ratio*)

$$d(\mathbf{v}) = \log p(\mathbf{v}|\text{EVENT}) - \log p(\mathbf{v}|\text{NONEVENT})$$

is then calculated. An instance v encoded as the vector $\mathbf{v}$ is labeled as EVENT or NONEVENT by examining the sign of $d(\mathbf{v})$. A positive difference $d(\mathbf{v})$ classifies v as EVENT; a negative value of $d(\mathbf{v})$ classifies v as NONEVENT. Should $d = 0$ the classifier is considered undecided and abstains.

		f_1	f_2	f_3
event seed vector	$\mathbf{s}^{(1)}$	3	1	8
event seed vector	$\mathbf{s}^{(2)}$	4	6	1
unlabeled mention vector	$\mathbf{v}$	2	0	7

$$p(\mathbf{v}|\text{event}) = \frac{12}{23} \cdot \left(\frac{3}{12}\right)^2 \left(\frac{1}{12}\right)^0 \left(\frac{8}{12}\right)^7 + \frac{11}{23} \cdot \left(\frac{4}{11}\right)^2 \left(\frac{6}{11}\right)^0 \left(\frac{1}{11}\right)^7 = 0.0019$$

Figure 1: Example of calculating the probability of unlabeled instance $\mathbf{v}$ under the event distribution composed of two event seeds $s^{(1)}$ and $s^{(2)}$.

Each test instance is composed of a term and the dependency triples it appears with in context in the test document. Therefore, an instance can be classified by **(i: *word*)**: Find the unlabeled feature vector in the training data corresponding to the term and apply the classifier to that vector, i.e. classify the instance based on the term's behavior summed across many occurrences in the training corpus; **(ii: *context*)**: Classify the instance based only on its immediate test context vector; or **(iii: *word+context*)**: For each model, multiply the probability information from the word vector (=i) and the context vector (=ii). In our experiments, all terms in the test corpus appeared at least once (80% appearing at least 500 times) in the training corpus, so there were no cases of unseen terms—not suprising with a training set 1,800 times larger than the test set. However, the ability to label an instance based only on its immediate context means that there is a backoff method in the case of unseen terms.

3 Experimental Results

3.1 Training, test, and seed word data

In order to train and test the model, we created two corpora and a lexicon of event and non-event seeds. The training corpus consisted of 156,000 newswire documents, ∼100 million words, from the Foreign Broadcast Information Service, Lexis

[3]The counts used here are the number of times a term is seen with any feature in the training corpus because the indexing tool used to calculate counts does not keep track of which instances appeared simultaneously with more than one feature. We do not expect this artifact to dramatically change the relative seed frequencies in our model.

Nexis, and other online news archives. The corpus was parsed using Janya's information extraction application, Semantex, which creates both shallow, non-recursive parsing structures and dependency links, and all (w_i, R, w_j) statistics were extracted as described in Section 2. From the 1.9 million patterns, $(w_i, R, *)$ and $(*, R, w_j)$ extracted from the corpus, the 48,353 that appeared more than 300 times were retained as features.

The test corpus was composed of 77 additional documents ($\sim$56K words), overlapping in time and content but not included in the training set. These were annotated by hand to mark event nominals. Specifically, every referential noun phrase headed by a non-proper noun was considered for whether it denoted an achievement, accomplishment, activity, or process (Parsons, 1990). Noun heads denoting any of these were marked as EVENT, and all others were left unmarked.

All documents were first marked by a junior annotator, and then a non-blind second pass was performed by a senior annotator (first author). Several semantic classes were difficult to annotate because they are particularly prone to coactivation, including terms denoting financial acts, legal acts, speech acts, and economic processes. In addition, for terms like *mission, plan, duty, tactic, policy*, it can be unclear whether they are hyponyms of EVENT or another abstract concept. In every case, however, the mention was labeled as an event or non-event depending on whether its use in that context appeared to be more or less event-like, respectively. Tests for the "event-y"ness of the context included whether an unambiguous word would be an acceptable substitute there (e.g. *funds* [=only non-event] for *expenditure* [either]).

To create the test data, the annotated documents were also parsed to automatically extract all common noun-headed NPs and the dependency triples they instantiate. Those with heads that aligned with the offsets of an event annotation were labeled as events; the remainder were labeled as non-events. Because of parsing errors, about 10% of annotated event instances were lost, that is remained unlabeled or were labeled as non-events. So, our results are based on the set of recoverable event nominals as a subset of all common-noun headed NPs that were extracted. In the test corpus there were 9,381 candidate instances, 1,579 (17%) events and 7,802 (83%) non-events. There were 2,319 unique term types; of these, 167 types (7%) appeared both as event tokens and non-event tokens. Some sample ambiguous terms include: *behavior, attempt, settlement, deal, violation, progress, sermon, expenditure*.

We constructed two lexicons of nominals to use as the seed terms. For events, we created a list of **95** terms, such as *election, war, assassination, dismissal*, primarily based on introspection combined with some checks on individual terms in WordNet and other dictionaries and using Google searches to judge how "event-y" the term was.

To create a list of non-events, we used WordNet and the British National Corpus. First, from the set of all lexemes that appear in only one synset in WordNet, all nouns were extracted along with the topmost hypernym they appear under. From these we retained those that both appeared on a lemmatized frequency list of the 6,318 words with more than 800 occurrences in the whole 100M-word BNC (Kilgarriff, 1997) and had one of the hypernyms GROUP, PSYCHOLOGICAL, ENTITY, POSSESSION. We also retained select terms from the categories STATE and PHENOMENON were labeled non-event seeds. Examples of the **295** non-event seeds are *corpse, electronics, bureaucracy, airport, cattle*.

Of the 9,381 test instances, 641 (6.8%) had a term that belonged to the seed list. With respect to types, 137 (5.9%) of the 2,319 term types in the test data also appeared on the seed lists.

3.2 Experiments

Experiments were performed to investigate the performance of our models, both when using original seed lists, and also when varying the content of the seed lists using a bootstrapping technique that relies on the probabilistic framework of the model. A 1,000-instance subset of the 9,381 test data instances was used as a validation set; the remaining 8,381 were used as evaluation data, on which we report all results (with the exception of Table 3 which is on the full test set).

EXP1: Results using original seed sets Probabilistic models for non-events and events were built from the full list of 295 non-event and 95 event seeds, respectively, as described above.

Table 1 (top half: original seed set) shows the results over the 8,381 evaluation data instances when using the three classification methods described above: (i) word, (ii) context, and (iii) word+context. The first row (ALL) reports scores where all undecided responses are marked as in-

			EVENT			NONEVENT			TOTAL			AVERAGE
		Input Vector	Correct	Acc (%)	Att (%)	Correct	Acc (%)	Att (%)	Correct	Acc (%)	Att (%)	Acc (%)
ORIGINAL SEED SET	ALL	word	1236	87.7	100.0	4217	60.5	100.0	5453	65.1	100.0	74.1
		context	627	44.5	100.0	2735	39.2	100.0	3362	40.1	100.0	41.9
		word+context	1251	**88.8**	100.0	4226	**60.6**	100.0	5477	**65.4**	100.0	**74.7**
	FAIR	word	1236	89.3	98.3	4217	60.7	99.6	5453	65.5	99.4	75.0
		context	627	69.4	64.2	2735	62.5	62.8	3362	63.6	63.0	65.9
		word+context	1251	**89.3**	99.5	4226	**60.7**	99.9	5477	**65.5**	99.8	**75.0**
BOOTSTRAPPED SEED SET	ALL	word	1110	78.8	100.0	5517	79.1	100.0	6627	79.1	100.0	79.0
		context	561	39.8	100.0	2975	42.7	100.0	3536	42.2	100.0	41.3
		word+context	1123	**79.8**	100.0	5539	**79.4**	100.0	6662	**79.5**	100.0	**79.6**
	FAIR	word	1110	80.2	98.3	5517	79.4	99.6	6627	79.5	99.4	79.8
		context	561	62.1	64.2	2975	67.9	62.8	3536	66.9	63.0	65.0
		word+context	1123	**80.2**	99.5	5539	**79.5**	99.9	6662	**79.7**	99.8	**79.9**
		LEX 1	1114	79.1	100.0	5074	72.8	100.0	6188	73.8	100.0	75.9
		total counts	1408			6973			8381			

Table 1: **(EXP1, EXP3)** Accuracies of classifiers in terms of correct classifications, % correct, and % attempted (if allowed to abstain), on the **evaluation test set**. **(Row 1)** Classifiers built from original seed set of size $(295, 95)$; **(Row 2)** Classifiers built from 15 iterations of bootstrapping; **(Row 3)** Classifier built from Lexicon 1. Accuracies in **bold** are those plotted in related Figures 2, 3(a) and 3(b).

correct. In the second row (FAIR), undecided answers ($d = 0$) are left out of the total, so the number of correct answers stays the same, but the percentage of correct answers increases.[4] Scores are measured in terms of accuracy on the EVENT instances, accuracy on the NONEVENT instances, TOTAL accuracy across all instances, and the simple AVERAGE of accuracies on non-events and events (last column). The AVERAGE score assumes that performance on non-events and events is equally important to us.

¿From EXP1, we see that the behavior of a term across an entire corpus is a better source of information about whether a particular instance of that term refers to an event than its immediate context. We can further infer that this is because the immediate context only provides definitive evidence for the models in 63.0% of cases; when the context model is not penalized for indecision, its accuracy improves considerably. Nonetheless, in combination with the word model, immediate context does not appear to provide much additional information over only the word. In other words, based only on a term's distribution in the past, one can make a reasonable prediction about how it will be used when it is seen again. Consequently, it seems that a well-constructed, i.e. domain customized, lexicon can classify nearly as well as a method that also takes context into account.

EXP2: Results on ACE 2005 event data In addition to using the data set created specifically for this project, we also used a subset of the annotated training data created for the ACE 2005 Event Detection and Recognition (VDR) task. Because only event mentions of specific types are marked in the ACE data, only recall of ACE event nominals can be measured rather than overall recall of event nominals and accuracy on non-event nominals. Results on the 1,934 nominal mentions of events (omitting cases of $d = 0$) are shown in Table 2. The performance of the hand-crafted Lexicon 1 on the ACE data, described in Section 3.3 below, is also included.

Input Vector	Acc (%)	Att (%)
word	96.1	97.2
context	72.8	63.1
word+context	95.5	98.9
LEX 1	76.5	100.0

Table 2: **(EXP2)** Results on ACE event nominals: %correct (accuracy) and %attempted, for our classifiers and LEX 1.

The fact that our method performs somewhat better on the ACE data than on our own data, while the lexicon approach is worse (7 points higher vs. 3 points lower, respectively) can likely be explained by the fact that in creating our introspective seed set for events, we consulted the annotation manual for ACE event types and attempted to include in our list any unambiguous seed terms that fit those types.

EXP3: Increasing seed set via Bootstrapping There are over 2,300 unlabeled vectors in the training data that correspond to the words that appear as lexical heads in the test data. These unlabeled training vectors can be powerfully leveraged using a simple bootstrapping algorithm to improve the individual models for non-events and events, as follows: **Step 1:** For each vector $\mathbf{v}$ in the unlabeled portion of training data, row-sum normalize

[4] Note that Att(%) does not change with bootstrapping—an artifact of the sparsity of certain feature vectors in the training and test data, and not the model's constituents seeds.

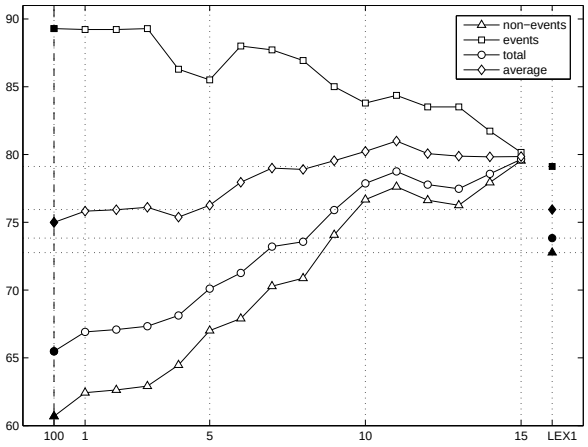


Figure 2: Accuracies vs. iterations of bootstrapping. Bold symbols on left denote classifier built from initial $(295, 95)$ seeds; and bold (disconnected) symbols at right are LEX 1.

it to produce $\tilde{\mathbf{v}}$ and compute a normalized measure of confidence of the algorithm's prediction, given by the magnitude of $d(\tilde{\mathbf{v}})$. **Step 2:** Add those vectors most confidently classified as either non-events or events to the seed set for non-events or events, according to the sign of $d(\tilde{\mathbf{v}})$. **Step 3:** Recalculate the model based on the new seed lists. **Step 4:** Repeat Steps 1–3 until either no more unlabeled vectors remain or the validation accuracy no longer increases.

In our experiments we added vectors to each model such that the ratio of the size of the seed sets remained constant, i.e. 50 non-events and 16 events were added at each iteration. Using our validation set, we determined that the bootstrapping should stop after 15 iterations (despite continuing for 21 iterations), at which point the average accuracy leveled out and then began to drop. After 15 iterations the seed set is of size $(295, 95) + (50, 16) \times 15 = (1045, 335)$. Figure 2 shows the change in the accuracy of the model as it is bootstrapped through 15 iterations.

TOTAL accuracy improves with bootstrapping, despite EVENT accuracy decreasing, because the test data is heavily populated with non-events, whose accuracy increases substantially. The AVERAGE accuracy also increases, which proves that bootstrapping is doing more than simply shifting the bias of the classifier to the majority class. The figure also shows that the final bootstrapped classifier comfortably outperforms Lexicon 1, impressive because the lexicon contains at least 13 times more terms than the seed lists.

EXP4: Bootstrapping with a reduced number of seeds The size of the original seed lists were chosen somewhat arbitrarily. In order to determine whether similar performance could be obtained using fewer seeds, i.e. less human effort, we experimented with reducing the size of the seed lexicons used to initialize the bootstrapping.

To do this, we randomly selected a fixed fraction, $f\%$, of the $(295, 95)$ available event and non-event seeds, and built a classifier from this subset of seeds (and discarded the remaining seeds). We then bootstrapped the classifier's models using the 4-step procedure described above, using candidate seed vectors from the unlabeled training corpus, and incrementing the number of seeds until the classifier consisted of $(295, 95)$ seeds. We then performed 15 additional bootstrapping iterations, each adding $(50, 16)$ seeds. Since the seeds making up the initial classifier are chosen stochastically, we repeated this entire process 10 times and report in Figures 3(a) and 3(b) the mean of the total and average accuracies for these 10 folds, respectively. Both plots have five traces, with each trace corresponding the fraction $f = (20, 40, 60, 80, 100)\%$ of labeled seeds used to build the initial models. As a point of reference, note that initializing with 100% of the seed lexicon corresponds to the first point of the traces in Figure 2 (where the x-axis is marked with $f = 100\%$).

Interestingly, there is no discernible difference in accuracy (total or average) for fractions f greater than 20%. However, upon bootstrapping we note the following trends. First, Figure 3(b) shows that using a larger initial seed set increases the maximum achievable accuracy, but this maximum occurs after a greater number bootstrapping iterations; indeed the maximum for 100% is achieved at 15 (or greater) iterations. This reflects the difference in rigidity of the initial models, with smaller initial models more easily misled by the seeds added by bootstrapping. Second, the final accuracies (total and average) are correlated with the initial seed set size, which is intuitively satisfying. Third, it appears from Figure 3(a) that the total accuracy at the model size (295,95) (or 100%) is in fact *anti*-correlated with the size of the initial seed set, with 20% performing best. This is correct, but highlights the sometimes misleading interpretation of the total accuracy: in this case the model is defaulting to classifying anything as a non-event (the majority class), and has a considerably impoverished event model.

If one wants to do as well as Lexicon 1 after 15 iterations of bootstrapping then one needs at least

	EVENT		NONEVENT		TOTAL		AVERAGE
	Corr	(%)	Corr	(%)	Corr	(%)	(%)
LEX 1	1256	**79.5**	5695	**73.0**	6951	**74.1**	**76.3**
LEX 2	1502	95.1	4495	57.6	5997	63.9	76.4
LEX 3	349	22.1	7220	92.5	7569	80.7	57.3
Total	1579		7802		9381		

Table 3: Accuracy of several lexicons, showing number and percentage of correct classifications on the **full test set**.

an initial seed set of size 60%. An alternative is to perform fewer iterations, but here we see that using 100% of the seeds comfortably achieves the highest total and average accuracies anyway.

3.3 Comparison with existing lexicons

In order to compare our weakly-supervised probabilistic method with a lexical lookup method based on very large hand-created lexical resources, we created three lexicons of event terms, which were used as very simple classifiers of the test data. If the test instance term belongs to the lexicon, it is labeled EVENT; otherwise, it is labeled as NONEVENT. The results on the full test set using these lexicons are shown in Table 3.

Lex 1 5,435 entries from NomLex (Macleod et al., 1998), FrameNet(Baker et al., 1998), CELEX (CEL, 1993), Timebank(Day et al., 2003).

Lex 2 13,659 entries from WordNet 2.0 hypernym classes EVENT, ACT, PROCESS, COGNITIVE PROCESS, & COMMUNICATION combined with Lex 1.

Lex 3 Combination of pre-existing lexicons in the information extraction application from WordNet, Oxford Advanced Learner's Dictionary, etc.

As shown in Tables 1 and 3, the relatively knowledge-poor method developed here using around 400 seeds performs well compared to the use of the much larger lexicons. For the task of detecting nominal events, using Lexicon 1 might be the quickest practical solution. In terms of extensibility to other semantic classes, domains, or languages lacking appropriate existing lexical resources, the advantage of our trainable method is clear. The primary requirement of this method is a dependency parser and a system user-developer who can provide a set of seeds for a class of interest and its complement. It should be possible in the next few years to create a dependency parser for a language with no existing linguistic resources (Klein and Manning, 2002). Rather than having to spend the considerable person-years it takes to create resources like FrameNet, CELEX, and WordNet, a better alternative will be to use weakly-supervised semantic labelers like the one described here.

4 Related Work

In recent years an array of new approaches have been developed using weakly-supervised techniques to train classifiers or learn lexical classes or synonyms, e.g. (Mihalcea, 2003; Riloff and Wiebe, 2003). Several approaches make use of dependency triples (Lin, 1998; Gorman and Curran, 2005). Our vector representation of the behavior of a word type across all its instances in a corpus is based on Lin (1998)'s DESCRIPTION OF A WORD.

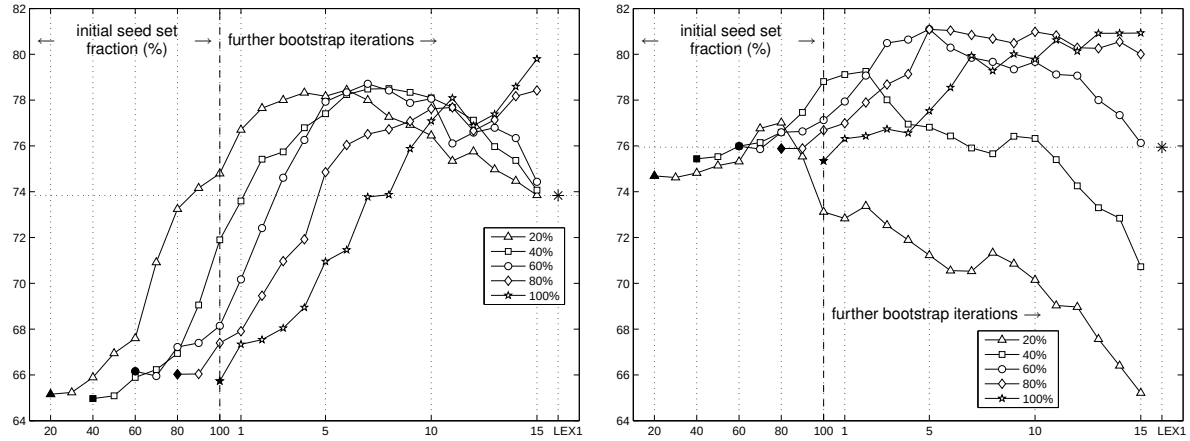
Yarowsky (1995) uses a conceptually similar technique for WSD that learns from a small set of seed examples and then increases recall by bootstrapping, evaluated on 12 idiosyncratically polysemous words. In that task, often a *single* disambiguating feature can be found in the context of a polysemous word instance, motivating his use of the decision list algorithm. In contrast, the goal here is to learn how event-like or non-event-like a *set* of contextual features together are. We do not expect that many individual features correlate unambiguously with references to events (or non-events), only that the presence of certain features make an event interpretation more or less likely. This justifies our probabilistic Bayesian approach, which performs well given its simplicity.

Thelen and Riloff (2002) use a bootstrapping algorithm to learn semantic lexicons of nouns for six semantic categories, one of which is EVENTS. For events, only 27% of the 1,000 learned words are correct. Their experiments were on a much smaller scale, however, using the 1,700 document MUC-4 data as a training corpus and using only 10 seeds per category.

Most prior work on event nominals does not try to classify them as events or non-events, but instead focuses on labeling the argument roles based on extrapolating information about the argument structure of the verbal root (Dahl et al., 1987; Lapata, 2002; Pradhan et al., 2004). Meyers, et al. (1998) describe how to extend a tool for extraction of verb-based events to corresponding nominalizations. Hull and Gomez (1996) design a set of rule-based algorithms to determine the sense of a nominalization and identify its arguments.

5 Conclusions

We have developed a novel algorithm for labeling nominals as events that combines WSD and lexical acquisition. After automatically bootstrapping the seed set, it performs better than static lexicons many times the original seed set size. Also,



(a) Total Accuracy (b) Average Accuracy

Figure 3: Accuracies of classifiers built from different-sized initial seed sets, and then bootstrapped onwards to the equivalent of 15 iterations as before. Total **(a)** and Average **(b)** accuracies highlight different aspects of the bootstrapping mechanism. Just as in Figure 2, the initial model is denoted with a bold symbol in the left part of the plot. Also for reference the relevant Lexicon 1 accuracy (LEX 1) is denoted with a $*$ at the far right.

it is more robust than lexical lookup as it can also classify unknown words based on their immediate context and can remain agnostic in the absence of sufficient evidence.

Future directions for this work include applying it to other semantic labeling tasks and to domains other than general news. An important unresolved issue is the difficulty of formulating an appropriate seed set to give good coverage of the complement of the class to be labeled without the use of a resource like WordNet.

References

C. Aone and M. Ramos-Santacruz. 2000. REES: A large-scale relation and event extraction system. In *6th ANLP*, pages 79–83.

C. F. Baker, C. J. Fillmore, and J. B. Lowe. 1998. The Berkeley FrameNet project. In *Proc. COLING-ACL*.

Centre of Lexical Information, Nijmegen, 1993. *CELEX English database*, E25, online edition.

A. Copestake and T. Briscoe. 1995. Semi-productive polysemy and sense extension. *Journal of Semantics*, 12:15–67.

D. Dahl, M. Palmer, and R. Passonneau. 1987. Nominalizations in PUNDIT. In *Proc. of the 25th ACL*.

D. Day, L. Ferro, R. Gaizauskas, P. Hanks, M. Lazo, J. Pustejovsky, R. Sauri, A. See, A. Setzer, and B. Sundheim. 2003. The TIMEBANK corpus. In *Corpus Linguistics 2003*, Lancaster UK.

J. Gorman and J. Curran. 2005. Approximate searching for distributional similarity. In *Proc. of the ACL-SIGLEX Workshop on Deep Lexical Acquisition*, pages 97–104.

R. Hull and F. Gomez. 1996. Semantic interpretation of nominalizations. In *Proc. of the 13th National Conf. on Artificial Intelligence*, pages 1062–1068.

A. Kilgarriff. 1997. Putting frequencies in the dictionary. *Int'l J. of Lexicography*, 10(2):135–155.

D. Klein and C. Manning. 2002. A generative constituent-context model for improved grammar induction. In *Proc. of the 40th ACL*.

M. Lapata. 2002. The disambiguation of nominalisations. *Computational Linguistics*, 28(3):357–388.

D. K. Lin. 1998. Automatic retrieval and clustering of similar words. In *Proc. of COLING-ACL '98*.

C. Macleod, R. Grishman, A. Meyers, L. Barrett, and R. Reeves. 1998. NOMLEX: A lexicon of nominalizations. In *Proc. of EURALEX'98*.

A. Meyers, C. Macleod, R. Yangarber, R. Grishman, L. Barrett, and R. Reeves. 1998. Using NOMLEX to produce nominalization patterns for information extraction. In *Proc. of the COLING-ACL Workshop on the Computational Treatment of Nominals*.

R. Mihalcea. 2003. Unsupervised natural language disambiguation using non-ambiguous words. In *Proc. of Recent Advances in Natural Language Processing*, pages 387–396.

T. Parsons. 1990. *Events in the Semantics of English*. MIT Press, Boston.

S. Pradhan, H. Sun, W. Ward, J. Martin, and D. Jurafsky. 2004. Parsing arguments of nominalizations in English and Chinese. In *Proc. of HLT-NAACL*.

E. Riloff and J. Wiebe. 2003. Learning extraction patterns for subjective expressions. In *Proc. EMNLP*.

H. Schütze. 1998. Automatic word sense disambiguation. *Computational Linguistics*, 24(1):97–124.

M. Thelen and E. Riloff. 2002. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *Proc. of EMNLP*.

D. Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proc. of the 33rd ACL*, pages 189–196.

A Bio-inspired Approach for Multi-Word Expression Extraction

Jianyong Duan, Ruzhan Lu
Weilin Wu, Yi Hu
Department of Computer Science
Shanghai Jiao Tong University
Shanghai, 200240, P.R. China
duanjy@hotmail.com
{lu-rz,wl-wu,huyi}@cs.sjtu.edu.cn

Yan Tian
School of Foreign Languages
Department of Computer Science
Shanghai Jiao Tong University
Shanghai, 200240, P.R. China
tianyan@sjtu.edu.cn

Abstract

This paper proposes a new approach for Multi-word Expression (MWE)extraction on the motivation of gene sequence alignment because textual sequence is similar to gene sequence in pattern analysis. Theory of Longest Common Subsequence (LCS) originates from computer science and has been established as affine gap model in Bioinformatics. We perform this developed LCS technique combined with linguistic criteria in MWE extraction. In comparison with traditional n-gram method, which is the major technique for MWE extraction, LCS approach is applied with great efficiency and performance guarantee. Experimental results show that LCS-based approach achieves better results than n-gram.

1 Introduction

Language is under continuous development. People enlarge vocabulary and let words carry more meanings. Meanwhile the language also develops larger lexical units to carry specific meanings; specifically MWE's, which include compounds, phrases, technical terms, idioms and collocations, etc. The MWE has relatively fixed pattern because every MWE denotes a whole concept. In computational view, the MWE repeats itself constantly in corpus(Taneli,2003).

The extraction of MWE plays an important role in several areas, such as machine translation (Pascale,1997), information extraction (Kalliopi,2000) etc. On the other hand, there is also a need for MWE extraction in a much more widespread scenario namely that of human translation and technical writing. Many efforts have been devoted to the study of MWE extraction (Beatrice,2003; Ivan,2002; Jordi,2001). These statistical methods detect MWE by frequency of candidate patterns. Linguistic information as a filtering strategy is also performed to improve precision by ranking their candidates (Violeta,2003; Stefan,2004; Arantza,2002). Some measures based on advance statistical methods are also used, such as mutual expectation with single statistic model (Paul,2005),C-value/NC-value method (Katerina,2000),etc.

Frequent information is the original data for further MWE extraction. Most approaches adopt n-gram technique(Daniel,1977; Satanjeev,2003; Makoto,1994). n-gram concerns about one sequence for each time. Every sequence can be cut into some segments with varied lengths because any length of segment has the possibility to become candidate MWE. The larger the context window is, the more difficulty its parameters acquire. Thus data sparseness problem deteriorates. Another problem arises from the flexible MWE which can be separated by an arbitrary number of blanks, for instance, "make. . . . . decision". These models cannot effectively distinguish all kinds of variations in flexible MWE.

On the consideration of relations between textual sequence and gene sequence, we propose a new bio-inspired approach for MWE identification. Both statistical and linguistic information are incorporated into this model.

2 Multi-word Expression

Multi-word Expression(in general, term) as the linguistic representation of concepts, also has some special statistical features. The component words of terms co-occur in the same context fre-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 176–182,
Sydney, July 2006.

quently. MWE extraction can be viewed as a problem of pattern extraction. It has two major phases. The first phase is to search the candidate MWEs by their frequent occurrence in the corpus. The second phase is to filter true MWEs from noise candidates. Filtering process involves linguistic knowledge and some intelligent observations.

MWE can be classified into strict patterns and flexible patterns by structures of their component words(Joaquim,1999). For example, a textual sequence s $= w_1 w_2 \cdots w_i \cdots w_n$, may contain two kinds of patterns:

Strict pattern: $\mathrm{p}_i = w_i w_{i+1} w_{i+2}$

Flexible pattern: $\mathrm{p}_j = w_i \sqcup w_{i+2} \sqcup w_{i+4}, p_k = w_i \sqcup \sqcup w_{i+3} w_{i+4}$

where $\sqcup$ denotes the variational or active element in pattern. The flexible pattern extraction is always a bottleneck for MWE extraction for lack of good knowledge of global solution.

3 Algorithms for MWE Extraction

3.1 Pure Mathematical Method

Although sequence alignment algorithm has been well-developed in bioinformatics (Michael,2003), (Knut,2000), (Hans,1999), it was rarely reported in MWE extraction. In fact, it also applies to MWE extraction especially for complex structures.

Algorithm.1.

1. Input:tokenlized textual sequences Q $= \{s_1, s_2, \cdots, s_n\}$
2. Initionalization : $pool,\ \Omega = \{\Omega_k\}, \Psi$
3. Computation:
   - I. Pairwise sequence alignment
     for all $s_i, s_j \in Q, s_i \neq s_j$
     $Similarity(s_i, s_j)$
     $Align(s_i, s_j) \overset{path(l_i,l_j)}{\longrightarrow} \{l_i,\ l_j,\ c_k\}$
     $pool \leftarrow\ pool \cup \{(l_i, c_k),\ (l_j,\ c_k)\}$
     $\Gamma \leftarrow \Gamma \cup c_k$
   - II. Creation of consistent set
     for all $c_k \in \Gamma,\ (l_i, c_k) \in pool$
     $\Omega_k \leftarrow \Omega_k + \{l_i\}$
     $pool \leftarrow\ pool - (l_i, c_k)$
   - III. Multiple sequence alignment
     for all Ω_k
     $star\_align(\Omega_k) \rightarrow MWU\ \Psi \leftarrow \Psi \cup MWU$
4. Output: Ψ

Our approach is directly inspired by gene sequence alignment as algorithm. 1. showed. The textual sequence should be preprocessed before input. For example, plurals recognition is a relatively simple task for computers which just need to check if the word accord with the general rule including rule (+s) and some alternative rules (-y + ies), etc. A set of tense forms, such as past, present and future forms, are also transformed into original forms. These tokenlized sequences will improve extraction quality.

Pairwise sequence alignment is a crucial step. Our algorithm uses local alignment for textual sequences. The similarity score between s$[1 \ldots i]$ and t$[1 \ldots i]$ can be computed by three arrays G$[i, j]$, E$[i, j]$,F$[i, j]$ and zero, where entry $\delta(x, y)$ means word x matches with word y; V$[i, j]$ denotes the best score of entry $\delta(x, y)$; G$[i, j]$ denotes s$[i]$ matched with $t[j]$:$\delta(s[i], t[j])$; $E[i, j]$ denotes a blank of string s matched with t$[j]$: $\delta(\sqcup, t[j])$; $F[i, j]$ denotes s$[i]$ matched with a blank of string t : $\delta(s[i], \sqcup)$.

Initialization:

$V[0, 0] = 0;\ V[i, 0] = E[i, 0] = 0;\ 1 \leq i \leq m.\ V[0, j] = F[0, j] = 0;\ 1 \leq j \leq n.$

A dynamic programming solution:

$$V[i, j] = \max\{G[i, j], E[i, j], G[i, j], 0\};$$

$$G[i, j] = \delta(i, j) + \max \begin{cases} G[i-1, j-1] \\ E[i-1, j-1] \\ F[i-1, j-1] \\ 0 \end{cases}$$

$$E[i, j] = \max \begin{cases} -(h+g) + G[i, j-1] \\ -g + E[i, j-1] \\ -(h+g) + F[i, j-1] \\ 0 \end{cases}$$

$$F[i, j] = \max \begin{cases} -(h+g) + G[i-1, j] \\ -(h+g) + E[i-1, j] \\ -g + F[i-1, j] \\ 0 \end{cases}$$

Here we explain the meaning of these arrays:

I. $G[i, j]$ includes the entry $\delta(i, j)$, it denotes the sum score is the last row plus the maximal score between prefix $s[1 \ldots i-1]$ and t$[1 \ldots j-1]$.

II. Otherwise the related prefixes $s[1 \ldots i]$ and t$[1 \ldots j-1]$ are needed[1]. They are used to check the first blank or additional blank in order to give appropriate penalty.

a. For $G[i, j-1]$ and $F[i, j-1]$, they don't end with a blank in string s. The blank $s[i]$ is the first blank. Its score is $G[i, j-1]$ (or $F[i, j-1]$) minus $(h+g)$.

b. For $E[i, j-1]$,The blank is the additional blank which should be only subtracted g.

In the maximum entry, it records the best score of optimum local alignment. This entry can be viewed as the started point of alignment. Then we backtrack entries by checking arrays which are generated from dynamic programming algorithm. When the score decrease to zero, alignment extension terminates. Finally, the similarity and alignment results are easily acquired.

Lots of aligned segments are extracted from pairwise alignment. Those segments with common component words (c_k) will be collected into the same set. It is called as consistent set for further multiple sequence alignment. These consistent sets collect similar sequences with score greater than certain threshold.

We perform star-alignment in multiple sequence alignment. The center sequence in the consistent set which has the highest score in comparison with others, is picked out from this set. Then all the other sequences gather to the center sequence with the technique of "once a blank, always a blank". These aligned sequences form common regions with n-column or a column. Every column contains one or more words. Calculation of dot-matrices is a widespread tool for common region analysis. Dot-plot agreement is developed to identify common patterns and reliably aligned regions in a set of related sequences. If several plots calculate consistently in a sequence set, it displays the similarity among them. It increases credibility of extracted pattern in this consistent set. Finally MWE with detailed pattern emerges from this aligned sequence set.

[1]Analysis approaches for $F[i, j]$ and $E[i, j]$ are the same, here only $E[i, j]$ is given its detailed explanation.

3.2 Linguistic Knowledge Combination

3.2.1 Heuristic Knowledge

Original candidate set is noise. Many meaningless patterns are extracted from corpus. Some linguistic rules (Argamon,1999) are introduced into our model. It is observed that candidate pattern should contain content words. Some patterns are only organized by pure function words, such as the most frequent patterns "the to", "of the". These patterns should be moved out from the candidate set. Filter table with certain words is also performed. For example, some words, like "then", cannot occur in the beginning position of MWE. These filters will reduce the number of noise patterns in great extent.

3.2.2 Embedded Base Phrase detection

Short textual sequence is apt to produce fragments of MWE because local alignment ends pattern extension when similarity score reduces to zero. The matched component words increase similarity score while unmatched words decrease it. The similarity scores of candidates in textual sequences are lower for lack of matched component words. Without accumulation of higher similarity score, pattern extension terminates quickly. Pattern extension becomes especially sensitive to unmatched words. Some isolated fragments are generated in this circumstance. One solution is to give higher scores for matched component words. It strengthens pattern extension ability at the expense of introducing noise.

We propose Embedded base phrase(EBP) detection as algorithm.2. It improves pattern extraction by giving lower penalty for longer base phrase. EBP is the base phrase in a gap (Changning,2000). It does not contain other phrase recursively. Good quality of MWE should avoid irrelative unit in its gap. The penalty function discerns the true EBP and irrelative unit in a gap only by length information. Longer gap means more irrelative unit. It builds a rough penalty model for lack of semantic information. We improve this model by POS information. POS tagged textual sequence is convenient to grammatical analysis. True EBP[2] gives comparatively lower penalty.

Algorithm.2.

1. Input: LCS of s_l, s_k

[2]The performance of our EBP tagger is 95% accuracy for base noun phrase and 90% accuracy for general use.

2. Check breakpoint in LCS

   i. Anchor neighbored common words and denote gaps
   for all $w_s = w_p, w_t = w_q$
   if $w_s \in l_s, w_t \in l_t, l_s \neq l_t$
   denote g_{st}, g_{pq}

   ii. Detect EBP in gaps
   $g_{st} \xrightarrow{EBP} g'_{st}, g_{pq} \xrightarrow{EBP} g'_{pq}$

   iii. Compute new similariy matrix in gaps
   $similarity(g'_{st}, g'_{pq})$

3. Link broken segment
   if $path(g'_{st}, g'_{pq})$
   $l_{st} = l_s + l_t, l_{pq} = l_p + l_q$

For textual sequence: $w_1 w_2 \cdots w_n$, and its corresponding POS tagged sequence: $t_1 t_2 \cdots t_n$, we suppose $[w_i \cdots w_j]$ is a gap from w_i to w_j in sequence $\cdots \; w_{i-1} [w_i \cdots w_j] w_j \; \cdots$. The corresponding tag sequence is $[t_i \cdots t_j]$. We only focus on EBP analysis in a gap instead of global sequence. Context Free Grammar (CFG) is employed in EBP detection. CFG rules follow this form:

$$
\begin{aligned}
&(1) EBP \leftarrow adj. + noun \\
&(2) EBP \leftarrow noun + "of" + noun \\
&(3) EBP \leftarrow adv. + adj. \\
&(4) EBP \leftarrow art. + adj. + noun \\
&\cdots
\end{aligned}
$$

The sequences inside breakpoint of LCS are analyzed by EBP detection. True base phrase will be given lower penalty. When the gap penalty for breakpoint is lower than threshold, the broken segment reunites. Based on experience knowledge, when the length of a gap is less than four words, EBP detection using CFG can gain good results. Lower penalty for true EBP will help MWE to emerge from noise pattern easily.

4 Experiments

4.1 Resources

A large amount of free texts are collected in order to meet the need of MWE extraction. These texts are downloaded from internet with various aspects including art, entertainment, military, business, etc. Our corpus size is 200, 000 sentences. The average sentence length is 15 words in corpus.

In addition, result evaluation is a hard job. Its difficulty comes from two aspects. Firstly, MWE identification for test corpus is a kind of labor-intensive business. The judgment of MWEs requires great efforts of domain expert. It is hard and boring to make a standard test corpus for MWE identification use. It is a bottleneck for large scales use. Secondly it relates to human cognition in psychological world. It is proved by experience that various opinions cannot simply be judged true or false. As a compromise way, gold standard set can be established by some accepted resources, for example, WordNet, as an online lexical reference system, including many compounds and phrases. Some terms extracted from dictionaries are also employed in our experiments. There are nearly 70,000 MWEs in our list.

4.2 Results and Discussion

4.2.1 Close Test

We created a closed test set of 8,000 sentences. MWEs in corpus are extracted by manual work. Every measure in both n-gram and LCS approaches complies with the same threshold, for example threshold for frequency is five times.Two conclusions are drawn from Tab.1.

Firstly, LCS has higher recall than n-gram but lower precision on the contrary. In close test set, LCS recall is higher than n-gram. LCS unifies all the cases of flexible patterns by GAM. However n-gram only considers limited flexible patterns because of model limitation. LCS nearly includes all the n-gram results. Higher recall decreases its precision to a certain extent because some flexible patterns are noisier than strict patterns. Flexible patterns tend to be more irrelevant than strict patterns. The GAM just provides a wiser choice for all flexible patterns by its gap penalty function. N-gram gives up analysis on many flexible patterns without further ado. N-gram ensures its precision by taking risk of MWE loss .

Secondly, advanced evaluation criterion can place more MWEs in the front rank of candidate list. Evaluation metrics for extracted patterns play an important role in MWE extraction. Many criteria, which are reported with better performances, are tested. MWE identification is similar to IR task. These measures have their own advantages to move interested patterns forward in the candidate list. For example, Frequency data contains much noise. True mutual infor-

Table 1: Close Test for N-gram and LCS Approaches

Measure	N-Gram			LCS		
	Precision (%)	Recall (%)	F-Measure (%)	Precision (%)	Recall (%)	F-Measure (%)
Frequency	35.2	38.0	36.0	32.1	48.2	38.4
TMI	44.7	56.2	49.1	43.2	62.1	51.4
ME	51.6	52.6	51.2	44.7	65.2	52.0
Rankratio	62.1	61.5	61.1	57.0	83.1	68.5

mation (TMI) concerns mutual information with logarithm(Manning,1999). Mutual expectation (ME) takes into account the relative probability of each word compared to the phrase(Joaquim,1999). Rankratio performs the best on both n-gram and LCS approaches because it provides all the contexts which associated with each word in the corpus and ranks them(Paul,2005). With the help of advanced statistic measures, the scores of MWEs are high enough to be detected from noisy patterns.

4.2.2 Open Test

In open test, we just show the extracted MWE numbers in different given corpus sizes. Two phenomena are observed in Fig.1.

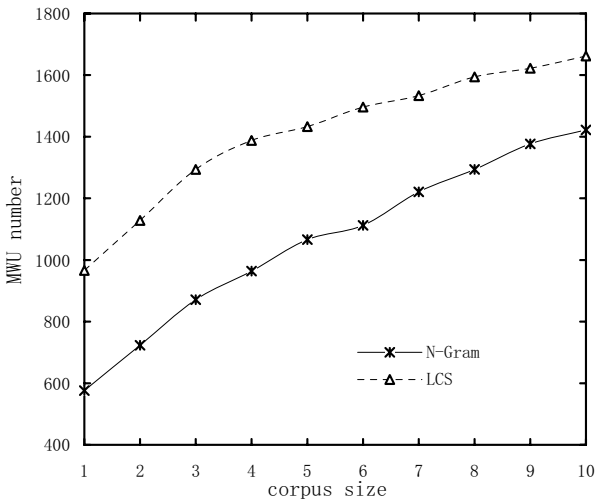


Figure 1: Open Test for N-gram and LCS Approaches

Firstly, with the enlargement of corpus size(every step of corpus size is 10,000 sentences), the detected MWE numbers increase in both approaches. When the corpus size reaches certain values, their increment speeds turn slower. It is reasonable on condition that MWE follow normal distribution. In the beginning, frequent MWEs are detected easily, and the number increases quickly. At a later phase, the detection goes into comparatively infrequent area. Mining these MWEs always need more corpus support. Lower increment speed appears.

Secondly, although LCS always keeps ahead in detecting MWE numbers, their gaps reduce with the increment of corpus size. LCS is sensitive to the MWE detection because of its alignment mechanism in which there is no difference between flexible pattern and strict pattern. In the beginning phase, LCS can detect MWEs which have high frequencies with flexible patterns. N-gram cannot effectively catch these flexible patterns. LCS detects a larger number of MWE than n-gram does. In the latter phase, many variable patterns for flexible MWE can also be observed, among which relatively strict patterns may appear in the larger corpus. They will be catched by n-gram. On the surface of observation, the discrepancy of detected numbers is gradually close to LCS. In nature, n-gram just makes up its limitation at the expense of corpus size because its detection mechanism for flexible patterns has no radical change.

5 Conclusion

In this article, our LCS-based approach is inspired by gene sequence alignment. In a new view, we reconsider MWE extraction task. These two tasks coincide with each other in pattern recognition. Some new phenomena in natural language are also observed. For example, we improve MWE mining result by EBP detection. Comparisons with variant n-gram approaches, which are the leading approaches, are performed for verifying the effectiveness of our approach. Although LCS approach results in better extraction model, a lot of improvements for more robust model are still needed.

Each innovation presented here only opens the way for more research. Some established theories between Computational Linguistics and Bioinformatics can be shared in a broader way.

6 Acknowledgements

The authors would like to thank three anonymous reviewers for their careful reading and helpful suggestions. This work is supported by National Natural Science Foundation of China (NSFC) (No.60496326) and 863 project of China (No.2001AA114210-11). Our thanks also go to Yushi Xu and Hui Liu for their coding and technical support.

References

Arantza Casillas, Raquel Martłnez , 2002. Aligning Multiword Terms Using a Hybrid Approach. Lecture Notes in Computer Science 2276: The 3rd International Conference of Computational Linguistics and Intelligent Text Processing.

Argamon, Shlomo, Ido Dagan and Yuval Krymolowski, 1999. A memory based approach to learning shallow natural language patterns. Journal of Experimental and Theoretical AI. 11, 369-390.

Beatrice Daille, 2003. Terminology Mining. Lecture Notes in Computer Science 2700: Extraction in the Web Era.

Changning Huang, Endong Xun, Zhou Ming, 2000. A Unified Statistical Model for the Identification of English BaseNP. The 38th Annual Meeting of the Association for Computational Linguistics.

Daniel S. Hirschberg, 1977. Algorithms for the Longest Common Subsequence Problem, Journal of the ACM, 24(4), 664-675.

Diana Binnenpoorte, Catia Cucchiarini, Lou Boves and Helmer Strik,2005. Multiword expressions in spoken language: An exploratory study on pronunciation variation. Computer Speech and Language,19(4):433-449

Hans Peter Lenhof, Burkhard Morgenstern, Knut Reinert, 1999. An exact solution for the segment-to-segment multiple sequence alignment problem. Bioinformatics. 15(3): 203-210.

Ivan A. Sag, Timothy Baldwin, Francis Bond, Ann A. Copestake, Dan Flickinger, 2002. Multiword Expressions: A Pain in the Neck for NLP. Lecture Notes in Computer Science 2276: The 3rd International Conference of Computational Linguistics and Intelligent Text Processing.

Jakob. H. Havgaard, R. Lyngs, G .D. Stormo and J. Gorodkin, 2005. Pairwise local structural alignment of RNA sequences with sequence similarity less than 40 percernt. Bioinfomrmatics. 21(9), 1815-1824.

Joaquim Ferreira da Silva, Gael Dias, Sylvie Guillore, Jose Gabriel Pereira Lopes, 1999. Using LocalMaxs Algorithm for the Extraction of Contiguous and Non-contiguous Multiword Lexical Units. The 9th Portuguese Conference on Artificial Intelligence.

Jordi Vivaldi, Llułs Marquez, Horacio Rodrłguez, 2001. Improving Term Extraction by System Combination Using Boosting. Lecture Notes in Computer Science 2167: The 12th European Conference on Machine Learning.

Kalliopi Zervanou and John McNaught, 2000. A Term-Based Methodology for Template Creation in Information Extraction. Lecture Notes in Computer Science 1835: Natural Language Processing.

Katerina Frantzi, Sophia Ananiadou, Hideki Mima, 2000. Automatic recognition of multi-word terms: the C-value/NC-value method. Int J Digit Libr. 3(2), 115C130.

Knut Reinert, Jens Stoye, Torsten Will, 2000. An iterative method for faster sum-of-pairs multiple sequence alignment. Bioinformatics. 16(9): 808-814.

Makoto Nagao, Shinsuke Mori, 1994. A New Method of N-gram Statistics for Large Number of n and Automatic Extraction of Words and Phrases from Large Text Data of Japanese. The 15th International Conference on Computational Linguistics.

Manning,C.D.,H.,Schutze,1999.Foundations of statistical natural language processing. MIT Press.

Marcus A. Zachariah, Gavin E. Crooks, Stephen R. Holbrook, Steven E. Brenner, 2005. A Generalized Affine Gap Model Significantly Improves Protein Sequence Alignment Accuracy. PROTEINS: Structure, Function, and Bioinformatics. 58(2), 329 - 338

Michael. Sammeth, B. Morgenstern, and J. Stoye, 2003. Divide-and-conquer multiple alignment with segment-based constraints. Bioinformatics. 19(2), 189-195.

Mike Paterson, Vlado Dancik ,1994. Longest Common Subsequences. Mathematical Foundations of Computer Science.

Pascale Fung, Kathleen Mckeown, 1997. A Technical Word and Term Translation Aid Using Noisy Parallel Corpora across Language Groups. Machine Translation. 12, 53C87.

Paul Deane, 2005. A Nonparametric Method for Extraction of Candidate Phrasal Terms. The 43rd Annual Meeting of the Association for Computational Linguistics.

Robertson, A.M. and Willett, P., 1998. Applications of n-grams in textual information systems. Journal of Documentation, 54(1), 48-69.

Satanjeev Banerjee, Ted Pedersen, 2003. The Design, Implementation, and Use of the Ngram Statistics Package. Lecture Notes in Computer Science 2588: The 4th International Conference of Computational Linguistics and Intelligent Text Processing.

Smith, T.F., Waterman, M.S., 1981. Identification of common molecular subsequences. J. Molecular Biology. 147(1), 195-197.

Stefan Diaconescu, 2004. Multiword Expression Translation Using Generative Dependency Grammar. Lecture Notes in Computer Science 3230: Advances in Natural Language Processing.

Suleiman H. Mustafa, 2004. Character contiguity in N-gram-based word matching: the case for Arabic text searching. Information Processing and Management. 41(4), 819-827.

Taneli Mielikainen, 2003. Frequency-Based Views to Pattern Collections. IFIP/SIAM Workshop on Discrete Mathematics and Data Mining.

Violeta Seretan, Luka Nerima, Eric Wehrl, 2003. Extraction of Multi-Word Collocations Using Syntactic Bigram Composition. International Conference on Recent Advances in NLP.

Towards A Modular Data Model For Multi-Layer Annotated Corpora

Richard Eckart
Department of English Linguistics
Darmstadt University of Technology
64289 Darmstadt, Germany
eckart@linglit.tu-darmstadt.de

Abstract

In this paper we discuss the current methods in the representation of corpora annotated at multiple levels of linguistic organization (so-called *multi-level* or *multi-layer* corpora). Taking five approaches which are representative of the current practice in this area, we discuss the commonalities and differences between them focusing on the underlying data models. The goal of the paper is to identify the common concerns in multi-layer corpus representation and processing so as to lay a foundation for a unifying, modular data model.

1 Introduction

Five approaches to representing multi-layer annotated corpora are reviewed in this paper. These reflect the current practice in the field and show the requirements typically posed on multi-layer corpus applications. Multi-layer annotated corpora keep annotations at different levels of linguistic organization separate from each other. Figure 1 illustrates two annotation layers on a transcription of an audio/video signal. One layer contains a functional annotation of a sentence in the transcription. The other contains a phrase structure annotation and Part-of-Speech tags for each word. Layers and signals are coordinated by a common timeline.

The motivation for this research is rooted in finding a proper data model for PACE-Ling (Sec. 2.2). The ultimate goal of our research is to create a modular extensible data model for multi-layer annotated corpora. To achieve this, we aim to create a data model based on the current state-of-the-art that covers all current requirements and

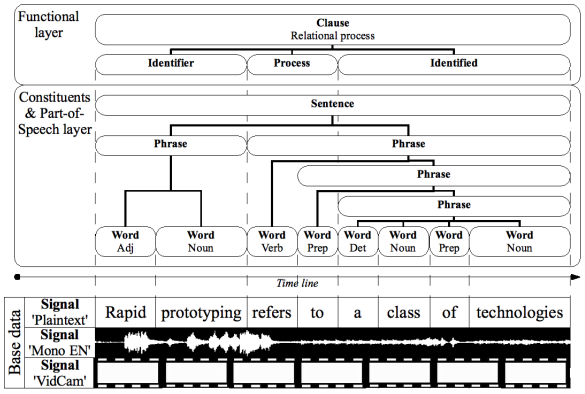


Figure 1: Multi-layer annotation on multi-modal base data

then decompose it into exchangeable components. We identify and discuss objects contained in four tiers commonly playing an important role in multi-layer corpus scenarios (see Fig. 2): *medial*, *locational*, *structural* and *featural* tiers. These are generalized categories that are in principle present in any multi-layer context, but come in different incarnations. Since query language and data model are closely related, common query requirements are also surveyed and examined for modular decomposition. While parts of the suggested data model and query operators are implemented by the projects discussed here, so far no comprehensive implementation exists.

2 Data models

There are three purposes data models can serve. The first purpose is *context suitability*. A data model used for this purpose must reflect as well as possible the data the user wants to query. The second purpose is *storage*. The data model used in the database backend can be very different from

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 183–190,
Sydney, July 2006.

the one exposed to the user, e.g. hierarchical structures may be stored in tables, indices might be kept to speed up queries, etc. The third purpose is *exchange* and *archival*. Here the data model, or rather the serialization of the data model, has to be easily parsable and follow a widely used standard.

Our review focuses on the suitability of data models for the first purpose. As extensions of the XML data model are used in most of the approaches reviewed here, a short introduction to this data model will be given first.

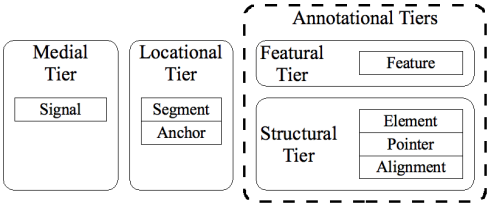


Figure 2: Tiers and objects

2.1 XML

Today XML has become the de-facto standard representation format for annotated text corpora. While the XML standard specifies a data model and serialization format for XML, a semantics is largely left to be defined for a particular application. Many data models can be mapped to the XML data model and serialized to XML (cf. Sec. 2.5).

The XML data model describes an ordered tree and defines several types of nodes. We examine a simplification of this data model here, limited to *elements*, *attributes* and *text nodes*. An element (*parent*) can contain *children*: elements and text nodes. Elements are named and can carry attributes, which are identified by a name and bear a value.

This data model is immediately suitable for simple text annotations. For example in a positional annotation, name-value pairs (*features*) can be assigned to tokens, which are obtained via tokenization of a text. These features and tokens can be represented by attributes and text nodes. The XML data model requires that both share a parent element which binds them together. Because the XML data model defines a tree, an additional root element is required to govern all positional annotation elements.

If the tree is constructed in such a way that one particular traversal strategy yields all tokens in their original order, then the data model is capable of covering all tiers: medial tier (textual base data), locational tier (sequential token order), structural tier (tokens) and featural tier (linguistic feature annotations). The structural tier can be expanded by adding additional elements en-route from the root element to the text nodes (*leaves*). In this way hierarchical structures can be modeled, for instance constituency structures. However, the XML data model covers these tiers only in a limited way. For example, tokens can not overlap each other without destroying the linear token order and thus sacrificing the temporal tier, a problem commonly known as *overlapping hierarchies*.

2.2 PACE-Ling

PACE-Ling (Bartsch et al., 05) aims at developing register profiles of texts from mechanical engineering (domain: data processing in construction) based on the multi-dimensional model of Systemic Functional Linguistics (SFL) (Halliday, 04).

The XML data model is a good foundation for this project as only written texts are analyzed, but SFL annotation requires multiple annotation layers with overlapping hierarchies. To solve this problem, the project applies a strategy known as *stand-off annotation*, first discussed in the context of SFL in (Teich et al., 05) and based on previous work by (Teich et al., 01). This strategy separates the annotation data from the base data and introduces references from the annotations to the base data, thus allowing to keep multiple layers of annotations on the same base data separate.

The tools developed in the project treat annotation data in XML from any source as separate annotation layers, provided the text nodes in each layer contain the same base data. The base data is extracted and kept in a text file and the annotation layers each in an XML file. The PACE-Ling data model substitutes text nodes from the XML data model by *segments*. Segments carry *start* and *end* attributes which specify the location of the text in the text file.

An important aspect of the PACE-Ling approach is minimal invasiveness. The minimally invasive change of only substituting text nodes by segments and leaving the rest of the original annotation file as it is, makes conversion between the original format and the format needed by the PACE-Ling tools very easy.

2.3 NITE XML Toolkit

The NITE XML toolkit (NXT) (Carletta et al., 04) was created with the intention to provide a framework for building applications working with annotated multi-modal data. NXT is based on the NITE Object Model (NOM) which is an extension of the XML data model. NOM features a similar separation of tiers as the PACE-Ling data model, but is more general.

NOM uses a continuous *timeline* to coordinate annotations. Instead of having dedicated segment elements, any annotation element can have special *start* and *end* attributes that anchor it to the timeline. This makes the data model less modular, because support for handling other locational strategies than a timeline can not be added by changing the semantics of segments (cf. Sec. 3.2).

NXT can deal with audio, video and textual base data, but due to being limited to the concept of a single common timeline, it is not possible to annotate a specific region in one video frame.

NOM introduces a new structural relation between annotation elements. Arbitrary links can be created by adding a *pointer* to an annotation element bearing a reference to another annotation element which designates the first annotation element to be a parent of the latter. Each pointer carries a role attribute describing its use.

Using pointers, arbitrary directed graphs can be overlaid on annotation layers and annotation elements can have multiple parents, one from the layer structure and any number of parents indicated by pointer references. This facilitates the reuse of annotations, e.g. when a number of annotations are kept that apply to words, the boundaries of words can be defined in one annotation layer and the other annotations can refer to that via pointers instead of defining the word boundaries explicitly in each layer. Using these pointers in queries is cumbersome, because they have to be processed one at a time (Evert et al., 03).

2.4 Deutsch Diachron Digital

The goal of *Deutsch Diachron Digital* (DDD) (Faulstich et al., 05) is the creation of a diachronic corpus, ranging from the earliest Old High German or Old Saxon texts from the 9th century up to Modern German at the end of the 19th century.

DDD requires each text to be available in several versions, ranging from the original facsimile over several transcription versions to translations into a modern language stage. This calls for a high degree of alignment between those versions as well as the annotations on those texts. Due to the vast amount of data involved in the project, the data model is not mapped to XML files, but to a SQL database for a better query performance.

The DDD data model can be seen as an extension of NOM. Because the corpus contains multiple versions of documents, coordination of annotations and base data along a single timeline is not sufficient. Therefore DDD segments refer to a specific version of a document.

DDD defines how *alignments* are modeled, thus elevating them from the level of structural annotation to an independent object in the structural tier: an alignment as a set of elements or segments, each of which is associated with a role.

Treating alignments as an independent object is reasonable because they are conceptually different from pointers and it facilitates providing an efficient storage for alignments.

2.5 ATLAS

The ATLAS project (Laprun et al., 02) implements a three tier data model model, resembling the separation of medial, locational and annotation tiers. This approach features two characteristic traits setting it apart from the others. First the data model is not inspired by XML, but by Annotation Graphs (AGs) (Bird & Liberman, 01). Second, it does not put any restriction on the kind of base data by leaving the semantics of segments and anchors undefined.

The ATLAS data model defines *signals*, *elements*, *attributes*, *pointers*, *segments* and *anchors*. Signals are base data objects (text, audio, etc.). Elements are related to each other only using pointers. While elements and pointers can be used to form trees, the ATLAS data model does not enforce this. As a result, the problem of overlapping hierarchies does not apply to the model. Elements are not contained within layers, instead they carry a type. However all elements of the same type can be interpreted as belonging to one layer. Segments do not carry start and end attributes, they carry a number of anchors. How exactly anchors are realized depends on the signals and is not specified in the data model.

The serialization format of ATLAS (AIF) is an XML dialect, but does not use the provisions for modeling trees present in the XML data model to

represent structural annotations as e.g. NXT does. The annotation data is stored as a flat set of elements, pointers, etc., which precludes the efficient use of existing tools like XPath to do structural queries. This is especially inconvenient as the ATLAS project does not provide a query language and query engine yet.

2.6 ISO 24610-1 - Feature Structures

The philosophy behind (ISO-24610-1, 06) is different from that of the four previous approaches. Here the base data is an XML document conforming to the TEI standard (Sperberg-McQueen & Burnard, 02). XML elements in the TEI base data can reference *feature stuctures*. A feature structure is a single-rooted graph, not necessarily a tree. The inner nodes of the graph are typed *elements*, the leaves are *values*, which can be shared amongst elements using *pointers* or can be obtained functionally from other values.

While in the four previously discussed approaches the annotations contain references to the base data in the leaves of the annotation structure, here the base data contains references to the root of the annotation structures. This is a powerful approach to identifying features of base data segments, but it is not very well suited for representing constituent hierarchies.

Feature structures put a layer of abstraction on top of the facilities provided by XML. XML validation schemes are used only to check the well-formedness of the serialization but not to validate the features structures. For this purpose *feature structure declarations* (FSD) have been defined.

3 A comprehensive data model

This section suggests a data model covering the objects that have been discussed in the context of the approaches presented in Sections 2.1-2.6. See Figure 3 for an overview.

3.1 Objects of the medial tier

We use the term *base data* for any data we want to annotate. A single instance of base data is called *signal*. Signals can be of many different kinds such as images (e.g. scans of facsimiles) or streams of text, audio or video data.

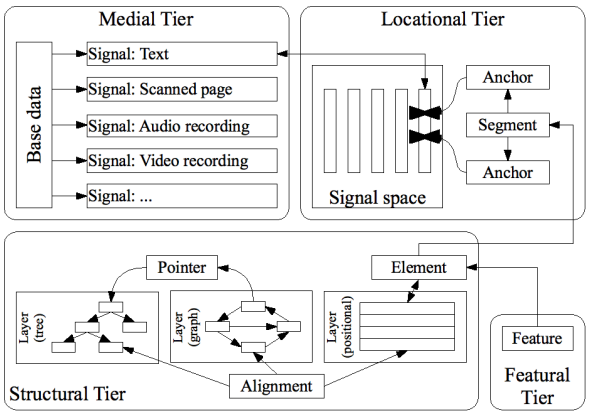


Figure 3: Comprehensive data model

3.2 Objects of the locational tier

Signals live in a virtual multi-dimensional *signal space*[1]. Each point of a signal is mapped to a unique point in signal space and vice versa. A *segment* identifies an area of signal space using a number of *anchors*, which uniquely identify points in signal space.

Depending on the kind of signal the dimensions of signal space have to be interpreted differently. For instance streams have a single dimension: time. At each point along the time axis, we may find a character or sound sample. Other kinds of signals can however have more dimensions: height, width, depth, etc. which can be continuous or discrete, bounded or open. For instance, a sheet of paper has two bounded and continuous dimensions: height and width. Thus a segment to capture a paragraph may have to describe a polygon. A single sheet of paper does not have a time dimension, however when multiple sheets are observed, these can be interpreted as a third dimension of discrete time.

3.3 Objects of the annotational tiers

An annotation *element* has a name and can have *features*, *pointers* and *segments*. A pointer is a typed directed reference to one or more elements. Elements relate to each other in different ways: directly by structural relations of the layer, pointers and alignments and indirectly by locational and medial relations (cf. Fig. 4).

An annotation *layer* contains elements and defines structural relations between them, e.g. *dominance* or *neighborhood* relations.

[1] (Laprun et al., 02) calls this *feature space*. This label is not used here to avoid suggesting a connection to the featural tier.

An *alignment* defines an equivalence class of elements, to each of which a *role* can be assigned.

Pointers can be used for structural relations that cross-cut the structural model of a layer or to create a relation across layer boundaries. Each pointer carries a role that specifies the kind of relation it models. Pointers allow an element to have multiple parents and to refer to other elements across annotation layers.

Features have a name and a value. They are always bound to an annotation element and cannot exist on their own. For the time being we use this simple definition of a feature, as it mirrors the concept of XML attributes. However, future work has to analyze if the ISO 24610 feature structures can and should be modelled as a part of the structural tier or if the featural tier should be extended.

4 Query

To make use of annotated corpora, query methods need to be defined. Depending on the data storage model that is used, different query languages are possible, e.g. XQuery for XML or SQL for relational databases. But these complicate query formulating because they are tailored to query a low level data storage model rather than a high level annotation data model.

A high level query language is necessary to get a good user acceptance and to achieve independence from lower level data models used to represent annotation data in an efficient way. NXT comes with NQL (Evert et al., 03), a sophisticated declarative high level query language. NQL is implemented in a completely new query engine instead of using XPath, XQuery or SQL. LPath, another recent development (Bird et al., 06), is a path-like query language. It is a linguistically motivated extension of XPath with additional axes and operators that allow additional queries and simplify others.

In some cases XML or SQL databases are simply not suited for a specific query. While we might be able to do regular expression matches on textual base data in a SQL or XML environment, doing a similar operation on video base data is beyond their scope.

The NXT project plans a translation of NQL to XQuery in order to use existing XQuery engines. LPath and DDD map high level query languages to SQL. (Grust et al., 04) are working on translating XQuery to SQL. The possibility of translating high level query languages into lower level query languages seems a good point for modularization.

4.1 Structural queries

Structural query operators are strongly tied to the structure of annotation layers, because they reflect the structural relations inside a layer. However, we also define structural relations such as alignments and pointers that exist independently of layers (cf. Sec. 3.3). The separation between pointers, alignments and different kinds of layers offers potential for modularization

Layers allowing only for positional annotations know only one structural relation: the neighborhood relation between two adjacent positions. Layers following the XML data model know parent-child relations and neighborhood relations. Layers with different internal structures may offer other relations. A number of possible relations is shown in Figure 4.

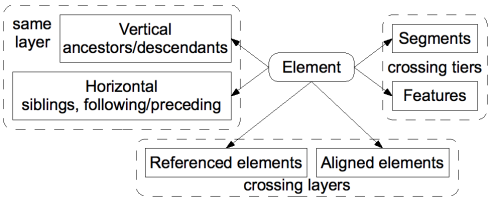


Figure 4: Structural relations and crossing to other tiers

While the implementation of query operators depends on the internal layer structure, the syntax does not necessarily have to be different. For instance a $following(a)$ operator of a positional layer will yield all elements following element a. A hierarchical layer can have two kinds of $following$ operators, one that only yields siblings following a and one yielding all elements following a. Here a choice has to be made if one of these operators is similar enough to the $following(a)$ to share that name without confusing the user.

Operators to follow pointers or alignments can be implemented independently of the layer structure.

XPath or LPath (Bird et al., 06) are path-like query languages specifically suited to access hierarchically structured data, but neither directly supports alignments, pointers or the locational tier. In the context of XQuery, XPath can be extended with user-defined functions that could be used to provide this access, but using such functions in path statements can become awkward. It may be a better idea to extend the path language instead.

Structural queries could look like this:

- Which noun phrases are inside verb phrases?
  `//VP//NP`
  Result: a set of annotation elements.

- Anaphora are annotated using a pointer with the role "anaphor". What do determiners in the corpus refer to?
  `//DET/=>anaphor`
  Result: a set of annotation elements.

- Translated elements are aligned in an alignment called "translation". What are the translations of the current element?
  `self/#translation`
  Result: a set of annotation elements.

4.2 Featural queries

If we use the simple definition of features from Section 3.3, there is only one operator native to the featural tier that can be used to access the annotation element associated with a feature. If we use the complex definition from ISO 24610, the operators of the featural tier are largely the same as in hierarchically structured annotation layers.

Operators to test the value of a feature can not strictly be assigned to the featural tier. Using the simple definition, the value of a feature is some typed atomic value. The query language has to provide generic operators to compare atomic values like strings or numbers with each other. E.g. XPath provides a weakly typed system that provides such operators.

Queries involving features could look like this:

- What is the value of the "PoS" feature of the current annotation element?
  `self/@PoS`
  Result: a string value.

- What elements have a feature called "PoS" with the value "N"?
  `//*[@PoS='N']`
  Result: a set of annotation elements.

4.3 Locational queries

Locational queries operate on segment data. The inner structure of segments reflects the structure of signal space and different kinds of signals require different operators. Most of the time operators working on single continuous dimensions, e.g. a timeline, will be used. An operator working on higher dimensions could be an intersection operator of two dimensional signal space areas (scan of a newspaper page, video frames, etc.).

Queries involving locations could look like this:

- What parts of segments a and b overlap?
  `overlap($a,$b)`
  Result: the empty set or a segment defining the overlapping part.

- Merge segments a and b.
  `merge($a, $b)`
  Result: if a and b overlap, the result is a new segment that covers both, otherwise the results is a set consisting of a and b.

- Is segment a following segment b?
  `is-following($a, $b)`
  Result: true or false.

Locational operators are probably best bundled into modules by the kind of locational structure they support: a module for sequential data such as text or audio, one for two-dimensional data such as pictures, and so on.

4.4 Medial queries

Medial query operators access base data, but often they take locational arguments or return locational information. When a medial operator is used to access textual base data, the result is a string. As with feature values, such a string could be evaluated by a query language that supports some primitive data types.

Assume there is a textual signal named 'plaintext'. Queries on base data could look like this:

- Where does the string "rapid" occur?
  `signal('plaintext')/'rapid'`
  Result: a set of segments.

- Where does the string "prototyping" occur to the right of the location of "rapid"?
  ```
  signal('plaintext')/
      'rapid'>>'prototyping'
  ```
  Result: a set of segments.

- What is the base data between offset 5 and 9 of the signal "plaintext"?
  `signal('plaintext')/<{5,9}>`
  Result: a portion of base data (e.g. a string).

If the base data is an audio or video stream, the type system of most query languages is likely to

be insufficient. In such a case a module providing support for audio or video storage should also provide necessary query operators and data type extensions to the query engine.

4.5 Projection between annotational and medial tiers

So far we have considered crossing the borders between the structural and featural tiers and between the locational and medial tiers. Now we examine the border between the locational and structural tier. An operator can be used to collect all locational data associated with an annotation element and its children:

```
seg(//S/VP/)
```

The result would be a set of potentially overlapping segments. Depending on the query, it will be necessary to merge overlapping segments to get a list of non-overlappping segments. Assume we have a recorded interview annotated for speakers and at some point speaker A and B speak at the same time. We want to listen to all parts of the interview in which speakers A or B speak. If we query without merging overlapping segments, we will hear the part in which both speak at the same time twice.

Similar decisions have to be made when projecting up from a segment into the structural layer. Figure 5 shows a hierarchical annotation structure. Only the elements $W1$, $W2$ and $W3$ bear segments that anchor them to the base data at the points A-D.

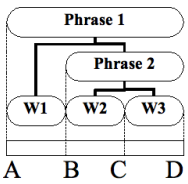


Figure 5: Example structure

When projecting up from the segment $\{B, D\}$ there are a number of potentially desirable results. Some are given here:

1. no result: because there is no annotation element that is anchored to $\{B, D\}$.

2. $W2$ and $W3$: because both are anchored to an area inside $\{B, D\}$.

3. *Phrase 2*, $W2$ and $W3$: because applying the `seg` operator to either element yields segments inside $\{B, D\}$.

4. *Phrase 2* only: because applying the `seg` operator to this element yields an area that covers exactly $\{B, D\}$.

5. *Phrase 1*, *Phrase 2*: because applying the `seg` operator to either element yields segments containing $\{B, D\}$.

The query language has to provide operators that enable the user to choose the desired result. Queries that yield the desired results could look like in Figure 6. Here the *same-extent* operator takes two sets of segments and returns those segments that are present in both lists and have the same start and end positions. The *anchored* operator takes an annotation element and returns *true* if the element is anchored. The *contains* operator takes two sets of segments a and b and returns all segments from set b that are contained in an area covered by any segment in set a. The *grow* operator takes a set of segments and returns a segment, which starts at the smallest offset and ends at the largest offset present in any segment of the input list. In the tests an empty set is interpreted as *false* and a non-empty set as *true*.

1. `//*[same-extent(seg(.), <{B,D}>)]`

2. `//*[anchored(.) and contains(<{B,D}>, seg(.))]`

3. `//*[contains(<{B,D}>, seg(.))]`

4. `//*[same-extent(grow(seg(.)), <{B,D}>)]`

5. `//*[contains(seg(.)), <{B,D}>]`

Figure 6: Projection examples

5 Conclusion

Corpus-based research projects often choose to implement custom tools and encoding formats. Small projects do not want to lose valuable time learning complex frameworks and adapting them to their needs. They often employ a custom XML format to be able to use existing XML processing tools like XQuery or XSLT processors.

ATLAS or NXT are very powerful, yet they suffer from lack of accessibility to programmers who have to adapt them to project-specific needs. Most specialized annotation editors do not build upon these frameworks and neither offer conversion tools between their data formats.

Projects such as DDD do not make use of the frameworks, because they are not easily extensible, e.g. with a SQL backend instead of an XML storage. Instead, again a high level query language is developed and a completely new framework is created which works with a SQL backend.

In the previous sections, objects from selected approaches with different foci in their work with annotated corpora have been collected and forged into a comprehensive data model. The potential for modularization of corpus annotation frameworks has been shown with respect to data models and query languages. As a next step, an existing framework should be taken and refactored into an extensible modular architecture. From a practical point of view reusing existing technology as much as possible is a desirable goal. This means reusing existing facilities provided for XML data, such as XPath, XQuery and XSchema and where necessary trying to extend them, instead of creating a new data model from scratch. For the annotational tiers, as LPath has shown, a good starting point to do so is to extend existing languages like XPath. Locational and medial operators seem to be best implemented as XQuery functions. The possibility to map between SQL and XML provides access to additional efficient resources for storing and querying annotation data. Support for various kinds of base data or locational information can be encapsulated in modules. Which modules exactly should be created and what they should cover in detail has to be further examined.

Acknowledgements

Many thanks go to Elke Teich and Peter Fankhauser for their support. Part of this research was financially supported by *Hessischer Innovationsfonds* and PACE (Partners for the Advancement of Collaborative Engineering Education).

References

S. Bartsch, R. Eckart, M. Holtz & E. Teich 2005. Corpus-based register profiling of texts form mechanical engineering In *Proceedings of Corpus Linguistics*, Birmingham, UK, July 2005.

S. Bird & M. Liberman 2001. A Formal Framework for Linguistic Annotation In *Speech Communication 33(1,2), pp 23-60*

S. Bird, Y. Chen, S. B. Davidson, H. Lee and Y. Zheng. 2006. Designing and Evaluating an XPath Dialect for Linguistic Queries. In *Proceedings of the 22nd International Conference on Data Engineering*, ICDE 2006, 3-8 April 2006, Atlanta, GA, USA

J. Carletta, D. McKelvie, A. Isard, A. Mengel, M. Klein & M.B. Møller 2004 A generic approach to software support for linguistic annotation using XML In *G. Sampson and D. McCarthy (eds.), Corpus Linguistics: Readings in a Widening Discipline*. London and NY: Continuum International.

S. Evert, J. Carletta, T. J. O'Donnell, J. Kilgour, A. Vögele & H. Voormann 2003. *The NITE Object Model* v2.1 http://www.ltg.ed.ac.uk/NITE/documents/NiteObjectModel.v2.1.pdf

L. C. Faulstich, U. Leser & A. Lüdeling 2005. Storing and querying historical texts in a relational database In *Informatik-Bericht 176*, Institut für Informatik, Humboldt-Universität zu Berlin, 2005.

T. Grust and S. Sakr and J. Teubner 2002. XQuery on SQL Hosts In *Proceedings of the 30th Int'l Conference on Very Large Data Bases (VLDB)* Toronto, Canada, Aug. 2004.

M.A.K. Halliday. 2004. *Introduction to Functional Grammar*. Arnold, London. Revised by CMIM Matthiessen

C. Laprun, J.G. Fiscus, J. Garofolo, S. Pajot 2002. A practical introduction to ATLAS In *Proceedings LREC 2002* Las Palmas http://www.nist.gov/speech/atlas/download/lrec2002-atlas.pdf

M. Laurent Romary (chair) and TC 37/SC 4/WG 2 2006. Language resource management - Feature structures - Part 1: Feature structure representation. In *ISO 24610-1*.

C. M. Sperberg-McQueen & L. Burnard, (eds.) 2002. *TEI P4: Guidelines for Electronic Text Encoding and Interchange*. Text Encoding Initiative Consortium. XML Version: Oxford, Providence, Charlottesville, Bergen

E. Teich, P. Fankhauser, R. Eckart, S. Bartsch, M. Holtz. 2005. Representing SFL-annotated corpora. In *Proceedings of the First Computational Systemic Functional Grammar Workshop (CSFG)*, Sydney, Australia.

E. Teich, S. Hansen, and P. Fankhauser. 2001. Representing and querying multi-layer corpora. In *Proceedings of the IRCS Workshop on Linguistic Databases*, pages 228-237, University of Pennsylvania, Philadelphia, 11-13 December.

A Modified Joint Source-Channel Model for Transliteration

Asif Ekbal
Comp. Sc. & Engg. Deptt.
Jadavpur University
India
ekbal_asif12@
yahoo.co.in

Sudip Kumar Naskar
Comp. Sc. & Engg. Deptt.
Jadavpur University
India
sudip_naskar@
hotmail.com

Sivaji Bandyopadhyay
Comp. Sc. & Engg. Deptt.
Jadavpur University
India
sivaji_cse_ju@
yahoo.com

Abstract

Most machine transliteration systems transliterate out of vocabulary (OOV) words through intermediate phonemic mapping. A framework has been presented that allows direct orthographical mapping between two languages that are of different origins employing different alphabet sets. A modified joint source–channel model along with a number of alternatives have been proposed. Aligned transliteration units along with their context are automatically derived from a bilingual training corpus to generate the collocational statistics. The transliteration units in Bengali words take the pattern C^+M where C represents a vowel or a consonant or a conjunct and M represents the vowel modifier or matra. The English transliteration units are of the form C*V* where C represents a consonant and V represents a vowel. A Bengali-English machine transliteration system has been developed based on the proposed models. The system has been trained to transliterate person names from Bengali to English. It uses the linguistic knowledge of possible conjuncts and diphthongs in Bengali and their equivalents in English. The system has been evaluated and it has been observed that the modified joint source-channel model performs best with a Word Agreement Ratio of 69.3% and a Transliteration Unit Agreement Ratio of 89.8%.

1 Introduction

In Natural Language Processing (NLP) application areas such as information retrieval, question answering systems and machine translation, there is an increasing need to translate OOV words from one language to another. They are translated through transliteration, the method of translating into another language by expressing the original foreign words using characters of the target language preserving the pronunciation in their original languages. Thus, the central problem in transliteration is predicting the pronunciation of the original word. Transliteration between two languages, that use the same set of alphabets, is trivial: the word is left as it is. However, for languages that use different alphabet sets, the names must be transliterated or rendered in the target language alphabets.

Technical terms and named entities make up the bulk of these OOV words. Named entities hold a very important place in NLP applications. Proper identification, classification and translation of named entities are very crucial in many NLP applications and pose a very big challenge to NLP researchers. Named entities are usually not found in bilingual dictionaries and they are very productive in nature. Translation of named entities is a tricky task: it involves both translation and transliteration. Transliteration is commonly used for named entities, even when the words could be translated. Different types of named entities are translated differently. Numerical and temporal expressions typically use a limited set of vocabulary words (e.g., names of months, days of the week etc.) and can be translated fairly easily using simple translation patterns. The named entity machine transliteration algorithms presented in this work

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 191–198,
Sydney, July 2006.

focus on person names, locations and organizations. A machine transliteration system that is trained on person names is very important in a multilingual country like India where large name collections like census data, electoral roll and railway reservation information must be available to multilingual citizens of the country in their vernacular. In the present work, the various proposed models have been evaluated on a training corpus of person names.

A hybrid neural network and knowledge-based system to generate multiple English spellings for Arabic personal names is described in (Arbabi et al., 1994). (Knight and Graehl, 1998) developed a phoneme-based statistical model using finite state transducer that implements transformation rules to do back-transliteration. (Stalls and Knight, 1998) adapted this approach for back transliteration from Arabic to English for English names. A spelling-based model is described in (Al-Onaizan and Knight, 2002a; Al-Onaizan and Knight, 2002c) that directly maps English letter sequences into Arabic letter sequences with associated probability that are trained on a small English/Arabic name list without the need for English pronunciations. The phonetics-based and spelling-based models have been linearly combined into a single transliteration model in (Al-Onaizan and Knight, 2002b) for transliteration of Arabic named entities into English.

Several phoneme-based techniques have been proposed in the recent past for machine transliteration using transformation-based learning algorithm (Meng et al., 2001; Jung et al., 2000; Vigra and Khudanpur, 2003). (Abduljaleel and Larkey, 2003) have presented a simple statistical technique to train an English-Arabic transliteration model from pairs of names. The two-stage training procedure first learns which n-gram segments should be added to unigram inventory for the source language, and then a second stage learns the translation model over this inventory. This technique requires no heuristic or linguistic knowledge of either language.

(Goto et al., 2003) described an English-Japanese transliteration method in which an English word is divided into conversion units that are partial English character strings in an English word and each English conversion unit is converted into a partial Japanese Katakana character string. It calculates the likelihood of a particular choice of letters of chunking into English conversion units for an English word by linking them to Katakana characters using syllables. Thus the English conversion units consider phonetic aspects. It considers the English and Japanese contextual information simultaneously to calculate the plausibility of conversion from each English conversion unit to various Japanese conversion units using a single probability model based on the maximum entropy method.

(Haizhou et al., 2004) presented a framework that allows direct orthographical mapping between English and Chinese through a joint source-channel model, called n-gram transliteration model. The orthographic alignment process is automated using the maximum likelihood approach, through the Expectation Maximization algorithm to derive aligned transliteration units from a bilingual dictionary. The joint source-channel model tries to capture how source and target names can be generated simultaneously, i.e., the context information in both the source and the target sides are taken into account.

A tuple n-gram transliteration model (Marino et al., 2005; Crego et al., 2005) has been log-linearly combined with feature functions to develop a statistical machine translation system for Spanish-to-English and English-to-Spanish translation tasks. The model approximates the joint probability between source and target languages by using trigrams.

The present work differs from (Goto et al., 2003; Haizhou et al., 2004) in the sense that identification of the transliteration units in the source language is done using regular expressions and no probabilistic model is used. The proposed modified joint source-channel model is similar to the model proposed by (Goto et. al., 2003) but it differs in the way the transliteration units and the contextual information are defined in the present work. No linguistic knowledge is used in (Goto et al., 2003; Haizhou et al., 2004) whereas the present work uses linguistic knowledge in the form of possible conjuncts and diphthongs in Bengali.

The paper is organized as follows. The machine transliteration problem has been formulated under both noisy-channel model and joint source-channel model in Section 2. A number of transliteration models based on collocation statistics including the modified joint source-channel model and their evaluation scheme have been proposed in Section 3. The Bengali-English machine transliteration scenario has been presented in Section 4. The proposed

models have been evaluated and the result of evaluation is reported in Section 5. The conclusion is drawn in Section 6.

2 Machine Transliteration and Joint Source-Channel Model

A transliteration system takes as input a character string in the source language and generates a character string in the target language as output. The process can be conceptualized as two levels of decoding: segmentation of the source string into transliteration units; and relating the source language transliteration units with units in the target language, by resolving different combinations of alignments and unit mappings. The problem of machine transliteration has been studied extensively in the paradigm of the noisy channel model.

For a given Bengali name B as the observed channel output, we have to find out the most likely English transliteration E that maximizes $P(E \mid B)$. Applying Bayes' rule, it means to find E to maximize

$$P(B,E) = P(B \mid E) * P(E) \quad (1)$$

with equivalent effect. This is equivalent to modelling two probability distributions: P(B|E), the probability of transliterating E to B through a noisy channel, which is also called transformation rules, and P(E), the probability distribution of source, which reflects what is considered good English transliteration in general. Likewiswe, in English to Bengali (E2B) transliteration, we could find B that maximizes

$$P(B,E) = P(E \mid B) * P(B) \quad (2)$$

for a given English name. In equations (1) and (2), P(B) and P(E) are usually estimated using n-gram language models. Inspired by research results of grapheme-to-phoneme research in speech synthesis literature, many have suggested phoneme-based approaches to resolving $P(B \mid E)$ and $P(E \mid B)$, which approximates the probability distribution by introducing a phonemic representation. In this way, names in the source language, say B, are converted into an intermediate phonemic representation P, and then the phonemic representation is further converted into the target language, say English E. In Bengali to English (B2E) transliteration, the phoneme-based approach can be formulated as $P(E \mid B) = P(E \mid P) * P(P \mid B)$ and conversely we have $P(B \mid E) = P(B \mid P) * P(P \mid E)$ for E2B back-transliteration.

However, phoneme-based approaches are limited by a major constraint that could compromise transliteration precision. The phoneme-based approach requires derivation of proper phonemic representation for names of different origins. One may need to prepare multiple language-dependent grapheme-to-phoneme(G2P) and phoneme-to-grapheme(P2G) conversion systems accordingly, and that is not easy to achieve.

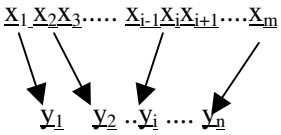
In view of close coupling of the source and target transliteration units, a joint source-channel model, or n-gram transliteration model (TM) has been proposed in (Haizhou et al., 2004). For K alligned transliteration units, we have

$$\begin{aligned} P(B,E) &= P(b_1, b_2 \ldots b_k, e_1, e_2 \ldots e_k) \\ &= P(\langle b,e\rangle_1, \langle b,e\rangle_2, \ldots \langle b,e\rangle_k) \\ &= \prod_{k=1}^{K} P(\langle b,e\rangle_k \mid \langle b,e\rangle_1^{k-1}) \end{aligned} \quad (3)$$

which provides an alternative to the phoneme-based approach for resolving equations (1) and (2) by eliminating the intermediate phonemic representation.

Unlike the noisy-channel model, the joint source-channel model does not try to capture how source names can be mapped to target names, but rather how source and target names can be generated simultaneously. In other words, a joint probability model is estimated that can be easily marginalized in order to yield conditional probability models for both transliteration and back-transliteration.

Suppose that we have a Bengali name $\alpha = x_1 x_2 \ldots x_m$ and an English transliteration $\beta = y_1 y_2 \ldots y_n$ where x_i, i = 1: m are Bengali transliteration units and y_j, j = 1: n are English transliteration units. An English transliteration unit may correspond to zero, one or more than one transliteration unit in Bengali. Often the values of m and n are different.



where there exists an alignment γ with $\langle b,e\rangle_1 = \langle x_1,y_1\rangle$; $\langle b,e\rangle_2 = \langle x_2x_3, y_2\rangle$; …. and $\langle b,e\rangle_k = \langle x_m,y_n\rangle$. A transliteration unit correspondence <b, e> is called a transliteration pair. Thus B2E transliteration can be formulated as

$$\overline{\beta} = \underset{\beta,\,\gamma}{\operatorname{argmax}}\, P(\alpha, \beta, \gamma) \quad (4)$$

and similarly the E2B back-transliteration as

$$\overline{\alpha} = \underset{\alpha, \gamma}{\operatorname{argmax}} P(\alpha, \beta, \gamma) \quad (5)$$

An *n*-gram transliteration model is defined as the conditional probability or transliteration probability of a transliteration pair $<b, e>_k$ depending on its immediate *n* predecessor pairs:

$$P(B, E) = P(\alpha, \beta, \gamma)$$

$$= \prod_{k=1}^{K} P(<b, e>_k \mid <b, e>_{k-n+1}^{k-1}) \quad (6)$$

3 Proposed Models and Evaluation Scheme

Machine transliteration has been viewed as a sense disambiguation problem. A number of transliteration models have been proposed that can generate the English transliteration from a Bengali word that is not registered in any bilingual or pronunciation dictionary. The Bengali word is divided into Transliteration Units (TU) that have the pattern C^+M, where C represents a vowel or a consonant or conjunct and M represents the vowel modifier or matra. An English word is divided into TUs that have the pattern C*V*, where C represents a consonant and V represents a vowel. The TUs are considered as the lexical units for machine transliteration. The system considers the Bengali and English contextual information in the form of collocated TUs simultaneously to calculate the plausibility of transliteration from each Bengali TU to various English candidate TUs and chooses the one with maximum probability. This is equivalent to choosing the most appropriate sense of a word in the source language to identify its representation in the target language. The system learns the mappings automatically from the bilingual training corpus guided by linguistic features. The output of this mapping process is a decision-list classifier with collocated TUs in the source language and their equivalent TUs in collocation in the target language along with the probability of each decision obtained from a training corpus. The machine transliteration of the input Bengali word is obtained using direct orthographic mapping by identifying the equivalent English TU for each Bengali TU in the input and then placing the English TUs in order. The various proposed models differ in the nature of collocational stastistics used during machine transliteration process: monogram model with no context, bigram model with previous (with respect to the current TU to be transliterated) source TU as the context, bigram model with next source TU as the context, bigram model with previous source and target TUs as the context (this is the joint source channel model), trigram model with previous and next source TUs as the context and the modified joint source-channel model with previous and next source TUs and the previous target TU as the context.

● Model A

In this model, no context is considered in either the source or the target side. This is essentially the monogram model.

$$P(B,E) = \prod_{k=1}^{K} P(<b,e>_k)$$

● Model B

This is essentially a bigram model with previous source TU, i.e., the source TU occurring to the left of the current TU to be transliterated, as the context.

$$P(B,E) = \prod_{k=1}^{K} P(<b,e>_k \mid b_{k-1})$$

●Model C

This is essentially a bigram model with next source TU, i.e., the source TU occurring to the right of the current TU to be transliterated, as the context.

$$P(B,E) = \prod_{k=1}^{K} P(<b,e>_k \mid b_{k+1})$$

● Model D

This is essentially the joint source-channel model where the previous TUs in both the source and the target sides are considered as the context. The previous TU on the target side refers to the transliterated TU to the immediate left of the current target TU to be transliterated.

$$P(B,E) = \prod_{k=1}^{K} P(<b,e>_k \mid <b,e>_{k-1})$$

● Model E

This is basically the trigram model where the previous and the next source TUs are considered as the context

$$P(B,E) = \prod_{k=1}^{K} P(<b,e>_k \mid b_{k-1}, b_{k+1})$$

● Model F

In this model, the previous and the next TUs in the source and the previous target TU are considered as the context. This is the modified joint source-channel model .

$$P(B,E) = \prod_{k=1}^{K} P(<b,e>_k \mid <b,e>_{k-1}, b_{k+1})$$

The performance of the system is evaluated in terms of Transliteration Unit Agreement Ratio (TUAR) and Word Agreement Ratio (WAR) following the evaluation scheme in (Goto et al., 2003). The evaluation parameter Character Agreement Ratio in (Goto et al., 2003) has been modified to Transliteration Unit Agreement Ratio as vowel modifier matra symbols in Bengali words are not independent and must always follow a consonant or a conjunct in a Transliteration Unit. Let, B be the input Bengali word, E be the English transliteration given by the user in open test and E' be the system generates the transliteration. TUAR is defined as, TUAR = (L-Err)/ L, where L is the number of TUs in E, and Err is the number of wrongly transliterated TUs in E' generated by the system. WAR is defined as, WAR= (S-Err$'$) / S, where S is the test sample size and Err$'$ is is the number of erroneous names generated by the system (when E' does not match with E). Each of these models has been evaluated with linguistic knowledge of the set of possible conjuncts and diphthongs in Bengali and their equivalents in English. It has been observed that the Modified Joint Source Channel Model with linguistic knowledge performs best in terms of Word Agreement Ratio and Transliteration Unit Agreement Ratio.

4 Bengali-English Machine Transliteration

Translation of named entities is a tricky task: it involves both translation and transliteration. Transliteration is commonly used for named entities, even when the words could be translated [জনতা দল (*janata dal*) is translated to *Janata Dal* (literal translation) although জনতা *(Janata)* and দল (*Dal)* are vocabulary words]. On the other hand যাদবপুর বিশ্ববিদ্যালয় (*jadavpur viswavidyalaya*) is translated to *Jadavpur University* in which যাদবপুর (*Jadavpur)* is transliterated to *Jadavpur* and বিশ্ববিদ্যালয় (*viswavidyalaya*) is translated to *University.*

A bilingual training corpus has been kept that contains entries mapping Bengali names to their respective English transliterations. To automatically analyze the bilingual training corpus to acquire knowledge in order to map new Bengali names to English, TUs are extracted from the Bengali names and the corresponding English names, and Bengali TUs are associated with their English counterparts.

Some examples are given below:

অভিনন্দন (*abhinandan*) → [অ । ভি । ন । ন্দ । ন]
abhinandan → [a | bhi | na | nda | n]

কৃষ্ণমূর্তি (*krishnamoorti*) → [কৃ । ষ্ণ । মূ । র্তি]
krishnamurthy → [kri | shna | mu | rthy]

শ্রীকান্ত (*srikant*) → [শ্রী । কা । ন্ত]
srikant → [sri | ka | nt]

After retrieving the transliteration units from a Bengali-English name pair, it associates the Bengali TUs to the English TUs along with the TUs in context.

For example, it derives the following transliteration pairs or rules from the name-pair:

রবীন্দ্রনাথ (*rabindranath)* → rabindranath

Source Language				**Target Language**	
previous TU	TU	next TU		previous TU	TU
-	র	বী	↔	-	ra
র	বী	ন্দ্র	↔	ra	bi
বী	ন্দ্র	না	↔	bi	ndra
ন্দ্র	না	থ	↔	ndra	na
না	থ	-	↔	na	th

But, in some cases, the number of transliteration units retrieved from the Bengali and English words may differ. The [বৃজমোহন (*brijmohan*) ↔ brijmohan] name pair yields 5 TUs in Bengali side and 4 TUs in English side [বৃ । জ । মো । হ । ন ↔ bri । jmo । ha । n]. In such cases, the system cannot align the TUs automatically and linguistic knowledge is used to resolve the confusion. A knowledge base that contains a list of Bengali conjuncts and diphthongs and their possible English representations has been kept. The hypothesis followed in the present work is that *the problem TU in the English side has always the maximum length.* If more than one English TU has the same length, then *system starts its analysis from the first one*. In the above example, the TUs *bri* and *jmo* have the same length. The system interacts with the knowledge base and ascertains that *bri* is valid and *jmo* cannot be a valid TU in English since there is no corresponding conjunct representation in Bengali. So *jmo* is split up into 2 TUs *j* and *mo*, and the system aligns the 5 TUs as [বৃ । জ । মো । হ । ন ↔ bri । j । mo । ha । n]. Similarly, [লোকনাথ (*loknath*) ↔ loknath] is initially split as [লো । ক । না । থ] ↔ lo । kna । th], and then as [lo । k । na । th] since *kna* has the maximum length and it does not have any valid conjunct representation in Bengali.

In some cases, the knowledge of Bengali diphthong resolves the problem. In the following example, [রা । ই । মা (*raima*) ↔ rai । ma], the number of TUs on both sides do not match. The English TU *rai* is chosen for analysis as its length is greater than the other TU *ma*. The vowel sequence *ai* corresponds to a diphthong in Bengali that has two valid representations < আই, ঐ >. The first representation signifies that a matra is associated to the previous character followed by the character ই. This matches the present Bengali input. Thus, the English vowel sequence *ai* is separated from the TU *rai* (rai → r । ai) and the intermediate form of the name pair appears to be [রা । ই । মা (*raima*) ↔ r । ai । ma]. Here, a *matra* is associated with the Bengali TU that corresponds to English TU *r* and so there must be a vowel attached with the TU *r*. TU *ai* is further splitted as *a* and *i* (ai → a । i) and the first one (i.e. *a)* is assimilated with the previous TU (i.e. *r*) and finally the name pair appears as: [রা । ই । মা (*raima*) ↔ ra । i । ma].

In the following two examples, the number of TUs on both sides does not match.

[দে । ব । রা । জ (*devraj*) ↔ de । vra । j]

[সো । ম । না । থ (*somnath*) ↔ so । mna । th]

It is observed that both *vr* and *mn* represent valid conjuncts in Bengali but these examples contain the constituent Bengali consonants in order and not the conjunct representation. During the training phase, if, for some conjuncts, examples with conjunct representation are outnumbered by examples with constituent consonants representation, the conjunct is removed from the linguistic knowledge base and training examples with such conjunct representation are moved to a Direct example base which contains the English words and their Bengali transliteration. The above two name pairs can then be realigned as

[দে । ব । রা । জ (*devraj*) ↔ de । v । ra । j]

[সো । ম । না । থ (*somnath*) ↔ so । m । na । th]

Otherwise, if such conjuncts are included in the linguistic knowledge base, training examples with constituent consonants representation are to be moved to the Direct example base.

The Bengali names and their English transliterations are split into TUs in such a way that, it results in a one-to-one correspondence after using the linguistic information. But in some cases there exits zero-to-one or many-to-one relationship. An example of Zero-to-One relationship [Φ → h] is the name-pair [আ । ল্লা (*alla*) ↔ a । lla । h] while the name-pair [আ । ই । ভি (*aivy*) ↔ i । vy] is an example of Many-to-One relationship [আ, ই → i]. These bilingual examples should also be included in the Direct example base.

In some cases, the linguistic knowledge apparently solves the mapping problem, but not always. From the name-pair [বরখা (*barkha*) ↔ barkha], the system initially generates the mapping [ব । র । খা ↔ ba । rkha] which is not one-to-one. Then it consults the linguistic knowledge base and breaks up the transliteration unit as (*rkha* → *rk* । *ha*) and generates the final

aligned transliteration pair [ব । র । খা ↔ ba । rk । ha] (since it finds out that *rk* has a valid conjunct representation in Bengali but not *rkh*), which is an incorrect transliteration pair to train the system. It should have been [ব । র । খা ↔ ba । r । kha]. Such type of errors can be detected by following the alignment process from the target side during the training phase. Such training examples may be either manually aligned or maintained in the Direct Example base.

5 Results of the Proposed Models

Approximately 6000 Indian person names have been collected and their English transliterations have been stored manually. This set acts as the training corpus on which the system is trained to generate the collocational statistics. These statistics serve as the decision list classifier to identify the target language TU given the source language TU and its context. The system also includes the linguistic knowledge in the form of valid conjuncts and diphthongs in Bengali and their English representation.

All the models have been tested with an open test corpus of about 1200 Bengali names that contains their English transliterations. The total number of transliteration units (TU) in these 1200 (Sample Size, i.e., S) Bengali names is 4755 (this is the value of L), i.e., on an average a Bengali name contains 4 TUs. The test set was collected from users and it was checked that it does not contain names that are present in the training set. The total number of transliteration unit errors (Err) in the system-generated transliterations and the total number of words erroneously generated ($Err^/$) by the system have been shown in Table 1 for each individual model. The models are evaluated on the basis of the two evaluation metrics, Word Agreement Ratio (WAR) and Transliteration Unit Agreement Ratio (TUAR). The results of the tests in terms of the evaluation metrics are shown in Table 2. The modified joint source-channel model (Model F) that incorporates linguistic knowledge performs best among all the models with a Word Agreement Ratio (WAR) of 69.3% and a Transliteration Unit Agreement Ratio (TUAR) of 89.8%. The joint source-channel model with linguistic knowledge (Model D) has not performed well in the Bengali-English machine transliteration whereas the trigram model (Model E) needs further attention as its result are comparable to the modified joint source-channel model (Model F). All the models were also tested for back-transliteration, i.e., English to Bengali transliteration, with an open test corpus of 1000 English names that contain their Bengali transliterations. The results of these tests in terms of the evaluation metrics WAR and TUAR are shown in Table 3. It is observed that the modified joint source-channel model performs best in back-transliteration with a WAR of 67.9% and a TUAR of 89%.

Model	Error in TUs (Err)	Error words ($Err^/$)
A	990	615
B	795	512
C	880	532
D	814	471
E	604	413
F	486	369

Table 1: Value of Err and $Err^/$ for each model (B2E transliteration)

Model	WAR (in %)	TUAR (in %)
A	48.8	79.2
B	57.4	83.3
C	55.7	81.5
D	60.8	82.9
E	65.6	87.3
F	69.3	89.8

Table 2: Results with Evaluation Metrics (B2E transliteration)

Model	WAR (in %)	TUAR (in %)
A	49.6	79.8
B	56.2	83.8
C	53.9	82.2
D	58.2	83.2
E	64.7	87.5
F	67.9	89.0

Table 3: Results with Evaluation Metrics (E2B transliteration)

6. Conclusion

It has been observed that the modified joint source-channel model with linguistic knowledge performs best in terms of Word Agreement Ratio (WAR) and Transliteration Unit Agreement Ratio (TUAR). Detailed examination of the

evaluation results reveals that Bengali has separate short and long vowels and the corresponding matra representation while these may be represented in English by the same vowel. It has been observed that most of the errors are at the *matra* level i.e., a short matra might have been replaced by a long matra or vice versa. More linguistic knowledge is necessary to disambiguate the short and the long vowels and the matra representation in Bengali. The system includes conjuncts and diphthongs as part of the linguistic knowledge base. Triphthongs or tetraphthongs usually do not appear in Indian names. But, inclusion of them will enable the system to transliterate those few names that may include them. The models are to be trained further on sets of additional person names from other geographic areas. Besides person names, location and organization names are also to be used for training the proposed models.

Acknowledgement

Our thanks go to Council of Scientific and Industrial Research, Human Resource Development Group, New Delhi, India for supporting Sudip Kumar Naskar under Senior Research Fellowship Award (9/96(402) 2003-EMR-I).

References

Abdul Jaleel Nasreen and Leah S. Larkey. 2003. *Statistical Transliteration for English-Arabic Cross Language Information Retrieval*. Proceedings of the Twelfth International Conference on Information and Knowledge Management (CIKM 2003), New Orleans, USA, 139-146.

Al-Onaizan Y. and Knight K. 2002a. *Named Entity Translation: Extended Abstract.* Proceedings of the Human Language Technology Conference (HLT 2002), 122-124.

Al-Onaizan Y. and Knight K.2002b. *Translating Named Entities Using Monolingual and Bilingual Resources.* Proceedings of the 40th Annual Meeting of the ACL (ACL 2002), 400-408.

Al-Onaizan Y. and Knight K. 2002c. *Machine Transliteration of Names in Arabic Text.* Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages.

Arbabi Mansur, Scott M. Fischthal, Vincent C. Cheng, and Elizabeth Bar. 1994. *Algorithms for Arabic name transliteration*. IBM Journal of Research and Development, 38(2): 183-193.

Crego J.M., Marino J.B. and A. de Gispert. 2005. *Reordered Search and Tuple Unfolding for Ngram-based SMT.* Proceedings of the MT-Summit X, Phuket, Thailand, 283-289.

Marino J. B., Banchs R., Crego J. M., A. de Gispert, P. Lambert, J. A. Fonollosa and M. Ruiz, *Bilingual N-gram Statistical Machine Translation.* Proceedings of the MT-Summit X, Phuket, Thailand, 275-282.

Goto I., N. Kato, N. Uratani, and T. Ehara. 2003. *Transliteration considering Context Information based on the Maximum Entropy Method*. Proceeding of the MT-Summit IX, New Orleans, USA, 125–132.

Haizhou Li, Zhang Min, Su Jian. 2004. *A Joint Source-Channel Model for Machine Transliteration*. Proceedings of the 42nd Annual Meeting of the ACL (ACL 2004), Barcelona, Spain, 159-166.

Jung Sung Young, Sung Lim Hong, and Eunok Paek. 2000. *An English to Korean Transliteration Model of Extended Markov Window*. Proceedings of COLING 2000, 1, 383-389.

Knight K. and J. Graehl. 1998. *Machine Transliteration*, Computational Linguistics, 24(4): 599-612.

Meng Helen M., Wai-Kit Lo, Berlin Chen and Karen Tang. 2001. *Generating Phonetic Cognates to handle Name Entities in English-Chinese Cross-language Spoken Document Retrieva*l. Proceedings of the Automatic Speech Recognition and Understanding (ASRU) Workshop, Trento, Italy.

Stalls, Bonnie Glover and Knight K. 1998. *Translating names and technical terms in Arabic text*. Proceedings of the COLING/ACL Workshop on Computational Approaches to Semitic Languages, Montral, Canada, 34-41.

Virga Paola and Sanjeev Khudanpur. 2003. *Transliteration of Proper Names in Crosslingual Information Retrieval*. Proceedings of the ACL 2003 Workshop on Multilingual and Mixed-language Named Entity Recognition, Sapporo, Japan, 57-60.

Chinese-English Term Translation Mining Based on Semantic Prediction

Gaolin Fang, Hao Yu, and Fumihito Nishino
Fujitsu Research and Development Center, Co., LTD. Beijing 100016, China
{glfang, yu, nishino}@cn.fujitsu.com

Abstract

Using abundant Web resources to mine Chinese term translations can be applied in many fields such as reading/writing assistant, machine translation and cross-language information retrieval. In mining English translations of Chinese terms, how to obtain effective Web pages and evaluate translation candidates are two challenging issues. In this paper, the approach based on semantic prediction is first proposed to obtain effective Web pages. The proposed method predicts possible English meanings according to each constituent unit of Chinese term, and expands these English items using semantically relevant knowledge for searching. The refined related terms are extracted from top retrieved documents through feedback learning to construct a new query expansion for acquiring more effective Web pages. For obtaining a correct translation list, a translation evaluation method in the weighted sum of multi-features is presented to rank these candidates estimated from effective Web pages. Experimental results demonstrate that the proposed method has good performance in Chinese-English term translation acquisition, and achieves 82.9% accuracy.

1 Introduction

The goal of Web-based Chinese-English (C-E) term translation mining is to acquire translations of terms or proper nouns which cannot be looked up in the dictionary from the Web using a statistical method, and then construct an application system for reading/writing assistant (e.g., 三国演义→The Romance of Three Kingdoms). During translating or writing foreign language articles, people usually encounter terms, but they cannot obtain native translations after many lookup efforts. Some skilled users perhaps resort to a Web search engine, but a large amount of retrieved irrelevant pages and redundant information hamper them to acquire effective information. Thus, it is necessary to provide a system to automatically mine translation knowledge of terms using abundant Web information so as to help users accurately read or write foreign language articles.

The system of Web-based term translation mining has many applications. 1) Reading/writing assistant. 2) The construction tool of bilingual or multilingual dictionary for machine translation. The system can not only provide translation candidates for compiling a lexicon, but also rescore the candidate list of the dictionary. We can also use English as a medium language to build a lexicon translation bridge between two languages with few bilingual annotations (e.g., Japanese and Chinese). 3) Provide the translations of unknown queries in cross-language information retrieval (CLIR). 4) As one of the typical application paradigms of the combination of CLIR and Web mining.

Automatic acquisition of bilingual translations has been extensively researched in the literature. The methods of acquiring translations are usually summarized as the following six categories. 1) Acquiring translations from parallel corpora. To reduce the workload of manual annotations, researchers have proposed different methods to automatically collect parallel corpora of different language versions from the Web (Kilgarriff, 2003). 2) Acquiring translations from non-parallel corpora (Fung, 1997; Rapp, 1999). It is based on the clue that the context of source term is very similar to that of target translation in a large amount of corpora. 3) Acquiring translations from a combination of translations of constituent words (Li et al., 2003). 4) Acquiring translations using cognate matching (Gey, 2004)

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 199–206,
Sydney, July 2006.

or transliteration (Seo et al., 2004). This method is very suitable for the translation between two languages with some intrinsic relationships, e.g., acquiring translations from Japanese to Chinese or from Korean to English. 5) Acquiring translations using anchor text information (Lu et al., 2004). 6) Acquiring translations from the Web. When people use Asia language (Chinese, Japanese, and Korean) to write, they often annotate associated English meanings after terms. With the development of Web and the open of accessible electronic documents, digital library, and scientific articles, these resources will become more and more abundant. Thus, acquiring term translations from the Web is a feasible and effective way. Nagata et al. (2001) proposed an empirical function of the byte distance between Japanese and English terms as an evaluation criterion to extract translations of Japanese words, and the results could be used as a Japanese-English dictionary.

Cheng et al. (2004) utilized the Web as the corpus source to translate English unknown queries for CLIR. They proposed context-vector and chi-square methods to determine Chinese translations for unknown query terms via mining of top 100 search-result pages from Web search engines.

Zhang and Vines (2004) proposed using a Web search engine to obtain translations of Chinese out-of-vocabulary terms from the Web to improve CLIR performance. The method used Chinese as query items, and retrieved previous 100 document snippets by Google, and then estimated possible translations using co-occurrence information.

From the review above, we know that previous related researches didn't concern the issue how to obtain effective Web pages with bilingual annotations, and they mainly utilized the frequency feature as the clue to mine the translation. In fact, previous 100 Web results seldom contain effective English equivalents. Apart from the frequency information, there are some other features such as distribution, length ratio, distance, keywords, key symbols and boundary information which have very important impacts on term translation mining. In this paper, the approach based on semantic prediction is proposed to obtain effective Web pages; for acquiring a correct translation list, the evaluation strategy in the weighted sum of multi-features is employed to rank the candidates.

The remainder of this paper is organized as follows. In Section 2, we give an overview of the system. Section 3 proposes effective Web page collection. In Section 4, we introduce translation candidate construction and noise solution. Section 5 presents candidate evaluation based on multi-features. Section 6 shows experimental results. The conclusion is drawn in the last section.

2 System Overview

The C-E term translation mining system based on semantic prediction is illustrated in Figure 1.

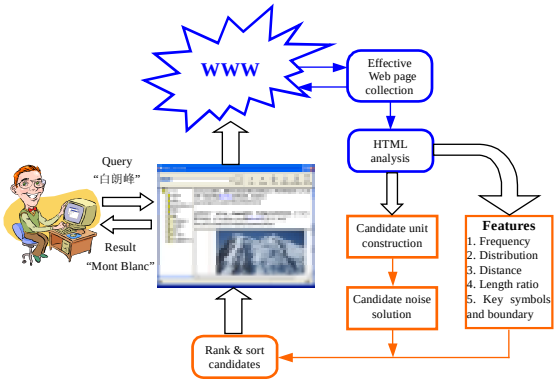


Figure 1. The Chinese-English term translation mining system based on semantic prediction

The system consists of two parts: Web page handling and term translation mining. Web page handling includes effective Web page collection and HTML analysis. The function of effective Web page collection is to collect these Web pages with bilingual annotations using semantic prediction, and then these pages are inputted into HTML analysis module, where possible features and text information are extracted. Term translation mining includes candidate unit construction, candidate noise solution, and rank&sort candidates. Translation candidates are formed through candidate unit construction module, and then we analyze their noises and propose the corresponding methods to handle them. At last, the approach using multi-features is employed to rank these candidates.

Correctly exploring all kinds of bilingual annotation forms on the Web can make a mining system extract comprehensive translation results. After analyzing a large amount of Web page examples, translation distribution forms is summarized as six categories in Figure 2: 1) Direct annotation (a). some have nothing (a1), and some have symbol marks (a2, a3) between the pair; 2) Separate annotation. There are English letters (b1) or some Chinese words (b2, b3) between the pair; 3) Subset form (c); 4) Table form (d); 5) List form (e); and 6) Explanation form (f).

Figure 2. The examples of translation distribution forms

3 Effective Web page collection

For mining the English translations of Chinese terms and proper names, we must obtain effective Web pages, that is, collecting these Web pages that contain not only Chinese characters but also the corresponding English equivalents. However, in a general Web search engine, when you input a Chinese technical term, the number of retrieved relevant Web pages is very large. It is infeasible to download all the Web pages because of a huge time-consuming process. If only the 100 abstracts of Web pages are used for the translation estimation just as in the previous work, effective English equivalent words are seldom contained for most Chinese terms in our experiments, for example: “三国演义, 三好学生, 百慕大三角, 车牌号”. In this paper, a feasible method based on semantic prediction is proposed to automatically acquire effective Web pages. In the proposed method, possible English meanings of every constituent unit of a Chinese term are predicted and further expanded by using semantically relevant knowledge, and these expansion units with the original query are inputted to search bilingual Web pages. In the retrieved top-20 Web pages, feedback learning is employed to extract more semantically-relevant terms by frequency and average length. The refined expansion terms, together with the original query, are once more sent to retrieve effective relevant Web pages.

3.1 Term expansion

Term expansion is to use predictive semantically-relevant terms of target language as the expansion of queries, and therefore resolve the issue that top retrieved Web pages seldom contain effective English annotations. Our idea is based on the assumption that the meanings of Chinese technical terms aren’t exactly known just through their constituent characters and words, but the closely related semantics and vocabulary information may be inferred and predicted. For example, the corresponding unit translations of a term “三国演义” are respectively: three(三), country, nation(国), act, practice(演), and meaning, justice(义). As seen from these English translations, we have a general impression of “things about three countries”. After expanding, the query item for the example above becomes "三国演义"+ (three | country | nation | act | practice | meaning | justice). The whole procedure consists of three steps: unit segmentation, item translation knowledge base construction, and expansion knowledge base evaluation.

Unit segmentation. Getting the constituent units of a technical term is a segmentation procedure. Because most Chinese terms consist of out-of-vocabulary words or meaningless characters, the performance using general word segmentation programs is not very desirable. In this paper, a segmentation method is employed to handle term segmentation so that possible meaningful constituent units are found. In the inner structure of proper nouns or terms, the rightmost unit usually contains a headword to reflect the major meaning of the term. Sometimes, the modifier starts from the leftmost point of a term to form a multi-character unit. As a result, forward maximum matching and backward maximum matching are respectively conducted on the term, and all the overlapped segmented units are added to candidate items. For example, for the term “abcd”, forward segmented units are “ab cd”, backward are “a bcd”, so “ab cd a bcd” will be viewed as our segmented items.

Item translation knowledge base construction. Because the segmented units of a technical term or proper name often consist of abbreviation items with shorter length, limited translations provided by general dictionaries often cannot satisfy the demand of translation prediction. Here, a semantic expansion based method is proposed to construct item translation knowledge base. In this method, we only keep these nouns or adjective items consisting of 1-3 characters in the dictionary. If an item length is greater than two characters and contains any item in the knowledge base, its translation will be added as translation candidates of this item. For example, the Chinese term “流通股” can be segmented into the units “流通” and “股”, where “股” has only two English meanings “section, thigh” in the dictionary. However, we can derive its meaning us-

ing the longer word including this item such as “股东, 股票”. Thus, their respective translations “stock, stockholder” are added into the knowledge base list of “股” (see Figure 3).

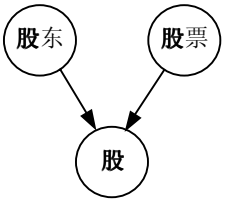


Figure 3. An expansion example in the dictionary knowledge base

Expansion knowledge base evaluation. To avoid over-expanding of translations for one item, using the retrieved number from the Web as our scoring criterion is employed to remove irrelevant expansion items and rank those possible candidates. For example, “股” and its expansion translation “stock” are combined as a new query “股 stock –股票”. It is sent to a general search engine like Google to obtain the count number, where only the co-occurrence of “股” and “stock” excluding the word “股票” is counted. The retrieved number is about 316000. If the occurrence number of an item is lower than a certain threshold (100), the evaluated translation will not be added to the item in the knowledge base. Those expanded candidates for the item in the dictionary are sorted through their retrieved number.

3.2 Feedback learning

Though pseudo-relevance feedback (PRF) has been successfully used in the information retrieval (IR), whether PRF in single-language IR or pre-translation PRF and post-translation PRF in CLIR, the feedback results are from source language to source language or target language to target language, that is, the language of feedback units is same as the retrieval language. Our novel is that the input language (Chinese) is different from the feedback target language (English), that is, realizing the feedback from source language to target language, and this feedback technique is also first applied to the term mining field.

After the expansion of semantic prediction, the predicted meaning of an item has some deviations with its actual sense, so the retrieved documents are perhaps not our expected results. In this paper, a PRF technique is employed to acquire more accurate, semantically relevant terms. At first, we collect top-20 documents from search results after term expansion, and then select target language units from these documents, get language units from these documents, which are highly related with the original query in source language. However, how to effectively select these units is a challenging issue. In the literature, researchers have proposed different methods such as Rocchio’s method or Robertson’s probabilistic method to solve this problem. After some experimental comparisons, a simple evaluation method using term frequency and average length is presented in this paper. The evaluation method is defined as follows:

$$w(t) = f(t) + \frac{1}{\Delta(t)+1}, \text{ where } \Delta(t) = \frac{\sum_{i=1}^{N} D_i(s,t)}{N} \quad (1)$$

$\Delta(t)$ represents the average length between the source word s and the target candidate t. If the greater that the average length is, the relevance degree between source terms and candidates will become lower. The purpose of adding $\Delta(t)$ to 1 is to avoid the divide overflow in the case that the average length is equal to zero. $D_i(s,t)$ denotes the byte distance between source words and target candidates, and N represents the total number of candidate occurrences in the estimated Web pages. This evaluation method is very suitable for the discrimination of these words with lower, but same term frequencies. In the ranked candidates after PRF feedback, top-5 candidates are selected as our refined expansion items. In the previous example, the refined expansion items are: Kingdoms, Three, Romance, Chinese, Traditional. These refined expansion terms, together with the original query, "三国演义"+(Kingdoms | Three | Romance | Chinese | Traditional) are once more sent to retrieve relevant results, which are viewed as effective Web pages used in the process of the following estimation.

4 Translation candidate construction and noise solution

The goal of translation candidate construction is to construct and mine all kinds of possible translation forms of terms from the Web, and effectively estimate their feature information such as frequency and distribution. In the transferred text, we locate the position of a query keyword, and then obtain a 100-byte window with keyword as the center. In this window, each English word is built as a beginning index, and then string candidates are constructed with the increase of string in the form of one English word unit. String candidates are indexed in the database with hash and binary search method. If there exists the same item as the inputted candidate, its frequency is increased by 1, otherwise, this candidate is added

to this position of the database. After handling one Web page, the distribution information is also estimated at the same time. In the programming implementation, the table of stop words and some heuristic rules of the beginning and end with respect to the keyword position are employed to accelerate the statistics process.

The aim of noise solution is to remove these irrelevant items and redundant information formed in the process of mining. These noises are defined as the following two categories.

1) Subset redundancy. The characteristic is that this item is a subset of one item, but its frequency is lower than that item. For example, "车牌号：License plate number (6), License plate (5)", where the candidate "License plate" belongs to subset redundancy. They should be removed.

2) Affix redundancy. The characteristic is that this item is the prefix or suffix of one item, but its frequency is greater than that item. For example, 1. "三国演义: Three Kingdoms (30), Romance of the Three Kingdoms (22), The Romance of Three Kingdoms (7)", 2. "蓝筹股: Blue Chip (35), Blue Chip Economic Indicators (10)". In Example 1, the item "Three Kingdoms" is suffix redundancy and should be removed. In Example 2, the term "Blue Chip" is in accord with the definition of prefix redundancy information, but this term is a correct translation candidate. Thus, the problem of affix redundancy information is so complex that we need an evaluation method to decide to retain or drop the candidate.

To deal with subset redundancy and affix redundancy information, sort-based subset deletion and mutual information methods are respectively proposed. More details refer to our previous paper (Fang et al., 2005).

5 Candidate evaluation based on multi-features

5.1 Possible features for translation pairs

Through analyzing mass Web pages, we obtain the following possible features that have important influences on term translation mining. They include: 1) candidate frequency and its distribution in different Web pages, 2) length ratio between source terms and target candidates (S-T), 3) distance between S-T, and 4) keywords, key symbols and boundary information between S-T.

1) Candidate frequency and its distribution

Translation candidate frequency is the most important feature and is the basis of decision-making. Only the terms whose frequencies are greater than a certain threshold are further considered as candidates in our system. Distribution feature reflects the occurrence information of one candidate in different Webs. If the distribution is very uniform, this candidate will more possibly become as the translation equivalent with a greater weight. This is also in accord with our intuition. For example, the translation candidates of the term "认股期权" include "put option" and "short put", and their frequencies are both 5. However, their distributions are "1, 1, 1, 1, 1" and "2, 2, 1". The distribution of "put option" is more uniform, so it will become as a translation candidate of "认股期权" with a greater weight.

2) Length ratio between S-T

The length ratio between S-T should satisfy certain constraints. Only the word number of a candidate falls within a certain range, the possibility of becoming a translation is great.

To estimate the length ratio relation between S-T, we conduct the statistics on the database with 5800 term translation pairs. For example, when Chinese term has three characters, i.e. W=3, the probability of English translations with two words is largest, about P(E=2 |W =3)= 78%, and there is nearly no occurrence out of the range of 1-4. Thus, different weights can be impacted on different candidates by using statistical distribution information of length ratio. The weight contributing to the evaluation function is set according to these estimated probabilities in the experiments.

3) Distance between S-T

Intuitively, if the distance between S-T is longer, the probability of being a translation pair will become smaller. Using this knowledge we can alleviate the effect of some noises through impacting different weights when we collect possible correct candidates far from the source term.

To estimate the distance between S-T, experiments are carried on 5800*200 pages with 5800 term pairs, and statistical results are depicted as the histogram of distances in Figure 4.

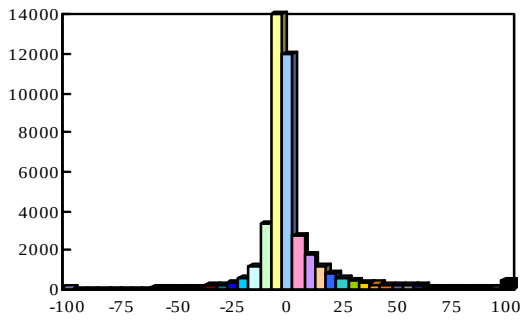


Figure 4. The histogram of distances between S-T

In the figure, negative value represents that English translation located in front of the Chinese term, and positive value represents English translation is behind the Chinese term. As shown from the figure, we know that most candidates are distributed in the range of -60-60 bytes, and few occurrences are out of this range. The numbers of translations appearing in front of the term and after the term are nearly equal. The curve looks like Gaussian probability distribution, so Gaussian models are proposed to model it. By the curve fitting, the parameters of Gaussian models are obtained, i.e. u=1 and sigma=2. Thus, the contribution probability of distance to the ranking function is formulized as $p_D(i,j) = \frac{1}{2\sqrt{2\pi}} e^{-(D(i,j)-1)^2/8}$, where D(*i*,*j*) represents the byte distance between the source term *i* and the candidate *j*.

4) Keywords, key symbols and boundary information between S-T

Some Chinese keywords or capital English abbreviation letters between S-T can provide an important clue for the acquisition of possible correct translations. These Chinese keywords include the words such as "中文叫，中文译为，中文名称，中文名称为，中文称为，或称为，又称为，英文叫，英文名为，英文称为，英文全称". The punctuations between S-T can also provide very strong constraints, for example, when the marks "（）( ) [ ]" exist, the probability of being a translation pair will greatly increase. Thus, correctly judging these cases can not only make translation finding results more comprehensive, but also increase the possibility that this candidate is as one of correct translations. Boundary information refers to the fact that the context of candidates on the Web has distinct mark information, for example, the position of transition from continuous Chinese to English, the place with bracket ellipsis and independent units in the HTML text.

5.2 Candidate evaluation method

After translation noise handling, we evaluate candidate translations so that possible candidates get higher scores. The method in the weighted sum of multi-features including: candidate frequency, distribution, length ratio, distance, keywords, key symbols and boundary information between S-T, is proposed to rank the candidates. The evaluation method is formulized as follows:

$$Score(t) = p_L(s,t)\sum_{i=1}^{N}[\lambda_1 \sum_j (p_D(i,j) + \delta(i,j)w) + \lambda_2 \max_j (p_D(i,j) + \delta(i,j)w)], \ \lambda_1 + \lambda_2 = 1 \quad (2)$$

In the equation, *Score(t)* is proportional to $p_L(s,t)$, *N* and $p_D(i,j)$. If the bigger these component values are, the more they contribute to the whole evaluation formula, and correspondingly the candidate has higher score. The length ratio relation $p_L(s,t)$ reflects the proportion relation between S-T as a whole, so its weight will be impacted on the *Score(t)* in the macro-view. The weights are trained through a large amount of technical terms and proper nouns, where each relation corresponds to one probability. *N* denotes the total number of Web pages that contain candidates, and partly reflects the distribution information of candidates in different Web pages. If the greater *N* is, the greater *Score(t)* will become. The distance relation $p_D(i,j)$ is defined as the distance contribution probability of the *j*th source-candidate pair on the *i*th Web pages, which is impacted on every word pair emerged on the Web in the point of micro-view. Its calculation formula is defined in Section 5.1. The weights of λ_1 and λ_2 represent the proportion of term frequency and term distribution, and λ_1 denotes the weight of the total number of one candidate occurrences, and λ_2 represents the weight of counting the nearest distance occurrence for each Web page. $\delta(i,j)w$ is the contribution probability of keywords, key symbols and boundary information. If there are predefined keywords, key symbols, and boundary information between S-T, i.e., $\delta(i,j) = 1$, then the evaluation formula will give a reward *w*, otherwise, $\delta(i,j) = 0$ indicate that there is no impact on the whole equation.

6 Experiments

Our experimental data consist of two sets: 400 C-E term pairs and 3511 C-E term pairs in the financial domain. There is no intersection between the two sets. Each term often consists of 2-8 Chinese characters, and the associated translation contains 2-5 English words. In the test set of 400 terms, there are more than one English translation for every Chinese term, and only one English translation for 3511 term pairs. In the test sets, Chinese terms are inputted to our system on batch, and their corresponding translations are viewed as a criterion to evaluate these mined candidates. The top n accuracy is defined as the

percentage of terms whose top n translations include correct translations in the term pairs. A series of experiments are conducted on the two test sets.

Experiments on the number of feedback pages: To obtain the best parameter of feedback Web pages that influence the whole system accuracy, we perform the experiments on the test set of 400 terms. The number of feedback Web pages is respectively set to 0, 10, 20, 30, and 40. N=1, 3, 5 represent the accuracies of top 1, 3, and 5. From the feedback pages, previous 5 semantically-relevant terms are extracted to construct a new query expansion for retrieving more effective Web pages. Translation candidates are mined from these effective pages, whose accuracy curves are depicted in Figure 5.

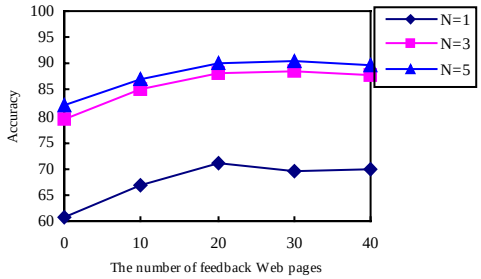


Figure 5. The number of feedback Web pages

As seen from the figure above, when the number of feedback Web pages is 20, the accuracy reaches the best. Thus, the feedback parameter in our experiments is set to 20.

Experiments on the parameter λ_1: In the candidate evaluation method using multi-features, the parameter of λ_1 need be chosen through the experiments. To obtain the best parameter, the experiments are set as follows. The accuracy of top 5 candidates is viewed as a performance criterion. The parameters are respectively set from 0 to 1 with the increase of 0.1 step. The results are listed in Figure 6. As seen from the figure, λ_1=0.4 is best parameter, and therefore λ_2 =0.6. In the following experiments, the parameters are all set to this value.

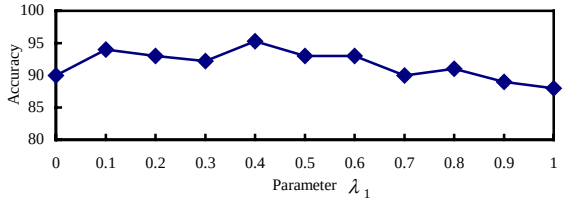


Figure 6. The relation between the parameter λ_1 and the accuracy

Experiments on the test set of 400 terms using different methods: The methods respectively without prediction(NP), with prediction(P), with prediction and feedback(PF) only using term frequency (TM), and with prediction and feedback using multi-features(PF+MF) are employed on the test set of 400 terms. The results are listed in Table 1. As seen from this table, if there is no semantic prediction, the obtained translations from Web pages are about 48% in the top 30 candidates. This is because general search engines will retrieve more relevant Chinese Web pages rather than those effective pages including English meanings. Thus, the semantic prediction method is employed. Experiments demonstrate the method with semantic prediction distinctly improves the accuracy, about 36.8%. To further improve the performance, the feedback learning technique is proposed, and it increases the average accuracy of 6.5%. Though TM is very effective in mining the term translation, the multi-feature method fully utilizes the context of candidates, and therefore obtains more accurate results, about 92.8% in the top 5 candidates.

Table 1. The term translation results using different methods

	Top30	**Top10**	**Top5**	**Top3**	**Top1**
NP	48.0	47.5	46.0	44.0	28.0
P	84.8	83.3	82.3	79.3	60.8
PF+TM	91.3	90.8	90.3	88.3	71.0
PF+MF	95.0	94.5	92.8	91.5	78.8

Experiments on a large vocabulary: To validate our system performance, experiments are carried on a large vocabulary of 3511 terms using different methods. One method is to use term frequency (TM) as an evaluation criterion, and the other method is to use multi-features (MF) as an evaluation criterion. Experimental results are shown as follows.

Table 2. The term translation results on a large vocabulary

	Top30	**Top10**	**Top5**	**Top3**	**Top1**
TM	82.5	81.2	78.3	73.5	49.4
MF	89.1	88.4	86.0	82.9	58.2

From Table 2, we know the accuracy with top 5 candidates is about 86.0%. The method using multi-features is better than that of using term frequency, and improves an average accuracy of 7.94%

Some examples of acquiring English translations of Chinese terms are provided in Table 3.

Only top 3 English translations are listed for each Chinese term.

Table 3. Some C-E mining examples

Chinese terms	The list of English translations (Top 3)
三国演义	The Three Kingdoms The Romance of the Three Kingdoms The Romance of Three Kingdoms
三好学生	Merit student "Three Goods" student Excellent League member
蓝筹股	Blue Chip Blue Chips Blue chip stocks
白朗峰	Mont Blanc Mont-Blanc Chamonix Mont-Blanc
百慕大三角	Burmuda Triangle Bermuda Triangle The Bermuda Triangle
车牌号	License plate number Vehicle plate number Vehicle identification no

7 Conclusions

In this paper, the method based on semantic prediction is first proposed to acquire effective Web pages. The proposed method predicts possible meanings according to each constituent unit of Chinese term, and expands these items for searching using semantically relevant knowledge, and then the refined related terms are extracted from top retrieved documents through feedback learning to construct a new query expansion for acquiring more effective Web pages. For obtaining a correct translation list, the translation evaluation method using multi-features is presented to rank these candidates. Experimental results show that this method has good performance in Chinese-English translation acquisition, about 82.9% accuracy in the top 3 candidates.

References

P.J. Cheng, J.W. Teng, R.C. Chen, et al. 2004. Translating unknown queries with web corpora for cross-language information retrieval, Proc. ACM SIGIR, pp. 146-153.

G.L. Fang, H. Yu, and F. Nishino. 2005. Web-Based Terminology Translation Mining, Proc. IJCNLP, pp. 1004-1016.

P. Fung. 1997. Finding terminology translations from nonparallel corpora, Proc. Fifth Annual Workshop on Very Large Corpora (WVLC'97), pp. 192-202.

F.C. Gey. 2004. Chinese and Korean topic search of Japanese news collections, In Working Notes of the Fourth NTCIR Workshop Meeting, Cross-Lingual Information Retrieval Task, pp. 214-218.

A. Kilgarriff and G. Grefenstette. 2003. Introduction to the special issue on the Web as corpus, Computational Linguistics, 29(3): 333-348.

H. Li, Y. Cao, and C. Li. 2003.Using bilingual web data to mine and rank translations, IEEE Intelligent Systems, 18(4): 54-59.

W.H. Lu, L.F. Chien, and H.J. Lee. 2004. Anchor text mining for translation of Web queries: A transitive translation approach, ACM Trans. Information System, 22(2): 242-269.

M. Nagata, T. Saito, and K. Suzuki. 2001. Using the web as a bilingual dictionary, Proc. ACL 2001 Workshop Data-Driven Methods in Machine Translation, pp. 95-102.

R. Rapp. 1999. Automatic identification of word translations from unrelated English and German corpora, Proc. 37th Annual Meeting Assoc. Computational Linguistics, pp. 519-526.

H.C. Seo, S.B. Kim, H.G. Lim and H.C. Rim. 2004. KUNLP system for NTCIR-4 Korean-English cross language information retrieval, In Working Notes of the Fourth NTCIR Workshop Meeting, Cross-Lingual Information Retrieval Task, pp. 103-109.

Y. Zhang and P. Vines. 2004. Using the web for automated translation extraction in cross-language information retrieval, Proc. ACM SIGIR, pp. 162-169.

Automatic Creation of Domain Templates

Elena Filatova[*], **Vasileios Hatzivassiloglou**[†] **and Kathleen McKeown**[*]

[*]Department of Computer Science
Columbia University
{filatova,kathy}@cs.columbia.edu

[†]Department of Computer Science
The University of Texas at Dallas
vh@hlt.utdallas.edu

Abstract

Recently, many Natural Language Processing (NLP) applications have improved the quality of their output by using various machine learning techniques to mine Information Extraction (IE) patterns for capturing information from the input text. Currently, to mine IE patterns one should know in advance the type of the information that should be captured by these patterns. In this work we propose a novel methodology for corpus analysis based on cross-examination of several document collections representing different instances of the same domain. We show that this methodology can be used for automatic domain template creation. As the problem of automatic domain template creation is rather new, there is no well-defined procedure for the evaluation of the domain template quality. Thus, we propose a methodology for identifying what information should be present in the template. Using this information we evaluate the automatically created domain templates through the text snippets retrieved according to the created templates.

1 Introduction

Open-ended question-answering (QA) systems typically produce a response containing a variety of specific facts proscribed by the question type. A biography, for example, might contain the date of birth, occupation, or nationality of the person in question (Duboue and McKeown, 2003; Zhou et al., 2004; Weischedel et al., 2004; Filatova and Prager, 2005). A definition may contain the genus of the term and characteristic attributes (Blair-Goldensohn et al., 2004). A response to a question about a terrorist attack might include the event, victims, perpetrator and date as the templates designed for the Message Understanding Conferences (Radev and McKeown, 1998; White et al., 2001) predicted. Furthermore, the type of information included varies depending on context. A biography of an actor would include movie names, while a biography of an inventor would include the names of inventions. A description of a terrorist event in Latin America in the eighties is different from the description of today's terrorist events.

How does one determine what facts are important for different kinds of responses? Often the types of facts that are important are hand encoded ahead of time by a human expert (e.g., as in the case of MUC templates). In this paper, we present an approach that allows a system to learn the types of facts that are appropriate for a particular response. We focus on acquiring fact-types for events, automatically producing a *template* that can guide the creation of responses to questions requiring a description of an event. The template can be tailored to a specific time period or country simply by changing the document collections from which learning takes place.

In this work, a *domain* is a set of events of a particular type; earthquakes and presidential elections are two such domains. Domains can be instantiated by several *instances* of events of that type (e.g., the earthquake in Japan in October 2004, the earthquake in Afghanistan in March 2002, etc.).[1] The granularity of domains and instances can be altered by examining data at different levels of detail, and domains can be hierarchically structured. An ideal template is a set of attribute-value pairs, with the attributes specifying particular functional roles important for the domain events.

In this paper we present a method of domain-independent on-the-fly template creation. Our method is completely automatic. As input it requires several document collections describing domain instances. We cross-examine the input instances, we identify verbs important for the majority of instances and relationships containing these verbs. We generalize across multiple domain instances to automatically determine which of these relations should be used in the template. We report on data collection efforts and results from four domains. We assess how well the automatically produced templates satisfy users' needs, as manifested by questions collected for these domains.

[1]Unfortunately, NLP terminology is not standardized across different tasks. Two NLP tasks most close to our research are Topic Detection and Tracking (TDT) (Fiscus et al., 1999) and Information Extraction (IE) (Marsh and Perzanowski, 1997). In TDT terminology, our domains are topics and our instances are events. In IE terminology, our domains are scenarios and our domain templates are scenario templates.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 207–214,
Sydney, July 2006.

2 Related Work

Our system automatically generates a template that captures the generally most important information for a particular domain and is reusable across multiple instances of that domain. Deciding what slots to include in the template, and what restrictions to place on their potential fillers, is a knowledge representation problem (Hobbs and Israel, 1994). Templates were used in the main IE competitions, the Message Understanding Conferences (Hobbs and Israel, 1994; Onyshkevych, 1994; Marsh and Perzanowski, 1997). One of the recent evaluations, ACE,[2] uses pre-defined frames connecting event types (e.g., *arrest*, *release*) to a set of attributes. The template *construction* task was not addressed by the participating systems. The domain templates were created manually by experts to capture the structure of the facts sought.

Although templates have been extensively used in information extraction, there has been little work on their automatic design. In the Conceptual Case Frame Acquisition project (Riloff and Schmelzenbach, 1998), extraction patterns, a domain semantic lexicon, and a list of conceptual roles and associated semantic categories for the domain are used to produce multiple-slot case frames with selectional restrictions. The system requires two sets of documents: those relevant to the domain and those irrelevant. Our approach does not require any domain-specific knowledge and uses only corpus-based statistics.

The GISTexter summarization system (Harabagiu and Maiorano, 2002) used statistics over an arbitrary document collection together with semantic relations from WordNet. The created templates heavily depend on the topical relations encoded in WordNet. The template models an input collection of documents. If there is only one domain instance described in the input than the template is created for this particular instance rather than for a domain. In our work, we learn domain templates by cross-examining several collections of documents on the same topic, aiming for a general domain template. We rely on relations cross-mentioned in different instances of the domain to automatically prioritize roles and relationships for selection.

Topic Themes (Harabagiu and Lăcătuşu, 2005) used for multi-document summarization merge various arguments corresponding to the same semantic roles for the semantically identical verb phrases (e.g., *arrests* and *placed under arrest*).

Atomic events also model an input document collection (Filatova and Hatzivassiloglou, 2003) and are created according to the statistics collected for co-occurrences of named entity pairs linked through actions. GISTexter, atomic events, and Topic Themes were used for modeling a collection of documents rather than a domain.

In other closely related work, Sudo et al. (2003) use frequent dependency subtrees as measured by TF*IDF to identify named entities and IE patterns important for a given domain. The goal of their work is to show how the techniques improve IE pattern acquisition. To do this, Sudo et al. constrain the retrieval of relevant documents for a MUC scenario and then use unsupervised learning over descriptions within these documents that match specific types of named entities (e.g., *Arresting Agency, Charge*), thus enabling learning of patterns for specific templates (e.g., the Arrest scenario). In contrast, the goal of our work is to show how similar techniques can be used to learn what information is important for a given domain or event and thus, should be included into the domain template. Our approach allows, for example, learning that an arrest along with other events (e.g., attack) is often part of a terrorist event. We do not assume any prior knowledge about domains. We demonstrate that frequent subtrees can be used not only to extract specific named entities for a given scenario but also to learn domain-important relations. These relations link domain actions and named entities as well as general nouns and words belonging to other syntactic categories.

Collier (1998) proposed a fully automatic method for creating templates for information extraction. The method relies on Luhn's (1957) idea of locating statistically significant words in a corpus and uses those to locate the sentences in which they occur. Then it extracts Subject-Verb-Object patterns in those sentences to identify the most important interactions in the input data. The system was constructed to create MUC templates for *terrorist attacks*. Our work also relies on corpus statistics, but we utilize arbitrary syntactic patterns and explicitly use multiple domain instances. Keeping domain instances separated, we cross-examine them and estimate the importance of a particular information type in the domain.

[2] http://www.nist.gov/speech/tests/ace/index.htm

3 Our Approach to Template Creation

After reading about presidential elections in different countries on different years, a reader has a general picture of this process. Later, when reading about a new presidential election, the reader already has in her mind a set of questions for which she expects answers. This process can be called *domain modeling*. The more instances of a particular domain a person has seen, the better understanding she has about what type of information should be expected in an unseen collection of documents discussing a new instance of this domain.

Thus, we propose to use a set of document collections describing different instances within one domain to learn the general characteristics of this domain. These characteristics can be then used to create a domain template. We test our system on four domains: airplane crashes, earthquakes, presidential elections, terrorist attacks.

4 Data Description

4.1 Training Data

To create training document collections we used BBC Advanced Search[3] and submitted queries of the type ⟨*domain title* + *country*⟩. For example, ⟨"presidential election" USA⟩.

In addition, we used BBC's Advanced Search date filter to constrain the results to different date periods of interest. For example, we used known dates of elections and allowed a search for articles published up to five days before or after each such date. At the same time for the terrorist attacks or earthquakes domain the time constraints we submitted were the day of the event plus ten days.

Thus, we identify several instances for each of our four domains, obtaining a document collection for *each* instance. E.g., for the earthquake domain we collected documents on the earthquakes in Afghanistan (March 25, 2002), India (January 26, 2001), Iran (December 26, 2003), Japan (October 26, 2004), and Peru (June 23, 2001). Using this procedure we retrieve training document collections for 9 instances of airplane crashes, 5 instances of earthquakes, 13 instances of presidential elections, and 6 instances of terrorist attacks.

4.2 Test Data

To test our system, we used document clusters from the Topic Detection and Tracking (TDT) corpus (Fiscus et al., 1999). Each TDT topic has a topic label, such as *Accidents* or *Natural Disasters*.[4] These categories are broader than our domains. Thus, we manually filtered the TDT topics relevant to our four training domains (e.g., Accidents matching Airplane Crashes). In this way, we obtained TDT document clusters for 2 instances of airplane crashes, 3 instances of earthquakes, 6 instances of presidential elections and 3 instances of terrorist attacks. The number of the documents corresponding to the instances varies greatly (from two documents for one of the earthquakes up to 156 documents for one of the terrorist attacks). This variation in the number of documents per topic is typical for the TDT corpus. Many of the current approaches of domain modeling collapse together different instances and make the decision on what information is important for a domain based on this generalized corpus (Collier, 1998; Barzilay and Lee, 2003; Sudo et al., 2003). We, on the other hand, propose to cross-examine these instances keeping them separated. Our goal is to eliminate dependence on how well the corpus is balanced and to avoid the possibility of greater impact on the domain template of those instances which have more documents.

5 Creating Templates

In this work we build domain templates around verbs which are estimated to be important for the domains. Using verbs as the starting point we identify semantic dependencies within sentences. In contrast to deep semantic analysis (Fillmore and Baker, 2001; Gildea and Jurafsky, 2002; Pradhan et al., 2004; Harabagiu and Lăcătuşu, 2005; Palmer et al., 2005) we rely only on corpus statistics. We extract the most frequent syntactic subtrees which connect verbs to the lexemes used in the same subtrees. These subtrees are used to create domain templates.

For each of the four domains described in Section 4, we automatically create domain templates using the following algorithm.

Step 1: *Estimate what verbs are important for the domain under investigation*. We initiate our algorithm by calculating the probabilities for all the verbs in the document collection for one domain — e.g., the collection containing all the instances in the domain of airplane crashes. We

[3] http://news.bbc.co.uk/shared/bsp/search2/advanced/news_ifs.stm

[4] In our experiments we analyze TDT topics used in TDT-2 and TDT-4 evaluations.

discard those verbs that are stop words (Salton, 1971). To take into consideration the distribution of a verb among different instances of the domain, we normalize this probability by its *VIF* value (verb instance frequency), specifying in how many domain instances this verb appears.

$$Score(vb_i) = \frac{count_{vb_i}}{\sum_{vb_j \in comb\ coll} count_{vb_j}} \times VIF(vb_i) \quad (1)$$

$$VIF(vb_i) = \frac{\#\ of\ domain\ instances\ containing\ vb_i}{\#\ of\ all\ domain\ instances} \quad (2)$$

These verbs are estimated to be the most important for the combined document collection for all the domain instances. Thus, we build the domain template around these verbs. Here are the top ten verbs for the *terrorist attack* domain:

killed, *told*, *found*, *injured*, *reported*, *happened*, *blamed*, *arrested*, *died*, *linked*.

Step 2: *Parse those sentences which contain the top 50 verbs*. After we identify the 50 most important verbs for the domain under analysis, we parse all the sentences in the domain document collection containing these verbs with the Stanford syntactic parser (Klein and Manning, 2002).

Step 3: *Identify most frequent subtrees containing the top 50 verbs*. A domain template should contain not only the most important actions for the domain, but also the entities that are linked to these actions or to each other through these actions. The lexemes referring to such entities can potentially be used within the domain template slots. Thus, we analyze those portions of the syntactic trees which contain the verbs themselves plus other lexemes used in the same subtrees as the verbs. To do this we use FREQuent Tree miner.[5] This software is an implementation of the algorithm presented by (Abe et al., 2002; Zaki, 2002), which extracts frequent ordered subtrees from a set of ordered trees. Following (Sudo et al., 2003) we are interested only in the lexemes which are near neighbors of the most frequent verbs. Thus, we look only for those subtrees which contain the verbs themselves and from four to ten tree nodes, where a node is either a syntactic tag or a lexeme with its tag. We analyze not only NPs which correspond to the subject or object of the verb, but other syntactic constituents as well. For example, PPs can potentially link the verb to locations or dates, and we want to include this information into the template. Table 1 contains a sample of subtrees for the *terrorist attack* domain mined from the sentences containing the verb *killed*. The first column of Table 1 shows how many nodes are in the subtree.

[5] http://chasen.org/~taku/software/freqt/

nodes	subtree
8	(SBAR(S(VP(VBD killed)(NP(QP(IN at))(NNS people)))))
8	(SBAR(S(VP(VBD killed)(NP(QP(JJS least))(NNS people)))))
5	(VP(ADVP)(VBD killed)(NP(NNS people)))
6	(VP(VBD killed)(NP(ADJP(JJ many))(NNS people)))
5	(VP(VP(VBD killed)(NP(NNS people))))
7	(VP(ADVP(NP))(VBD killed)(NP(CD 34)(NNS people)))
6	(VP(ADVP)(VBD killed)(NP(CD 34)(NNS people)))

Table 1: Sample subtrees for the *terrorist attack* domain.

Step 4: *Substitute named entities with their respective tags*. We are interested in analyzing a whole domain, not just an instance of this domain. Thus, we substitute all the named entities with their respective tags, and all the exact numbers with the tag NUMBER. We speculate that subtrees similar to those presented in Table 1 can be extracted from a document collection representing any instance of a terrorist attack, with the only difference being the exact number of causalities. Later, however, we analyze the domain instances separately to identity information typical for the domain. The procedure of substituting named entities with their respective tags previously proved to be useful for various tasks (Barzilay and Lee, 2003; Sudo et al., 2003; Filatova and Prager, 2005). To get named entity tags we used BBN's IdentiFinder (Bikel et al., 1999).

Step 5: *Merge together the frequent subtrees*. Finally, we merge together those subtrees which are identical according to the information encoded within them. This is a key step in our algorithm which allows us to bring together subtrees from different instances of the same domain. For example, the information rendered by all the subtrees from the bottom part of Table 1 is identical. Thus, these subtrees can be merged into one which contains the longest common pattern:

(VBD killed)(NP(NUMBER)(NNS people))

After this merging procedure we keep only those subtrees for which each of the domain instances has at least one of the subtrees from the initial set of subtrees. This subtree should be used in the instance at least twice. At this step, we make sure that we keep in the template only the information which is generally important for the domain rather than only for a fraction of instances in this domain. We also remove all the syntactic tags as we want to make this pattern as general for the domain as possible. A pattern without syntactic dependencies contains a verb together with a prospective tem-

plate slot corresponding to this verb:

killed: (NUMBER) (NNS people)

In the above example, the prospective template slots appear after the verb *killed*. In other cases the domain slots appear in front of the verb. Two examples of such slots, for the *presidential election* and *earthquake* domains, are shown below:

(PERSON) won
(NN earthquake) struck

The above examples show that it is not enough to analyze only named entities, general nouns contain important information as well. We term the structure consisting of a verb together with the associated slots a *slot structure*. Here is a part of the slot structure we get for the verb *killed* after cross-examination of the *terrorist attack* instances:

killed (NUMBER) (NNS people)
(PERSON) killed
(NN suicide) killed

Slot structures are similar to verb frames, which are manually created for the PropBank annotation (Palmer et al., 2005).[6] An example of the PropBank frame for the verb *to kill* is:

Roleset kill.01 "cause to die":
Arg0:killer
Arg1:corpse
Arg2:instrument

The difference between the slot structure extracted by our algorithm and the PropBank frame slots is that the frame slots assign a semantic role to each slot, while our algorithm gives either the type of the named entity that should fill in this slot or puts a particular noun into the slot (e.g., ORGANIZATION, earthquake, people). An ideal domain template should include semantic information but this problem is outside of the scope of this paper.

Step 6: *Creating domain templates*. After we get all the frequent subtrees containing the top 50 domain verbs, we merge all the subtrees corresponding to the same verb and create a slot structure for every verb as described in Step 5. The union of such slot structures created for all the important verbs in the domain is called the *domain template*. From the created templates we remove the slots which are used in all the domains. For example,

(PERSON) told.

□

The presented algorithm can be used to create a template for any domain. It does not require predefined domain or world knowledge. We learn domain templates from cross-examining document collections describing different instances of the domain of interest.

[6] http://www.cs.rochester.edu/~gildea/Verbs/

6 Evaluation

The task we deal with is new and there is no well-defined and standardized evaluation procedure for it. Sudo et al. (2003) evaluated how well their IE patterns captured named entities of three predefined types. We are interested in evaluating how well we capture the major actions as well as their constituent parts.

There is no set of domain templates which are built according to a unique set of principles against which we could compare our automatically created templates. Thus, we need to create a gold standard. In Section 6.1, we describe how the gold standard is created. Then, in Section 6.2, we evaluate the quality of the automatically created templates by extracting clauses corresponding to the templates and verifying how many answers from the questions in the gold standard are answered by the extracted clauses.

6.1 Stage 1. Information Included into Templates: Interannotator Agreement

To create a gold standard we asked people to create a list of questions which indicate what is important for the domain description. Our decision to aim for the lists of questions and not for the templates themselves is based on the following considerations: first, not all of our subjects are familiar with the field of IE and thus, do not necessarily know what an IE template is; second, our goal for this evaluation is to estimate interannotator agreement for capturing the important aspects for the domain and not how well the subjects agree on the template structure.

We asked our subjects to think of their experience of reading newswire articles about various domains.[7] Based on what they remember from this experience, we asked them to come up with a list of questions about a particular domain. We asked them to come up with at most 20 questions covering the information they will be looking for given an unseen news article about a new event in the domain. We did not give them any input information about the domain but allowed them to use any sources to learn more information about the domain.

We had ten subjects, each of which created one list of questions for one of the four domains under

[7] We thank Rod Adams, Cosmin-Adrian Bejan, Sasha Blair-Goldensohn, Cyril Cerovic, David Elson, David Evans, Ovidiu Fortu, Agustin Gravano, Lokesh Shresta, John Yundt-Pacheco and Kapil Thadani for the submitted questions.

	Jaccard metric		
Domain	$subj_1$ and $subj_2$ (and $subj_3$)	$subj_1$ and MUC	$subj_2$ and MUC
Airplane crash	0.54	-	-
Earthquake	0.68	-	-
Presidential Election	0.32	-	-
Terrorist Attack	0.50	0.63	0.59

Table 2: Creating gold standard. Jaccard metric values for interannotator agreement.

analysis. Thus, for the *earthquake* and *terrorist attack* domains we got two lists of questions; for the *airplane crash* and *presidential election* domains we got three lists of questions.

After the questions lists were created we studied the agreement among annotators on what information they consider is important for the domain and thus, should be included in the template. We matched the questions created by different annotators for the same domain. For some of the questions we had to make a judgement call on whether it is a match or not. For example, the following question created by one of the annotators for the *earthquake* domain was:

> *Did the earthquake occur in a well-known area for earthquakes (e.g. along the San Andreas fault), or in an unexpected location?*

We matched this question to the following three questions created by the other annotator:

> *What is the geological localization?*
> *Is it near a fault line?*
> *Is it near volcanoes?*

Usually, the degree of interannotator agreement is estimated using Kappa. For this task, though, Kappa statistics cannot be used as they require knowledge of the expected or chance agreement, which is not applicable to this task (Fleiss et al., 1981). To measure interannotator agreement we use the Jaccard metric, which does not require knowledge of the expected or chance agreement. Table 2 shows the values of Jaccard metric for interannotator agreement calculated for all four domains. Jaccard metric values are calculated as

$$Jaccard(domain^d) = \frac{|QS_i^d \cap QS_j^d|}{|QS_i^d \cup QS_j^d|} \quad (3)$$

where QS_i^d and QS_j^d are the sets of questions created by subjects i and j for domain d. For the *airplane crash* and *presidential election* domains we averaged the three pairwise Jaccard metric values.

The scores in Table 2 show that for some domains the agreement is quite high (e.g., *earthquake*), while for other domains (e.g., *presidential election*) it is twice as low. This difference in scores can be explained by the complexity of the domains and by the differences in understanding of these domains by different subjects. The scores for the *presidential election* domain are predictably low as in different countries the roles of presidents are very different: in some countries the president is the head of the government with a lot of power, while in other countries the president is merely a ceremonial figure. In some countries the presidents are elected by general voting while in other countries, the presidents are elected by parliaments. These variations in the domain cause the subjects to be interested in different issues of the domain. Another issue that might influence the interannotator agreement is the distribution of the presidential election process in time. For example, one of our subjects was clearly interested in the pre-voting situation, such as debates between the candidates, while another subject was interested only in the outcome of the presidential election.

For the *terrorist attack* domain we also compared the lists of questions we got from our subjects with the *terrorist attack* template created by experts for the MUC competition. In this template we treated every slot as a separate question, excluding the first two slots which captured information about the text from which the template fillers were extracted and not about the domain. The results for this comparison are included in Table 2.

Differences in domain complexity were studied by IE researchers. Bagga (1997) suggests a classification methodology to predict the syntactic complexity of the domain-related facts. Huttunen et al. (2002) analyze how component subevents of the domain are linked together and discuss the factors which contribute to the domain complexity.

6.2 Stage 2. Quality of the Automatically Created Templates

In section 6.1 we showed that not all the domains are equal. For some of the domains it is much easier to come to a consensus about what slots should be present in the domain template than for others. In this section we describe the evaluation of the four automatically created templates.

Automatically created templates consist of slot structures and are not easily readable by human annotators. Thus, instead of direct evaluation of the template quality, we evaluate the clauses extracted according to the created templates and

check whether these clauses contain the answers to the questions created by the subjects during the first stage of the evaluation. We extract the clauses corresponding to the test instances according to the following procedure:

1. Identify all the simple clauses in the documents corresponding to a particular test instance (respective TDT topic). For example, for the sentence

   Her husband, Robert, survived Thursday's explosion in a Yemeni harbor that killed at least six crew members and injured 35.

   only one part is output:

   that killed at least six crew members and injured 35
2. For every domain template slot check all the simple clauses in the instance (TDT topic) under analysis. Find the shortest clause (or sequence of clauses) which includes both the verb and other words extracted for this slot in their respective order. Add this clause to the list of extracted clauses unless this clause has been already added to this list.
3. Keep adding clauses to the list of extracted clauses till all the template slots are analyzed or the size of the list exceeds 20 clauses.

The key step in the above algorithm is Step 2. By choosing the shortest simple clause or sequence of simple clauses corresponding to a particular template slot, we reduce the possibility of adding more information to the output than is necessary to cover each particular slot.

In Step 3 we keep only the first twenty clauses so that the length of the output which potentially contains an answer to the question of interest is not larger than the number of questions provided by each subject. The templates are created from the slot structures extracted for the top 50 verbs. The higher the estimated score of the verb (Eq. 1) for the domain the closer to the top of the template the slot structure corresponding to this verb will be. We assume that the important information is more likely to be covered by the slot structures that are placed near the top of the template.

The evaluation results for the automatically created templates are presented in Figure 1. We calculate what average percentage of the questions is covered by the outputs created according to the domain templates. For every domain, we present the percentage of the covered questions separately for each annotator and for the intersection of questions (Section 6.1).

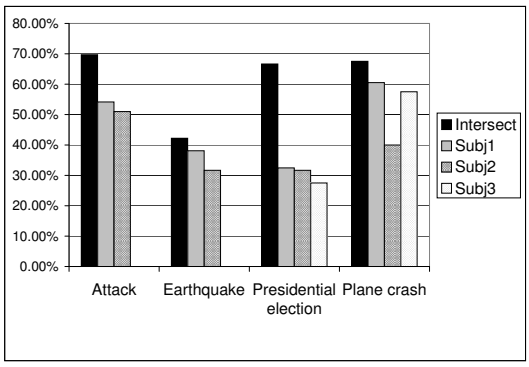


Figure 1: Evaluation results.

For the questions common for all the annotators we capture about 70% of the answers for three out of four domains. After studying the results we noticed that for the *earthquake* domain some questions did not result in a template slot and thus, could not be covered by the extracted clauses. Here are two of such questions:

Is it near a fault line?
Is it near volcanoes?

According to the template creation procedure, which is centered around verbs, the chances that extracted clauses would contain answers to these questions are low. Indeed, only one of the three sentence sets extracted for the three TDT earthquake topics contain an answer to one of these questions.

Poor results for the *presidential election* domain could be predicted from the Jaccard metric value for interannotator agreement (Table 2). There is considerable discrepancy in the questions created by human annotators which can be attributed to the great variation in the *presidential election* domain itself. It must be also noted that most of the questions created for the *presidential election* domain were clearly referring to the democratic election procedure, while some of the TDT topics categorized as *Elections* were about either election fraud or about opposition taking over power without the formal resignation of the previous president.

Overall, this evaluation shows that using automatically created domain templates we extract sentences which contain a substantial part of the important information expressed in questions for that domain. For those domains which have small diversity our coverage can be significantly higher.

7 Conclusions

In this paper, we presented a robust method for data-driven discovery of the important fact-types

for a given domain. In contrast to supervised methods, the fact-types are not pre-specified. The resulting slot structures can subsequently be used to guide the generation of responses to questions about new instances of the same domain. Our approach features the use of corpus statistics derived from both lexical and syntactic analysis across documents. A comparison of our system output for four domains of interest shows that our approach can reliably predict the majority of information that humans have indicated are of interest. Our method is flexible: analyzing document collections from different time periods or locations, we can learn domain descriptions that are tailored to those time periods and locations.

Acknowledgements. We would like to thank Rebecca Passonneau and Julia Hirschberg for the fruitful discussions at the early stages of this work; Vasilis Vassalos for his suggestions on the evaluation instructions; Michel Galley, Agustin Gravano, Panagiotis Ipeirotis and Kapil Thadani for their enormous help with evaluation.

This material is based upon work supported in part by the Advanced Research Development Agency (ARDA) under Contract No. NBCHC040040 and in part by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-06-C-0023. Any opinions, findings and conclusions expressed in this material are those of the authors and do not necessarily reflect the views of ARDA and DARPA.

References

Kenji Abe, Shinji Kawasoe, Tatsuya Asai, Hiroki Arimura, and Setsuo Arikawa. 2002. Optimized substructure discovery for semi-structured data. In *Proc. of PKDD*.

Amit Bagga. 1997. Analyzing the complexity of a domain with respect to an Information Extraction task. In *Proc. 7th MUC*.

Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *Proc. of HLT/NAACL*.

Daniel Bikel, Richard Schwartz, and Ralph Weischedel. 1999. An algorithm that learns what's in a name. *Machine Learning Journal Special Issue on Natural Language Learning*, 34:211–231.

Sasha Blair-Goldensohn, Kathleen McKeown, and Andrew Hazen Schlaikjer, 2004. *Answering Definitional Questions: A Hybrid Approach*. AAAI Press.

Robin Collier. 1998. *Automatic Template Creation for Information Extraction*. Ph.D. thesis, University of Sheffield.

Pablo Duboue and Kathleen McKeown. 2003. Statistical acquisition of content selection rules for natural language generation. In *Proc. of EMNLP*.

Elena Filatova and Vasileios Hatzivassiloglou. 2003. Domain-independent detection, extraction, and labeling of atomic events. In *Proc. of RANLP*.

Elena Filatova and John Prager. 2005. Tell me what you do and I'll tell you what you are: Learning occupation-related activities for biographies. In *Proc. of EMNLP/HLT*.

Charles Fillmore and Collin Baker. 2001. Frame semantics for text understanding. In *Proc. of WordNet and Other Lexical Resources Workshop, NAACL*.

Jon Fiscus, George Doddington, John Garofolo, and Alvin Martin. 1999. NIST's 1998 topic detection and tracking evaluation (TDT2). In *Proc. of the 1999 DARPA Broadcast News Workshop*, pages 19–24.

Joseph Fleiss, Bruce Levin, and Myunghee Cho Paik, 1981. *Statistical Methods for Rates and Proportions*. J. Wiley.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Sanda Harabagiu and Finley Lăcătuşu. 2005. Topic themes for multi-document summarization. In *Proc. of SIGIR*.

Sanda Harabagiu and Steven Maiorano. 2002. Multi-document summarization with GISTexter. In *Proc. of LREC*.

Jerry Hobbs and David Israel. 1994. Principles of template design. In *Proc. of the HLT Workshop*.

Silja Huttunen, Roman Yangarber, and Ralph Grishman. 2002. Complexity of event structure in IE scenarios. In *Proc. of COLING*.

Dan Klein and Christopher Manning. 2002. Fast exact inference with a factored model for natural language parsing. In *Proc. of NIPS*.

Hans Luhn. 1957. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of Research and Development*, 1:309–317.

Elaine Marsh and Dennis Perzanowski. 1997. MUC-7 evaluation of IE technology: Overview of results. In *Proc. of the 7th MUC*.

Boyan Onyshkevych. 1994. Issues and methodology for template design for information extraction system. In *Proc. of the HLT Workshop*.

Martha Palmer, Dan Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James Martin, and Daniel Jurafsky. 2004. Shallow semantic parsing using support vector machines. In *Proc. of HLT/NAACL*.

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

Ellen Riloff and Mark Schmelzenbach. 1998. An empirical approach to conceptual case frame acquisition. In *Proc. of the 6th Workshop on Very Large Corpora*.

Gerard Salton, 1971. *The SMART retrieval system*. Prentice-Hall, NJ.

Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. 2003. An improved extraction pattern representation model for automatic IE pattern acquisition. In *Proc. of ACL*.

Ralph Weischedel, Jinxi Xu, and Ana Licuanan, 2004. *Hybrid Approach to Answering Biographical Questions*. AAAI Press.

Michael White, Tanya Korelsky, Claire Cardie, Vincent Ng, David Pierce, and Kiri Wagstaff. 2001. Multi-document summarization via information extraction. In *Proc. of HLT*.

Mohammed Zaki. 2002. Efficiently mining frequent trees in a forest. In *Proc. of SIGKDD*.

Liang Zhou, Miruna Ticrea, and Eduard Hovy. 2004. Multi-document biography summarization. In *Proc. of EMNLP*.

Using Lexical Dependency and Ontological Knowledge to Improve a Detailed Syntactic and Semantic Tagger of English

Andrew Finch
NiCT*ATR†
Kyoto, Japan
andrew.finch@atr.jp

Ezra Black
Epimenides Corp.
New York, USA
ezra.black@epimenides.com

Young-Sook Hwang
ETRI
Seoul, Korea
yshwang7@etri.re.kr

Eiichiro Sumita
NiCT-ATR
Kyoto, Japan
eiichiro.sumita@atr.jp

Abstract

This paper presents a detailed study of the integration of knowledge from both dependency parses and hierarchical word ontologies into a maximum-entropy-based tagging model that simultaneously labels words with both syntax and semantics. Our findings show that information from both these sources can lead to strong improvements in overall system accuracy: dependency knowledge improved performance over all classes of word, and knowledge of the position of a word in an ontological hierarchy increased accuracy for words not seen in the training data. The resulting tagger offers the highest reported tagging accuracy on this tagset to date.

1 Introduction

Part-of-speech (POS) tagging has been one of the fundamental areas of research in natural language processing for many years. Most of the prior research has focussed on the task of labeling text with tags that reflect the words' syntactic role in the sentence. In parallel to this, the task of word sense disambiguation (WSD), the process of deciding in which semantic sense the word is being used, has been actively researched. This paper addresses a combination of these two fields, that is: labeling running words with tags that comprise, in addition to their syntactic function, a broad semantic class that signifies the semantics of the word in the context of the sentence, but does not necessarily provide information that is sufficiently fine-grained as to disambiguate its sense. This differs from what is commonly meant by WSD in that although each word may have many "senses" (by senses here, we mean the set of semantic labels the word may take), these senses are not specific to the word itself but are drawn from a vocabulary applicable to the subset of all types in the corpus that may have the same semantics.

In order to perform this task, we draw on research from several related fields, and exploit publicly available linguistic resources, namely the WordNet database (Fellbaum, 1998). Our aim is to simultaneously disambiguate the semantics of the words being tagged while tagging their POS syntax. We treat the task as fundamentally a POS tagging task, with a larger, more ambiguous tag set. However, as we will show later, the 'n-gram' feature set traditionally employed to perform POS tagging, while basically competent, is not up to this challenge, and needs to be augmented by features specifically targeted at semantic disambiguation.

2 Related Work

Our work is a synthesis of POS tagging and WSD, and as such, research from both these fields is directly relevant here.

The basic engine used to perform the tagging in these experiments is a direct descendent of the maximum entropy (ME) tagger of (Ratnaparkhi, 1996) which in turn is related to the taggers of (Kupiec, 1992) and (Merialdo, 1994). The ME approach is well-suited to this kind of labeling because it allows the use of a wide variety of features without the necessity to explicitly model the interactions between them.

The literature on WSD is extensive. For a good overview we direct the reader to (Nancy and Jean, 1998). Typically, the local context around the

*National Institute of Information and Communications Technology

†ATR Spoken Language Communication Research Labs

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 215–222,
Sydney, July 2006.

word to be sense-tagged is used to disambiguate the sense (Yarowsky, 1993), and it is common for linguistic resources such as WordNet (Li et al., 1995; Mihalcea and Moldovan, 1998; Ramakrishnan and Prithviraj, 2004), or bilingual data (Li and Li, 2002) to be employed as well as more long-range context. An ME-system for WSD that operates on similar principles to our system (Suarez, 2002) was based on an array of local features that included the words/POS tags/lemmas occurring in a window of +/-3 words of the word being disambiguated. (Lamjiri et al., 2004) also developed an ME-based system that used a very simple set of features: the article before; the POS before and after; the preposition before and after, and the syntactic category before and after the word being labeled. The features used in both of these approaches resemble those present in the feature set of a standard n-gram tagger, such as the one used as the baseline for the experiments in this paper. The semantic tags we use can be seen as a form of semantic categorization acting in a similar manner to the semantic class of a word in the system of (Lamjiri et al., 2004). The major difference is that with a left-to-right beam-search tagger, labeled context to the right of the word being labeled is not available for use in the feature set.

Although POS tag information has been utilized in WSD techniques (e.g. (Suarez, 2002)), there has been relatively little work addressing the problem of assigning a part-of-speech tag to a word together with its semantics, despite the fact that the tasks involve a similar process of label disambiguation for a word in running text.

3 Experimental Data

The primary corpus used for the experiments presented in this paper is the ATR General English Treebank. This consists of 518,080 words (approximately 20 words per sentence, on average) of text annotated with a detailed semantic and syntactic tagset.

To understand the nature of the task involved in the experiments presented in this paper, one needs some familiarity with the ATR General English Tagset. For detailed presentations, see (Black et al., 1996b; Black et al., 1996a; Black and Finch, 2001). An apercu can be gained, however, from Figure 1, which shows two sample sentences from the ATR Treebank (and originally from a Chinese take–out food flier), tagged with respect to the ATR General English Tagset. Each verb, noun, adjective and adverb in the ATR tagset includes a semantic label, chosen from 42 noun/adjective/adverb categories and 29 verb/verbal categories, some overlap existing between these category sets. Proper nouns, plus certain adjectives and certain numerical expressions, are further categorized via an additional 35 "proper–noun" categories. These semantic categories are intended for any "Standard–American–English" text, in any domain. Sample categories include: "physical.attribute" (nouns/adjectives/adverbs), "alter" (verbs/verbals), "interpersonal.act" (nouns/adjectives/adverbs/verbs/verbals), "orgname" (proper nouns), and "zipcode" (numericals). They were developed by the ATR grammarian and then proven and refined via day–in–day–out tagging for six months at ATR by two human "treebankers", then via four months of tagset–testing–only work at Lancaster University (UK) by five treebankers, with daily interactions among treebankers, and between the treebankers and the ATR grammarian. The semantic categorization is, of course, in addition to an extensive syntactic classification, involving some 165 basic syntactic tags.

The test corpus has been designed specifically to cope with the ambiguity of the tagset. It is possible to correctly assign any one of a number of 'allowable' tags to a word in context. For example, the tag of the word *battle* in the phrase "a legal battle" could be either NN1PROBLEM or NN1INTER-ACT, indicating that the semantics is either a problem, or an inter-personal action. The test corpus consists of 53,367 words sampled from the same domains as, and in approximately the same proportions as the training data, and labeled with a set of up to 6 allowable tags for each word. During testing, only if the predicted tag fails to match any of the allowed tags is it considered an error.

4 Tagging Model

4.1 ME Model

Our tagging framework is based on a maximum entropy model of the following form:

$$p(t,c) = \gamma \prod_{k=0}^{K} \alpha_k^{f_k(c,t)} p_0 \qquad (1)$$

where:

```
(_( Please_RRCONCESSIVE Mention_VVIVERBAL-ACT this_DD1 coupon_NN1DOCUMENT
when_CSWHEN ordering_VVGINTER-ACT

OR_CCOR ONE_MC1WORD FREE_JJMONEY FANTAIL_NN1ANIMAL SHRIMPS_NN1FOOD
```

Figure 1: Two ATR Treebank Sentences from a Take–Out Food Flier

- t is tag being predicted;
- c is the context of t;
- γ is a normalization coefficient that ensures: $\Sigma_{t=0}^{L}\gamma\prod_{k=0}^{K}\alpha_k^{f_k(c,t)}p_0 = 1$;
- K is the number of features in the model;
- L is the number of tags in our tag set;
- α_k is the weight of feature f_k;
- f_k are feature functions and $f_k \epsilon \{0,1\}$;
- p_0 is the default tagging model (in our case, the uniform distribution, since all of the information in the model is specified using ME constraints).

Our baseline model contains the following feature predecate set:

w_0	t_{-1}	pos_0	$pref_1(w_0)$
w_{-1}	t_{-2}	pos_{-1}	$pref_2(w_0)$
w_{-2}		pos_{-2}	$pref_3(w_0)$
w_{+1}		pos_{+1}	$suff_1(w_0)$
w_{+2}		pos_{+2}	$suff_2(w_0)$
			$suff_3(w_0)$

where:

- w_n is the word at offset n relative to the word whose tag is being predicted;
- t_n is the tag at offset n;
- pos_n is the syntax-only tag at offset n assigned by a syntax-only tagger;
- $pref_n(w_0)$ is the first n characters of w_0;
- $suff_n(w_0)$ is the last n characters of w_0;

This feature set contains a typical selection of n-gram and basic morphological features. When the tagger is trained in tested on the UPENN treebank (Marcus et al., 1994), its accuracy (excluding the pos_n features) is over 96%, close to the state of the art on this task. (Black et al., 1996b) adopted a two-stage approach to prediction, first predicting syntax, then semantics given the syntax, whereas in (Black et al., 1998) both syntax and semantics were predicted together in one step. In using syntactic tags as features, we take a softer approach to the two-stage process. The tagger has access to accurate syntactic information; however, it is not necessarily constrained to accept this choice of syntax. Rather, it is able to decide both syntax and semantics while taking semantic context into account. In order to find the most probable sequence of tags, we tag in a left-to-right manner using a beam-search algorithm.

4.2 Feature selection

For reasons of practicability, it is not always possible to use the full set of features in a model: often it is necessary to control the number of features to reduce resource requirements during training. We use mutual information (MI) to select the most useful feature predicates (for more details, see (Rosenfeld, 1996)). It can be viewed as a means of determining how much information a given predicate provides when used to predict an outcome.

That is, we use the following formula to gauge a feature's usefulness to the model:

$$I(f;T) = \sum_{f\in\{0,1\}}\sum_{t\in T} p(f,t)log\frac{p(f,t)}{p(f)p(t)} \quad (2)$$

where:

- $t \in T$ is a tag in the tagset;
- $f \in \{0,1\}$ is the value of any kind of predicate feature.

Using mutual information is not without its shortcomings. It does not take into account any of the interactions between features. It is possible for a feature to be pronounced useful by this procedure, whereas in fact it is merely giving the same information as another feature but in different form. Nonetheless this technique is invaluable in practice. It is possible to eliminate features

which provide little or no benefit to the model, thus speeding up the training. In some cases it even allows a model to be trained where it would not otherwise be possible to train one. For the purposes of our experiments, we use the top 50,000 predicates for each model to form the feature set.

5 External Knowledge Sources

5.1 Lexical Dependencies

Features derived from n-grams of words and tags in the immediate vicinity of the word being tagged have underpinned the world of POS tagging for many years (Kupiec, 1992; Merialdo, 1994; Ratnaparkhi, 1996), and have proven to be useful features in WSD (Yarowsky, 1993). Lower-order n-grams which are closer to word being tagged offer the greatest predictive power (Black et al., 1998). However, in the field of WSD, relational information extracted from grammatical analysis of the sentence has been employed to good effect, and in particular, subject-object relationships between verbs and nouns have been shown be effective in disambiguating semantics (Nancy and Jean, 1998). We take the broader view that dependency relationships in general between any classes of words may help, and use the ME training process to weed out the irrelevant relationships. The principle is exactly the same as when using a word in the local context as a feature, except that the word in this case has a grammatical relationship with the word being tagged, and can be outside the local neighborhood of the word being tagged. For both types of dependency, we encoded the model constraints $f_{stl}(d)$ as boolean functions of the form:

$$f_{stl}(d) = \begin{cases} 1 & \text{if } d.s = s \wedge d.t = t \wedge d.l = l \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where:

- d is a lexical dependency, consisting of a source word (the word being tagged) $d.s$, a target word $d.t$ and a label $d.l$
- s and t (words), and l (link label) are specific to the feature

We generated two distinct features for each dependency. The source and target were exchanged to create these features. This was to allow the models to capture the bidirectional nature of the dependencies. For example, when tagging a verb, the model should be aware of the dependent object, and conversely when tagging that object, the model should have a feature imposing a constraint arising from the identity of the dependent verb.

5.1.1 Dependencies from the CMU Link Grammar

We parsed our corpus using the parser detailed in (Grinberg et al., 1995). The dependencies output by this parser are labeled with the type of dependency (connector) involved. For example, subjects (connector type S) and direct objects of verbs (O) are explicitly marked by the process (a full list of connectors is provided in the paper). We used all of the dependencies output by the parser as features in the models.

5.1.2 Dependencies from Phrasal Structure

It is possible to extract lexical dependencies from a phrase-structure parse. The procedure is explained in detail in (Collins, 1996). In essence, each non-terminal node in the parse tree is assigned a head word, which is the head of one of its children denoted the 'head child'. Dependencies are established between this headword and the heads of each of the children (except for the head child). In these experiments we used the MXPOST tagger (Ratnaparkhi, 1996) combined with Collins' parser (Collins, 1996) to assign parse trees to the corpus. The parser had a 98.9% coverage of the sentences in our corpora. Again, all of the dependencies output by the parser were used as features in the models.

5.2 Hierarchical Word Ontologies

In this section we consider the effect of features derived from hierarchical sets of words. The primary advantage is that we are able to construct these hierarchies using knowledge from outside the training corpus of the tagger itself, and thereby glean knowledge about rare words. In these experiments we use the human annotated word taxonomy of hypernyms (IS-A relations) in the WordNet database, and an automatically acquired ontology made by clustering words in a large corpus of unannotated text.

We have chosen to use hierarchical schemes for both the automatic and manually acquired ontologies because this offers the opportunity to combat data-sparseness issues by allowing features derived from all levels of the hierarchy to be used. The process of training the model is able to de-

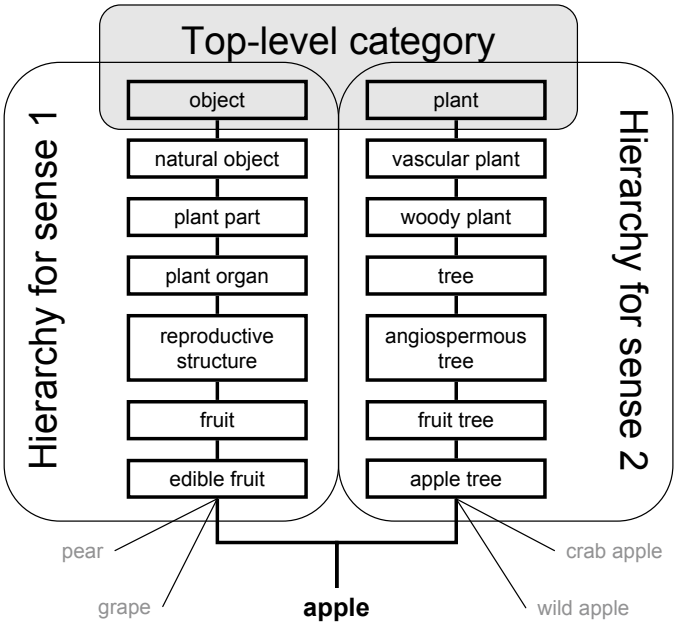


Figure 2: The WordNet taxonomy for both (WordNet) senses of the word *apple*

cide the levels of granularity that are most useful for disambiguation. For the purposes of generating features for the ME tagger we treat both types of hierarchy in the same fashion. One of these features is illustrated in Figure 5.3. Each predicate is effectively a question which asks whether the word (or word being used in a particular sense in the case of the WordNet hierarchy) is a descendent of the node to which the predicate applies. These predicates become more and more general as one moves up the hierarchy. For example in the hierarchy shown in Figure 5.2, looking at the nodes on the right hand branch, the lowest node represents the class of **apple trees** whereas the top node represents the class of all **plants**.

We expect these hierarchies to be particularly useful when tagging out of vocabulary words (OOV's). The identity of the word being tagged is by far the most important feature in our baseline model. When tagging an OOV this information is not available to the tagger. The automatic clustering has been trained on 100 times as much data as our tagger, and therefore will have information about words that tagger has not seen during training. To illustrate this point, suppose that we are tagging the OOV *pomegranate*. This word is in the WordNet database, and is in the same synset as the 'fruit' sense of the word *apple*. It is reasonable to assume that the model will have learned (from the many examples of all fruit words) that the predicate representing membership of this **fruit** synset should, if true, favor the selection of the correct tag for fruit words: NN1FOOD. The predicate will be true for the word *pomegranate* which will thereby benefit from the model's knowledge of how to tag the other words in its class. Even if this is not so at this level in the hierarchy, it is likely to be so at some level of granularity. Precisely which levels of detail are useful will be learned by the model during training.

5.2.1 Automatic Clustering of Text

We used the automatic agglomerative mutual-information-based clustering method of (Ushioda, 1996) to form hierarchical clusters from approximately 50 million words of tokenized, unannotated text drawn from similar domains as the treebank used to train the tagger. Figure 5.2 shows the position of the word *apple* within the hierarchy of clusters. This example highlights both the strengths and weaknesses of this approach. One strength is that the process of clustering proceeds in a purely objective fashion and associations between words that may not have been considered by a human annotator are present. Moreover, the clustering process considers all types that actually occur in the corpus, and not just those words that might appear in a dictionary (we will return to this later). A major problem with this approach is that

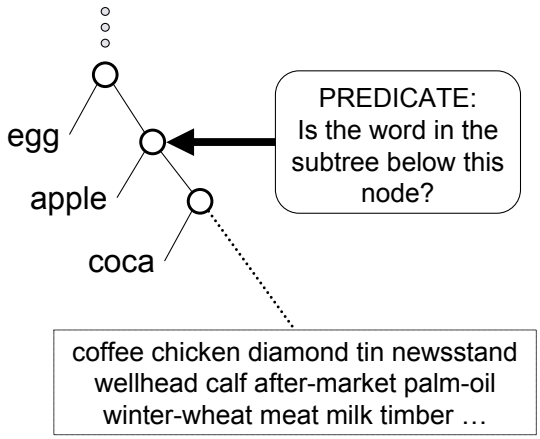


Figure 3: The dendrogram for the automatically acquired ontology, showing the word *apple*

the clusters tend to contain a lot of noise. Rare words can easily find themselves members of clusters to which they do not seem to belong, by virtue of the fact that there are too few examples of the word to allow the clustering to work well for these words. This problem can be mitigated somewhat by simply increasing the size of the text that is clustered. However the clustering process is computationally expensive. Another problem is that a word may only be a member of a single cluster; thus typically the cluster set assigned to a word will only be appropriate for that word when used in its most common sense.

Approximately 93% of running words in the test corpus, and 95% in the training corpus were covered by the words in the clusters (when restricted to verbs, nouns, adjectives and adverbs, these figures were 94.5% and 95.2% respectively). Approximately 81% of the words in the vocabulary from the test corpus were covered, and 71% of the training corpus vocabulary was covered.

5.2.2 WordNet Taxonomy

For this class of features, we used the hypernym taxonomy of WordNet (Fellbaum, 1998). Figure 5.2 shows the WordNet hypernym taxonomy for the two senses of the word *apple* that are in the database. The set of predicates query membership of all levels of the taxonomy for all WordNet senses of the word being tagged. An example of one such predicate is shown in the figure.

Only 63% of running words in both the training and the test corpus were covered by the words in the clusters. Although this figure appears low, it can be explained by the fact that WordNet only contains entries for words that have senses in certain parts of speech. Some very frequent classes of words, for example determiners, are not in WordNet. The coverage of only nouns, verbs, adjectives and adverbs in running text is 94.5% for both training and test sets. Moreover, approximately 84% of the words in the vocabulary from the test corpus were covered, and 79% on the training corpus. Thus, the effective coverage of WordNet on the important classes of words is similar to that of the automatic clustering method.

6 Experimental Results

The results of our experiments are shown in Table 1. The task of assigning semantic and syntactic tags is considerably more difficult than simply assigning syntactic tags due to the inherent ambiguity of the tagset. To gauge the level of human performance on this task, experiments were conducted to determine inter-annotator consistency; in addition, annotator accuracy was measured on 5,000 words of data. Both the agreement and accuracy were found to be approximately 97%, with all of the inconsistencies and tagging errors arising from the semantic component of the tags. 97% accuracy is therefore an approximate upper bound for the performance one would expect from an automatic tagger. As a point of reference for a lower bound, the overall accuracy of a tagger which uses only a single feature representing the identity of the word being tagged is approximately 73%.

The overall baseline accuracy was 82.58% with only 30.58% of OOV's being tagged correctly. Of the two lexical dependency-based approaches,

the features derived from Collins' parser were the most effective, improving accuracy by 0.8% overall. To put the magnitude of this gain into perspective, dropping the features for the identity of the previous word from the baseline model, only degraded performance by 0.2%. The features from the link grammar parser were handicapped due to the fact that only 31% of the sentences were able to be parsed. When the model (Model 3 in Table 1) was evaluated on only the parsable portion on the test set, the accuracy obtained was roughly comparable to that using the dependencies from Collins' parses. To control for the differences between these parseable sentences and the full test set, Model 4 was tested on the same 31% of sentence that parsed. Its accuracy was within 0.2% of the accuracy on the whole test set in all cases. Neither of the lexical dependency-based approaches had a particularly strong effect on the performance on OOV's. This is in line with our intuition, since these features rely on the identity of the word being tagged, and the performance gain we see is due to the improvement in labeling accuracy of the context around the OOV.

In contrast to this, for the word-ontology-based feature sets, one would hope to see a marked improvement on OOV's, since these features were designed specifically to address this issue. We do see a strong response to these features in the accuracy of the models. The overall accuracy when using the automatically acquired ontology is only 0.1% higher than the accuracy using dependencies from Collins' parser. However the accuracy on OOV's jumps 3.5% to 35.08% compared to just 0.7% for Model 4. Performance for both clustering techniques was quite similar, with the WordNet taxonomical features being slightly more useful, especially for OOV's. One possible explanation for this is that overall, the coverage of both techniques is similar, but for rarer words, the MI clustering can be inconsistent due to lack of data (for an example, see Figure 5.2: the word *newsstand* is a member of a cluster of words that appear to be commodities), whereas the WordNet clustering remains consistent even for rare words. It seems reasonable to expect, however, that the automatic method would do better if trained on more data. Furthermore, all uses of words can be covered by automatic clustering, whereas for example, the common use of the word *apple* as a company name is beyond the scope of WordNet.

In Model 7 we combined the best lexical dependency feature set (Model 4) with the best clustering feature set (Model 6) to investigate the amount of information overlap existing between the feature sets. Models 4 and 6 improved the baseline performance by 0.8% and 1.3% respectively. In combination, accuracy was increased by 2.3%, 0.2% more than the sum of the component models' gains. This is very encouraging and indicates that these models provide independent information, with virtually all of the benefit from both models manifesting itself in the combined model.

7 Conclusion

We have described a method for simultaneously labeling the syntax and semantics of words in running text. We develop this method starting from a state-of-the-art maximum entropy POS tagger which itself outperforms previous attempts to tag this data (Black et al., 1996b). We augment this tagging model with two distinct types of knowledge: the identity of dependent words in the sentence, and word class membership information of the word being tagged. We define the features in such a manner that the useful lexical dependencies are selected by the model, as is the granularity of the word classes used. Our experimental results show that large gains in performance are obtained using each of the techniques. The dependent words boosted overall performance, especially when tagging verbs. The hierarchical ontology-based approaches also increased overall performance, but with particular emphasis on OOV's, the intended target for this feature set. Moreover, when features from both knowledge sources were applied in combination, the gains were cumulative, indicating little overlap.

Visual inspection the output of the tagger on held-out data suggests there are many remaining errors arising from special cases that might be better handled by models separate from the main tagging model. In particular, numerical expressions and named entities cause OOV errors that the techniques presented in this paper are unable to handle. In future work we would like to address these issues, and also evaluate our system when used as a component of a WSD system, and when integrated within a machine translation system.

#	Model	Accuracy (± c.i.)	OOV's	Nouns	Verbs	Adj/Adv
1	Baseline	82.58± 0.32	30.58	68.47	74.32	70.99
2	+ Dependencies (link grammar)	82.74± 0.32	30.92	68.18	74.96	73.02
3	As above (only parsed sentences)	83.59± 0.53	30.92	69.16	77.21	73.52
4	+ Dependencies (Collins' parser)	83.37± 0.31	31.24	69.36	75.78	72.62
5	+ Automatically acquired ontology	83.71± 0.31	35.08	71.89	75.83	75.34
6	+ WordNet ontology	83.90± 0.31	36.18	72.28	76.29	74.47
7	+ Model 4 + Model 6	84.90± 0.31	37.02	72.80	78.36	76.16

Table 1: Tagging accuracy (%), '+' being shorthand for "Baseline +", 'c.i.' denotes the confidence interval of the mean at a 95% significance level, calculated using bootstrap resampling.

References

E. Black and A. Finch. 2001. Developing and proving effective broad-coverage semantic-and-syntactic tagsets for natural language: The atr approach. In *Proceedings of ICCPOL-2001*.

E. Black, S. Eubank, H. Kashioka, R. Garside, G. Leech, and D. Magerman. 1996a. Beyond skeleton parsing: producing a comprehensive large–scale general–english treebank with full grammatical analysis. In *Proceedings of the 16th Annual Conference on Computational Linguistics*, pages 107–112, Copenhagen.

E. Black, S. Eubank, H. Kashioka, and J. Saia. 1996b. Reinventing part-of-speech tagging. *Journal of Natural Language Processing (Japan)*, 5:1.

Ezra Black, Andrew Finch, and Hideki Kashioka. 1998. Trigger-pair predictors in parsing and tagging. In *Proceedings, 36th Annual Meeting of the Association for Computational Linguistics, 17th Annual Conference on Computational Linguistics*, Montreal, Canada.

Michael John Collins. 1996. A new statistical parser based on bigram lexical dependencies. In Arivind Joshi and Martha Palmer, editors, *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, pages 184–191, San Francisco. Morgan Kaufmann Publishers.

C. Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Dennis Grinberg, John Lafferty, and Daniel Sleator. 1995. A robust parsing algorithm for LINK grammars. Technical Report CMU-CS-TR-95-125, CMU, Pittsburgh, PA.

J. Kupiec. 1992. Robust part-of-speech tagging using a hidden markov model. *Computer Speech and Language*, 6:225–242.

A. K. Lamjiri, O. El Demerdash, and L.Kosseim. 2004. Simple features for statistical word sense disambiguation. In *Proc. ACL 2004 – Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text (Senseval-3)*, Barcelona, Spain, July. ACL-2004.

C. Li and H. Li. 2002. Word translation disambiguation using bilingual bootstrapping.

Xiaobin Li, Stan Szpakowicz, and Stan Matwin. 1995. A wordnet-based algorithm for word sense disambiguation. In *IJCAI*, pages 1368–1374.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1994. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330.

B. Merialdo. 1994. Tagging english text with a probabilistic model. *Computational Linguistics*, 20(2):155–172.

Rada Mihalcea and Dan I. Moldovan. 1998. Word sense disambiguation based on semantic density. In Sanda Harabagiu, editor, *Use of WordNet in Natural Language Processing Systems: Proceedings of the Conference*, pages 16–22. Association for Computational Linguistics, Somerset, New Jersey.

I. Nancy and V. Jean. 1998. Word sense disambiguation: The state of the art. *Computational Linguistics*, 24:1:1–40.

G. Ramakrishnan and B. Prithviraj. 2004. Soft word sense disambiguation. In *International Conference on Global Wordnet (GWC 04)*, Brno, Czeck Republic.

A. Ratnaparkhi. 1996. A maximum entropy part-of-speech tagger. In *Proceedings of the Empirical Methods in Natural Language Processing Conference*.

R. Rosenfeld. 1996. A maximum entropy approach to adaptive statistical language modelling. *Computer Speech and Language*, 10:187–228.

A. Suarez. 2002. A maximum entropy-based word sense disambiguation system. In *Proc. International Conference on Computational Linguistics*.

A. Ushioda. 1996. Hierarchical clustering of words. In *In Proceedings of COLING 96*, pages 1159–1162.

D. Yarowsky. 1993. One sense per collocation. In *In the Proceedings of ARPA Human Language Technology Workshop*.

The Benefit of Stochastic PP Attachment to a Rule-Based Parser

Kilian A. Foth and Wolfgang Menzel
Department of Informatics
Hamburg University
D-22527 Hamburg
Germany
foth|menzel@nats.informatik.uni-hamburg.de

Abstract

To study PP attachment disambiguation as a benchmark for empirical methods in natural language processing it has often been reduced to a binary decision problem (between verb or noun attachment) in a particular syntactic configuration. A parser, however, must solve the more general task of deciding between more than two alternatives in many different contexts. We combine the attachment predictions made by a simple model of lexical attraction with a full-fledged parser of German to determine the actual benefit of the subtask to parsing. We show that the combination of data-driven and rule-based components can reduce the number of all parsing errors by 14% and raise the attachment accuracy for dependency parsing of German to an unprecedented 92%.

1 Introduction

Most NLP applications are either data-driven (classification tasks are solved by comparing possible solutions to previous problems and their solutions) or rule-based (general rules are formulated which must be applicable to all cases that might be encountered). Both methods face obvious problems: The data-driven approach is at the mercy of its training set and cannot easily avoid mistakes that result from biased or scarce data. On the other hand, the rule-based approach depends entirely on the ability of a computational linguist to anticipate every construction that might ever occur. These handicaps are part of the reason why, despite great advances, many tasks in computational linguistics still cannot be performed nearly as well by computers as by human informants.

Applied to the subtask of syntax analysis, the dichotomy manifests itself in the existence of *learnt* and *handwritten* grammars of natural languages. A great many formalisms have been advanced that fall into either of the two variants, but even the best of them cannot be said to interpret arbitrary input consistently in the same way that a human reader would. Because the handicaps of different methods are to some degree complementary, it seems likely that a combination of approaches could yield better results than either alone. We therefore integrate a data-driven classifier for the special task of PP attachment into an existing rule-based parser and measure the effect that the additional information has on the overall accuracy.

2 Motivation

PP attachment disambiguation has often been studied as a benchmark test for empirical methods in natural language processing. Prepositions allow subordination to many different attachment sites, and the choice between them is influenced by factors from many different linguistic levels, which are generally subject to preferential rather than rigorous regularities. For this reason, PP attachment is a comparatively difficult subtask for rule-based syntax analysis and has often been attacked by statistical methods.

Because probabilistic approaches solve PP attachment as a natural subtask of parsing anyhow, the obvious application of a PP attacher is to integrate it into a rule-based system. Perhaps surprisingly, so far this has rarely been done. One reason for this is that many rule-driven syntax analyzers provide no obvious way to integrate uncertain, statistical information into their decisions. Another is the traditional emphasis on PP attachment as a binary classification task; since (Hindle and Rooth, 1991), research has concentrated on resolving the ambiguity in the category pattern 'V+N+P+N', i.e. predicting the PP attachment to either the verb or the first noun. It is often assumed that the correct attachment is always among these

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 223–230,
Sydney, July 2006.

two options, so that all problem instances can be solved correctly despite the simplification. This task is sufficient to measure the relative quality of different probability models, but it is quite different from what a parser must actually do: It is easier because the set of possible answers is pre-filtered so that only a binary decision remains, and the baseline performance for pure guessing is already 50%. But it is harder because it does not provide the predictor with all the information needed to solve many doubtful cases; (Hindle and Rooth, 1991) found that human arbiters consistently reach a higher agreement when they are given the entire sentence rather than just the four words concerned.

Instead of the accuracy of PP attachers in the *isolated* decision between two words, we investigate the problem of *situated* PP attachment. In this task, all nouns and verbs in a sentence are potential attachment points for a preposition; the computer must find suitable attachments for one or more prepositions in parallel, while building a globally coherent syntax structure at the same time.

3 Methods

Statistical PP attachment is based on the observation that the identities of content words can be used to predict which prepositional phrases modify which words, and achieve better-than-chance accuracy. This is apparently because, as heads of their respective phrases, they are representative enough that they can serve as a crude approximation of the semantic structure that could be derived from the phrases. Consider the following example (the last sentence in our test set):

Die Firmen müssen noch die Bedenken der EU-Kommission gegen die Fusion ausräumen. (The companies have yet to address the Commission's concerns about the merger.)

In this sentence, the preferred analysis will pair the preposition 'gegen' *(against, about, versus)* with the noun 'Bedenken' *(concerns)*, since the proposition is clearly that the concerns pertain to the merger. A syntax tree of this interpretation is shown in Figure 1. Note that there are at least three different syntactically plausible attachment sites for the preposition. In fact, there are even more, since a parser can make no initial assumptions about the global structure of the syntax tree that it will construct; for instance, the possibility that 'gegen' attaches to the noun 'Firmen' *(companies)* cannot be ruled out when beginning to parse.

3.1 WCDG

For the following experiments, we used the dependency parser of German described in (Foth et al., 2005). This system is especially suited to our goals for several reasons. Firstly, the parser achieves the highest published dependency-based accuracy on unrestricted written German input, but still has a comparatively high error rate for prepositions. In particular, it mis-attaches the preposition 'gegen' in the example sentence. Second, although rule-based in nature, it uses numerical penalties to arbitrate between different disambiguation rules. It is therefore easy to add another rule of varying strength, which depends on the output of an external statistical predictor, to guide the parser when it has no other means of making an attachment decision. Finally, the parser and grammar are freely available for use and modification (`http://nats-www.informatik.uni-hamburg.de/download`).

Weighted Constraint Dependency Grammar (Schröder, 2002) models syntax structure as labelled dependency trees as shown in the example. A grammar in this formalism is written as a set of *constraints* that license well-formed partial syntax structures. For instance, general projectivity rules ensure that the dependency tree corresponds to a properly nested syntax structure without crossing brackets[1]. Other constraints require an auxiliary verb to be modified by a full verb, or prescribe morphosyntactical agreement between a determiner and its regent (the word modified by the determiner). Although the *Constraint Satisfaction Problem* that this formalism defines is, in theory, infeasibly hard, it can nevertheless be solved approximatively with heuristic solution methods, and achieve competitive parsing accuracy.

To allow the resolution of true ambiguity (the existence of different structures neither of which is strictly ungrammatical), *weighted* constraints can be written that the solution *should* satisfy, if this is possible. The goal is then to build the structure that violates as few constraints as possible, and preferentially violates weak rather than strong constraints. This allows preferences to be expressed rather than hard rules. For instance, agreement constraints could actually be declared as violable, since typing errors, reformulations, etc. can

[1] Some constructions of German actually violate this property; exceptions in the projectivity constraints deal with these cases.

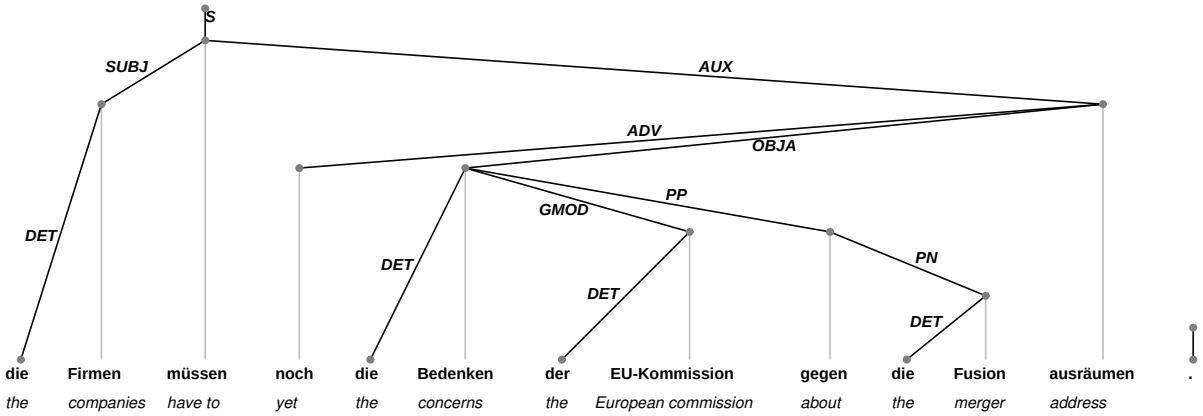


Figure 1: Correct syntax analysis of the example sentence.

and do actually lead to mis-inflected phrases. In this way robustness against many types of error can be achieved while still preferring the correct variant. For more about the WCDG parser, see (Schröder, 2002; Foth and Menzel, 2006) .

The grammar of German available for this parser relies heavily on weighted constraints both to cope with many kinds of imperfect input and to resolve true ambiguities. For the example sentence, it retrieves the desired dependencies except for constructing the implausible dependency 'ausräumen'+'gegen' *(address against)*. Let us briefly review the relevant constraints that cause this error:

- General structural, valence and agreement constraints determine the macro structure of the sentence in the desired way. For instance, the finite and the full verb must combine to form an auxiliary phrase, because this is the only way of accounting for all words while satisfying valence and category constraints. For the same reasons both determiners must be paired with their respective nouns. Also, the prepositional phrase itself is correctly predicted.

- General category constraints ensure that the preposition can attach to nouns and verbs, but not, say, to a determiner or to punctuation.

- A weak constraint on adjuncts says that adjuncts are usually close to their regent. The penalty of this constraint varies according to the length of the dependency that it is applied to, so that shorter dependencies are generally preferred.

- A slightly stronger constraint prefers attachment of the preposition to the verb, since overall verb attachment is more common than noun attachment in German. Therefore, the verb attachment leads to the globally best solution for this sentence.

There are no lexicalized rules that capture the particular plausibility of the phrase 'Bedenken gegen' *(concerns about)*. A constraint that describes this individual word pair would be trivial to write, but it is not feasible to model the general phenomenon in this way; thousands of constraints would be needed just to reflect the more important collocations in a language, and the exact set of collocating words is impossible to predict accurately. Data-driven information would be much more suitable for curing this lexical blind spot.

3.2 The Collocation Measure

The usual way to retrieve the lexical preference of a word such as 'Bedenken' for 'gegen' is to obtain a large corpus and assume that it is representative of the entire language; in particular, that collocations in this corpus are representative of collocations that will be encountered in future input. The assumption is of course not entirely true, but it can nevertheless be preferable to rely on such uncertain knowledge rather than remain undecided, on the reasonable assumption that it will lead to more correct than wrong decisions. Note that the same reasoning applies to many of the violable constraints in a WCDG: although they do not hold on *all* possible structures, they hold more often than they fail, and therefore can be useful for analysing unknown input.

Different measures have been used to gauge the strength of a lexical preference, but in general the efficacy of the statistical approach depends more on the suitability of the training corpus than on details of the collocation measure. Since our focus

is not on finding the best extraction method, but on judging the benefit of statistical components to parsing, we employ a collocation measure related to the idea of *mutual information*: a collocation between a word w and a preposition p is judged more likely the more often it appears, and the less often its component words appear. By normalizing against the total number t of utterances we derive a measure of Lexical Attraction for each possible collocation:

$$LA(w,p) := \frac{f_{w+p}}{t} \Big/ \left(\frac{f_w}{t} \cdot \frac{f_p}{t} \right)$$

For instance, if we assume that the word 'Bedenken' occurs in one out of 2,000 sentences of German and the word 'gegen' occurs in one sentence out of 31 (these figures were taken from the unsupervised experiment described later), then pure chance would make the two words co-occur in one sentence out of 62,000. If the LA score is higher than 1, i. e. we observe a much higher frequency of co-occurrences in a large corpus, we can assume that the two events are not statistically independent — in other words, that there is a positive correlation between the two words. Conversely, we would expect a much lower score for the implausible collocation 'Bedenken'+'für', indicating a dispreference for this attachment.

4 Experiments

4.1 Sources

To obtain the counts to base our estimates of attraction on, we first turned to the dependency treebank that accompanies the WCDG parsing suite. This corpus contains some 59,000 sentences with 1,000,000 words with complete syntactic annotations, 61% of which are drawn from online technical newscasts, 33% from literature and 6% from law texts. We used the entire corpus except for the test set as a source for counting PP attachments directly. All verbs, nouns and prepositions were first reduced to their base forms in order to reduce the parameter space. Compound nouns were reduced to their base nouns, so that 'EU-Kommission' is treated the same as 'Kommission', on the assumption that the compound exerts similar attractions as the base noun. In contrast, German verbs with prefixes usually differ markedly in their preferences from the base verb. Since forms of verbs such as 'ausräumen' *(address)* can be split into two parts ('NP räumte NP aus'), such separated verbs were reassembled before stemming.

(w,p)	f_{w+p}	f_w	LA
'Firma'+'gegen'	72	76492	0.03
'Bedenken'+'gegen'	1529	9618	4.96
'Kommission'+'gegen'	223	52415	0.13
'ausräumen'+'gegen'	130	2342	1.73

(where $f_p = 566068$, $t = 17657329$)

Table 1: Example calculation of lexical attraction.

Although the information retrieved from complete syntax trees is valuable, it is clearly insufficient for estimating many valid collocations. In particular, even for a comparatively strong collocation such as 'Bedenken'+'gegen' we can expect only very few instances. (There are, in fact, 4 such instances, well above chance level but still a very small number.) Therefore we used the archived text from 18 volumes of the newspaper *tageszeitung* as a second source. This corpus contains about 295,000,000 words and should allow us to detect many more collocations. In fact, we do find 2338 instances of 'Bedenken'+'gegen' in the same sentence.

Of course, since we have no syntactic annotations for this corpus (and it would be infeasible to create them even by fully automatic parsing), not all of these instances may indicate a syntactic dependency. (Ratnaparkhi, 1998) solved this problem by regarding only prepositions in syntactically unambiguous configurations. Unfortunately, his patterns cannot directly be applied to German sentences because of their freer word order. As an approximation it would be possible to count only pairs of adjacent content words and prepositions. However, this would introduce systematic biases into the counts, because nouns do in fact very often occur adjacently to prepositions that modify them, but many verbs do not. For instance, the phrase 'jmd. anklagen wegen etw.' *(to sue s.o. for s.th.)* gives rise to a strong collocation between the verb 'anklagen' and the preposition 'wegen'; however, in the predominant sentence types of German, the two words are virtually never adjacent, because either the preposition kernel or the direct object must intervene. Therefore, we relax the adjacency condition for verb attachment and also count prepositions that occur within a fixed distance of their suspected regent.

Table 1 shows the detailed values when judging the example sentence according to the unparsed corpus. The strong collocation that we would expect for 'Bedenken'+'gegen' is indeed

Value of i	Recall for V	for N	overall
1	96.2%	39.8%	65.2%
2	96.2%	52.0%	71.9%
5	88.8%	66.3%	76.4%
8	80.0%	79.6%	79.8%
10	67.5%	82.7%	75.8%

Table 2: Influence of noun factor on solving isolated attachment decisions.

observed, with a value of 4.96. However, the verb attachment also has a score above 1, indicating that 'gegen'+'ausräumen' *(to address about)* are also positively correlated. This is almost certainly a misleading figure, since those two words do not form a plausible verb phrase; it is much more probable that the very strong, in fact idiomatic, correlation 'Bedenken ausräumen' *(to address concerns)* causes many co-occurrences of all *three* words. Therefore our figures falsely suggest that 'gegen' would often attach to 'ausräumen', when it is in fact the direct object of that verb that it is attracted to.

(Volk, 2002) already suggested that this counting method introduced a general bias toward verb attachment, and when comparing the results for very frequent words (for which more reliable evidence is available from the treebank) we find that verb attachments are in fact systematically overestimated. We therefore adopted his approach and artificially inflated all noun+preposition counts by a constant factor i. To estimate an appropriate value for this factor, we extracted 178 instances of the standard verb+noun+preposition configuration from our corpus, of which 80 were verb attachments (V) and 98 were noun attachments (N).

Table 2 shows the performance of the predictor for this binary decision task. Taken as it is, it retrieves most verb attachments, but less than half of the noun attachments, while higher values of i can improve the recall both for noun attachments and overall. The performance achieved falls somewhat short of the highest figures reported previously for PP attachment for German (Volk, 2002); this is at least in part due to our simple model that ignores the kernel noun of the PP. However, it could well be good enough to be integrated into a full parser and provide a benefit to it. Also, the syntactical configuration in this standard benchmark is not the predominant one in complete German sentences; in fact fewer than 10% of all prepositions occur in this context. The best performance on the triple task is therefore not guaranteed to be the best choice for full parsing. In our experiments, we used a value of $i = 8$, which seems to be suited best to our grammar.

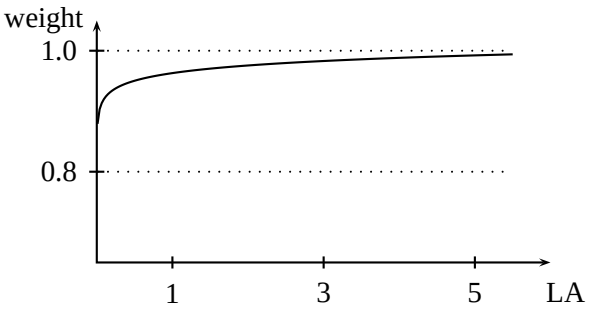


Figure 2: Mapping lexical attraction values to penalties

4.2 Integration Method

To add our simple collocation model to the parser, it is sufficient to write a single variable-strength constraint that judges each PP dependency by how strong the lexical attraction between the regent and the dependent is. The only question is how to map our lexical attraction values to penalties for this constraint. Their predicted relative order of plausibility should of course be reflected, so that dependencies with a high lexical attraction are preferred over those with lower lexical attraction. At the same time, the information should not be given too much weight compared to the existing grammar rules, since it is heuristic in nature and should certainly not override important principles such as valence or agreement. The penalties of WCDG constraints range from 0.0 (hard constraint) through 1.0 (a constraint with this penalty has no effect whatsoever and is only useful for debugging).

We chose an inverse mapping based on the logarithm of lexical attraction (cf. Figure 2):

$$p(w,p) = \frac{\max(1,\min(0.8,1-(2-\log_3(LA(w,p)))/50))}{\mu}$$

where μ is a normalization constant that scales the highest occurring value of LA to 1. For instance, this mapping will interpret a strong lexical attraction of 5 as the penalty 0.989 (almost perfect) and a lexical attraction of only 0.5 as the penalty 0.95 (somewhat dispreferred). The overall range of PP attachment penalties is limited to the interval $[0.8 - 1.0]$, which ensures that the judgement of the statistical module will usually come into play only when no other evidence is available; preliminary experiments showed that a stronger integration of the component yields no additional advantage. In any case, the exact figure depends closely on the valuation of the existing constraints of the grammar and is of little importance as such.

Label	occurred	retrieved	errors	accuracy
PP	1892	1285	607	67.9%
ADV	1137	951	186	83.6%
OBJA	775	675	100	87.1%
APP	659	567	92	86.0%
SUBJ	1338	1251	87	93.5%
S	1098	1022	76	93.1%
KON	481	406	75	84.4%
REL	167	107	60	64.1%
overall	17719	16073	1646	90.7

Table 3: Performance of the original parser on the test set.

Besides adding the new constraint 'PP attachment' to the grammar, we also disabled several of the existing constraints that apply to prepositions, since we assume that our lexicalized model is superior to the unlexicalized assumptions that the grammar writers had made so far. For instance, the constraint mentioned in Section 3 that globally prefers verb attachment to noun attachment is essentially a crude approximation of lexical attraction, whose task is now taken over entirely by the statistical predictor. We also assume that lexical preference exerts a stronger influence on attachment than mere linear distance; therefore we changed the distance constraint so that it exempts prepositions from the normal distance penalties imposed on adjuncts.

4.3 Corpus

For our parsing experiments, we used the first 1,000 sentences of technical newscasts from the dependency treebank mentioned above. This test set has an average sentence length of 17.7 words, and from previous experiments we estimate that it is comparable in difficulty to the NEGRA corpus to within 1% of accuracy. Although online articles and newspaper copy follow some different conventions, we assume the two text types are similar enough that collocations extracted from one can be used to predict attachments in the other.

For parsing we used the heuristic transformation-based search described in (Foth et al., 2000). Table 3 illustrates the structural accuracy[2] of the unmodified system for various subordination types. For instance, of the 1892 dependency edges with the label 'PP' in the gold standard, 1285 are attached correctly by the parser, while 607 receive an incorrect regent. We see that PP attachment decisions are particularly prone to errors both in absolute and in relative terms.

[2]Note that the WCDG parser always succeeds in assigning exactly one regent to each word, so that there is no difference between precision and recall. We refer to structural accuracy as the ratio of words which have been attached correctly to all words.

Method	PP accuracy	overall accuracy
baseline	67.9%	90.7%
supervised	79.4%	91.9%
unsupervised	78.3%	91.9%
backed-off	78.9%	92.2%

Table 4: Structural accuracy of PP edges and all edges.

4.4 Results

We trained the PP attachment predictor both with the counts acquired from the dependency treebank (supervised) and those from the newspaper corpus (unsupervised). We also tested a mode of operation that uses the more reliable data from the treebank, but backs off to unsupervised counts if the hypothetical regent was seen fewer than 1,000 times in training.

Table 4 shows the results when parsing with the augmented grammar. Both the overall structural accuracy and the accuracy of PP edges are given; note that these figures result from the general subordination task, therefore they correspond to Table 3 and not to Table 2. As expected, lexicalized preference information for prepositions yields a large benefit to full parsing: the attachment error rate is decreased by 34% for prepositions, and by 14% overall. In this experiment, where much more unsupervised training data was available, supervised and unsupervised training achieved almost the same level of performance (although many individual sentences were parsed differently).

A particular concern with corpus-based decision methods is their applicability beyond the training corpus. In our case, the majority of the material for supervised training was taken from the same newscast collection as the test set. However, comparable results are also achieved when applying the parser to the standard test set from the NEGRA corpus of German, as used by (Schiehlen, 2004; Foth et al., 2005): adding the PP predictor trained on our dependency treebank raises the overall attachment accuracy from 89.3% to 90.6%. This successful reuse indicates that lexical preference between prepositions and function words is largely independent of text type.

5 Related Work

(Hindle and Rooth, 1991) first proposed solving the prepositional attachment task with the help of statistical information, and also defined the prevalent formulation as a binary decision problem with three words involved. (Ratnaparkhi et al., 1994)

extended the problem instances to quadruples by also considering the kernel noun of the PP, and used maximum entropy models to estimate the preferences.

Both supervised and unsupervised training procedures for PP attachment have been investigated and compared in a number of studies, with supervised methods usually being slightly superior (Ratnaparkhi, 1998; Pantel and Lin, 2000), with the notable exception of (Volk, 2002), who obtained a worse accuracy in the supervised case, obviously caused by the limited size of the available treebank. Combining both methods can lead to a further improvement (Volk, 2002; Kokkinakis, 2000), a finding confirmed by our experiments.

Supervised training methods already applied to PP attachment range from stochastic maximum likelihood (Collins and Brooks, 1995) or maximum entropy models (Ratnaparkhi et al., 1994) to the induction of transformation rules (Brill and Resnik, 1994), decision trees (Stetina and Nagao, 1997) and connectionist models (Sopena et al., 1998). The state-of-the-art is set by (Stetina and Nagao, 1997) who generalize corpus observations to semantically similar words as they can be derived from the WordNet hierarchy.

The best result for German achieved so far is the accuracy of 80.89% obtained by (Volk, 2002). Note, however, that our goal was not to optimize the performance of PP attachment in isolation but to quantify the contribution it can make to the performance of a full parser for unrestricted text.

The accuracy of PP attachment has rarely been evaluated as a subtask of full parsing. (Merlo et al., 1997) evaluate the attachment of multiple prepositions in the same sentence for English; 85.3% accuracy is achieved for the first PP, 69.6% for the second and 43.6% for the third. This is still rather different from our setup, where PP attachment is fully integrated into the parsing problem. Closer to our evaluation scenario comes (Collins, 1999) who reports 82.3%/81.51% recall/precision on PP modifications for his lexicalized stochastic parser of English. However, no analysis has been carried out to determine which model components contributed to this result.

A more application-oriented view has been adopted by (Schwartz et al., 2003), who devised an unsupervised method to extract positive and negative lexical evidence for attachment preferences in English from a bilingual, aligned English-Japanese corpus. They used this information to reattach PPs in a machine translation system, reporting an improvement in translation quality when translating into Japanese (where PP attachment is not ambiguous and therefore matters) and a decrease when translating into Spanish (where attachment ambiguities are close to the original ones and therefore need not be resolved).

Parsing results for German have been published a number of times. Combining treebank transformation techniques with a suffix analysis, (Dubey, 2005) trained a probabilistic parser and reached a labelled F-score of 76.3% on phrase structure annotations for a subset of the sentences used here (with a maximum length of 40). For dependency parsing a labelled accuracy of 87.34% and an unlabelled one of 90.38% has been achieved by applying the dependency parser described in (McDonald et al., 2005) to German data. This system is based on a procedure for online large margin learning and considers a huge number of locally available features, which allows it to determine the optimal attachment fully deterministically. Using a stochastic variant of Constraint Dependency Grammar (Wang and Harper, 2004) reached a 92.4% labelled F-score on the Penn Treebank, which slightly outperforms (Collins, 1999) who reports 92.0% on dependency structures automatically derived from phrase structure results.

6 Conclusions and future work

Corpus-based data has been shown to provide a significant benefit when used to guide a rule-based dependency parser of German, reducing the error rate for situated PP attachment by one third. Prepositions still remain the largest source of attachment errors; many reasons can be tracked down for individual errors, such as faulty POS tagging, misinterpreted global sentence structure, genuinely ambiguous constructions, failure of the attraction heuristics, or simply lack of processing time. However, considering that even human arbiters often agree only on 90% of PP attachments, the results appear promising. In particular, many attachment errors that strongly disagree with human intuition (such as in the example sentence) were in fact prevented. Thus, the addition of a corpus-based knowledge source to the system yielded a much greater benefit than could have been achieved with the same effort by writing individual constraints.

One obvious further task is to improve our simple-minded model of lexical attraction. For instance, some remaining errors suggest that taking the kernel noun into account would yield a higher attachment precision; this will require a redesign of the extraction tools to keep the parameter space manageable. Also, other subordination types than 'PP' may benefit from similar knowledge; e.g., in many German sentences the roles of subject and object are syntactically ambiguous and can only be understood correctly through world knowledge. This is another area in which synergy between lexical attraction estimates and general symbolic rules appears possible.

References

E. Brill and P. Resnik. 1994. A rule-based approach to prepositional phrase attachment disambiguation. In *Proc. 15th Int. Conf. on Computational Linguistics*, pages 1198 – 1204, Kyoto, Japan.

M. Collins and J. Brooks. 1995. Prepositional attachment through a backed-off model. In *Proc. of the 3rd Workshop on Very Large Corpora*, pages 27–38, Somerset, New Jersey.

M. Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Phd thesis, University of Pennsylvania, Philadephia, PA.

A. Dubey. 2005. What to do when lexicalization fails: parsing German with suffix analysis and smoothing. In *Proc. 43rd Annual Meeting of the ACL*, Ann Arbor, MI.

K. Foth and W. Menzel. 2006. Hybrid parsing: Using probabilistic models as predictors for a symbolic parser. In *Proc. 21st Int. Conf. on Computational Linguistics, Coling-ACL-2006*, Sydney.

K. Foth, W. Menzel, and I. Schröder. 2000. A Transformation-based Parsing Technique with Anytime Properties. In *4th Int. Workshop on Parsing Technologies, IWPT-2000*, pages 89 – 100.

K. Foth, M. Daum, and W. Menzel. 2005. Parsing unrestricted German text with defeasible constraints. In H. Christiansen, P. R. Skadhauge, and J. Villadsen, editors, *Constraint Solving and Language Processing*, volume 3438 of *LNAI*, pages 140–157. Springer-Verlag, Berlin.

D. Hindle and M. Rooth. 1991. Structural Ambiguity and Lexical Relations. In *Meeting of the Association for Computational Linguistics*, pages 229–236.

D. Kokkinakis. 2000. Supervised pp-attachment disambiguation for swedish; (combining unsupervised supervised training data). *Nordic Journal of Linguistics*, 3.

R. McDonald, F. Pereira, K. Ribarov, and J. Hajic. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proc. Human Language Technology Conference / Conference on Empirical Methods in Natural Language Processing, HLT/EMNLP-2005*, Vancouver, B.C.

P. Merlo, M. Crocker, and C. Berthouzoz. 1997. Attaching Multiple Prepositional Phrases: Generalized Backed-off Estimation. In *Proc. 2nd Conf. on Empirical Methods in NLP*, pages 149–155, Providence, R.I.

P. Pantel and D. Lin. 2000. An unsupervised approach to prepositional phrase attachment using contextually similar words. In *Proc. 38th Meeting of the ACL*, pages 101–108, Hong Kong.

A. Ratnaparkhi, J. Reynar, and S. Roukos. 1994. A Maximum Entropy Model for Prepositional Phrase Attachment. In *Proc. ARPA Workshop on Human Language Technology*, pages 250 –255.

A. Ratnaparkhi. 1998. Statistical models for unsupervised prepositional phrase attachment. In *Proc. 17th Int. Conf. on Computational Linguistics*, pages 1079–1085, Montreal.

M. Schiehlen. 2004. Annotation Strategies for Probabilistic Parsing in German. In *Proceedings of COLING 2004*, pages 390–396, Geneva, Switzerland, Aug 23–Aug 27. COLING.

I. Schröder. 2002. *Natural Language Parsing with Graded Constraints*. Ph.D. thesis, Department of Informatics, Hamburg University, Hamburg, Germany.

L. Schwartz, T. Aikawa, and C. Quirk. 2003. Disambiguation of english PP-attachment using multilingual aligned data. In *Machine Translation Summit IX*, New Orleans, Louisiana, USA.

J. M. Sopena, A. LLoberas, and J. L. Moliner. 1998. A connectionist approach to prepositional phrase attachment for real world texts. In *Proc. 17th Int. Conf. on Computational Linguistics*, pages 1233–1237, Montreal.

J. Stetina and M. Nagao. 1997. Corpus based PP attachment ambiguity resolution with a semantic dictionary. In Jou Shou and Kenneth Church, editors, *Proc. 5th Workshop on Very Large Corpora*, pages 66–80, Hong Kong.

M. Volk. 2002. Combining Unsupervised and Supervised Methods for PP Attachment Disambiguation. In *Proc. of COLING-2002*, Taipeh.

W. Wang and M. P. Harper. 2004. A statistical constraint dependency grammar (CDG) parser. In *Proc. ACL Workshop Incremental Parsing: Bringing Engineering and Cognition Together*, pages 42–49, Barcelona, Spain.

Using Bilingual Comparable Corpora and Semi-supervised Clustering for Topic Tracking

Fumiyo Fukumoto
Interdisciplinary Graduate
School of Medicine and Engineering
Univ. of Yamanashi
fukumoto@yamanashi.ac.jp

Yoshimi Suzuki
Interdisciplinary Graduate
School of Medicine and Engineering
Univ. of Yamanashi
ysuzuki@yamanashi.ac.jp

Abstract

We address the problem dealing with *skewed data*, and propose a method for estimating effective training stories for the topic tracking task. For a small number of labelled positive stories, we extract story pairs which consist of positive and its associated stories from bilingual comparable corpora. To overcome the problem of a large number of labelled negative stories, we classify them into some clusters. This is done by using k-means with EM. The results on the TDT corpora show the effectiveness of the method.

1 Introduction

With the exponential growth of information on the Internet, it is becoming increasingly difficult to find and organize *relevant* materials. Topic Tracking defined by the TDT project is a research area to attack the problem. It starts from a few sample stories and finds all subsequent stories that discuss the target topic. Here, a topic in the TDT context is something that happens at a specific place and time associated with some specific actions. A wide range of statistical and ML techniques have been applied to topic tracking(Carbonell et. al, 1999; Oard, 1999; Franz, 2001; Larkey, 2004). The main task of these techniques is to tune the parameters or the threshold to produce optimal results. However, parameter tuning is a tricky issue for tracking(Yang, 2000) because the number of initial positive training stories is very small (one to four), and topics are localized in space and time. For example, 'Taipei Mayoral Elections' and 'U.S. Mid-term Elections' are topics, but 'Elections' is not a topic. Therefore, the system needs to estimate whether or not the test stories are the same topic with few information about the topic. Moreover, the training data is *skewed data*, i.e. there is a large number of labelled negative stories compared to positive ones. The system thus needs to balance the amount of positive and negative training stories not to hamper the accuracy of estimation.

In this paper, we propose a method for estimating efficient training stories for topic tracking. For a small number of labelled positive stories, we use bilingual comparable corpora (TDT1-3 English and Japanese newspapers, Mainichi and Yomiuri Shimbun). Our hypothesis using bilingual corpora is that many of the broadcasting station from one country report local events more frequently and in more detail than overseas' broadcasting stations, even if it is a world-wide famous ones. Let us take a look at some topic from the TDT corpora. A topic, 'Kobe Japan quake' from the TDT1 is a world-wide famous one, and 89 stories are included in the TDT1. However, Mainichi and Yomiuri Japanese newspapers have much more stories from the same period of time, i.e. 5,029 and 4,883 stories for each. These observations show that it is crucial to investigate the use of bilingual comparable corpora based on the NL techniques in terms of collecting more information about some specific topics. We extract Japanese stories which are relevant to the positive English stories using English-Japanese bilingual corpora, together with the EDR bilingual dictionary. The associated story is the result of alignment of a Japanese term association with an English term association.

For a large number of labelled negative stories, we classify them into some clusters using labelled positive stories. We used a semi-supervised clustering technique which combines

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 231–238,
Sydney, July 2006.

labeled and unlabeled stories during clustering. Our goal for semi-supervised clustering is to classify negative stories into clusters where each cluster is $meaningful$ in terms of class distribution provided by one cluster of positive training stories. We introduce k-means clustering that can be viewed as instances of the EM algorithm, and classify negative stories into clusters. In general, the number of clusters k for the k-means algorithm is not given beforehand. We thus use the Bayesian Information Criterion (BIC) as the splitting criterion, and select the proper number for k.

2 Related Work

Most of the work which addresses the small number of positive training stories applies statistical techniques based on word distribution and ML techniques. Allan et. al explored on-line adaptive filtering approaches based on the threshold strategy to tackle the problem(Allan et. al, 1998). The basic idea behind their work is that stories closer together in the stream are more likely to discuss related topics than stories further apart. The method is based on unsupervised learning techniques except for its incremental nature. When a tracking query is first created from the N_t training stories, it is also given a threshold. During the tracking phase, if a story S scores over that threshold, S is regarded to be relevant and the query is regenerated as if S were among the N_t training stories. This method was tested using the TDT1 corpus and it was found that the adaptive approach is highly successful. But adding more than four training stories provided only little help, although in their approach, 12 training stories were added. The method proposed in this paper is similar to Allan's method, however our method for collecting relevant stories is based on story pairs which are extracted from bilingual comparable corpora.

The methods for finding bilingual story pairs are well studied in the cross-language IR task, or MT systems/bilingual lexicons(Dagan, 1997). Much of the previous work uses cosine similarity between story term vectors with some weighting techniques(Allan et. al, 1998) such as TF-IDF, or cross-language similarities of terms. However, most of them rely on only two stories in question to estimate whether or not they are about the same topic. We use $multiple\text{-}links$ among stories to produce optimal results.

In the TDT tracking task, classifying negative stories into $meaningful$ groups is also an important issue to track topics, since a large number of labelled negative stories are available in the TDT context. Basu et. al. proposed a method using k-means clustering with the EM algorithm, where labeled data provides prior information about the conditional distribution of hidden category labels(Basu, 2002). They reported that the method outperformed the standard random seeding and COP-k-means(Wagstaff, 2001). Our method shares the basic idea with Basu et. al. An important difference with their method is that our method does not require the number of clusters k in advance, since it is determined during clustering. We use the BIC as the splitting criterion, and estimate the proper number for k. It is an important feature because in the tracking task, no knowledge of the number of topics in the negative training stories is available.

3 System Description

The system consists of four procedures: extracting bilingual story pairs, extracting monolingual story pairs, clustering negative stories, and tracking.

3.1 Extracting Bilingual Story Pairs

We extract story pairs which consist of positive English story and its associated Japanese stories using the TDT English and Mainichi and Yomiuri Japanese corpora. To address the optimal positive English and their associated Japanese stories, we combine the output of similarities(multiple-links). The idea comes from speech recognition where two outputs are combined to yield a better result in average. Fig.1 illustrates multiple-links. The TDT English corpus consists of training and test stories. Training stories are further divided into positive(black box) and negative stories(doted box). Arrows in Fig.1 refer to an edge with similarity value between stories. In Fig.1, for example, whether the story J_2 discusses the target topic, and is related to E_1 or not is determined by not only the value of similarity between E_1 and J_2, but also the similarities between J_2 and J_4, E_1 and J_4.

Extracting story pairs is summarized as follows: Let initial $positive$ training stories $E_1, \cdots, E_m$ be initial node, and each Japanese stories $J_1, \cdots, J_{m'}$ be node or terminal node in the graph G. We calculate cosine similarities between $E_i (1 \leq i \leq m)$ and $J_j (1 \leq j \leq m')$[1]. In a similar way, we calcu-

[1] m' refers to the difference of dates between English and

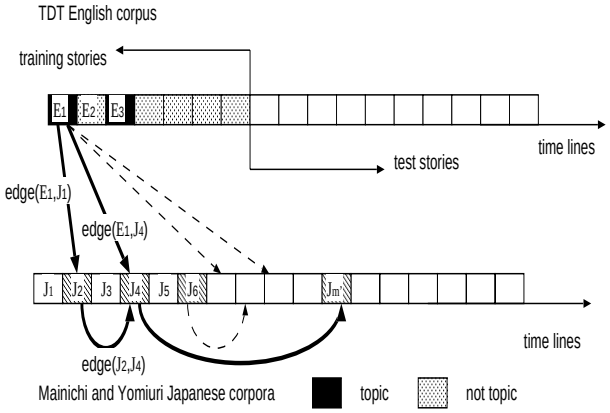


Figure 1: Multiple-links among stories

late similarities between J_k and $J_l (1 \leq k, l \leq m')$. If the value of similarity between nodes is larger than a certain threshold, we connect them by an edge(bold arrow in Fig.1). Next, we delete an edge which is not a constituent of maximal connected sub-graph(doted arrow in Fig.1). After eliminating edges, we extract pairs of initial positive English story E_i and Japanese story J_j as a linked story pair, and add associated Japanese story J_j to the training stories. In Fig.1, E_1, J_2, and J_4 are extracted. The procedure for calculating cosine similarities between E_i and J_j consists of two sub-steps: extracting terms, and estimating bilingual term correspondences.

Extracting terms

The first step to calculate similarity between E_i and J_j is to align a Japanese term with its associated English term using the bilingual dictionary, EDR. However, this naive method suffers from frequent failure due to incompleteness of the bilingual dictionary. Let us take a look at the Mainichi Japanese newspaper stories. The total number of terms(words) from Oct. 1, 1998 to Dec. 31, 1998, was 528,726. Of these, 370,013 terms are not included in the EDR bilingual dictionary. For example, ' エンデバー (Endeavour)' which is a key term for the topic 'Shuttle Endeavour mission for space station' from the TDT3 corpus is not included in the EDR bilingual dictionary. New terms which fail to segment by during a morphological analysis are also a problem in calculating similarities between stories in monolingual data. For example, a proper noun '首都大学東京'(Tokyo Metropolitan Univ.) is divided into three terms, '首都' (Metropolitan), '大学 (Univ.)', and '東京 (Tokyo)'. To tackle these problems, we conducted term extraction from a large collection of English and Japanese corpora. There are several techniques for term extraction(Chen, 1996). We used n-gram model with Church-Gale smoothing, since Chen reported that it outperforms all existing methods on bigram models produced from large training data. The length of the extracted terms does not have a fixed range[2]. We thus applied the normalization strategy which is shown in Eq.(1) to each length of the terms to bring the probability value into the range [0,1]. We extracted terms whose probability value is greater than a certain threshold. Words from the TDT English(Japanese newspaper) corpora are identified if they match the extracted terms.

$$sim_{new} = \frac{sim_{old} - sim_{min}}{sim_{max} - sim_{min}} \quad (1)$$

Table 1: t_E and t_J matrix

		t_E	
		$t_E \in s'_E$	$t_E \notin s'_E$
t_J	$t_J \in S'^i_J$	a	b
	$t_J \notin S'^i_J$	c	d

Bilingual term correspondences

The second step to calculate similarity between E_i and J_j is to estimate bilingual term correspondences using χ^2 statistics. We estimated bilingual term correspondences with a large collection of English and Japanese data. More precisely, let E_i be an English story ($1 \leq i \leq n$), where n is the number of stories in the collection, and S^i_J denote the set of Japanese stories with cosine similarities higher than a certain threshold value θ: $S^i_J = \{J_j \mid \cos(E_i, J_j) \geq \theta\}$. Then, we concatenate constituent Japanese stories of S^i_J into one story S'^i_J, and construct a pseudo-parallel corpus PPC_{EJ} of English and Japanese stories: $PPC_{EJ} = \{ \{ E_i, S'^i_J \} \mid S^i_J \neq 0 \}$. Suppose that there are two criteria, monolingual term t_E in English story and t_J in Japanese story. We can determine whether or not a particular term belongs to a particular story. Consequently, terms are divided into four classes, as shown in Table 1. Based on the contingency table of co-occurence frequencies of t_E and t_J, we estimate bilingual term correspondences according to the statistical measure χ^2.

$$\chi^2(t_E, t_J) = \frac{(ad - bc)^2}{(a+b)(a+c)(b+d)(c+d)} \quad (2)$$

Japanese story pairs.

[2]We set at most five noun words.

We extract term t_J as a pair of t_E which satisfies maximum value of χ^2, i.e. $\max_{t_J \in T_J}\ \chi^2(t_E,t_J)$, where $T_J = \{t_J \mid \chi^2(t_E,t_J)\}$. For the extracted English and Japanese term pairs, we conducted semi-automatic acquisition, i.e. we manually selected bilingual term pairs, since our source data is not a clean parallel corpus, but an artificially generated noisy pseudo-parallel corpus, it is difficult to compile bilingual terms full-automatically(Dagan, 1997). Finally, we align a Japanese term with its associated English term using the selected bilingual term correspondences, and again calculate cosine similarities between Japanese and English stories.

3.2 Extracting Monolingual Story Pairs

We noted above that our source data is not a clean parallel corpus. Thus the difference of dates between bilingual stories is one of the key factors to improve the performance of extracting story pairs, i.e. stories closer together in the timeline are more likely to discuss related subjects. We therefore applied a method for extracting bilingual story pairs from stories closer in the timelines. However, this often hampers our basic motivation for using bilingual corpora: bilingual corpora helps to collect more information about the target topic. We therefore extracted monolingual(Japanese) story pairs and added them to the training stories. Extracting Japanese monolingual story pairs is quite simple: Let J_j ($1 \leq j \leq m'$) be the extracted Japanese story in the procedure, extracting bilingual story pairs. We calculate cosine similarities between J_j and J_k($1 \leq k \leq n$). If the value of similarity between them is larger than a certain threshold, we add J_k to the training stories.

3.3 Clustering Negative Stories

Our method for classifying negative stories into some clusters is based on Basu et. al.'s method(Basu, 2002) which uses k-means with the EM algorithm. K-means is a clustering algorithm based on iterative relocation that partitions a dataset into the number of k clusters, locally minimizing the average squared distance between the data points and the cluster centers(centroids). Suppose we classify $X = \{\ x_1, \cdots, x_N\}$, $x_i \in R^d$ into k clusters: one is the cluster which consists of positive stories, and other k-1 clusters consist of negative stories. Here, which clusters does each negative story belong to? The EM is a method of finding the maximum-likelihood estimate(MLE) of the parameters of an underlying distribution from a set of observed data that has missing value. K-means is essentially an EM on a mixture of k Gaussians under certain assumptions. In the standard k-means without any initial supervision, the k-means are chosen randomly in the initial M-step and the stories are assigned to the nearest means in the subsequent E-step. For positive training stories, the initial labels are kept unchanged throughout the algorithm, whereas the conditional distribution for the negative stories are re-estimated at every E-step. We select the number of k initial stories: one is the cluster center of positive stories, and other k-1 stories are negative stories which have the top k-1 smallest value between the negative story and the cluster center of positive stories. In Basu et. al's method, the number of k is given by a user. However, for negative training stories, the number of clusters is not given beforehand. We thus developed an algorithm for estimating k. It goes into action after each run of k means[3], making decisions about which sets of clusters should be chosen in order to better fit the data. The splitting decision is done by computing the Bayesian Information Criterion which is shown in Eq.(3).

$$BIC(k=l) \quad = \quad \hat{ll}_l(X) - \frac{p_l}{2} \cdot \log N \qquad (3)$$

where $\hat{ll}_l(X)$ is the log-likelihood of X according to the number of k is l, N is the total number of training stories, and p_l is the number of parameters in $k = l$. We set p_l to the sum of k class probabilities, $\sum_{m=1}^{k} \hat{ll}(X_m)$, the number of $n \cdot k$ centroid coordinates, and the MLE for the variance, $\hat{\rho}^2$. Here, n is the number of dimensions. $\hat{\rho}^2$, under the identical spherical Gaussian assumption, is:

$$\hat{\rho}^2 \quad = \quad \frac{1}{N-k} \sum_i (x_i - \mu_i)^2 \qquad (4)$$

where μ_i denotes i-th partition center. The probabilities are:

$$\hat{P}(x_i) \quad = \quad \frac{R_i}{N} \cdot \frac{1}{\sqrt{2\pi}\hat{\rho}^n} exp(-\frac{1}{2\hat{\rho}^2} \mid\mid x_i - \mu_i \mid\mid^2) \qquad (5)$$

R_i is the number of stories that have μ_i as their closest centroid. The log-likelihood of $ll(X)$

[3]We set the maximum number of k to 100 in the experiment.

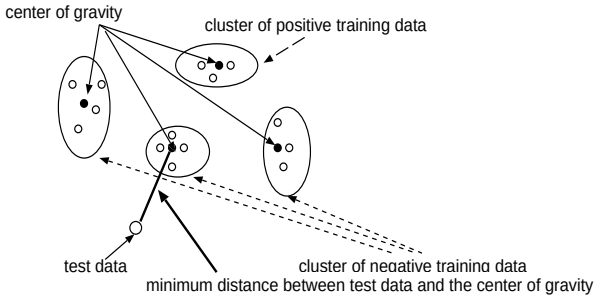


Figure 2: Each cluster and a test story

is $\log \prod_i P(x_i)$. It is taken at the maximum-likelihood point(story), and thus, focusing just on the set $X_m \subseteq X$ which belongs to the centroid m and plugging in the MLE yields:

$$\hat{ll}(X_m) = -\frac{R_m}{2}\log(2\pi) - \frac{R_m \cdot n}{2}\log(\hat{\rho^2}) - \frac{R_m - k}{2} + R_m \log R_m - R_m \log N \quad (1 \leq m \leq k) \quad (6)$$

We choose the number of k whose value of BIC is highest.

3.4 Tracking

Each story is represented as a vector of terms with $tf \cdot idf$ weights in an n dimensional space, where n is the number of terms in the collection. Whether or not each test story is positive is judged using the distance (measured by cosine similarity) between a vector representation of the test story and each centroid $\mathbf{g}$ of the clusters. Fig.2 illustrates each cluster and a test story in the tracking procedure. Fig.2 shows that negative training stories are classified into three groups. The centroid $\mathbf{g}$ for each cluster is calculated as follows:

$$\mathbf{g} = (g_1, \cdots, g_n) = (\frac{1}{p}\sum_{i=1}^{p} x_{i1}, \cdots, \frac{1}{p}\sum_{i=1}^{p} x_{in}) \quad (7)$$

where x_{ij} $(1 \leq j \leq n)$ is the $tf \cdot idf$ weighted value of term j in the story $\mathbf{x}_i$. The test story is judged by using these centroids. If the value of cosine similarity between the test story and the centroid with positive stories is smallest among others, the test story is declared to be positive. In Fig.2, the test story is regarded as negative, since the value between them is smallest. This procedure, is repeated until the last test story is judged.

4 Experiments

4.1 Creating Japanese Corpus

We chose the TDT3 English corpora as our gold standard corpora. TDT3 consists of 34,600 stories with 60 manually identified topics. We then created Japanese corpora (Mainichi and Yomiuri newspapers) to evaluate the method. We annotated the total number of 66,420 stories from Oct.1, to Dec.31, 1998, against the 60 topics. Each story was labelled according to whether the story discussed the topic or not. Not all the topics were present in the Japanese corpora. We therefore collected 1 topic from the TDT1 and 2 topics from the TDT2, each of which occurred in Japan, and added them in the experiment. TDT1 is collected from the same period of dates as the TDT3, and the first story of 'Kobe Japan Quake' topic starts from Jan. 16th. We annotated 174,384 stories of Japanese corpora from the same period for the topic. Table 2 shows 24 topics which are included in the Japanese corpora. 'TDT' refers to the evaluation data, TDT1, 2, or 3. 'ID' denotes topic number defined by the TDT. 'OnT.'(On-Topic) refers to the number of stories discussing the topic. Bold font stands for the topic which happened in Japan. The evaluation of annotation is made by three humans. The classification is determined to be correct if the majority of three human judges agree.

4.2 Experiments Set Up

The English data we used for extracting terms is Reuters'96 corpus(806,791 stories) including TDT1 and TDT3 corpora. The Japanese data was 1,874,947 stories from 14 years(from 1991 to 2004) Mainichi newspapers(1,499,936 stories), and 3 years(1994, 1995, and 1998) Yomiuri newspapers(375,011 stories). All Japanese stories were tagged by the morphological analysis Chasen(Matsumoto, 1997). English stories were tagged by a part-of-speech tagger(Schmid, 1995), and stop word removal. We applied n-gram model with Church-Gale smoothing to noun words, and selected terms whose probabilities are higher than a certain threshold[4]. As a result, we obtained 338,554 Japanese and 130,397 English terms. We used the EDR bilingual dictionary, and translated Japanese terms into English. Some of the words had no translation. For these, we estimated term correspondences. Each story is represented as a vector of terms with $tf \cdot idf$ weights. We calculated story similarities and extracted story pairs between positive and its associated stories[5]. In

[4]The threshold value for both English and Japanese was 0.800. It was empirically determined.

[5]The threshold value for bilingual story pair was 0.65, and that for monolingual was 0.48. The difference of dates between bilingual stories was ± 4.

Table 2: Topic Name

TDT	ID	Topic name	OnT.	TDT	ID	Topic name	OnT.
1	15	**Kobe Japan quake**	9,912				
2	31015	**Japan Apology to Korea**	28	2	31023	**Kyoto Energy Protocol**	40
3	30001	Cambodian government coalition	48	3	30003	Pinochet trial	165
3	30006	NBA labor disputes	44	3	30014	Nigerian gas line fire	6
3	30017	North Korean food shortages	23	3	30018	Tony Blair visits China in Oct.	7
3	30022	Chinese dissidents sentenced	21	3	30030	Taipei Mayoral elections	353
3	30031	Shuttle Endeavour mission for space station	17	3	30033	Euro Introduced	152
3	30034	Indonesia-East Timor conflict	34	3	30038	Olympic bribery scandal	35
3	30041	**Jiang's Historic Visit to Japan**	111	3	30042	PanAm lockerbie bombing trial	13
3	30047	Space station module Zarya launched	30	3	30048	IMF bailout of Brazil	28
3	30049	North Korean nuclear facility?	111	3	30050	U.S. Mid-term elections	123
3	30053	Clinton's Gaza trip	74	3	30055	D'Alema's new Italian government	37
3	30057	India train derailment	12				

the tracking, we used the extracted terms together with all verbs, adjectives, and numbers, and represented each story as a vector of these with $tf{\cdot}idf$ weights.

We set the evaluation measures used in the TDT benchmark evaluations. 'Miss' denotes Miss rate, which is the ratio of the stories that were judged as YES but were not evaluated as such for the run in question. 'F/A' shows false alarm rate, which is the ratio of the stories judged as NO but were evaluated as YES. The DET curve plots misses and false alarms, and better performance is indicated by curves more to the lower left of the graph. The detection cost function(C_{Det}) is defined by Eq.(8).

$$\begin{aligned} C_{Det} &= (C_{Miss} * P_{Miss} * P_{Target} + \\ &\quad C_{Fa} * P_{Fa} * (1 - P_{Target})) \\ P_{Miss} &= \#Misses/\#Targets \\ P_{Fa} &= \#FalseAlarms/\#NonTargets \end{aligned} \quad (8)$$

C_{Miss}, C_{Fa}, and P_{Target} are the costs of a missed detection, false alarm, and priori probability of finding a target, respectively. C_{Miss}, C_{Fa}, and P_{Target} are usually set to 10, 1, and 0.02, respectively. The normalized cost function is defined by Eq.(9), and lower cost scores indicate better performance.

$$\begin{aligned} (C_{Det})_{Norm} &= C_{Det}/MIN(C_{Miss} * P_{Target}, C_{Fa} \\ &\quad *(1 - P_{Target})) \end{aligned} \quad (9)$$

4.3 Basic Results

Table 3 summaries the tracking results. MIN denotes $MIN(C_{Det})_{Norm}$ which is the value of $(C_{Det})_{Norm}$ at the best possible threshold. N_t is the number of initial positive training stories. We recall that we used subset of the topics defined by the TDT. We thus implemented Allan's method(Allan et. al, 1998) which is similar to our method, and compared the results. It is based on a tracking query which is created from the top 10 most commonly occurring features in the N_t stories, with weight equal to the number of times the term occurred in those stories multiplied by its incremental idf value. They used a shallow tagger and selected all nouns, verbs, adjectives, and numbers. We added the extracted terms to these part-of-speech words to make their results comparable with the results by our method. 'Baseline' in Table 3 shows the best result with their method among varying threshold values of similarity between queries and test stories. We can see that the performance of our method was competitive to the baseline at every N_t value.

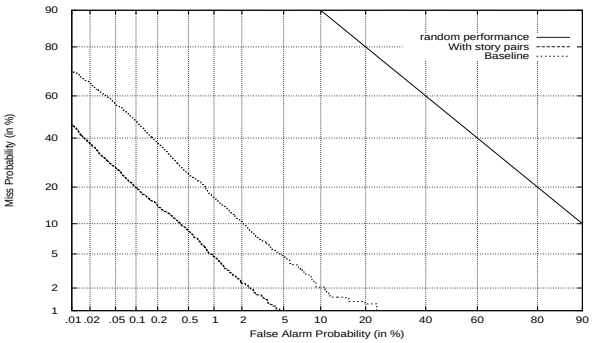


Figure 3: Tracking result(23 topics)

Fig.3 shows DET curves by both our method and Allan's method(baseline) for 23 topics from the TDT2 and 3. Fig.4 illustrates the results for 3 topics from TDT2 and 3 which occurred in Japan. To make some comparison possible, only the N_t = 4 is given for each. Both Figs. show that we have an advantage using bilingual comparable corpora.

4.4 The Effect of Story Pairs

The contribution of the extracted story pairs, especially the use of two types of story pairs, bilingual and monolingual, is best explained by looking at the two results: (i) the tracking results with two types of story pairs, with only English and

Table 3: Basic results

TDT1 (Kobe Japan Quake)

Baseline							Bilingual corpora & clustering						
N_t	Miss	F/A	Recall	Precision	F	MIN	N_t	Miss	F/A	Recall	Precision	F	MIN
1	27%	.15%	73%	67%	.70	.055	1	10%	.42%	90%	74%	.81	.023
2	20%	.12%	80%	73%	.76	.042	2	6%	.27%	93%	76%	.83	.013
4	9%	.09%	91%	80%	.85	.039	4	5%	.18%	96%	81%	.88	.012

TDT2 & TDT3(23 topics)

Baseline							Bilingual corpora & clustering						
N_t	Miss	F/A	Recall	Precision	F	MIN	N_t	Miss	F/A	Recall	Precision	F	MIN
1	41%	.17%	59%	60%	.60	.089	1	29%	.25%	71%	54%	.61	.059
2	40%	.16%	60%	62%	.61	.072	2	27%	.25%	73%	55%	.63	.054
4	29%	.12%	71%	72%	.71	.057	4	20%	.13%	80%	73%	.76	.041

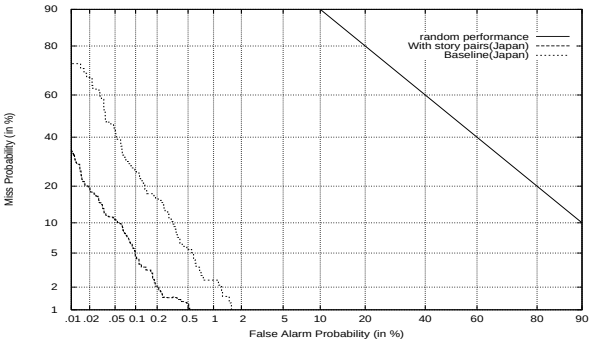


Figure 4: 3 topics concerning to Japan

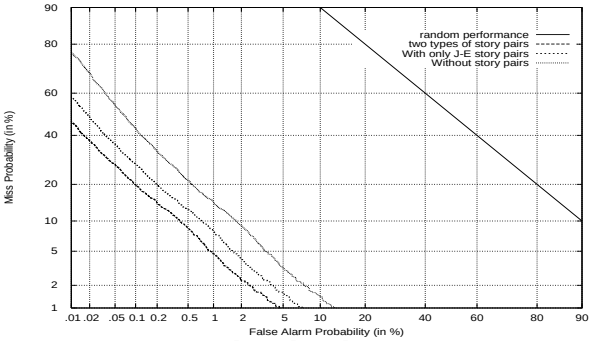


Figure 5: With and without story pairs

Japanese stories in question, and without story pairs, and (ii) the results of story pairs by varying values of N_t. Fig.5 illustrates DET curves for 23 topics, N_t=4.

As can be clearly seen from Fig.5, the result with story pairs improves the overall performance, especially the result with two types of story pairs was better than that with only English and Japanese stories in question. Table 4 shows the performance of story pairs which consist of positive and its associated story. Each result denotes micro-averaged scores. 'Rec.' is the ratio of correct story pair assignments by the system divided by the total number of correct assignments. 'Prec.' is the ratio of correct story pair assignments by the system divided by the total number of system's assignments. Table 4 shows that the system with two types of story pairs correctly extracted stories related to the target topic even for a small number of positive training stories, since the ratio of Prec. in N_t = 1 is 0.82. However, each recall value in Table 4 is low. One solution is to use an incremental approach, i.e. by repeating story pairs extraction, new story pairs that are not extracted previously may be extracted. This is a rich space for further exploration.

Table 4: Performance of story pairs(24 topics)

	Two types of story pairs			J-E story pairs		
N_t	Rec.	Prec.	F	Rec.	Prec.	F
1	30%	82%	.439	28%	80%	.415
2	36%	85%	.506	33%	82%	.471
4	45%	88%	.595	42%	79%	.548

The effect of story pairs for the tracking task also depends on the performance of bilingual term correspondences. We obtained 1,823 English and Japanese term pairs in all when a period of days was $\pm$4. Fig.6 illustrates the result using different period of days($\pm$1 to $\pm$10). For example, '$\pm$1' shows that the difference of dates between English and Japanese story pairs is less than $\pm$1. Y-axis shows the precision which is the ratio of correct term pairs by the system divided by the total number of system's assignments. Fig.6 shows that the difference of dates between bilingual story pairs, affects the overall performance.

4.5 The Effect of k-means with EM

The contribution of k-means with EM for classifying negative stories is explained by looking at the result without classifying negative stories. We calculated the centroid using all negative training stories, and a test story is judged to be negative or

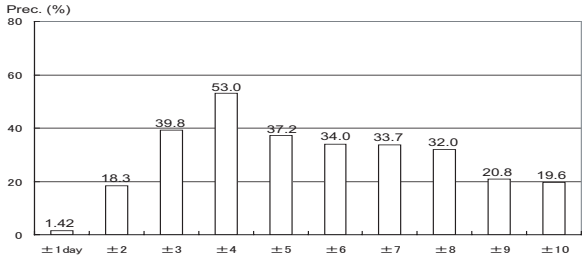


Figure 6: Prec. with different period of days

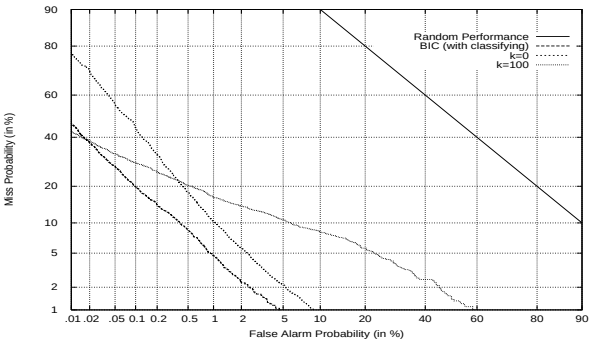


Figure 7: BIC v.s. fixed k for k-means with EM

positive by calculating cosine similarities between the test story and each centroid of negative and positive stories. Further, to examine the effect of using the BIC, we compared with choosing a pre-defined k, i.e. k=10, 50, and 100. Fig.7 illustrates part of the result for k=100. We can see that the method without classifying negative stories(k=0) does not perform as well and results in a high miss rate. This result is not surprising, because the size of negative training stories is large compared with that of positive ones, and therefore, the test story is erroneously judged as NO. Furthermore, the result indicates that we need to run BIC, as the result was better than the results with choosing any number of pre-defined k, i.e. k=10, 50, and 100. We also found that there was no correlation between the number of negative training stories for each of the 24 topics and the number of clusters k obtained by the BIC. The minimum number of clusters k was 44, and the maximum was 100.

5 Conclusion

In this paper, we addressed the issue of the difference in sizes between positive and negative training stories for the tracking task, and investigated the use of bilingual comparable corpora and semi-supervised clustering. The empirical results were encouraging. Future work includes (i) extending the method to an incremental approach for extracting story pairs, (ii) comparing our clustering method with the other existing methods such as X-means(Pelleg, 2000), and (iii) applying the method to the TDT4 for quantitative evaluation.

Acknowledgments

This work was supported by the Grant-in-aid for the JSPS, Support Center for Advanced Telecommunications Technology Research, and International Communications Foundation.

References

J.Allan and R.Papka and V.Lavrenko, *On-line new event detection and tracking*, Proc. of the DARPA Workshop, 1998.

J.Allan and V.Lavrenko and R.Nallapti, *UMass at TDT 2002*, Proc. of TDT Workshop, 2002.

S.Basu and A.Banerjee and R.Mooney, *Semi-supervised clustering by seeding*, Proc. of ICML'02, 2002.

J.Carbonell et. al, *CMU report on TDT-2: segmentation, detection and tracking*, Proc. of the DARPA Workshop, 1999.

S.F.Chen and J.Goodman, *An empirical study of smoothing techniques for language modeling*, Proc. of the ACL'96, pp. 310-318, 1996.

N.Collier and H.Hirakawa and A.Kumano, *Machine translation vs. dictionary term translation - a comparison for English-Japanese news article alignment*, Proc. of COLING'02, pp. 263-267, 2002.

I.Dagan and K.Church, *Termight: Coordinating humans and machines in bilingual terminology acquisition*, Journal of MT, Vol. 20, No. 1, pp. 89-107, 1997.

M.Franz and J.S.McCarley, *Unsupervised and supervised clustering for topic tracking*, Proc. of SIGIR'01, pp. 310-317, 2001.

L.S.Larkey et. al, *Language-specific model in multilingual topic tracking*, Proc. of SIGIR'04, pp. 402-409, 2004.

Y.Matsumoto et. al, *Japanese morphological analysis system chasen manual*, NAIST Technical Report, 1997.

D.W.Oard, *Topic tracking with the PRISE information retrieval system*, *Proc. of the DARPA Workshop*, pp. 94-101, 1999.

D.Pelleg and A.Moore, *X-means: Extending K-means with efficient estimation of the number of clusters*, *Proc. of ICML'00*, pp. 727-734, 2000.

H.Schmid, *Improvements in part-of-speech tagging with an application to german*, *Proc. of the EACL SIGDAT Workshop*, 1995.

K.Wagstaff et. al, *Constrained K-means clustering with background knowledge*, *Proc. of ICML'01*, pp. 577-584, 2001.

Y.Yang et. al, *Improving text categorization methods for event tracking*, *Proc. of SIGIR'00*, pp. 65-72, 2000.

Robust Word Sense Translation by EM Learning of Frame Semantics

Pascale Fung and Benfeng Chen
Human Language Technology Center
Department of Electrical & Electronic Engineering
University of Science & Technology (HKUST)
Clear Water Bay
Hong Kong
{pascale,bfchen}@ee.ust.hk

Abstract

We propose a robust method of automatically constructing a bilingual word sense dictionary from readily available monolingual ontologies by using estimation-maximization, without any annotated training data or manual tuning. We demonstrate our method on the English FrameNet and Chinese HowNet structures. Owing to the robustness of EM iterations in improving translation likelihoods, our word sense translation accuracies are very high, at 82% on average, for the 11 most ambiguous words in the English FrameNet with 5 senses or more. We also carried out a pilot study on using this automatically generated bilingual word sense dictionary to choose the best translation candidates and show the first significant evidence that frame semantics are useful for translation disambiguation. Translation disambiguation accuracy using frame semantics is 75%, compared to 15% by using dictionary glossing only. These results demonstrate the great potential for future application of bilingual frame semantics to machine translation tasks.

1 Introduction

As early as in the 1950s, semantic nets were invented as an "interlingua" for machine translation.

The "semantic net" or "semantic map" that humans possess in the cognitive process is a structure of concept classes and lexicon (Illes and Francis 1999). In addition, the frame-semantic representation of predicate-argument relations has gained much attention in the research community. The Berkeley FrameNet (Baker et al. 1998) is such an example.

We suggest that in addition to dictionaries, bilingual frame semantics (word sense dictionary) is a useful resource for lexical selection in the translation process of a statistical machine translation system. Manual inspection of the contrastive error analysis data from a state-of-the-art SMT system showed that around 20% of the error sentences produced could have been avoided if the correct predicate argument information was used (Och et al. 2003). Therefore, frame semantics can provide another layer of translation disambiguation in these systems.

We therefore propose to generate a bilingual frame semantics mapping (word sense dictionary), simulating the "semantic map" in a bilingual speaker. Other questions of interest to us include how concept classes in English and Chinese break down and map to each other.

This paper is organized as follows. In section 2, we present the one-frame-two-languages idea of bilingual frame semantics representation. In section 3, we explain the EM algorithm for generating a bilingual ontology fully automatically. In section 4, we present an evaluation on word sense translation. Section 5 describes an evaluation on how well bilingual frame semantics can improve translation disambiguation. We then discuss related work in section 6, conclude in section 7, and finally discuss future work in section 8.

2 One Frame Two Languages

The challenge of translation disambiguation is to select the target word *cl** with the correct semantic *frame f--(cl,f),* among the multitude of translation candidates Pr*(cl|el)*. We suggest that while a source word in the input sentence might have multiple translation candidates, the correct target word must have the same sense, i.e., belong to the same semantic frame, as the source word (i.e. *Pr(cl,f|el,f)* is high). For example, "burn| 烫

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 239–246,
Sydney, July 2006.

(tang)" carries the "*cause_harm|damage*" sense, whereas "burn|烧(shao)" carries the "*heat|cooking*" sense. The source sentence "*My hands are burned*" has the "*cause_harm|damage*" sense, therefore the correct translation of "burn" is "烫(tang)" not "烧(shao)". The frame semantics information of the source word can thus lead to the best translation candidate.

Whereas some translation ambiguities are preserved over languages, most are not. In particular, for languages as different as English and Chinese, there is little overlap between how lexicon is broken-down (Ploux and Ji 2003). Some cognitive scientists suggest that a bilingual speaker tends to group concepts in a single semantic map and simply attach different words in English and Chinese to the categories in this map.

Based on the above, we propose the one-frame-two-languages idea for constructing a bilingual word sense dictionary from monolingual ontologies.

FrameNet (Baker et al. 1998) is a collection of lexical entries grouped by frame semantics. Each lexical entry represents an individual word sense, and is associated with semantic roles and some annotated sentences. Lexical entries with the same semantic roles are grouped into a "frame" and the semantic roles are called "frame elements". Each frame in FrameNet is a concept class and a single word sense belongs to only one frame. However, the Chinese HowNet represents a hierarchical view of lexical semantics in Chinese.

HowNet (Dong and Dong 2000) is a Chinese ontology with a graph structure of word senses called "concepts", and each concept contains 7 fields including lexical entries in Chinese, English gloss, POS tags for the word in Chinese and English, and a definition of the concept including its category and semantic relations (Dong and Dong, 2000). Whereas HowNet concepts correspond roughly to FrameNet lexical entries, its semantic relations do not correspond directly to FrameNet semantic roles.

A bilingual frame, as shown in Figure 1, simulates the semantic system of a bilingual speaker by having lexical items in two languages attached to the frame.

3 Automatic Generation of Bilingual Frame Semantics

To choose "burn|烫(tang)" instead of "burn|烧(shao)" in the translation of "*My hands are burned*", we need to know that "烫(tang)" belongs to the "*cause_harm*" frame, but "烧(shao)" belongs to the "*heat*" frame. In other words, we need to have a bilingual frame semantics ontology. Much like a dictionary, this bilingual ontology forms part of the translation "lexicon", and can be used either by human translators or automatic translation systems.

Such a bilingual frame semantics ontology also provides a simulation of the "concept lexicon" of a bilingual person, as suggested by cognitive scientists.

Figure 1 shows an example of a bilingual frame that *possibly* corresponds to the semantic structure in a bilingual person.

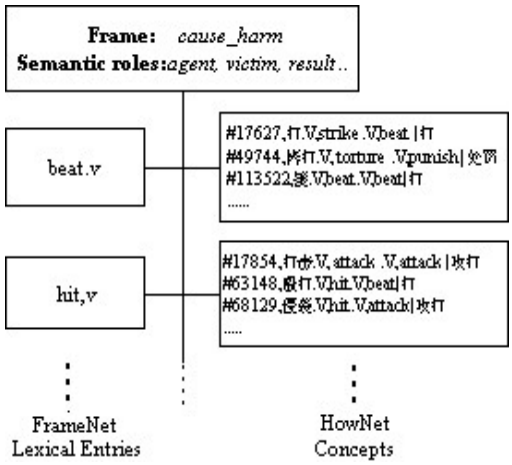


Figure 1. An example bilingual frame

We previously proposed using the Chinese HowNet and a bilingual lexicon to map the English FrameNet into a bilingual BiFrameNet (Fung and Chen 2004). We used a combination of frame size thresholding and taxonomy distance to obtain the final alignment between FrameNet frames and HowNet categories, to generate the BiFrameNet.

Our previous algorithm had the disadvantage of requiring the ad hoc tuning of thresholds. This results in poor performance on lexical entries from small frames (i.e. frames with very few lexical entries). The tuning process also means that a development set of annotated data is needed. In this paper, we propose a fully automatic estimation-maximization algorithm instead, to generate a similar FrameNet to HowNet

bilingual ontology, without requiring any annotated data or manual tuning. As such, our method can be applied to ontologies of any structure, and is not restricted to FrameNet or HowNet.

Our approach is based on the following assumptions:

1. A source semantic frame is mapped to a target semantic frame if many word senses in the two frames translate to each other;
2. A source word sense translates into a target word sense if their parent frames map to each other.

The semantic frame in FrameNet is defined as a single frame, whereas in HowNet it is defined as the category. The formulae of our proposed algorithm are listed in Figure 2.

Variable definitions:
cl : Chinese lexeme .
cf : Chinese frame.
(cl, cf) : the word sense entry in cf .
el : English lexeme .
ef : English frame.
(el, ef) : the word sense entry in ef .
(All variables are assumed to be independent of each other.)

Model parameters:
Pr$(cl|el)$: bilingual word pair probability from dictionary
Pr$(cf|ef)$: Chinese to English frame mapping probability.
Pr$(cl,cf|el,ef)$: Chinese to English word sense translation probability.

(1) Word senses that belong to mapped frames are translated to each other:

$$\Pr(cl, cf \mid el, ef) = \frac{\Pr(cl \mid el) \cdot \Pr(cf \mid ef)}{\sum_{\forall cf} \Pr(cl, cf \mid el, ef)}$$

where we assume the a priori probability

$$\Pr(cl) = 1 \quad \forall cl$$

(2) Frames that have translated word senses are mapped to each other:

$$\Pr(cf \mid ef) = \frac{\sum_{\forall el} \sum_{\forall cl} \Pr(cl, cf \mid el, ef)}{\sum_{\forall cf} \sum_{\forall el} \sum_{\forall cl} \Pr(cl, cf \mid el, ef)}$$

Figure 2. The bilingual frame semantics formulae

In the initialization step of our EM algorithm, all English words in FrameNet are glossed into Chinese using a bilingual lexicon with uniform probabilities Pr$(cl|el)$. Next, we apply the EM algorithm to align FrameNet frames and HowNet categories. By using EM, we improve the probabilities of frame mapping in Pr$(cf|ef)$ and word sense translations in $Pr(cl,cf|el,ef)$ iteratively: We estimate sense translations based on uniform bilingual dictionary probabilities $Pr(cl|el)$ first. The frame mappings are maximized by using the estimated sense translation. The *a priori* lexical probability $Pr(cl)$ is assumed to be one for all Chinese words. Underlining the correctness of our algorithm, we note that the overall likelihoods of the model parameters in our algorithm improve until convergence after 11 iterations. We use the alignment output after the convergence step. That is, we obtain all word sense translations and frame mapping from the EM algorithm:

$$(cl, cf)^* = \arg\max_{(cl,cf)} \Pr(cl, cf \mid el, ef) \quad \forall (el, ef)$$

$$cf^* = \arg\max_{cf} \Pr(cf \mid ef) \quad \forall ef$$

The mapping between FrameNet frames and HowNet categories is obviously not one-to-one since the two languages are different. The initial and final mappings before and after EM iterations are shown in Figures 3a,b and 4a,b. Each point (i,j) in Figures 3a and b represents an alignment between FrameNet frame i to HowNet category j. Before EM iterations, each English lexical item is glossed into its (multiple) Chinese translations by a bilingual dictionary. The parent frame of the English lexical item and those of *all* its Chinese translations are aligned to form an initial mapping. This initial mapping shows that each English FrameNet frame is aligned to an average of 56 Chinese HowNet categories. This mapping is clearly noisy. After EM iterations, each English frame is aligned to 5 Chinese categories on average, and each Chinese category is aligned to 1.58 English frames on average.

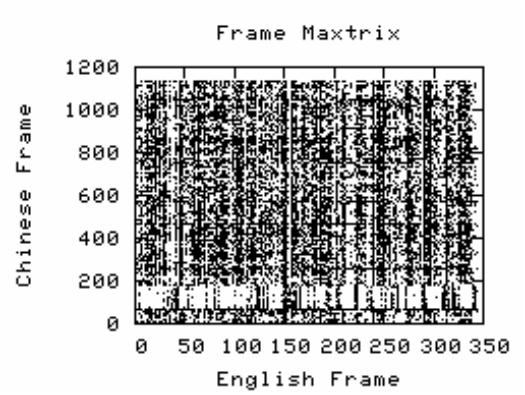


Figure 3a. FrameNet to HowNet mapping before EM iterations.

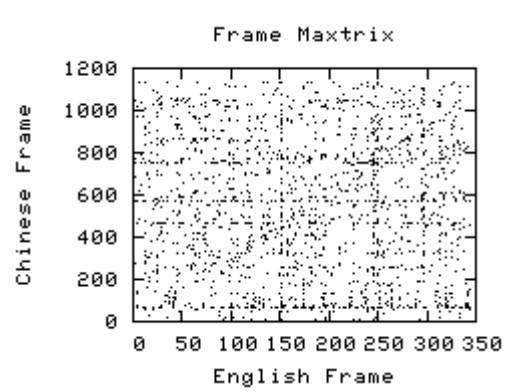


Figure 3b. FrameNet to HowNet mapping after EM iterations.

We also plot the histograms of one-to-X mapping between FrameNet frames and HowNet categories before and after EM iterations in Figure 4. The horizontal axis is the number X in one-to-X mapping between English and Chinese frames. The vertical axis is the occurrence frequency. For example, point (*i*,*j*) represents that there are *j* frames in English mapping to *i* categories in Chinese. Figure 4 shows that using lexical glossing only, there are a large number of frames that are aligned to over 150 of Chinese categories, while only a small number of English frames align to relatively few Chinese categories. After EM iterations, the majority of the English frames align to only a few Chinese categories, significantly improving the frame mapping across the two languages.

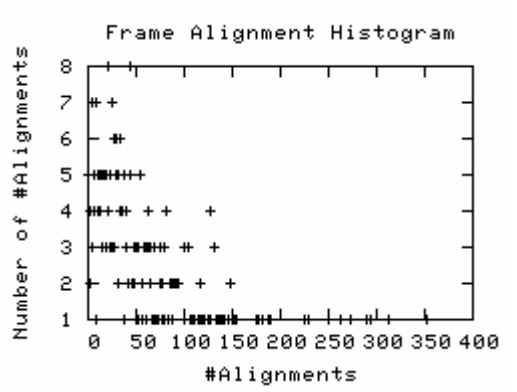


Figure 4a. Histogram of one-to-X mappings between English frames and Chinese categories. Most English frames align to a lot of Chinese categories before EM learning.

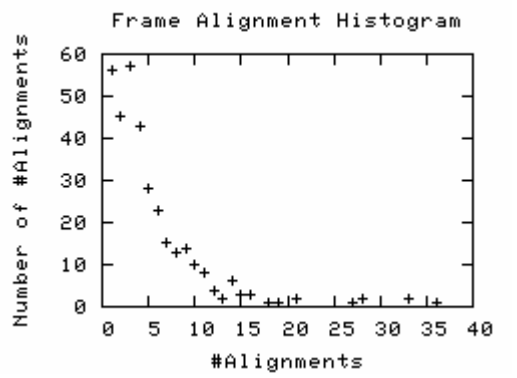


Figure 4b. Histograms of one-to-X mappings between English frames and Chinese categories. Most English frames only align to a few Chinese categories after EM learning.

The above plots demonstrate the difference between FrameNet and HowNet structures. For example, "boy.n" belongs to "*attention_getting*" and "*people*" frames in FrameNet. "boy.n|attention_getting" should translate into "茶房/waiter" in Chinese, whereas "boy.n|people" has the sense of "男孩/male child". However, in HowNet, both "茶房/waiter" and "男孩/male child" belong to the same category, human|人.

```
burn.v,cause_harm --> 辣.v,damage|损害
burn.v,cause_harm --> 烫.v,damage|损害
burn.v,cause_harm --> 嘘.v,damage|损害
burn.v,cause_harm --> 蜇.v,damage|损害
burn.v,experience_bodily_harm --> 烧
伤.v,wounded|受伤
burn.v,heat --> 烧.v,cook|烹调
```

Figure 5. Example word sense translation of the English verb "*burn*" in our bilingual frame semantics mapping.

An example of word sense translation from our algorithm output is shown in Figure 5. The word sense translations of the FrameNet lexical entries represent the simulated semantic world of a bilingual person who uses the same semantic structure but with lexical access in two languages. For example, the frame "*cause_harm*" now contains the bilingual word sense pair "burn.v,cause_harm --> 辣.v,damage|损害 "; and the frame "*experience_bodily_harm*" contains the bilingual word sense pair "burn.v,experience_bodily_harm --> 烧伤.v,wounded|受伤".

4 Robust Word Sense Translation Using Frame Semantics

We evaluate the accuracy of word sense translation in our automatically generated bilingual ontology, by testing on the most ambiguous lexical entries in FrameNet, i.e. words with the highest number of frames. These words and some of their sense translations are shown in Table 1 below.

tie.n,clothing -> 襻.n,part|部件 tie.v,cause_confinement -> 拘束.v,restrain|制止 tie.v,cognitive_connection -> 联结.v,connect|连接
make.n,type -> 性质.n,attribute|属性 make.v,building -> 建造.v,build|建造 make.v,causation -> 令.v,CauseToDo|使动
roll.v,body-movement -> 摇动.v,wave|摆动 roll.v,mass_motion -> 翻滚.v,roll|滚 roll.v,reshaping -> 卷.v,FormChange|形变
feel.n,sensation -> 手感.n,experience|感受 feel.v,perception_active -> 觉得.v,perception|感知 feel.v,seeking -> 摸.v,LookFor|寻

Table 1. Example word sense translation output

The word sense translation accuracies of the above words are shown in Table 2. The results are highly positive given that those from previous work in word translation disambiguation using bootstrapping methods (Li and Li, 2003; Yarowsky 1995) achieved 80-90% accuracy in disambiguating between only two senses per word[1].

The only susceptibility of our algorithm is in its reliance on bilingual dictionaries. The sense translations of the words "tie", "roll", and "look" are relatively less accurate due to the absence of certain translations in the dictionaries we used. For example, the "bread/food" sense of the word "roll" is not found in the bilingual dictionaries at all.

English word	**Number of frames/senses in FrameNet**	**Sense translation accuracy**
tie	8	64%
make	7	100%
roll	6	55%
feel	6	88%
can	5	81%
run	5	100%
shower	5	100%
burn	5	91%
pack	5	85%
drop	5	76%
look	5	64%
Average	**5.6**	**82%**

Table 2. Translation accuracies of the most ambiguous words in FrameNet

We compare our results to that of our previous work (Fung and Chen 2004), by using the same bilingual lexicon. Table 3 shows that we have improved the accuracy of word sense translation using the current method.

lexical entry	Parent frame	Accuracy (Fung & Chen 2004)	**Accuracy (this paper)**
beat.v	cause_harm	88.9%	**100%**
move.v	motion	100%	**100%**
bright.a	light_emission	79.1%	**100%**
hold.v	containing	22.4%	**100%**
fall.v	motion_directional	87%	**100%**
issue.v	emanating	31.1%	**100%**

Table 3. Our method improves word sense translation precision over Fung and Chen (2004).

We note in particular that whereas the previous algorithm in Fung and Chen (2004) does not

[1] We are not able to evaluate our algorithm on the same set of words as in (Li & Li 2003; Yarowsky 1995) since these words do not have entries in FrameNet.

perform well on lexical entries from small frames (e.g. on "hold.v" and "issue.v") due to ad hoc manual thresholding, the current method is fully automatic and therefore more robust. In Fung and Chen (2004), semantic frames are mapped to each other if their lexical entries translate to each other *above a certain threshold*. If the frames are small and therefore do not contain many lexical entries, then these frames might not be correctly mapped. If the parent concept classes are not correctly mapped, then word sense translation accuracy suffers.

The main advantage of our algorithm over our work in 2004 lies in the hill-climbing iterations of the EM algorithm. In the proposed algorithm, all concept classes are mapped *with a certain probability*, so no mapping is filtered out prematurely. As the algorithm iterates, it is more probable for the correct bilingual word sense to be translated to each other, and it is also more probable for the bilingual concept classes to be mapped to each other. After convergence of the algorithm, the output probabilities are optimal and the translation results are more accurate.

5 Towards Translation Disambiguation using Frame Semantics

As translation disambiguation forms the core of various machine translation strategies, we are interested in studying whether the generated bilingual frame semantics can complement existing resources, such as bilingual dictionaries, for translation disambiguation.

The semantic frame of the predicate verb and the argument structures in a sentence can be identified by the syntactic structure, part-of-speech tags, head word, and other features in the sentence. The predicate verb translation corresponds to the word sense translation we described in the previous sections, *Pr(cl,cf | el,ef)*.

We intend to evaluate the effectiveness of bilingual frame semantics mapping in disambiguating between translation candidates. For the evaluation set, we use 202 randomly selected example sentences from FrameNet, which have been annotated with predicate-argument structures.

In the first step of the experiment, for each predicate word *(el,ef)*, we find all its translation candidates of the predicate word in each sentence, and annotate them with their HowNet categories to form a translated word sense *Pr(cl,cf|el,ef)*. For the example sentence in Figure 6, there are altogether 147 word sense translations for *(hold,detaining)*.

Under South African law police could HOLD the man for questioning for up to 48 hours before seeking the permission of magistrates for an extension
##HOLD,detaining
召开,engage|从事--
牵,guide|引导--
以为,regard|认为--
束缚,restrain|制止--
装,load|装载 --
装,pretend|假装--
手持,hold|拿--
…
拿,hold|拿 --
拿,occupy|占领 --
握住,hold|拿 --
占,occupy|占领 --
…
持,hold|拿 --
握,hold|拿 --
…
操,hold|拿 --
操,speak|说 --
承,KeepOn|使继续 --
…
开,function|活动 --
开,manage|管理 --
 # 扣留,detain|扣住 ++
要塞,facilities|设施 --
拥有,own|有 –

Figure 6. A FrameNet example sentence and predicate verb translations {Pr(cl,cf|el,ef)}.

We then find the word sense translation with the highest probability among all HowNet and FrameNet class mappings from our EM algorithm:

$$cl^* = \arg\max_{cl} \Pr(cl, cf \mid el, ef)$$
$$= \arg\max_{cl} \frac{\Pr(cl \mid el) \cdot \Pr(cf \mid ef)}{\sum_{\forall cf} \Pr(cl, cf \mid el, ef)}$$

An example *(el,ef)* is (*hold, detaining)* and the *cl*=argmax P(cl,cf|el,ef)* found by our program is *扣留*. *(cl,cf)** in this case is *(扣留,detain|扣住)*.

Human evaluators then look at the set of *{cl*}* and mark *cl** as either true translations or erroneous. The accuracy of word sense translations on this evaluation set of example sentences is at 74.9%.

In comparison, we also look at *Pr(cl|el)*, translation based on bilingual dictionary only, and find

$$cl^* = \arg\max_{cl} \Pr(cl \mid el) = \arg\max_{cl} \Pr(cl, el)$$

The translation accuracy of using bilingual dictionary only, is at a predictable low 15.8%.

Our results are the first significant evidence of, in addition to bilingual dictionaries, bilingual frame semantics is a useful resource for the translation disambiguation task.

6 Related Work

The most relevant previous works include word sense translation and translation disambiguation (Li & Li 2003; Cao & Li 2002; Koehn and Knight 2000; Kikui 1999; Fung et al., 1999), frame semantic induction (Green et al., 2004; Fung & Chen 2004), and bilingual semantic mapping (Fung & Chen 2004; Huang et al. 2004; Ploux & Ji, 2003, Ngai et al., 2002; Palmer & Wu 1995). Other than the English FrameNet (Baker et al, 1998), we also note the construction of the Spanish FrameNet (Subirats & Petruck, 2003), the Japanese FrameNet (Ikeda 1998), and the German FrameNet (Boas, 2002). In terms of learning method, Chen and Palmer (2004) also used EM learning to cluster Chinese verb senses.

Word Sense Translation

Previous word sense translation methods are based on using context information to improve translation. These methods look at the context words and discourse surrounding the source word and use methods ranging from boostrapping (Li & Li 2003), EM iterations (Cao and Li, 2002; Koehn and Knight 2000), and the cohesive relation between the source sentence and translation candidates (Fung et al. 1999; Kikui 1999).

Our proposed translation disambiguation method compares favorably to (Li & Li 2003) in that we obtain an average of 82% precision on words with multiple senses, whereas they obtained precisions of 80-90% on words with two senses. Our results also compare favorably to (Fung et al. 1999; Kikui 1999) as the precision of our predicate verb in the input sentence translation disambiguation is about 75% whereas their precisions range from 40% to 80%, albeit on an independent set of words.

Automatic Generation of Frame Semantics

Green et al. (2004) induced SemFrame automatically and compared it favorably to the hand-constructed FrameNet (83.2% precision in covering the FrameNet frames). They map WordNet and LDOCE, two semantic resources, to obtain SemFrame. Burchardt et al. (2005) used FrameNet in combination with WordNet to extend coverage.

Bilingual Semantic Mapping

Ploux and Ji, (2003) proposed a spatial model for matching semantic values between French and English. Palmer and Wu (1995) studied the mapping of *change-of-state* English verbs to Chinese. Dorr et al. (2002) described a technique for the construction of a Chinese-English verb lexicon based on HowNet and the English LCS Verb Database (LVD). They created links between HowNet concepts and LVD verb classes using both statistics and a manually constructed "seed mapping" of thematic classes between HowNet and LVD. Ngai et al. (2002) induced bilingual semantic network from WordNet and HowNet. They used lexical neighborhood information in a word-vector based approach to create the alignment between WordNet and HowNet classes without any manual annotation.

7 Conclusion

Based on the one-frame-two-languages idea, which stems from the hypothesis of the mind of a bilingual speaker, we propose automatically generating a bilingual word sense dictionary or ontology. The bilingual ontology is generated from iteratively estimating and maximizing the probability of a word translation given frame mapping, and that of frame mapping given word translations. We have shown that for the most ambiguous 11 words in the English FrameNet, the average word sense translation accuracy is 82%. Applying the bilingual ontology mapping to translation disambiguation of predicate verbs in another evaluation, the accuracy of our method is at an encouraging 75%, significantly better than the 15% accuracy of using bilingual dictionary only. Most importantly, we have demonstrated that bilingual frame semantics is potentially useful for cross-lingual retrieval, machine-aided and machine translation.

8 Future Work

Our evaluation exercise has shown the promise of using bilingual frame semantics for translation task. We are currently carrying out further work in the aspects of (1) improving the accuracy of source word frame identification and (2) incorporating bilingual frame semantics in a full fledged

machine translation system. In addition, FrameNet has a relatively poor coverage of lexical entries. It would be necessary to apply either semi-automatic or automatic methods such as those in (Burchardt et al. 2005, Green et al 2004) to extend FrameNet coverage for final application to machine translation tasks. Last but not the least, we are interested in applying our method to other ontologies such as the one used for the Propbank data, as well as to other language pairs.

Acknowledgement

This work was partially supported by CERG# HKUST6206/03E and CERG# HKUST6213/02E of the Hong Kong Research Grants Council. We thank Yang, Yongsheng for his help in the final draft of the paper, and the anonymous reviewers for their useful comments.

References

Collin F. Baker, Charles J. Fillmore and John B. Lowe. (1998).The Berkeley FrameNet project. In *Proceedings of the COLING-ACL*, Montreal, Canada.

Hans C. Boas. (2002). Bilingual FrameNet Dictionaries for Machine Translation. In *Proceedings of the Third International Conference on Language Resources and Evaluation*. Las Palmas, Spain. Vol. IV: 1364-1371 2002.

A. Burchardt, K. Erk, A. Frank. (2005). A WordNet Detour to FrameNet. In *Proceedings of the 2nd GermaNet Workshop*, 2005.

Cao, Yunbo and Hang Li. (2002). Base Noun Phrase Translation Using Web Data and the EM Algorithm. In COLING 2002. Taipei, 2002.

Jinying Chen and Martha Palmer. (2004). Chinese Verb Sense Discrimination Using EM Clustering Model with Rich Linguistic Features. In ACL 2004. Barcelona, Spain, 2004.

Dong, Zhendong., and Dong, Qiang. (2002) HowNet [online]. Available at http://www.keenage.com/zhiwang/e_zhiwang.html

Bonnie J. Dorr, Gina-Anne Levow, and Dekang Lin.(2002).Construction of a Chinese-English Verb Lexicon for Machine Translation. In *Machine Translation, Special Issue on Embedded MT*, 17:1-2.

Pascale Fung and Benfeng Chen. (2004). BiFrameNet: Bilingual Frame Semantics Resource Construction by Cross-lingual Induction. In *COLING 2004*. Geneva, Switzerland. August 2004.

Pascale Fung, Liu, Xiaohu, and Cheung, Chi Shun. (1999). Mixed-Language Query Disambiguation. In *Proceedings of ACL '99*, Maryland: June 1999.

Daniel Gildea and Daniel Jurafsky.(2002).Automatic Labeling of Semantic Roles. In *Computational Linguistics*, Vol 28.3: 245-288.

Rebecca Green, Bonnie Dorr, Philip Resnik. (2004). Inducing Frame Semantic Verb Classes from WordNet and LDOCE. In *ACL 2004*.

François Grosjean in Lesley Milroy and Pieter Muysken (editors). One Speaker, Two Languages. *Cambridge University Press*, 1995.

Chu-Ren Huang, Ru-Yng Chang, Hsiang-Pin Lee. (2004).Sinica BOW (Bilingual Ontological Wordnet): Integration of Bilingual WordNet and SUMO. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, (2004).

Judy Illes and Wendy S. Francis. (1999). Convergent cortical representation of semantic processing in bilinguals. In *Brain and Language*, 70(3):347-363, 1999.

Satoko Ikeda. (1998). Manual response set in a stroop-like task involving categorization of English and Japanese words indicates a common semantic representation. In *Perceptual and Motor Skills*, 87(2):467-474, 1998.

Liu Qun and Li, Sujian. (2002). Word Similarity Computing Based on How-net. In *Computational Linguistics and Chinese Language Processing*, Vol.7, No.2, August 2002, pp.59-76

Philipp Koehn and Kevin Knight. (2000). Estimating Word Translation Probabilities from Unrelated Monolingual Corpora Using the EM Algorithm. In *AAAI/IAAI 2000*: 711-715

Grace Ngai, Marine Carpuat, and Pascale Fung. (2002). Identifying Concepts Across Languages: A First Step towards a Corpus-based Approach to Automatic Ontology Alignment. In *Proceedings of COLING-02*, Taipei, Taiwan.

Franz Och et al. (2003). http://www.clsp.jhu.edu/ws2003/groups/translate/

Martha Palmer and Wu Zhibiao**.** (1995).Verb Semantics for English-Chinese Translation. In *Machine Translation* 10: 59-92, 1995.

Sabine Ploux and Hyungsuk Ji. (2003). A Model for Matching Semantic Maps between Languages (French/English, English/French). In *Computational Linguistics* 29(2):155-178, 2003.

Carlos Subitrats and Miriam Petriuck. (2003). Suprirse: Spanish FrameNet. *Workshop on Frame Semantics, International Congress of Linguists*, July 2003.

Coreference handling in XMG

Claire Gardent
CNRS/LORIA
615, rue du jardin botanique, B.P. 101
54602 Villers lès Nancy CEDEX
France
Claire.Gardent@loria.fr

Yannick Parmentier
INRIA Lorraine
615, rue du jardin botanique, B.P. 101
54602 Villers lès Nancy CEDEX
France
Yannick.Parmentier@loria.fr

Abstract

We claim that existing specification languages for tree based grammars fail to adequately support identifier managment. We then show that XMG (*eXtensible MetaGrammar*) provides a sophisticated treatment of identifiers which is effective in supporting a linguist-friendly grammar design.

1 Specifying tree-based grammars

Whilst the development of standard unification-based grammars is well supported by the design of formalisms such as PATR-II, Ale or TDL (Krieger and Schafer, 1994), the situation is less well established for Tree-Based Grammars such as Tree Adjoining Grammars (Joshi and Schabes, 1997), Tree Description Grammars (Kallmeyer, 1996) or Interaction Grammars (Perrier, 2003).

Roughly, two main types of specification formalism for Tree-Based Grammars can be distinguished: formalisms based on tree fragments and non monotonic inheritance and formalisms based on tree descriptions and monotonic inheritance.

The tree fragment approach is advocated in (Evans et al., 1995) which proposes to encode lexicalised TAGs using the DATR representation language[1]. In this approach, tree fragments are combined within a non monotonic inheritance hierarchy. Furthermore, new fragments can be derived from existing ones by means of lexical rules. This first approach suffers from the procedural character of non-monotonic inheritance. In specifying the grammar, the grammar writer must keep in mind the order in which non-monotonic statements have been made so as to be able to predict how explicit statements interact with defaults and non-monotonic inheritance in determining the final output. When developing a large coverage grammar, this rapidly become extremely cumbersome. Moreover, as (Candito, 1996) remarks, non-monotonicity may result in an information loss which makes it impossible to express the relation existing for instance between an active object and the corresponding passive subject.

The approach based on tree descriptions (often called, the *metagrammar approach*) obviates the procedural character of the non-monotonic approach by taking tree descriptions rather than trees to be the basic units (Candito, 1996; Xia et al., 1999; Vijay-Shanker and Schabes, 1992). In essence, tree fragments are described using tree descriptions and tree descriptions are combined through conjunction or inheritance. The idea is that the minimal models satisfying the resulting descriptions are TAG elementary trees. In some cases, lexical rules are also used to derive new trees from existing ones.

One main drawback with this second type of approach concerns the management of node identifiers. Either nodes are represented by nameless variables and node identification is forced by well-formedness constraints e.g., *wff*-constraints on trees and *wff*-constraints given by the input tree description (cf. e.g., (Duchier and Gardent, 1999)) or nodes are named and nodes with identical names are forced to denote the same entity. The first option is unrealistic when developing a large core grammar as it is easy to omit a necessary constraint and thereby permit overgeneration (the description will be satisfied by more trees than intended). The second option greatly degrades

[1] A tree based approach is also used in(Becker, 2000) but this time in combination with metarules. In that particular approach, procedural aspects also come into play as the order in which metarules apply affect the results.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 247–254,
Sydney, July 2006.

modularity as the grammar writer must remember which names were used where and with which interpretation. As we shall see below, it also has the undesirable effect that the same tree fragment cannot be used twice in a given tree description. Nevertheless, this is the option that is adopted in most grammar formalisms and grammar compilers (Candito, 1996; Xia et al., 1999; Gaiffe et al., 2002).

In this paper, we present an approach which remedies these shortcomings by combining monotonic inheritance of tree descriptions with an *explicit management of identifier scope and identifiers equality*[2]. The proposed approach thus eschews both the inconvenients induced by a non monotonic framework (by using tree descriptions rather than trees) and those resulting from a global treatment of identifiers (by providing greater expressivity wrt identifiers).

Specifically, we show that the proposed approach supports several ways of identifying (node or feature) values, we motivate this multiplicity and we identify the linguistic and/or technical criteria for choosing among the various possibilities.

The paper starts in section 2 by introducing the syntax of the XMG formalism. In section 3, we show that XMG provides four different ways of identifying two (node or variable) identifiers. In section 4, we motivate each of these four different ways and indicate when each of them can and should be used.

2 The XMG formalism

We start by briefly introducing XMG (eXtended MetaGrammar). First, we show that it supports the description and the combination of *blocks* consisting of tree fragments and/or semantic representations. Then, we show that it supports a sophisticated treatment of identifiers.

2.1 Defining blocks

At the syntactic level, the basic units are tree descriptions which are specified using the following tree logic:

$$\begin{aligned} \textit{Description} ::= \; & x \rightarrow y \mid x \rightarrow^{+} y \mid x \rightarrow^{*} y \mid \\ & x \prec y \mid x \prec^{+} y \mid x \prec^{*} y \mid \\ & x[f{:}E] \mid x = y \mid \\ & \textit{Description} \wedge \textit{Description} \end{aligned} \tag{1}$$

where x, y represent node variables, $\rightarrow$ immediate dominance (x is directly above y), $\rightarrow^{+}$ strict dominance (x is above y), and $\rightarrow^{*}$ large dominance[3] (x is above or equal to y). Similarly $\prec$ denotes immediate precedence, $\prec^{+}$ strict precedence, and $\prec^{*}$ large precedence. Finally $x[f{:}E]$ constrains feature f with associated expression E on node x, and $x = y$ indicates node identification.

The XMG formalism also supports the association of semantic representations with elementary trees. The semantic representation language is a flat semantic representation language (Bos, 1995) with the following syntax:

$$\begin{aligned} \textit{Description} ::= \; & \ell{:}p(E_1, ..., E_n) \mid \\ & \neg\ell{:}p(E_1, ..., E_n) \mid E_i \ll E_j \\ & \textit{Description} \wedge \textit{Description} \end{aligned} \tag{2}$$

where ℓ is a label, p is a predicate and $E_1, .., E_n$ are parameters. Further, $\neg$ denotes negation and $E_i \ll E_j$ expresses a scope constraint between E_i and E_j (E_j is in the scope of E_i).

2.2 Combining blocks

As in other existing tree-based formalisms, in XMG, blocks can be combined using inheritance. However, XMG additionally supports block conjunction and block disjunction.

Specifically, a *Class* associates a name with a content:

$$\textit{Class} \quad ::= \quad \textit{Name} \rightarrow \{\, \textit{Content} \,\} \tag{3}$$

A *Content* is either a *Description* (i.e., a tree description, a semantic formula or both), a class name, a conjunction or a disjunction of class name:

$$\begin{aligned} \textit{Content} \quad ::= \quad & \textit{Description} \mid \textit{Name} \mid \\ & \textit{Name} \vee \textit{Name} \mid \textit{Name} \wedge \textit{Name} \end{aligned} \tag{4}$$

Further, XMG allows *multiple* inheritance: a given class can *import* or inherit one or more classes (written C_i here):

[2] Recently, (Villemonte de la Clergerie, 2005) has proposed a highly compact representation formalism for tree-based grammars which also features explicit identifier management. His approach differs from ours in that it includes neither a colouring mechanism (cf. section 3.4) nor interfaces (cf. section 3.3).

[3] By large, we mean the transitive reflexive closure of dominance.

$$\textit{Class} \quad ::= \quad \textit{Name} \angle C_1 \wedge \ldots \wedge C_n \rightarrow \{\, \textit{Content} \,\} \qquad (5)$$

The semantic of the *import* instruction is to include the description of the *imported* class within the current one. This makes it possible to refine a class e.g., by adding information to a node or by adding new nodes[4].

2.3 Managing identifiers

We now introduce the treatment of identifiers supported by XMG. We show in particular, that it integrates:

- a convenient way of managing identifier scope based on *import/export declarations* inspired from standard Object Oriented Programming techniques (section 2.3.1);
- an alternative means of identifying feature values based on the use of *unification*
- *polarity*- (here called *colour*-) based node identification as first proposed in (Muskens and Krahmer, 1998) and later used in (Duchier and Thater, 1999; Perrier, 2000).

The next sections will detail the linguistic and technical motivations behind this variety of identifier handling techniques.

2.3.1 Import/Export declaration

In XMG, the default scope of an identifier is the class in which it is declared. However, *export* specifications can be used to extend the scope of a given identifier outside its declaration class. The export of identifier ?X ouside class A is written :[5]

$$\mathtt{A}_{?\mathtt{X}} \quad \rightarrow \quad \{\ldots ?\mathtt{X} \ldots\}$$

Export declarations interact with inheritance, conjunction and disjunction specifications as follows (where A, B, C are classes):

Inheritance: if the class A is *imported* either directly or indirectly by a class B, then ?X is visible in B. In case of multiple inheritance e.g., if $\mathtt{B} \angle C_1 \wedge \ldots \wedge C_n$, then all identifiers exported by $C_1 \wedge \ldots \wedge C_n$ are visible from B *provided they have distinct names*. In other words, if two (or more) classes in $C_1 \wedge \ldots \wedge C_n$ export the *same* identifier ?X, then ?X is not directly visible from B. It can be accessed though using the dot operator. First A is identified with a local identifier (e.g., ?T = A), then ?T.?X can be used to refer to the identifier ?X exported by A.

Conjunction: if classes A and B are *conjoined* inside a class C, then all the identifiers exported by A or B are visible within C using the *dot* operator.

Disjunction: if classes A and B are *disjoined* inside a class C, then all the identifiers exported by A or B are visible within C using the *dot* operator. However in this case, both A and B have to be associated with the same local identifier.

In sum, export/import declarations permit extending/restricting the scope of an identifier within a branch of the inheritance hierarchy whilst the dot operator allows explicit access to an inherited identifier in case the inheriting class either displays multiple inheritance or is combined by conjunction or disjunction with other classes. More specifically, inheritance allows implicit coreference (the use of an imported name ensures coreference with the object referred to when declaring this name) and the dot operator explicit coreference (through an explicit equality statement e.g., ?A.?X = ?B.?Y).

2.3.2 Class interface

In XMG, a class can be associated with a **class interface** i.e., with a feature structure. Furthermore, when two classes are related either by inheritance or by combination (conjunction or disjunction), their interfaces are *unified*. Hence class interfaces can be used to ensure the unification of identifiers across classes.

Here is an illustrating example:

$$\begin{array}{lcl} \mathtt{A} & \rightarrow & \{\ldots ?\mathtt{X} \ldots\} * = [n1 = ?\mathtt{X}] \\ \mathtt{B} & \rightarrow & \{\ldots ?\mathtt{Y} \ldots\} * = [n1 = ?\mathtt{Y}] \end{array}$$

In A (resp. B), the *local* identifier ?X (resp. ?Y) is associated with an interface feature named $n1$. If

[4] Note that disjunctive inheritance is not supported which would allow a block to be defined as importing one or more classes from a given set of imported classes

[5] Similarly, *import* declaration can be used to restrict the set of accessible identifiers to a subset of it.

these two classes are combined either by conjunction or by inheritance, their interfaces are unified and as a result, the local identifiers ?X and ?Y are unified. In the case of a disjunction, the interface of the current class (C here) is non deterministically unified with that of A or B.

In practice, interface-based identification of values is particularly useful when two distinct features need to be assigned the same value. In (Gardent, 2006) for instance, it is used to identify the semantic index associated with e.g., the subject node of a verbal tree and the corresponding semantic index in the semantic representation associated with that tree.

2.3.3 Colouring nodes

Finally, XMG provides a very economical way of identifying node variables based on the use of *colours* (also called *polarities* in the literature). The idea is that node variables are associated with a specific colour and that this colouring will either prevent or trigger node identifications based on the following identification rules:

	$\bullet_B$	$\bullet_R$	$\circ_W$	$\bot$
$\bullet_B$	$\bot$	$\bot$	$\bullet_B$	$\bot$
$\bullet_R$	$\bot$	$\bot$	$\bot$	$\bot$
$\circ_W$	$\bullet_B$	$\bot$	$\circ_W$	$\bot$
$\bot$	$\bot$	$\bot$	$\bot$	$\bot$

and on the requirement that valid trees only have red or black nodes. In effect, node colouring enforces the following constraints : (i) a white node must be identified with a black node, (ii) a red node cannot be identified with any other node and (iii) a black node may be identified with one or more white nodes.

Contrary to other means of value identification, colours are restricted to node identifiers. Hence they are best used to induce node identification in those contexts where neither inheritance nor explicit identification are appropriate (see section 4).

3 XMG at work

Recall (section 1) that one main problem when developing a factorised specification of tree based grammars is to ensure a consistent treatment of identifiers and in particular, of identifier unification. That is, when combining two units of information, the grammar writer must ensure that her specification correctly states which objects are the same and which are distinct.

In what follows, we show that XMG supports four different ways of identifying objects. We illustrate this by demonstrating that the following tree can be obtained in four different ways:

Figure 1: A tree that can be derived in four ways

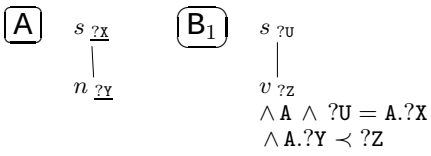
In section 4, we will show that these four ways of identifying nodes and/or features values support both explicitness and economy thereby reducing the risks of specification errors.

3.1 Using explicit identification

The most basic way to identify two identifiers is to explicitly state their identity. Thus the above tree can be produced by combining the following two classes[6] :

$$\begin{aligned} \mathtt{A}_{?X,?Y} \rightarrow \quad & \{\, ?\mathtt{X}\,[cat : s] \rightarrow ?\mathtt{Y}\,[cat : n]\,\} \\ \mathtt{B}_1 \rightarrow \quad & \{\, ?\mathtt{U}\,[cat : s] \rightarrow ?\mathtt{Z}\,[cat : v] \\ & \wedge\ \mathtt{A} \wedge\ ?\mathtt{U} = \mathtt{A}.?\mathtt{X}\ \wedge\ \mathtt{A}.?\mathtt{Y} \prec ?\mathtt{Z}\,\} \end{aligned}$$

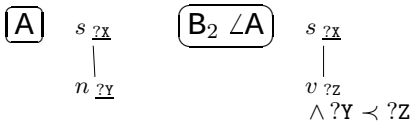
To improve readability, we use from now on a graphical representation. For instance, the classes above are represented as follows (exported identifiers are underlined and boxed letters indicate class names):



Thus, the class A describes the left branch of the tree in Figure 1 and the class $\mathtt{B}_1$ its right branch. The root of A and B are named ?X and ?U respectively. Since ?X is exported, ?X is visible in $\mathtt{B}_1$. The explicit identification ?U=A.?X then enforces that the two roots are identified thus constraining the solution to be the tree given in Figure 1.

3.2 Using inheritance

Using inheritance instead of conjunction, the same nodes identification can be obtained in a more economical way. We reuse the same class A as before, but we now define a class $\mathtt{B}_2$ as a sub-class of A:



[6]Here and in what follows, we abbreviate the conjunction of a class identifi cation ?T = A and a dot notation T.?X to A.?X. That is,

$$?\mathtt{T} = \mathtt{A} \wedge \mathtt{T}.?\mathtt{X} \rightarrow_{abbrev} \mathtt{A}.?\mathtt{X}$$

Since the identifiers ?X and ?Y are exported by A, they are visible in B_2. Thus, in the latter we only have to indicate the precedence relation between ?Y and ?Z.

In sum, the main difference between explicit identification and identification through simple exports, is that whilst inheritance of exported identifiers gives direct access to these identifiers, class combination requires the use of a prefix and dot statement. Note nevertheless that with the latter, identifiers conflicts are a lot less likely to appear.

3.3 Using interfaces

A third possibility is to use interfaces to force node identifications as illustrated in figure 2.

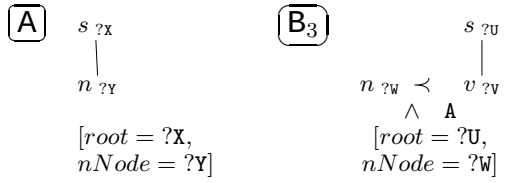
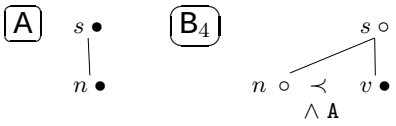


Figure 2: Structure sharing using interfaces

Class A is the same as before except that the identifiers ?X and ?Y are no longer exported. Instead they are associated with the interface features $root$ and $nNode$ respectively. Similarly, class B_3 associates the identifiers (?U and ?V) with the interface features $root$ and $nNode$. As the tree fragment of class B_3 is conjoined with A, the interface features of A and B_3 are unified so that ?X is identified with ?U and ?Y with ?V.

3.4 Using node colours

Finally, colours can be used as illustrated in the Figure below:



Now, class B_4 contains three nodes: two white ones whose categories are s and n and which must be identified with compatible black nodes in A; and a black node that may but need not be identified with a white one. To satisfy these constraints, the black s node in A must be identified with the white s node in B and similarly for the n nodes. The result is again the tree given in Figure 1.

Note that in this case, none of the identifiers need to be exported. Importantly, the use of colours supports a very economical way of forcing nodes identification. Indeed, nodes that are identified through colouration need neither be exported nor even be named.

4 Which choice when?

As shown in the previous section, XMG allows four ways of identifying values (i.e., nodes or feature values): through simple exports, through explicit identification, through colour constraints and through the interface. We now identify when each of these four possibilities is best used.

4.1 Exports

As shown in section 2.3, an identifier ?X can be explicitly exported by a class C with the effect that, within all classes that inherit from C, all occurrences of ?X denote the same object.

In essence, exports supports variable naming that is global to a branch of the inheritance hierarchy. It is possible to name an identifier within a given class C and to reuse it within any other class that inherits from C. Thus the empirical difficulty associated with the use of exported identifiers is that associated with global names. That is, the grammar writer must remember the names used and their intended interpretation. When developing a large size grammar, this rapidly makes grammar writing, maintenance and debugging an extremely difficult task. Hence global identifiers should be use sparingly.

But although non trivial (this was in fact one of the main motivations for developing XMG), this empirical limitation is not the only one. There are two additional formal restrictions which prevent a general use of exported identifiers.

First, as remarked upon in (Crabbe and Duchier, 2004), global names do not support multiple use of the same class within a class. For instance, consider the case illustrated in Figure 3.

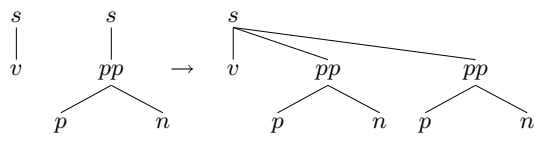


Figure 3: Case of double prepositional phrase.

In this case, the aim is to produce the elementary tree for a verb taking two prepositional arguments such as *parler à quelqu'un de quelque chose* (*to tell someone about something*). Ideally, this is done by combining the verbal fragment on the left

with two occurrences of the PP class in the middle to yield the tree on the right. However if, as is likely in a large size metagrammar, any of the pp, the p or the n node bears an exported identifier, then the two occurrences of this node will be identified so that the resulting tree will be that given in (4).

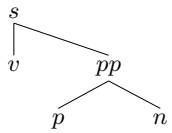


Figure 4: Double prepositional phrase with exported identifiers.

We will see below how colours permit a natural account of such cases.

Second, exported modifiers do not support identifier unification in cases of conjunction, disjunction and multiple inheritance. That is, in each of the three cases below, the various occurrences of ?X are not identified.

$$\mathsf{C}_1\ ?\mathtt{X} \wedge \mathsf{C}_2\ ?\mathtt{X}$$
$$\mathsf{C}_1\ ?\mathtt{X} \vee \mathsf{C}_2\ ?\mathtt{X}$$
$$\mathsf{C}_3\ ?\mathtt{X} \measuredangle \mathsf{C}_1\ ?\mathtt{X} \wedge \mathsf{C}_2\ ?\mathtt{X}$$

In such cases, the multiple occurrences of ?X need to be explicitly identified (see below).

In practice then, the safest use of simple exports (ie without explicit identifier equalities) consists in using them

- in combination with inheritance only and
- within a linguistically motivated subpart of the inheritance hierarchy

4.2 Colours

As discussed in section 2.3, node identifications can be based on colours. In particular, if a node is white, it *must* be identified with a black node.

The main advantage of this particular identification mechanism is that it is extremely economical. Not only is there no longer any need to remember names, there is in fact no need to chose a name. When developing a metagrammar containing several hundreds of nodes, this is a welcome feature.

This "no-name" aspect of the colour mechanism is in particular very useful when a given class needs to be combined with many other classes. For instance, in SEMFRAG (Gardent, 2006), the semantic index of a semantic functor (*i.e.*, a verb, an adjective, a preposition or a predicative noun) needs to be projected from the anchor to the root node as illustrated in Figure 5. This can be done, as shown in the figure by conjoining C_{Sem} with C_V or C_A and letting the colour unify the appropriate nodes.

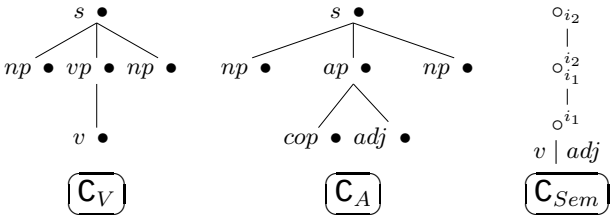


Figure 5: Case of semantic projections.

Colouring also solves the problem raised by the multiple reuse of the same class in the definition of a given class. The colouring will be as shown in Figure 6. Since the pp, p and n nodes are black, their two occurrences cannot be identified. The two white s nodes however will both be unified with the black one thus yielding the expected tree.

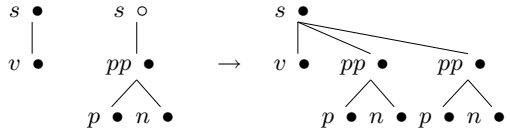


Figure 6: Case of double prepositional phrase with coloured descriptions.

As for exports however, colours cannot always be used to force identifications.

First, colours can only be used in combination with conjunction or inheritance of non exported identifiers. Indeed, inheritance does not allow the identification of two different objects. Hence if a class C containing a white node named ?X inherits from another class C' exporting a black node also named ?X, compilation will fail as a given identifier can only have one colour[7]. In contrast, when solving a description containing the conjunction of a black and a white node (where these two nodes have either no names or distinct names), the well formedness constraint on coloured tree will ensure that these two nodes are in fact the same (since a tree containing a white node is ill formed).

Second, colour based identification is non deterministic. For instance, in Figure 5, if the lowest

[7]However, different occurrences of the same unnamed node can have distinct colours.

node b of $\mathtt{C}_{Sem}$ was not labelled $cat = v \mid adj$, $\mathtt{C}_A \wedge \mathtt{C}_{Sem}$ would yield not one but two trees: one where b is identified with the cop node and the other where it is identified with the adj one. In other words, colour based unification is only possible in cases where node decorations (or explicit node identifications) are sufficiently rich to constrain the possible unifications.

To sum up, colours are useful in situations where:

- a given class needs to be combined with many other classes – in this case it is unlikely that the names used in all classes to be combined are consistent (ie that they are the same for information that must be unified and that they are different for information that must not) and
- the nodes to be identified are unambiguous (the white and the black nodes contain enough information so that it is clear which white node must be identified with which black one)

4.3 Interfaces

Interfaces provide another mechanism for global naming. They are particularly useful in cases where two fundamentally different objects contain non-node identifiers that must be unified.

Recall (cf. section 4.2) that exported identifiers are best used within restricted, linguistically well defined hierarchies. In a case where the objects containing the two identifiers to be identified are different, these will belong to distinct part of the inheritance hierarchy hence identifier export is not a good option.

Node colouring is another possibility but as the name indicates, it only works for *nodes*, not for feature values.

In such a situation then, interfaces come in handy. This is the case for instance, when combining a semantic representation with a tree. The semantic formula and the tree are distinct objects but in the approach to semantic construction described in (Gardent and Kallmeyer, 2003), they share some semantic indices. For instance, the subject node in the tree is labelled with an index feature whose value must be (in an active form tree) that of the first argument occurring in the semantic representation. The encoding of the required coreference can be sketched as follows:

$$\begin{aligned}
\mathtt{Subj} &\rightarrow \{\ldots \mathtt{?X} \ldots\}* = [subjectIdx = \mathtt{?X}] \\
\mathtt{Sem} &\rightarrow \{\ldots \mathtt{?Y} \ldots\}* = [arg1 = \mathtt{?Y}] \\
\mathtt{Tree} &\rightarrow \mathtt{Subj}* = [subjectIdx = \mathtt{?Z}] \wedge \\
&\qquad \mathtt{Sem}* = [arg1 = \mathtt{?Z}]
\end{aligned}$$

The first two lines show the naming of the identifiers ?X and ?Y through the interface, the third illustrates how unification can be used to identify the values named by the interface: since the same variable ?Z is the value of the two features $arg1$ and $subjectIdx$, the corresponding values in the Subj and Sem classes are identified.

4.4 Explicit identification of exported identifiers

The explicit identification of exported identifiers is the last resort solution. It is not subject to any of the restrictions listed above and can be combined with conjunction, disjunction and inheritance. It is however uneconomical and complexifies grammar writing (since every node identification must be explicitly declared). Hence it should be used as little as possible.

In practice, explicit identification of exported identifiers is useful :

- to further constrain colour based identification (when the feature information present in the nodes does not suffice to force identification of the appropriate nodes)
- to model general principles that apply to several subtrees in a given hierarchy

The second point is illustrated by Subject/Verb agreement. Suppose that in the metagrammar, we want to have a separate class to enforce this agreement. This class consists of a subject node ?SubjAgr bearing agreement feature ?X and of a verb node ?VerbAgr bearing the same agreement feature. It must then be combined with all verbal elementary trees described by the metagrammar whereby in each such combination the nodes ?SubjAgr, ?VerbAgr must be identified with the subject and the verb node respectively. This is a typical case of multiple inheritance because both the subject and the verb nodes are specified by inheritance and ?SubjAgr, ?VerbAgr must be further inherited. Since nodes must be identified and multiple inheritance occur, simple identifier exports cannot be used (cf. section 2.3.1). If colours cannot be sufficiently

	Pros	Cons	Practice
Export	Economy	Name management Not with multiple inheritance Not with conjunction Not with disjunction Not with multiple reuse	Use in linguistically motivated sub-hierarchy
Colours	Economy ++ Multiple reuse OK	Non deterministic Not with inheritance and identically named identifi ers	Use when a given class combines with many classes
Interface	Global	Name management	Use for Syntax/Semantic interface
Explicit identifi cation	Usable in all cases	Uneconomical	Last Resort solution

Figure 7: Summary of the pros and cons of sharing mechanisms.

constrained by features, then the only solution left is explicit node identification.

Figure 7 summarises the pros and the cons of each approach.

5 Conclusion

In this paper, we have introduced a specification formalism for Tree-Based Grammars and shown that its expressivity helps solving specification problems which might be encountered when developping a large scale tree-based grammar.

This formalism has been implemented within the XMG system and successfully used to encode both a core TAG for French (Crabbe, 2005; Gardent, 2006) and a core Interaction Grammar (Perrier, 2003). We are currently exploring ways in which the XMG formalism could be extended to automatically enforce linguistically-based well-formedness principles such as for instance, a kind of Head Feature Principle for TAG.

References

T. Becker. 2000. Patterns in metarules. In A. Abeille and O. Rambow, editors, *Tree Adjoining Grammars: formal, computational and linguistic aspects*. CSLI publications, Stanford.

J. Bos. 1995. Predicate Logic Unplugged. In *Proceedings of the 10th Amsterdam Colloquium, Amsterdam*.

M.H. Candito. 1996. A principle-based hierarchical representation of LTAGs. In *Proceedings of COLING'96, Kopenhagen.*

B. Crabbe and D. Duchier. 2004. Metagrammar Redux. In *Proceedings of CSLP 2004, Copenhagen.*

B. Crabbe. 2005. *Représentation informatique de grammaires fortement lexicalisées : Application à la grammaire d'arbres adjoints.* Ph.D. thesis, Université Nancy 2.

D. Duchier and C. Gardent. 1999. A constraint based treatment of descriptions. In *Proceedings of the 3rd IWCS, Tilburg*.

Denys Duchier and Stefan Thater. 1999. Parsing with tree descriptions: a constraint-based approach. In *NLULP*, pages 17–32, Las Cruces, New Mexico.

R. Evans, G. Gazdar, and D. Weir. 1995. Encoding lexicalized tree adjoining grammars with a nonmonotonic inheritance hierarchy. In *Proceedings of the 33rd Annual Meeting of the ACL, 77-84*.

B. Gaiffe, B. Crabbe, and A. Roussanaly. 2002. A new metagrammar compiler. In *Proceedings of TAG+6, Venice*.

C. Gardent and L. Kallmeyer. 2003. Semantic construction in FTAG. In *Proceedings of EACL'03, Budapest*.

C. Gardent. 2006. Intégration d'une dimension sémantique dans les grammaires d'arbres adjoints. In *Actes de La 13ème édition de la conférence sur le TALN (TALN 2006)*.

A. Joshi and Y. Schabes. 1997. Tree-adjoining grammars. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 69 – 124. Springer, Berlin, New York.

L. Kallmeyer. 1996. Tree description grammars. In *Results of the 3rd KONVENS Conference*, pages 330 – 341. Mouton de Gruyter ed., Hawthorne, NY, USA.

H.-U. Krieger and U. Schafer. 1994. TDL – a type description language for constraint-based grammars. In *Proceedings of COLING-94, pp. 893–899.*

R. Muskens and E. Krahmer. 1998. Description theory, ltags and underspecifi ed semantics. In *TAG'4*.

G. Perrier. 2000. Interaction grammars. In *Proceedings of 18th International Conference on Computational Linguistics (CoLing 2000)*, Sarrebrcken.

G. Perrier. 2003. Les grammaires d'interaction. HDR en informatique, Université Nancy 2.

K. Vijay-Shanker and Y. Schabes. 1992. Structure sharing in lexicalized tree adjoining grammars. In *Proceedings of COLING'92, Nantes, pp. 205 - 212*.

E. Villemonte de la Clergerie. 2005. DyALog: a tabular logic programming based environment for NLP. In *Proceedings of CSLP'05*, Barcelona.

F. Xia, M. Palmer, and K. Vijay-Shanker. 1999. Toward semi-automating grammar development. In *Proc. of NLPRS-99, Beijing, China*.

Conceptual Coherence in the Generation of Referring Expressions

Albert Gatt
Department of Computing Science
University of Aberdeen
agatt@csd.abdn.ac.uk

Kees van Deemter
Department of Computing Science
University of Aberdeen
kvdeemte@csd.abdn.ac.uk

Abstract

One of the challenges in the automatic generation of referring expressions is to identify a set of domain entities coherently, that is, from the same conceptual perspective. We describe and evaluate an algorithm that generates a conceptually coherent description of a target set. The design of the algorithm is motivated by the results of psycholinguistic experiments.

1 Introduction

Algorithms for the Generation of Referring Expressions (GRE) seek a set of properties that distinguish an intended referent from its distractors in a knowledge base. Much of the GRE literature has focused on developing efficient content determination strategies that output the best available description according to some interpretation of the Gricean maxims (Dale and Reiter, 1995), especially Brevity. Work on reference to sets has also proceeded within this general framework (van Deemter, 2002; Gardent, 2002; Horacek, 2004).

One problem that has not received much attention is that of *conceptual coherence* in the generation of plural references, i.e. the ascription of related properties to elements of a set, so that the resulting description constitutes a coherent cover for the plurality. As an example, consider a reference to $\{e_1, e_3\}$ in Table 1 using the Incremental Algorithm (IA) (Dale and Reiter, 1995). IA searches along an ordered list of attributes, selecting properties of the intended referents that remove some distractors. Assuming the ordering in the top row, IA would yield *the postgraduate and the chef*, which is fine in case **occupation** is the *relevant* attribute in the discourse, but otherwise is arguably worse than an alternative like *the italian and the maltese*, because it is more difficult to see what a postgraduate and a chef have in common.

	type	occupation	nationality
e_1	man	postgraduate	maltese
e_2	man	undergraduate	greek
e_3	man	chef	italian

Table 1: Example domain

Such examples lead us to hypothesise the following constraint:

> **Conceptual Coherence Constraint (**CC**):** As far as possible, describe objects using related properties.

Related issues have been raised in the formal semantics literature. Aloni (2002) argues that an appropriate answer to a question of the form *'Wh x?'* must conceptualise the different instantiations of *x* using a perspective which is relevant given the hearer's information state and the context. Kronfeld (1989) distinguishes a description's *functional relevance* – i.e. its success in distinguishing a referent – from its *conversational relevance*, which arises in part from implicatures. In our example, describing e_1 as *the postgraduate* carries the implicature that the entity's academic role is relevant. When two entities are described using contrasting properties, say *the student and the italian*, the contrast may be misleading for the listener.

Any attempt to port these observations to the GRE scenario must do so without sacrificing logical completeness. While a GRE algorithm should attempt to find the most coherent description available, it should not fail in the absence of a coherent set of properties. This paper aims to achieve a dual goal. First (§2), we will show that the CC can be explained and modelled in terms of lexical semantic forces within a description, a claim supported by the results of two experiments. Our focus on 'low-level', lexical, determinants of adequacy constitutes a departure from the standard Gricean view. Second, we describe an algorithm

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 255–262,
Sydney, July 2006.

motivated by the experimental findings (§3) which seeks to find the most coherent description available in a domain according to CC.

2 Empirical evidence

We take as paradigmatic the case where a plural reference involves disjunction/union, that is, has the logical form $\lambda x\,(p(x) \vee q(x))$, realised as a description of the form *the* N_1 *and the* N_2. By hypothesis, the case where all referents can be described using identical properties (logically, a conjunction), is a limiting case of CC.

Previous work on plural anaphor processing has shown that pronoun resolution is easier when antecedents are ontologically similar (e.g. all humans) (Kaup et al., 2002; Koh and Clifton, 2002). Reference to a heterogeneous set increases processing difficulty.

Our experiments extended these findings to full definite NP reference. Throughout, we used a *distributional* definition of similarity, as defined by Lin (1998), which was found to be highly correlated to people's preferences for disjunctive descriptions (Gatt and van Deemter, 2005). The similarity of two arbitrary objects a and b is a function of the information gained by giving a joint description of a and b in terms of what they have in common, compared to describing a and b separately. The relevant data in the lexical domain is the grammatical environment in which words occur. This information is represented as a set of triples $\langle rel, w, w' \rangle$, where rel is a grammatical relation, w the word of interest and w' its co-argument in rel (e.g. $\langle$ *premodifies, dog, domestic* $\rangle$). Let $F(w)$ be a list of such triples. The information content of this set is defined as mutual information $I(F(w))$ (Church and Hanks, 1990). The similarity of two words w_1 and w_2, of the same grammatical category, is:

$$\sigma(w_1, w_2) = \frac{2 \times I(F(w_1) \cap F(w_2))}{I(F(w_1)) + I(F(w_2))} \quad (1)$$

For example, if *premodifies* is one of the relevant grammatical relations, then *dog* and *cat* might occur several times in a corpus with the same premodifiers (*tame*, *domestic*, etc). Thus, $\sigma(dog, cat)$ is large because in a corpus, they often occur in the same contexts and there is considerable information gain in a description of their common data.

Rather than using a hand-crafted ontology to infer similarity, this definition looks at real language use. It covers ontological similarity to the extent that ontologically similar objects are talked about in the same contexts, but also cuts across ontological distinctions (for example *newspaper* and *journalist* might turn out to be very similar).

Condition	a	b	c	distractor
HDS	spanner	chisel	plug	thimble
LDS	toothbrush	knife	ashtray	clock

Figure 1: Conditions in Experiment 1

We use the information contained in the SketchEngine database[1] (Kilgarriff, 2003), a largescale implementation of Lin's theory based on the BNC, which contains grammatical triples in the form of *Word Sketches* for each word, with each triple accompanied by a salience value indicating the likelihood of occurrence of the word with its argument in a grammatical relation. Each word also has a thesaurus entry, containing a ranked list of words of the same category, ordered by their similarity to the head word.

2.1 Experiment 1

In Experiment 1, participants were placed in a situation where they were buying objects from an online store. They saw scenarios containing four pictures of objects, three of which (the targets) were identically priced. Participants referred to them by completing a 2-sentence discourse:

S1 The *object1* and the *object 2* cost *amount*.

S2 The *object3* also costs *amount*.

If similarity is a constraint on referential coherence in plural references, then if two targets are similar (and dissimilar to the third), a plural reference to them in S1 should be more likely, with the third entity referred to in S2.

Materials, design and procedure All the pictures were artefacts selected from a set of drawings normed in a picture-naming task with British English speakers (Barry et al., 1997).

Each trial consisted of the four pictures arranged in an array on a screen. Of the three targets (a, b, c), c was always an object whose name in the norms was *dis*similar to that of a and b. The semantic similarity of (nouns denoting) a and b was manipulated as a factor with two levels: **High Distributional Similarity (HDS)** meant that b occurred among the top 50 most similar items to a in its Sketchengine thesaurus entry. **Low DS (LDS))**

[1] `http://www.sketchengine.co.uk`

meant that b did not occur in the top 500 entries for a. Examples are shown in Figure 2.1.

Visual Similarity (VS) of a and b was also controlled. Pairs of pictures were first normed with a group who rated them on a 10-point scale based on their visual properties. High-VS (HVS) pairs had a mean rating ≥ 6; Low-VS LVS) pairs had mean ratings ≤ 2. Two sets of materials were constructed, for a total of $2\ (DS) \times 2\ (VS) \times 2 = 8$ trials.

29 self-reported native or fluent speakers of English completed the experiment over the web. To complete the sentences, participants clicked on the objects in the order they wished to refer to them. Nouns appeared in the next available space[2].

Results and discussion Responses were coded according to whether objects a and b were referred to in the plural subject of S1 ($a + b$ responses) or not ($a - b$ responses). If our hypothesis is correct, there should be a higher proportion of $a + b$ responses in the HDS condition. We did not expect an effect of VS. In what follows, we report by-subjects Friedman analyses (χ^2_1); by-items analyses (χ^2_2); and by-subjects sign tests (Z) on proportions of responses for pairwise comparisons.

Response frequencies across conditions differed reliably by subjects ($\chi^2_1 = 46.124, p < .001$). The frequency of $a + b$ responses in S1 was reliably higher than that of $a - b$ in the HDS condition ($\chi^2_2 = 41.371, p < .001$), but not the HVS condition ($\chi^2_2 = 1.755, ns$). Pairwise comparisons between HDS and LDS showed a significantly higher proportion of $a + b$ responses in the former ($Z = 4.48, p < .001$); the difference was barely significant across VS conditions ($Z = 1.9, p = .06$).

The results show that, given a clear choice of entities to refer to in a plurality, people are more likely to describe similar entities in a plural description. However, these results raise two further questions. First, given a choice of distinguishing properties for individuals making up a target set, will participants follow the predictions of the CC? (In other words, is distributional similarity relevant for content determination?) Second, does the similarity effect carry over to modifiers, such as adjectives, or is the CC exclusively a constraint on types?

[2]Earler replications involving typing yielded parallel results and high conformity between the words used and those predicted by the picture norms.

	Three millionaires with a passion for antiques were spotted dining at a London restaurant.
e_1	One of the men, a Rumanian, is a dealer$_i$.
e_2	The second, a prince$_j$, is a collector$_i$.
e_3	The third, a duke$_j$, is a bachelor.
	The XXXX were both accompanied by servants, but the bachelor wasn't .

Figure 2: Example discourses

2.2 Experiment 2

Experiment 2 was a sentence continuation task, designed to closely approximate content determination in GRE. Participants saw a series of discourses, in which three entities (e_1, e_2, e_3) were introduced, each with two distinguishing properties. The final sentence in each discourse had a missing plural subject NP referring to two of these. The context made it clear which of the three entities had to be referred to. Our hypothesis was that participants would prefer to use semantically similar properties for the plural reference, *even if* dissimilar properties were also available.

Materials, design and procedure Materials consisted of 24 discourses, such as those in Figure 2.2. After an initial introductory sentence, the 3 entities were introduced in separate sentences. In all discourses, the pairs $\{e_1, e_2\}$ and $\{e_2, e_3\}$ could be described using either pairwise similar or dissimilar properties (similar pairs are coindexed in the figure). In half the discourses, the distinguishing properties of each entity were *nouns*; thus, although all three entities belonged to the same ontological category (e.g. all human), they had distinct types (e.g. *duke*, *prince*, *bachelor*). In the other half, entities were of the same type, that is the NPs introducing them had the same nominal head, but had distinguishing adjectival modifiers. For counterbalancing, two versions of each discourse were constructed, such that, if $\{e_1, e_2\}$ was the target set in Version 1, then $\{e_2, e_3\}$ was the target in Version 2. Twelve filler items requiring singular reference in the continuation were also included. The order in which the entities were introduced was randomised across participants, as was the order of trials. The experiment was completed by 18 native speakers of English, selected from the Aberdeen NLG Group database. They were randomly assigned to either Version 1 or 2.

Results and discussion Responses were coded 1 if the semantically similar properties were used (e.g. *the prince and the duke* in Fig. 2.2); 2 if the

similar properties were used together with other properties (e.g. *the prince and the bachelor duke*); 3 if a superordinate term was used to replace the similar properties (e.g. *the noblemen*); 4 otherwise (e.g. *The duke and the collector*).

Response types differed significantly in the nominal condition both by subjects ($\chi^2_1 = 45.89, p < .001$) and by items ($\chi^2_2 = 287.9, p < .001$). Differences were also reliable in the modifier condition ($\chi^2_1 = 36.3, p < .001$, $\chi^2_2 = 199.2, p < .001$). However, the trends across conditions were opposed, with more items in the 1 response category in the nominal condition (53.7%) and more in the 4 category in the modifier condition (47.2%). Recoding responses as binary ('similar' = 1,2,3; 'dissimilar' = 4) showed a significant difference in proportions for the nominal category ($\chi^2 = 4.78, p = .03$), but not the modifier category. Pairwise comparisons showed a significantly larger proportion of 1 ($Z = 2.7, p = .007$) and 2 responses ($Z = 2.54, p = .01$) in the nominal compared to the modifier condition.

The results suggest that in a referential task, participants are likely to conform to the CC, but that the CC operates mainly on nouns, and less so on (adjectival) modifiers. Nouns (or types, as we shall sometimes call them) have the function of categorising objects; thus similar types facilitate the mental representation of a plurality in a conceptually coherent way. According to the definition in (1), this is because similarity of two types implies a greater likelihood of their being used in the same predicate-argument structures. As a result, it is easier to map the elements of a plurality to a common role in a sentence. A related proposal has been made by Moxey and Sanford (1995), whose *Scenario Mapping Principle* holds that a plural reference is licensed to the extent that the elements of the plurality can be mapped to a common role in the discourse. This is influenced by how easy it is to conceive of such a role for the referents. Our results can be viewed as providing a handle on the notion of 'ease of conception of a common role'; in particular we propose that likelihood of occurrence in the same linguistic contexts directly reflects the extent to which two types can be mapped to a single plural role.

As regards modifiers, while it is probably premature to suggest that CC plays no role in modifier selection, it is likely that modifiers play a different role from nouns. Previous work has shown that restrictions on the plausibility of adjective-noun combinations exist (Lapata et al., 1999), and that using unlikely combinations (e.g. *the immaculate kitchen* rather than *the spotless kitchen*) impacts processing in online tasks (Murphy, 1984). Unlike types, which have a categorisation function, modifiers have the role of adding information about an element of a category. This would partially explain the experimental results: When elements of a plurality have identical types (as in the modifier version of our experiment), the CC is already satisfied, and selection of modifiers would presumably depend on respecting adjective-noun combination restrictions. Further research is required to verify this, although the algorithm presented below makes use of the Sketch Engine database to take modifier-noun combinations into account.

id	base type	occupation	specialisation	girth
e_1	woman	professor	physicist	plump
e_2	woman	lecturer	geologist	thin
e_3	man	lecturer	biologist	plump
e_4	man		chemist	thin

Table 2: An example knowledge base

3 An algorithm for referring to sets

Our next task is to port the results to GRE. The main ingredient to achieve conceptual coherence will be the definition of semantic similarity. In what follows, all examples will be drawn from the domain in Table 3.

We make the following assumptions. There is a set U of domain entities, properties of which are specified in a KB as attribute-value pairs. We assume a distinction between *types*, that is, any property that can be realised as a noun; and *modifiers*, or non-types. Given a set of target referents $R \subseteq U$, the algorithm described below generates a description D in Disjunctive Normal Form (DNF), having the following properties:

1. Any disjunct in D contains a 'type' property, i.e. a property realisable as a head noun.
2. If D has two or more disjuncts, each a conjunction containing at least one type, then the disjoined types should be as similar as possible, given the information in the KB and the *completeness* requirement: that the algorithm find a distinguishing description whenever one exists.

We first make our interpretation of the CC more precise. Let T be the set of types in the KB, and let $\sigma(t, t')$ be the (symmetrical) similarity between any two types t and t'. These determine a semantic space $\mathbb{S} = \langle T, \sigma \rangle$. We define the notion of a perspective as follows.

Definition 1. Perspective
A perspective $\mathcal{P}$ is a convex subset of $\mathbb{S}$, i.e.:

$$\forall t, t', t'' \in T :$$
$$\{t, t'\} \subseteq \mathcal{P} \wedge \sigma(t, t'') \geq \sigma(t, t') \rightarrow t'' \in \mathcal{P}$$

The aims of the algorithm are to describe elements of R using types from the same perspective, failing which, it attempts to minimise the distance between the perspectives from which types are selected in the disjunctions of D. Distance between perspectives is defined below.

3.1 Finding perspectives

The system makes use of the SketchEngine database as its primary knowledge source. Since the definition of similarity applies to words, rather than properties, the first step is to generate all possible lexicalisations of the available attribute-value pairs in the domain. In this paper, we simplify by assuming a one-to-one mapping between properties and words.

Another requirement is to distinguish between type properties (the set T), and non-types (M)[3]. The Thesaurus is used to find pairwise similarity of types in order to group them into related clusters. Word Sketches are used to find, for each type, the modifiers in the KB that are appropriate to the type, on the basis of the associated salience values. For example, in Table 3, e_3 has *plump* as the value for **girth**, which combines more felicitously with *man*, than with *biologist*.

Types are clustered using the algorithm described in Gatt (2006). For each type t, the algorithm finds its nearest neighbour n_t in semantic space. Clusters are then found by recursively grouping elements with their nearest neighbours. If t, t' have a common nearest neighbour n, then $\{t, t', n\}$ is a cluster. Clearly, the resulting sets are convex in the sense of Definition 1. Each modifier is assigned to a cluster by finding in its Word Sketch the type with which it co-occurs with the greatest salience value. Thus, a cluster is a pair

[3]This is determined using corpus-derived information. Note that T and M need not be disjoint, and entities can have more than one type property

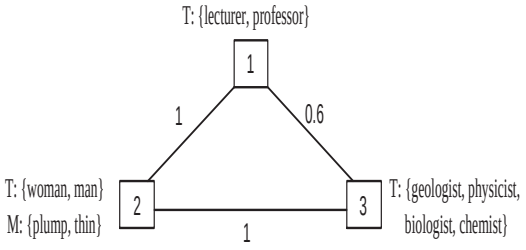


Figure 3: Perspective Graph

$\langle \mathcal{P}, M' \rangle$ where $\mathcal{P}$ is a perspective, and $M' \subseteq M$. The distance $\delta(A, B)$ between two clusters A and B is defined straightforwardly in terms of the distance between their perspectives $\mathcal{P}_A$ and $\mathcal{P}_B$:

$$\delta(A, B) = \frac{1}{1 + \frac{\sum_{x \in \mathcal{P}_A, y \in \mathcal{P}_B} \sigma(x,y)}{|\mathcal{P}_A \times \mathcal{P}_B|}} \qquad (2)$$

Finally, a weighted, connected graph $\mathcal{G} = \langle V, E, \delta \rangle$ is created, where V is the set of clusters, and E is the set of edges with edge weights defined as the semantic distance between perspectives. Figure 3.1 shows the graph constructed for the domain in Table 3.

We now define the coherence of a description more precisely. Given a DNF description D, we shall say that a perspective $\mathcal{P}$ is *realised in* D if there is at least one type $t \in \mathcal{P}$ which is in D. Let $\mathbb{P}_D$ be the set of perspectives realised in D. Since $\mathcal{G}$ is connected, $\mathbb{P}_D$ determines a connected subgraph of $\mathcal{G}$. The *total weight* of D, $w(D)$ is the sum of weights of the edges in $\mathbb{P}_D$.

Definition 2. Maximal coherence
A description D is *maximally coherent* iff there is no description D' coextensive with D such that $w(D) > w(D')$.

(Note that several descriptions of the same referent may all be maximally coherent.)

3.2 Content determination

The core of the content determination procedure maintains the DNF description D as an associative array, such that for any $r \in R$, $D[r]$ is a conjunction of properties true of r. Given a cluster $\langle \mathcal{P}, M \rangle$, the procedure searches incrementally first through $\mathcal{P}$, and then M, selecting properties that are true of at least one referent and exclude some distractors, as in the IA (Dale and Reiter, 1995).

By Definition 2, the task of the algorithm is to minimise the total weight $w(D)$. If $\mathbb{P}_D$ is the

set of perspectives represented in D on termination, then *maximal coherence* would require $\mathbb{P}_D$ to be the subgraph of $\mathcal{G}$ with the lowest total cost from which a distinguishing description could be constructed. Under this interpretation, $\mathbb{P}_D$ corresponds to a Shortest Connection, or Steiner, Network. Finding such networks is known to be NP-Hard. Therefore, we adopt a weaker (greedy) interpretation. Under the new definition, if D is the only description for R, then it trivially satisfies maximal coherence. Otherwise, the algorithm aims to maximise *local coherence*.

Definition 3. Local coherence
A description D is *locally coherent* iff:

a. **either** D is maximally coherent **or**

b. there is no D' coextensive with D, obtained by replacing types from some perspective in $\mathbb{P}_D$ with types from another perspective such that $w(D) > w(D')$.

Our implementation of this idea begins the search for distinguishing properties by identifying the vertex of $\mathcal{G}$ which contains the greatest number of referents in its extension. This constitutes the root node of the search path. For each node of the graph it visits, the algorithm searches for properties that are true of some subset of R, and removes some distractors, maintaining a set N of the perspectives which are represented in D up to the current point. The crucial choice points arise when a new node (perspective) needs to be visited in the graph. At each such point, the next node n to be visited is the one which minimises the total weight of N, that is:

$$\min_{n \in V} \sum_{u \in N} w(u, n) \tag{3}$$

The results of this procedure closely approximate maximal coherence, because the algorithm starts with the vertex most likely to distinguish the referents, and then greedily proceeds to those nodes which minimise $w(D)$ given the current state, that is, taking all previously used nodes into account.

As an example of the output, we will take $R = \{e_1, e_3, e_4\}$ as the intended referents in Table 3. First, the algorithm determines the cluster with the greatest number of referents in its extension. In this case, there is a tie between clusters 2 and 3 in Figure 3.1, since all three entities have type properties in these clusters. In either case, the entities are distinguishable from a single cluster. If cluster 3 is selected as the root, the output is $\lambda x\,[physicist(x) \vee biologist(x) \vee chemist(x)]$. In case the algorithm selects cluster 2 as the root node the final output is the logical form $\lambda x\,[man(x) \vee (woman(x) \wedge plump(x))]$.

There is an alternative description that the algorithm does not consider. An algorithm that aimed for conciseness would generate $\lambda x\,[professor(x) \vee man(x)]$ (*the professor and the men*), which does not satisfy local coherence. These examples therefore highlight the possible tension between the avoidance of redundancy and achieving coherence. It is to an investigation of this tension that we now turn.

4 Evaluation

It has been known at least since Dale and Reiter (1995) that the best distinguishing description is not always the shortest one. Yet, brevity plays a part in all GRE algorithms, sometimes in a strict form (Dale, 1989), or by letting the algorithm *approximate* the shortest description (for example, in the Dale and Reiter's IA). This is also true of references to sets, the clearest example being Gardent's constraint based approach, which always finds the description with the smallest number of logical operators. Such proposals do not take coherence (in our sense of the word) into account. This raises obvious questions about the relative importance of brevity and coherence in reference to sets.

The evaluation took the form of an experiment to compare the output of our *Coherence Model* with the family of algorithms that have placed Brevity at the centre of content determination. Participants were asked to compare pairs of descriptions of one and the same target set, selecting the one they found most natural. Each description could either be optimally brief or not ($\pm b$) and also either optimally coherent or not ($\pm c$). Non-brief descriptions, took the form *the A, the B and the C*. Brief descriptions 'aggregated' two disjuncts into one (e.g. *the A and the D's* where D comprises the union of B and C). We expected to find that:

H1 $+c$ descriptions are preferred over $-c$.

H2 $(+c, -b)$ descriptions are preferred over ones that are $(-c, +b)$.

H3 $+b$ descriptions are preferred over $-b$.

Confirmation of H1 would be interpreted as evidence that, by taking coherence into account, our

	Three old manuscripts were auctioned at Sotheby's.
e_1	One of them is a book, a biography of a composer.
e_2	The second, a sailor's journal, was published in the form of a pamphlet. It is a record of a voyage.
e_3	The third, another pamphlet, is an essay by Hume.
$(+c, -b)$	The biography, the journal and the essay were sold to a collector.
$(+c, +b)$	The book and the pamphlets were sold to a collector.
$(-c, +b)$	The biography and the pamphlets were sold to a collector.
$(-c, -b)$	The book, the record and the essay were sold to a collector.

Figure 4: Example domain in the evaluation

algorithm is on the right track. If H3 were confirmed, then earlier algorithms were (also) on the right track by taking brevity into account. Confirmation of H2 would be interpreted as meaning that, in references to sets, conceptual coherence is more important than brevity (defined as the number of disjuncts in a disjunctive reference to a set).

Materials, design and procedure Six discourses were constructed, each introducing three entities. Each set of three could be described using all 4 possible combinations of $\pm b \times \pm c$ (see Figure 4). Entities were human in two of the discourses, and artefacts of various kinds in the remainder. Properties of entities were introduced textually; the order of presentation was randomised. A forced-choice task was used. Each discourse was presented with 2 possible continuations consisting of a sentence with a plural subject NP, and participants were asked to indicate the one they found most natural. The 6 comparisons corresponded to 6 sub-conditions:

C1. **Coherence constant**
- a. $(+c, -b)$ vs. $(+c, +b)$
- b. $(-c, -b)$ vs. $(-c, +b)$

C2. **Brevity constant**
- a. $(+c, -b)$ vs. $(-c, -b)$
- b. $(+c, +b)$ vs. $(-c, +b)$

C3. **Tradeoff/control**
- a. $(+c, -b)$ vs. $(-c, +b)$
- b. $(-c, -b)$ vs. $(+c, +b)$

Participants saw each discourse in a single condition. They were randomly divided into six groups, so that each discourse was used for a different condition in each group. 39 native English speakers, all undergraduates at the University of Aberdeen, took part in the study.

Results and discussion Results were coded according to whether a participant's choice was $\pm b$ and/or $\pm c$. Table 4 displays response proportions. Overall, the conditions had a significant impact on responses, both by subjects (Friedman $\chi^2 = 107.3, p < .001$) and by items ($\chi^2 = 30.2, p < .001$). When coherence was kept constant (C1a and C1b), the likelihood of a response being $+b$ was no different from $-b$ (C1a: $\chi^2 = .023, p = .8$; C1b: $\chi^2 = .64, p = .4$); the conditions C1a and C1b did not differ significantly ($\chi^2 = .46, p = .5$). By contrast, conditions where brevity was kept constant (C2a and C2b) resulted in very significantly higher proportions of $+c$ choices (C2a: $\chi^2 = 16.03, p < .001$; C2b: $\chi^2 = 13.56, p < .001$). No difference was observed between C2a and C2b ($\chi^2 = .08, p = .8$). In the tradeoff case (C3a), participants were much more likely to select a $+c$ description than a $+b$ one ($\chi^2 = 39.0, p < .001$); a majority opted for the $(+b, +c)$ description in the control case ($\chi^2 = 39.0, p < .001$).

	C1a	C1b	C2a	C2b	C3a	C3b
$+b$	51.3	43.6	–	–	30.8	76.9
$+c$	–	–	82.1	79.5	69.2	76.9

Table 3: Response proportions (%)

The results strongly support H1 and H2, since participants' choices are impacted by Coherence. They do not indicate a preference for brief descriptions, a finding that echoes Jordan's (2000), to the effect that speakers often relinquish brevity in favour of observing task or discourse constraints. Since this experiment compared our algorithm against the current state of the art in references to sets, these results do not necessarily warrant the affirmation of the null hypothesis in the case of H3. We limited Brevity to number of disjuncts, omitting negation, and varying only between length 2 or 3. Longer or more complex descriptions might evince different tendencies. Nevertheless, the results show a strong impact of Coherence, compared to (a kind of) brevity, in strong support of the algorithm presented above, as a realisation of the Coherence Model.

5 Conclusions and future work

This paper started with an empirical investigation of conceptual coherence in reference, which led to a definition of *local* coherence as the basis for a new greedy algorithm that tries to minimise the semantic distance between the perspectives repre-

sented in a description. The evaluation strongly supports our Coherence Model.

We are extending this work in two directions. First, we are investigating similarity effects *across noun phrases*, and their impact on text readability. Finding an impact of such factors would make this model a useful complement to current theories of discourse, which usually interpret coherence in terms of discourse/sentential structure.

Second, we intend to relinquish the assumption of a one-to-one correspondence between properties and words (cf. Siddharthan and Copestake (2004)), making use of the fact that words can be disambiguated by nearby words that are similar. To use a well-worn example: the 'financial institution' sense of *bank* might not make *the river and its bank* lexically incoherent as a description of a piece of scenery, since the word *river* might cause the hearer to focus on the aquatic reading of the word anyway.

6 Acknowledgements

Thanks to Ielka van der Sluis, Imtiaz Khan, Ehud Reiter, Chris Mellish, Graeme Ritchie and Judith Masthoff for useful comments. This work is part of the TUNA project (`http://www.csd.abdn.ac.uk/research/tuna`), supported by EPSRC grant no. GR/S13330/01

References

M. Aloni. 2002. Questions under cover. In D. Barker-Plummer, D. Beaver, J. van Benthem, and P. Scotto de Luzio, editors, *Words, Proofs, and Diagrams*. CSLI, Stanford, Ca.

C. Barry, C. M. Morrison, and A. W. Ellis. 1997. Naming the snodgrass and vanderwart pictures. *Quarterly Journal of Experimental Psychology*, 50A(3):560–585.

K. W. Church and P. Hanks. 1990. Word association norms, mutual information and lexicography. *Computational Linguistics*, 16(1):22–29.

R. Dale and E. Reiter. 1995. Computational interpretation of the Gricean maxims in the generation of referring expressions. *Cognitive Science*, 19(8):233–263.

Robert Dale. 1989. Cooking up referring expressions. In *Proc. 27th Annual Meeting of the Association for Computational Linguistics.*

C. Gardent. 2002. Generating minimal definite descriptions. In *Proc. 40th Annual Meeting of the Association for Computational Linguistics.*

A. Gatt and K. van Deemter. 2005. Semantic similarity and the generation of referring expressions: A first report. In *Proceedings of the 6th International Workshop on Computational Semantics, IWCS-6.*

A. Gatt. 2006. Structuring knowledge for reference generation: A clustering algorithm. In *Proc. 11th Conference of the European Chapter of the Association for Computational Linguistics.*

H. Horacek. 2004. On referring to sets of objects naturally. In *Proc. 3rd International Conference on Natural Language Generation.*

P. W. Jordan. 2000. Can nominal expressions achieve multiple goals? In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics.*

B. Kaup, S. Kelter, and C. Habel. 2002. Representing referents of plural expressions and resolving plural anaphors. *Language and Cognitive Processes*, 17(4):405–450.

A. Kilgarriff. 2003. Thesauruses for natural language processing. In *Proc. NLP-KE, Beijing.*

S. Koh and C. Clifton. 2002. Resolution of the antecedent of a plural pronoun: Ontological categories and predicate symmetry. *Journal of Memory and Language*, 46:830–844.

A. Kronfeld. 1989. Conversationally relevant descriptions. In *Proc. 27th Annual Meeting of the Association for Computational Linguistics.*

M. Lapata, S. McDonald, and F. Keller. 1999. Determinants of adjective-noun plausibility. In *Proc. 9th Conference of the European Chapter of the Association for Computational Linguistics.*

D. Lin. 1998. An information-theoretic definition of similarity. In *Proc. International Conference on Machine Learning.*

L. Moxey and A. Sanford. 1995. Notes on plural reference and the scenario-mapping principle in comprehension. In C.Habel and G.Rickheit, editors, *Focus and cohesion in discourse.* de Gruyter, Berlin.

G.L. Murphy. 1984. Establishing and accessing referents in discourse. *Memory and Cognition*, 12:489–497.

A. Siddharthan and A. Copestake. 2004. Generating referring expressions in open domains. In *Proc. 42nd Annual Meeting of the Association for Computational Linguistics.*

K. van Deemter. 2002. Generating referring expressions: Boolean extensions of the incremental algorithm. *Computational Linguistics*, 28(1):37–52.

Discriminative Reranking for Semantic Parsing

Ruifang Ge Raymond J. Mooney
Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712
{grf,mooney}@cs.utexas.edu

Abstract

Semantic parsing is the task of mapping natural language sentences to complete formal meaning representations. The performance of semantic parsing can be potentially improved by using discriminative reranking, which explores arbitrary global features. In this paper, we investigate discriminative reranking upon a baseline semantic parser, SCISSOR, where the composition of meaning representations is guided by syntax. We examine if features used for syntactic parsing can be adapted for semantic parsing by creating similar semantic features based on the mapping between syntax and semantics. We report experimental results on two real applications, an interpreter for coaching instructions in robotic soccer and a natural-language database interface. The results show that reranking can improve the performance on the coaching interpreter, but not on the database interface.

1 Introduction

A long-standing challenge within natural language processing has been to understand the meaning of natural language sentences. In comparison with shallow semantic analysis tasks, such as word-sense disambiguation (Ide and Jeanéronis, 1998) and semantic role labeling (Gildea and Jurafsky, 2002; Carreras and Màrquez, 2005), which only partially tackle this problem by identifying the meanings of target words or finding semantic roles of predicates, semantic parsing (Kate et al., 2005; Ge and Mooney, 2005; Zettlemoyer and Collins, 2005) pursues a more ambitious goal – mapping natural language sentences to complete formal meaning representations (MRs), where the meaning of each part of a sentence is analyzed, including noun phrases, verb phrases, negation, quantifiers and so on. Semantic parsing enables logic reasoning and is critical in many practical tasks, such as speech understanding (Zue and Glass, 2000), question answering (Lev et al., 2004) and advice taking (Kuhlmann et al., 2004).

Ge and Mooney (2005) introduced an approach, SCISSOR, where the composition of meaning representations is guided by syntax. First, a statistical parser is used to generate a semantically-augmented parse tree (SAPT), where each internal node includes both a syntactic and semantic label. Once a SAPT is generated, an additional meaning-composition process guided by the tree structure is used to translate it into a final formal meaning representation.

The performance of semantic parsing can be potentially improved by using discriminative reranking, which explores arbitrary global features. While reranking has benefited many tagging and parsing tasks (Collins, 2000; Collins, 2002c; Charniak and Johnson, 2005) including semantic role labeling (Toutanova et al., 2005), it has not yet been applied to semantic parsing. In this paper, we investigate the effect of discriminative reranking to semantic parsing.

We examine if the features used in reranking syntactic parses can be adapted for semantic parsing, more concretely, for reranking the top SAPTs from the baseline model SCISSOR. The syntactic features introduced by Collins (2000) for syntactic parsing are extended with similar semantic features, based on the coupling of syntax and semantics. We present experimental results on two corpora: an interpreter for coaching instructions

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 263–270,
Sydney, July 2006.

in robotic soccer (CLANG) and a natural-language database interface (GeoQuery). The best reranking model significantly improves F-measure on CLANG from 82.3% to 85.1% (15.8% relative error reduction), however, it fails to show improvements on GEOQUERY.

2 Background

2.1 Application Domains

2.1.1 CLANG: the RoboCup Coach Language

RoboCup (`www.robocup.org`) is an international AI research initiative using robotic soccer as its primary domain. In the Coach Competition, teams of agents compete on a simulated soccer field and receive advice from a team coach in a formal language called CLANG. In CLANG, tactics and behaviors are expressed in terms of if-then rules. As described in Chen et al. (2003), its grammar consists of 37 non-terminal symbols and 133 productions. Negation and quantifiers like *all* are included in the language. Below is a sample rule with its English gloss:

```
((bpos (penalty-area our))
 (do (player-except our {4})
     (pos (half our))))
```

"If the ball is in our penalty area, all our players except player 4 should stay in our half."

2.1.2 GEOQUERY: a DB Query Language

GEOQUERY is a logical query language for a small database of U.S. geography containing about 800 facts. The GEOQUERY language consists of Prolog queries augmented with several meta-predicates (Zelle and Mooney, 1996). Negation and quantifiers like *all* and *each* are included in the language. Below is a sample query with its English gloss:

```
answer(A,count(B,(city(B),loc(B,C),
       const(C,countryid(usa))),A))
```

"How many cities are there in the US?"

2.2 SCISSOR: the Baseline Model

SCISSOR is based on a fairly standard approach to compositional semantics (Jurafsky and Martin, 2000). First, a statistical parser is used to construct a semantically-augmented parse tree that captures the semantic interpretation of individual words and the basic predicate-argument structure of a sentence. Next, a recursive deterministic procedure is used to compose the MR of a parent node from the MR of its children following the tree structure.

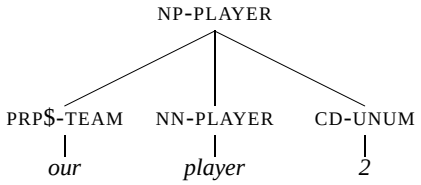


Figure 1: A SAPT for describing a simple CLANG concept PLAYER .

Figure 1 shows the SAPT for a simple natural language phrase describing the concept PLAYER in CLANG. We can see that each internal node in the parse tree is annotated with a semantic label (shown after dashes) representing *concepts* in an application domain; when a node is semantically vacuous in the application domain, it is assigned with the semantic label NULL. The semantic labels on words and non-terminal nodes represent the meanings of these words and constituents respectively. For example, the word *our* represents a TEAM concept in CLANG with the value *our*, whereas the constituent OUR PLAYER 2 represents a PLAYER concept. Some *type concepts* do not take arguments, like *team* and *unum* (uniform number), while some concepts, which we refer to as *predicates*, take an ordered list of arguments, like *player* which requires both a TEAM and a UNUM as its arguments.

SAPTs are given to a meaning composition process to compose meaning, guided by both tree structures and domain predicate-argument requirements. In figure 1, the MR of *our* and *2* would fill the arguments of PLAYER to generate the MR of the whole constituent PLAYER(OUR,2) using this process.

SCISSOR is implemented by augmenting Collins' (1997) head-driven parsing model II to incorporate the generation of semantic labels on internal nodes. In a head-driven parsing model, a tree can be seen as generated by expanding non-terminals with grammar rules recursively. To deal with the sparse data problem, the expansion of a non-terminal (parent) is decomposed into primitive steps: a child is chosen as the head and is generated first, and then the other children (modifiers) are generated independently

BACK-OFFLEVEL	$\mathcal{P}_{L1}(L_i\|...)$
1	P,H,w,t,Δ,LC
2	P,H,t,Δ,LC
3	P,H,Δ,LC
4	P,H
5	P

Table 1: Extended back-off levels for the semantic parameter $\mathcal{P}_{L1}(L_i|...)$, using the same notation as in Ge and Mooney (2005). The symbols P, H and L_i are the semantic label of the parent , head, and the ith left child, w is the head word of the parent, t is the semantic label of the head word, δ is the distance between the head and the modifier, and LC is the left semantic subcat.

constrained by the head. Here, we only describe changes made to SCISSOR for reranking, for a full description of SCISSOR see Ge and Mooney (2005).

In SCISSOR, the generation of semantic labels on modifiers are constrained by semantic subcategorization frames, for which data can be very sparse. An example of a semantic subcat in Figure 1 is that the head PLAYER associated with NN requires a TEAM as its modifier. Although this constraint improves SCISSOR's precision, which is important for semantic parsing, it also limits its recall. To generate plenty of candidate SAPTs for reranking, we extended the back-off levels for the parameters generating semantic labels of modifiers. The new set is shown in Table 1 using the parameters for the generation of the left-side modifiers as an example. The back-off levels 4 and 5 are newly added by removing the constraints from the semantic subcat. Although the best SAPTs found by the model may not be as precise as before, we expect that reranking can improve the results and rank correct SAPTs higher.

2.3 The Averaged Perceptron Reranking Model

Averaged perceptron (Collins, 2002a) has been successfully applied to several tagging and parsing reranking tasks (Collins, 2002c; Collins, 2002a), and in this paper, we employed it in reranking semantic parses generated by the base semantic parser SCISSOR. The model is composed of three parts (Collins, 2002a): a set of candidate SAPTs GEN, which is the top n SAPTs of a sentence from SCISSOR; a function Φ that maps a sentence

Inputs: A set of training examples (x_i, y_i^*), $i = 1...n$, where x_i is a sentence, and y_i^* is a candidate SAPT that has the highest similarity score with the gold-standard SAPT
Initialization: Set $\bar{W} = 0$
Algorithm:
For $t = 1...T, i = 1...n$
Calculate $y_i = \arg\max_{y \in GEN(x_i)} \Phi(x_i, y) \cdot \bar{W}$
If $(y_i \neq y_i^*)$ then $\bar{W} = \bar{W} + \Phi(x_i, y_i^*) - \Phi(x_i, y_i)$
Output: The parameter vector $\bar{W}$

Figure 2: The perceptron training algorithm.

x and its SAPT y into a feature vector $\Phi(x, y) \in \mathbb{R}^d$; and a weight vector $\bar{W}$ associated with the set of features. Each feature in a feature vector is a function on a SAPT that maps the SAPT to a real value. The SAPT with the highest score under a parameter vector $\bar{W}$ is outputted, where the score is calculated as:

$$score(x, y) = \Phi(x, y) \cdot \bar{W} \quad (1)$$

The perceptron training algorithm for estimating the parameter vector $\bar{W}$ is shown in Figure 2. For a full description of the algorithm, see (Collins, 2002a). The averaged perceptron, a variant of the perceptron algorithm is often used in testing to decrease generalization errors on unseen test examples, where the parameter vectors used in testing is the average of each parameter vector generated during the training process.

3 Features for Reranking SAPTs

In our setting, reranking models discriminate between SAPTs that can lead to correct MRs and those that can not. Intuitively, both syntactic and semantic features describing the syntactic and semantic substructures of a SAPT would be good indicators of the SAPT's correctness.

The syntactic features introduced by Collins (2000) for reranking syntactic parse trees have been proven successfully in both English and Spanish (Cowan and Collins, 2005). We examine if these syntactic features can be adapted for semantic parsing by creating similar semantic features. In the following section, we first briefly describe the syntactic features introduced by Collins (2000), and then introduce two adapted semantic feature sets. A SAPT in CLANG is shown in Figure 3 for illustrating the features throughout this section.

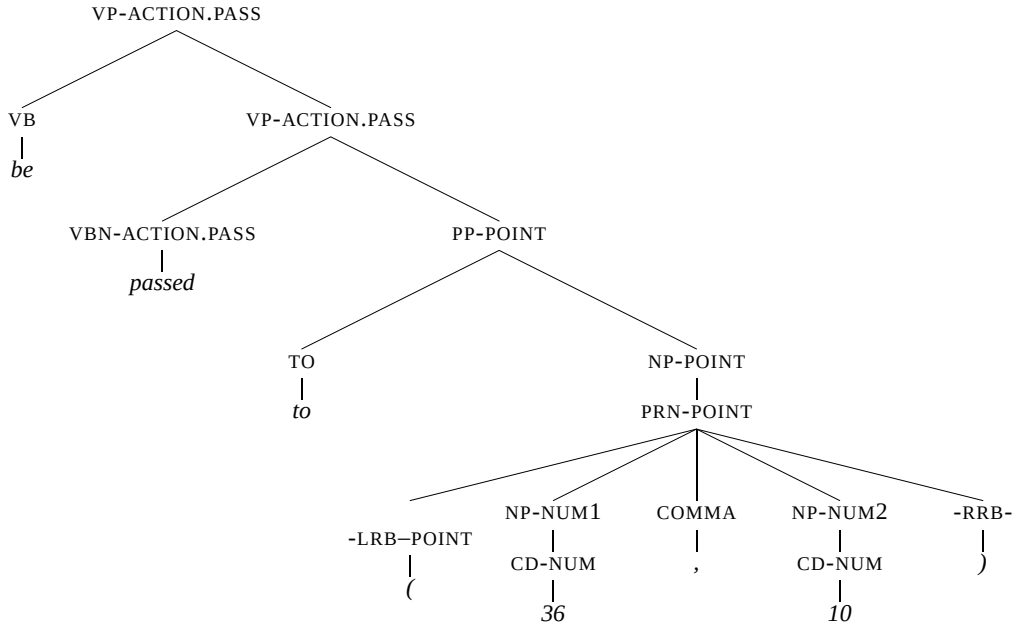


Figure 3: A SAPT for illustrating the reranking features, where the syntactic label "," is replaced by COMMA for a clearer description of features, and the NULL semantic labels are not shown. The head of the rule "PRN-POINT→ -LRB–POINT NP-NUM1 COMMA NP-NUM2 -RRB-" is -LRB–POINT. The semantic labels NUM1 and NUM2 are meta concepts in CLANG specifying the semantic role filled since NUM can fill multiple semantic roles in the predicate POINT.

3.1 Syntactic Features

All syntactic features introduced by Collins (2000) are included for reranking SAPTs. While the full description of all the features is beyond the scope of this paper, we still introduce several feature types here for the convenience of introducing semantic features later.

1. Rules. These are the counts of unique syntactic context-free rules in a SAPT. The example in Figure 3 has the feature f(PRN→ -LRB- NP COMMA NP -RRB-)=1.
2. Bigrams. These are the counts of unique bigrams of syntactic labels in a constituent. They are also featured with the syntactic label of the constituent, and the bigram's relative direction (*left*, *right*) to the head of the constituent. The example in Figure 3 has the feature f(NP COMMA, *right*, PRN)=1.
3. Grandparent Rules. These are the same as Rules, but also include the syntactic label above a rule. The example in Figure 3 has the feature f([PRN→ -LRB- NP COMMA NP -RRB-], NP)=1, where NP is the syntactic label above the rule "PRN→ -LRB- NP COMMA NP -RRB-".
4. Grandparent Bigrams. These are the same as Bigrams, but also include the syntactic label above the constituent containing a bigram. The example in Figure 3 has the feature f([NP COMMA, *right*, PRN], NP)=1, where NP is the syntactic label above the constituent PRN.

3.2 Semantic Features

3.2.1 Semantic Feature Set I

A similar semantic feature type is introduced for each syntactic feature type used by Collins (2000) by replacing syntactic labels with semantic ones (with the semantic label NULL not included). The corresponding semantic feature types for the features in Section 3.1 are:

1. Rules. The example in Figure 3 has the feature f(POINT→ POINT NUM1 NUM2)=1.
2. Bigrams. The example in Figure 3 has the feature f(NUM1 NUM2, *right*, POINT)=1, where the bigram "NUM1 NUM2"appears to the right of the head POINT.
3. Grandparent Rules. The example in Figure 3 has the feature f([POINT→ POINT NUM1 NUM2], POINT)=1, where the last POINT is

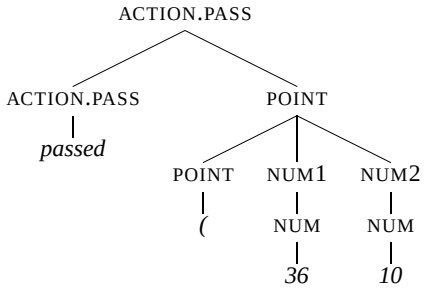


Figure 4: The tree generated by removing purely-syntactic nodes from the SAPT in Figure 3 (with syntactic labels omitted.)

the semantic label above the semantic rule "POINT→ POINT NUM1 NUM2".

4. Grandparent Bigrams. The example in Figure 3 has the feature f([NUM1 NUM2, *right*, POINT], POINT)=1, where the last POINT is the semantic label above the POINT associated with PRN.

3.2.2 Semantic Feature Set II

Purely-syntactic structures in SAPTs exist with no meaning composition involved, such as the expansions from NP to PRN, and from PP to "TO NP" in Figure 3. One possible drawback of the semantic features derived directly from SAPTs as in Section 3.2.1 is that they could include features with no meaning composition involved, which are intuitively not very useful. For example, the nodes with purely-syntactic expansions mentioned above would trigger a semantic rule feature with meaning unchanged (from POINT to POINT). Another possible drawback of these features is that the features covering broader context could potentially fail to capture the real high-level meaning composition information. For example, the Grandparent Rule example in Section 3.2.1 has POINT as the semantic grandparent of a POINT composition, but not the real one ACTION.PASS.

To address these problems, another semantic feature set is introduced by deriving semantic features from trees where purely-syntactic nodes of SAPTs are removed (the resulting tree for the SAPT in Figure 3 is shown in Figure 4). In this tree representation, the example in Figure 4 would have the Grandparent Rule feature f([POINT→ POINT NUM1 NUM2], ACTION.PASS)=1, with the correct semantic grandparent ACTION.PASS included.

4 Experimental Evaluation

4.1 Experimental Methodology

Two corpora of natural language sentences paired with MRs were used in the reranking experiments. For CLANG, 300 pieces of coaching advice were randomly selected from the log files of the 2003 RoboCup Coach Competition. Each formal instruction was translated into English by one of four annotators (Kate et al., 2005). The average length of an natural language sentence in this corpus is 22.52 words. For GEOQUERY, 250 questions were collected by asking undergraduate students to generate English queries for the given database. Queries were then manually translated into logical form (Zelle and Mooney, 1996). The average length of a natural language sentence in this corpus is 6.87 words.

We adopted standard 10-fold cross validation for evaluation: 9/10 of the whole dataset was used for training (training set), and 1/10 for testing (test set). To train a reranking model on a training set, a separate "internal" 10-fold cross validation over the training set was employed to generate n-best SAPTs for each training example using a baseline learner, where each training set was again separated into 10 folds with 9/10 for training the baseline learner, and 1/10 for producing the n-best SAPTs for training the reranker. Reranking models trained in this way ensure that the n-best SAPTs for each training example are not generated by a baseline model that has already seen that example. To test a reranking model on a test set, a baseline model trained on a whole training set was used to generate n-best SAPTs for each test example, and then the reranking model trained with the above method was used to choose a best SAPT from the candidate SAPTs.

The performance of semantic parsing was measured in terms of *precision* (the percentage of completed MRs that were correct), *recall* (the percentage of all sentences whose MRs were correctly generated) and F-measure (the harmonic mean of precision and recall). Since even a single mistake in an MR could totally change the meaning of an example (e.g. having OUR in an MR instead of OPPONENT in CLANG), no partial credit was given for examples with partially-correct SAPTs.

Averaged perceptron (Collins, 2002a), which has been successfully applied to several tagging and parsing reranking tasks (Collins, 2002c; Collins, 2002a), was employed for training rerank-

	CLANG			GEOQUERY		
	P	R	F	P	R	F
SCISSOR	**89.5**	73.7	80.8	**98.5**	74.4	84.8
SCISSOR+	87.0	**78.0**	**82.3**	95.5	**77.2**	**85.4**

Table 2: The performance of the baseline model SCISSOR+ compared with SCISSOR (with the best result in bold), where P = precision, R = recall, and F = F-measure.

n	1	2	5	10	20	50
CLANG	78.0	81.3	83.0	84.0	85.0	85.3
GEOQUERY	77.2	77.6	80.0	81.2	81.6	81.6

Table 3: Oracle recalls on CLANG and GEOQUERY as a function of number *n* of *n*-best SAPTs.

ing models. To choose the correct SAPT of a training example required for training the averaged perceptron, we selected a SAPT that results in the correct MR; if multiple such SAPTs exist, the one with the highest baseline score was chosen. Since no partial credit was awarded in evaluation, a training example was discarded if it had no correct SAPT. Rerankers were trained on the 50-best SAPTs provided by SCISSOR, and the number of perceptron iterations over the training examples was limited to 10. Typically, in order to avoid over-fitting, reranking features are filtered by removing those occurring in less than some minimal number of training examples. We only removed features that never occurred in the training data since experiments with higher cut-offs failed to show any improvements.

4.2 Results

4.2.1 Baseline Results

Table 2 shows the results comparing the baseline learner SCISSOR using both the back-off parameters in Ge and Mooney (2005) (SCISSOR) and the revised parameters in Section 2.2 (SCISSOR+). As we expected, SCISSOR+ has better recall and worse precision than SCISSOR on both corpora due to the additional levels of back-off. SCISSOR+ is used as the baseline model for all reranking experiments in the next section.

Table 3 gives oracle recalls for CLANG and GEOQUERY where an oracle picks the correct parse from the n-best SAPTs if *any* of them are correct. Results are shown for increasing values of n. The trends for CLANG and GEOQUERY are different: small values of n show significant improvements for CLANG, while a larger n is needed to improve results for GEOQUERY.

4.2.2 Reranking Results

In this section, we describe the experiments with reranking models utilizing different feature sets. All models include the score assigned to a SAPT by the baseline model as a special feature.

Table 4 shows results using different feature sets derived directly from SAPTs. In general, reranking improves the performance of semantic parsing on CLANG, but not on GEOQUERY. This could be explained by the different oracle recall trends of CLANG and GEOQUERY. We can see that in Table 3, even a small n can increase the oracle score on CLANG significantly, but not on GEOQUERY. With the baseline score included as a feature, correct SAPTs closer to the top are more likely to be reranked to the top than the ones in the back, thus CLANG is more likely to have more sentences reranked correct than GEOQUERY. On CLANG, using the semantic feature set alone achieves the best improvements over the baseline with 2.8% absolute improvement in F-measure (15.8% relative error reduction), which is significant at the 95% confidence level using a paired Student's t-test. Nevertheless, the difference between SEM_1 and SYN+SEM_1 is very small (only one example). Using syntactic features alone only slightly improves the results because the syntactic features do not directly discriminate between correct and incorrect meaning representations. To put this in perspective, Charniak and Johnson (2005) reported that reranking improves the F-measure of syntactic parsing from 89.7% to 91.0% with a *50*-best oracle F-measure score of 96.8%.

Table 5 compares results using semantic features directly derived from SAPTs (SEM_1), and from trees with purely-syntactic nodes removed (SEM_2). It compares reranking models using these

	CLANG			GEOQUERY		
	P	R	F	P	R	F
SCISSOR+	87.0	78.0	82.3	**95.5**	**77.2**	**85.4**
SYN	87.7	78.7	83.0	**95.5**	**77.2**	**85.4**
SEM_1	**90.0(23.1)**	**80.7(12.3)**	**85.1(15.8)**	95.5	76.8	85.1
$SYN+SEM_1$	89.6	80.3	84.7	95.5	76.4	84.9

Table 4: Reranking results on CLANG and GEOQUERY using different feature sets derived directly from SAPTs (with the best results in bold and relative error reduction in parentheses). The reranking model SYN uses the syntactic feature set in Section 3.1, SEM_1 uses the semantic feature set in Section 3.2.1, and $SYN+SEM_1$ uses both.

	CLANG			GEOQUERY		
	P	R	F	P	R	F
SEM_1	**90.0**	**80.7**	**85.1**	95.5	76.8	85.1
SEM_2	88.1	79.0	83.3	**96.0**	**77.2**	**85.6**
SEM_1+SEM_2	88.5	79.3	83.7	95.5	76.4	84.9
$SYN+SEM_1$	89.6	80.3	84.7	95.5	76.4	84.9
$SYN+SEM_2$	88.1	79.0	83.3	95.5	76.8	85.1
$SYN+SEM_1+SEM_2$	88.9	79.7	84.0	95.5	76.4	84.9

Table 5: Reranking results on CLANG and GEOQUERY comparing semantic features derived directly from SAPTs, and semantic features from trees with purely-syntactic nodes removed. The symbol SEM_1 and SEM_2 refer to the semantic feature sets in Section 3.2.1 and 3.2.1 respectively, and SYN refers to the syntactic feature set in Section 3.1.

feature sets alone and together, and using them along with the syntactic feature set (SYN) alone and together. Overall, SEM_1 provides better results than SEM_2 on CLANG and slightly worse results on GEOQUERY (only in one sentence), regardless of whether or not syntactic features are included. Using both semantic feature sets does not improve the results over just using SEM_1. On one hand, the better performance of SEM_1 on CLANG contradicts our expectation because of the reasons discussed in Section 3.2.2; the reason behind this needs to be investigated. On the other hand, however, it also suggests that the semantic features derived directly from SAPTs can provide good evidence for semantic correctness, even with redundant purely syntactically motivated features.

We have also informally experimented with smoothed semantic features utilizing domain ontology given by CLANG, which did not show improvements over reranking models not using these features.

5 Conclusion

We have applied discriminative reranking to semantic parsing, where reranking features are developed from features for reranking syntactic parses based on the coupling of syntax and semantics. The best reranking model significantly improves F-measure on a Robocup coaching task (CLANG) from 82.3% to 85.1%, while it fails to improve the performance on a geography database query task (GEOQUERY).

Future work includes further investigation of the reasons behind the different utility of reranking for the CLANG and GEOQUERY tasks. We also plan to explore other types of reranking features, such as the features used in semantic role labeling (SRL) (Gildea and Jurafsky, 2002; Carreras and Màrquez, 2005), like the path between a target predicate and its argument, and kernel methods (Collins, 2002b). Experimenting with other effective reranking algorithms, such as SVMs (Joachims, 2002) and MaxEnt (Charniak and Johnson, 2005), is also a direction of our future research.

6 Acknowledgements

We would like to thank Rohit J. Kate and anonymous reviewers for their insightful comments. This research was supported by Defense Ad-

vanced Research Projects Agency under grant HR0011-04-1-0007.

References

Xavier Carreras and Luís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proc. of 9th Conf. on Computational Natural Language Learning (CoNLL-2005)*, pages 152–164, Ann Arbor, MI, June.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proc. of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL-05)*, pages 173–180, Ann Arbor, MI, June.

Mao Chen, Ehsan Foroughi, Fredrik Heintz, Spiros Kapetanakis, Kostas Kostiadis, Johan Kummeneje, Itsuki Noda, Oliver Obst, Patrick Riley, Timo Steffens, Yi Wang, and Xiang Yin. 2003. Users manual: RoboCup soccer server manual for soccer server version 7.07 and later. Available at `http://sourceforge.net/projects/sserver/`.

Michael J. Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proc. of the 35th Annual Meeting of the Association for Computational Linguistics (ACL-97)*, pages 16–23.

Michael Collins. 2000. Discriminative reranking for natural language parsing. In *Proc. of 17th Intl. Conf. on Machine Learning (ICML-2000)*, pages 175–182, Stanford, CA, June.

Michael Collins. 2002a. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proc. of the 2002 Conf. on Empirical Methods in Natural Language Processing (EMNLP-02)*, Philadelphia, PA, July.

Michael Collins. 2002b. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, pages 263–270, Philadelphia, PA, July.

Michael Collins. 2002c. Ranking algorithms for named-entity extraction: Boosting and the voted perceptron. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, pages 489–496, Philadelphia, PA.

Brooke Cowan and Michael Collins. 2005. Morphology and reranking for the statistical parsing of Spanish. In *Proc. of the Human Language Technology Conf. and Conf. on Empirical Methods in Natural Language Processing (HLT/EMNLP-05)*, Vancouver, B.C., Canada, October.

Ruifang Ge and Raymond J. Mooney. 2005. A statistical semantic parser that integrates syntax and semantics. In *Proc. of 9th Conf. on Computational Natural Language Learning (CoNLL-2005)*, pages 9–16, Ann Arbor, MI, July.

Daniel Gildea and Daniel Jurafsky. 2002. Automated labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Nancy A. Ide and Jeanéronis. 1998. Introduction to the special issue on word sense disambiguation: The state of the art. *Computational Linguistics*, 24(1):1–40.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proc. of 8th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining (KDD-2002)*, Edmonton, Canada.

Daniel Jurafsky and James H. Martin. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, Upper Saddle River, NJ.

R. J. Kate, Y. W. Wong, and R. J. Mooney. 2005. Learning to transform natural to formal languages. In *Proc. of 20th Natl. Conf. on Artificial Intelligence (AAAI-2005)*, pages 1062–1068, Pittsburgh, PA, July.

Gregory Kuhlmann, Peter Stone, Raymond J. Mooney, and Jude W. Shavlik. 2004. Guiding a reinforcement learner with natural language advice: Initial results in RoboCup soccer. In *Proc. of the AAAI-04 Workshop on Supervisory Control of Learning and Adaptive Systems*, San Jose, CA, July.

Iddo Lev, Bill MacCartney, Christopher D. Manning, and Roger Levy. 2004. Solving logic puzzles: From robust processing to precise semantics. In *Proc. of 2nd Workshop on Text Meaning and Interpretation, ACL-04*, Barcelona, Spain.

Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. 2005. Joint learning improves semantic role labeling. In *Proc. of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL-05)*, Ann Arbor, MI, June.

John M. Zelle and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proc. of 13th Natl. Conf. on Artificial Intelligence (AAAI-96)*, pages 1050–1055, Portland, OR, August.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proc. of 21th Conf. on Uncertainty in Artificial Intelligence (UAI-2005)*, Edinburgh, Scotland, July.

Victor W. Zue and James R. Glass. 2000. Conversational interfaces: Advances and challenges. In *Proc. of the IEEE*, volume 88(8), pages 1166–1180.

Multilingual Lexical Database Generation from parallel texts in 20 European languages
with endogenous resources

GIGUET EMMANUEL
GREYC CNRS UMR 6072
Université de Caen
14032 Caen Cedex – France
giguet@info.unicaen.fr

LUQUET Pierre-Sylvain
GREYC CNRS UMR 6072
Université de Caen
14032 Caen Cedex – France
psluquet@info.unicaen.fr

Abstract

This paper deals with multilingual database generation from parallel corpora. The idea is to contribute to the enrichment of lexical databases for languages with few linguistic resources. Our approach is endogenous: it relies on the raw texts only, it does not require external linguistic resources such as stemmers or taggers. The system produces alignments for the 20 European languages of the 'Acquis Communautaire' Corpus.

1 Introduction

1.1 Automatic processing of bilingual and multilingual corpora

Processing bilingual and multilingual corpora constitutes a major area of investigation in natural language processing. The linguistic and translational information that is available make them a valuable resource for translators, lexicographers as well as terminologists. They constitute the nucleus of example-based machine translation and translation memory systems.

Another field of interest is the constitution of multilingual lexical databases such as the project planned by the European Commission's Joint Research Centre (JRC) or the more established Papillon project. Multilingual lexical databases are databases for structured lexical data which can be used either by humans (e.g. to define their own dictionaries) or by natural language processing (NLP) applications.

Parallel corpora are freely available for research purposes and their increasing size demands the exploration of automatic methods. The 'Acquis Communautaire' (AC) Corpus is such a corpus. Many research teams are involved in the JRC project for the enrichment of a multilingual lexical database. The aim of the project is to reach an automatic extraction of lexical tuples from the AC Corpus.

The AC document collection was constituted when ten new countries joined the European Union in 2004. They had to translate an existing collection of about ten thousand legal documents covering a large variety of subject areas. The 'Acquis Communautaire' Corpus exists as a parallel text in 20 languages. The JRC has collected large parts of this document collection, has converted it to XML, and provide sentence alignments for most language pairs (Steinberger et al., 2006).

1.2 Alignment approaches

Alignment becomes an important issue for research on bilingual and multilingual corpora. Existing alignment methods define a continuum going from purely statistical methods to linguistic ones. A major point of divergence is the granularity of the proposed alignments (entire texts, paragraphs, sentences, clauses, words) which often depends on the application.

In a coarse-grained alignment task, punctuation or formatting can be sufficient. At finer-grained levels, methods are more sophisticated and combine linguistic clues with statistical ones. Statistical alignment methods at sentence level have been thoroughly investigated (Gale & Church, 1991a/ 1991b ; Brown et al., 1991 ; Kay & Röscheisen, 1993). Others use various linguistic information (Simard et al., 1992 ; Papageorgiou et al., 1994). Purely statistical alignment methods are proposed at word level (Gale & Church, 1991a ; Kitamura & Matsumoto, 1995). (Tiedemann, 1993 ; Boutsis & Piperidis, 1996 ; Piperidis et al., 1997) combine statistical and linguistic information for the same task. Some methods make alignment suggestions at an intermediate level between sentence and word

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 271–278,
Sydney, July 2006.

and word (Smadja, 1992 ; Smadja et al., 1996 ; Kupiec, 1993 ; Kumano & Hirakawa, 1994 ; Boutsis & Piperidis, 1998).

A common problem is the delimitation and spotting of the units to be matched. This is not a real problem for methods aiming at alignments at a high level of granularity (paragraphs, sentences) where unit delimiters are clear. It becomes more difficult for lower levels of granularity (Simard, 2003), where correspondences between graphically delimited words are not always satisfactory.

2 The multi-grained endogenous alignment approach

The approach proposed here deals with the spotting of *multi-grained* translation equivalents. We do not adopt very rigid constraints concerning the size of linguistic units involved, in order to account for the flexibility of language and translation divergences. Alignment links can then be established at various levels, from sentences to words and obeying no other constraints than the maximum size of candidate alignment sequences and their minimum frequency of occurrence.

The approach is endogenous since the input is used as the only used linguistic resource. It is the multilingual parallel AC corpus itself. It does not contain any syntactical annotation, and the texts have not been lemmatised. In this approach, no classical linguistic resources are required. The input texts have been segmented and aligned at sentence level by the JRC. Inflectional divergencies of isolated words are taken into account without external linguistic information (lexicon) and without linguistic parsers (stemmer or tagger). The morphology is learnt automatically using an endogenous parsing module integrated in the alignment tool based on (Déjean, 1998).

We adopt a minimalist approach, in the line of GREYC. In the JRC project, many languages do not have available linguistic resources for automatic processing, neither inflectional or syntactical annotation, nor surface syntactic analysis or lexical resources (machine-readable dictionaries etc.). Therefore we can not use a large amount of a priori knowledge on these languages.

3 Considerations on the Corpus

3.1 Corpus definition

Concretely, the texts constituting the AC corpus (Steinberger et al., 2006) are legal documents translated in several languages and aligned at sentence level. Here is a description of the parallel corpus, in the 20 languages available:

- Czech: 7106 documents
- Danish: 8223 documents
- German: 8249 documents
- Greek: 8003 documents
- English: 8240 documents
- Spanish: 8207 documents
- Estonian: 7844 documents
- Finnish: 8189 documents
- French: 8254 documents
- Hungarian: 7535 documents
- Italian: 8249 documents,
- Lithuanian: 7520 documents
- Latvian: 7867 documents
- Maltese: 6136 documents
- Dutch: 8247 documents
- Polish: 7768 documents
- Portuguese: 8210 documents
- Slovakian: 6963 documents
- Slovene:7821 documents
- Swedish: 8233 documents

The documents contained in the archives are XML files, UTF-8 encoding, containing information on "sentence" segmentation. Each file is stamped with a unique identifier (the celex identifier). It refers to a unique document. Here is an excerpt of the document 31967R0741, in Czech.

```
<document celex="31967R0741" lang="cs"
    ver="1.0">
  <title>
    <P sid="1">NAŘÍZENÍ RADY č.
      741/67/EHS ze dne 24. října
      1967 o příspěvcích ze zá-
      ruční sekce Evropského
      orientačního a záručního
      fondu</P>
  </title>
  <text>
    <P sid="2">NAŘÍZENÍ RADY č.
      741/67/EHS</P>
    <P sid="3">ze dne 24. října
      1967</P>
    <P sid="4">o příspěvcích ze zá-
      ruční sekce Evropského
      orientačního a záručního
      fondu</P>
    <P sid="5">RADA EVROPS-
      KÝCH SPOLEČENST-
      VÍ,</P>
    <P sid="6">s ohledem na Smlou-
      vu o založení Evropského
      hospodářského společenst-
      ví, a zejména na článek 43
      této smlouvy,</P>
    <P sid="7">s ohledem na návrh
      Komise,</P>
    <P sid="8">s ohledem na stano-
      visko Shromáždění1,</P>
```

```
<P sid="9">vzhledem k tomu, že
    zavedením režimu jednot-
    ných a povinných náhrad při
    vývozu do třetích zemí od
    zavedení jednotné organiza-
    ce trhu pro zemědělské pro-
    dukty, jež ve značné míře
    existuje od 1. července
    1967, vyšlo kritérium nejnižší
    průměrné náhrady stanove-
    né pro financování náhrad
    podle čl. 3 odst. 1 písm. a)
    nařízení č. 25 o financování
    společné zemědělské poli-
    tiky2 z používání;</P>
[…]
```

Sentence alignments files are also provided with the corpus for 111 language pairs. The XML files encoded in UTF-8 are about 2M packed and 10M unpacked. Here is an excerpt of the alignment file of the document 31967R0741, for the language pair Czech-Danish.

```
<document celexid="31967R0741">
  <title1>NAŘÍZENÍ RADY č.
      741/67/EHS ze dne 24. října 1967
      o příspěvcích ze záruční sekce Ev-
      ropského orientačního a záručního
      fondu</title1>
  <title2>Raadets forordning nr.
      741/67/EOEF af 24. oktober 1967
      om stoette fra Den europaeiske
      Udviklings- og Garantifond for
      Landbruget, garantisek-
      tionen</title2>
  <link type="1-2" xtargets="2;2 3" />
  <link type="1-1" xtargets="3;4" />
  <link type="1-1" xtargets="4;5" />
  <link type="1-1" xtargets="5;6" />
   […]
  <link type="1-1" xtargets="49;53" />
  <link type="2-1" xtargets="50 51;54" />
  <link type="1-1" xtargets="52;55" />
</document>
```

In this file, the xtargets "ids" refer to the <P sid="…"> of the Czech and Danish translations of the document 31967R0741.

The current version of our alignment system deals with one language pair at a time, whatever the languages are. The algorithm takes as input a corpus of bitexts aligned at sentence level. Usually, the alignment at this level outputs aligned windows containing from 0 to 2 segments. One-to-one mapping corresponds to a standard output (see link types "1-1" above). An empty window corresponds to a case of addition in the source language or to a case of omission in the target language. One-to-two mapping corresponds to split sentences (see link types "1-2" and "2-1" above).

Formally, each bitext is a quadruple < T1, T2, Fs, C> where T1 and T2 are the two texts, Fs is the function that reduces T1 to an element set Fs(T1) and also reduces T2 to an element set Fs(T2), and C is a subset of the Cartesian product of Fs(T1) x Fs(T2) (Harris, 1988).

Different standards define the encoding of parallel text alignments. Our system natively handles TMX and XCES format, with UTF-8 or UTF-16 encoding.

4 The Resolution Method

The resolution method is composed of two stages, based on two underlying hypotheses. The first stage handles the document grain. The second stage handles the corpus grain.

4.1 Hypotheses

hypothesis 1 : let's consider a bitext composed of the texts T_1 and T_2. If a sequence S_1 is repeated several times in T_1 and in well-defined sentences[1], there are many chances that a repeated sequence S_2 corresponding to the translation of S_1 occurs in the corresponding aligned sentences in T_2.

hypothesis 2 : let's consider a corpus of bitexts, composed of two languages L_1 and L_2. There is no guarantee for a sequence S_1 which is repeated in many texts of language L_1 to have a unique translation in the corresponding texts of language L_2.

4.2 Stage 1 : Bitext analysis

The first stage handles the document scale. Thus it is applied on each document, individually. There is no interaction at the corpus level.

Determining the multi-grained sequences to be aligned

First, we consider the two languages of the document independently, the source language L_1 and the target language L_2. For each language, we compute the repeated sequences as well as their frequency.

The algorithm based on suffix arrays does not retain the sub-sequences of a repeated sequence if they are as frequent as the sequence itself. For instance, if "*subjects*" appears with the same frequency than "*healthy subjects*" we retain only the second sequence. On the contrary, if "*disease*" occurs more frequently than "*thyroid disease*" we retain both.

[1] Here, « sentences » can be generalized as « textual segments »

When computing the frequency of a repeated sequence, the offset of each occurrence is memorized. So the output of this processing stage is a list of sequences with their frequency and the offset list in the document.

"thyroid cancer": list of segments where the sequence appears
45, 46, 46, 48, 51, 51, …

Handling inflections

Inflectional divergencies of isolated words are taken into account without external linguistic information (lexicon) and without linguistic parsers (stemmer or tagger). The morphology is learnt automatically using an endogenous approach derived from (Déjean, 1998). The algorithm is reversible: it allows to compute prefixes the same way, with reversed word list as input.

The basic idea is to approximate the border between the nucleus and the suffixes. The border matches the position where the number of distinct letters preceding a suffix of length n is greater than the number of distinct letters preceding a suffix of length n-1.

For instance, in the first English document of our corpus, "*g*" is preceded by 4 distinct letters, "*ng*" by 2 and "*ing*" by 10: "*ing*" is probably a suffix. In the first Greek document, "*ά*" is preceded by 5 letters, "*κά*" by 1 and "*ικά*" by 10. "*ικά*" is probably a suffix.

The algorithm can generate some wrong morphemes, from a strictly linguistic point of view. But at this stage, no filtering is done in order to check their validity. We let the alignment algorithm do the job with the help of contextual information.

Vectorial representation of the sequences

An orthonormal space is then considered in order to explore the existence of possible translation relations between the sequences, and in order to define translation couples. The existence of translation relations between sequences is approximated by the cosine of vectors associated to them, in this space.

The links in the alignment file allow the construction of this orthonormal space. This space has n_o dimensions, where n_o is the number of non-empty links. Alignment links with empty sets (type="0-?" or type="?-0") corresponds to cases of omission or addition in one language.

Every repeated sequence is seen as a vector in this space. For the construction of this vector, we first pick up the segment offset in the document for each repeated sequence.

"thyroid cancer": list of segments where the sequence appears
45, 46, 46, 48, 51, 51

Then we convert this list in a n_L-dimension vector v_L, where n_L is the number of textual segments of the document of language L. Each dimension contains the number of occurrences present in the segment.

"*thyroid cancer*" : *associated with a vector of n_L dimensions*.

1	2	…	**45**	**46**	47	48	49	50	51	…	n_L
0	0		1	2	0	1	0	0	2		0

With the help of the alignment file, we can now make the projection of the vector v_L in the n_o-dimension vector v_o. For instance, if the link <link type="2-1" xtargets="45 46;45" /> is located at rank r=40 in the alignment file and if English is the first language (L=*en*), then $v_o[40] = v_{en}[45] + v_{en}[46]$.

Sequence alignment

For each sequence of L_1 to be aligned, we look for the existence of a translation relation between it and every L_2 sequence to be aligned. The existence of a translation relation between two sequences is approximated by the cosine of the vectors associated to them.

The cosine is a mathematical tool used in in Natural Language Processing for various purposes, e.g. (Roy & Beust, 2004) uses the cosine for thematic categorisation of texts. The cosine is obtained by dividing the scalar product of two vectors with the product of their norms.

$$\cos(x_i,y_i)=\frac{\sum x_i \cdot y_i}{\sqrt{\sum x_i^2} \times \sqrt{\sum y_i^2}}$$

We note that the cosine is never negative as vectors coordinates are always positive. The sequences proposed for the alignment are those that obtain the largest cosine. We do not propose an alignment if the best cosine is inferior to a certain threshold.

4.3 Stage 2 : Corpus management

The second stage handles the corpus grain and merges the information found at document grain, in the first stage.

Handling the Corpus Dimension

The bitext corpus is not a bag of aligned sentences and is not considered as if it were. It is a bag of bitexts, each bitext containing a bag of aligned sentences.

Considering the bitext level (or document grain) is useful for several reasons. First, for operational sake. The greedy algorithm for repeated sequence extraction has a cubic complexity. It is better to apply it on the document unit rather than on the corpus unit. But this is not the main reason.

Second, the alignment algorithm between sequences relies on the principle of translation coherence: a repeated sequence in L1 has many chances to be translated by the same sequence in L2 in the same text. This hypothesis holds inside the document but not in the corpus: a polysemic term can be translated in different ways according to the document genre or domain.

Third, the confidence in the generated alignments is improved if the results obtained by the execution of the process on several documents share compatible alignments.

Alignment Filtering and Ranking

The filtering process accepts terms which have been produced (1) by the execution on at least two documents, (2) by the execution on solely one document if the aligned terms correspond to the same character string or if the frequency of the terms is greater than an empirical threshold function. This threshold is proportional to the inverse term length since there are fewer complex repeated terms than simple terms.

The ranking process sorts candidates using the product of the term frequency by the number of output agreements.

5 Results

The results concern an alignment task between English and the 19 other languages of the AC-Corpus. For each language pair, we considered 500 bitexts of the AC Corpus. We join in annexes A, B, and C some sample of this results. Annex A deals with English-French parallel texts, Annex B deals with English-Spanish parallel texts and finally Annex C deals with English-German ones. We discuss in the following lines of the English-French alignment.

Among the correct alignments, we find domain dependant lexical terms:

- legal terms of the EEC (*EEC initial verification /vérification primitive CEE, Regulation (EEC) No/règlement (CEE) n°*),
- specialty terms (*rear-view mirrors / rétroviseurs, poultry/volaille*).

We also find invariant terms (*km/h/km/h, kg/kg, mortem/mortem*).

We encounter alignments at different grain*:* territory/territoire Member States/États membres, Whereas/Considérant que, fresh poultrymeat/viandes fraîches de volaille, Having regard to the Opinion of the/vu l'avis.

The wrong alignments mainly come from candidates that have not been confirmed by running on several documents (column ndoc=1): *on/la commercialisation des.*

A permanent dedicated web site will be open in March 2006 to detail all the results for each language pair. The URL is http://users.info.unicaen.fr/~giguet/alignment.

5.1 Discussion

First, the results are similar to those obtained on the Greek/English scientific corpus.

Second, it is sometimes difficult to choose between distinct proposals for a same term when the grain vary: *Member/membre~* Member State~/membre~ *Member States/États membres State/membre State~/membre~*. There is a problem both in the definition of terms and in the ability of an automatic process to choose between the components of the terms.

Third, thematic terms of the corpus are not always aligned, since they are not repeated. Corefence is used instead, thanks to nominal anaphora, acronyms, and also lexical reductions. Accuracy depends on the document domain. In the medical domain, acronyms are aligned but not their expansion. However, we consider that this problem has to be solved by an anaphora resolution system, not by this alignment algorithm.

6 Conclusion

We showed that it is possible to contribute to the processing of languages for which few linguistic resources are available. We propose a solution to the spotting of multi-grained translation from parallel corpora. The results are surprisingly good and encourage us to improve the method, in order to reach a semi-automatic construction of a multilingual lexical database.

The endogenous approach allows to handle inflectional variations. We also show the importance of using the proper knowledge at the proper level (sentence grain, document grain and corpus grain). An improvement would be to calculate inflectional variations at corpus grain rather than at document grain. Therefore, it is possible to plug any external and exogenous component in our architecture to improve the overall quality.

The size of this "massive compilation" (we work with a 20 languages corpora) implies the design of specific strategies in order to handle it properly and quite efficiently. Special efforts have been done in order to manage the AC Corpus from our document management platform, WIMS.

The next improvement is to precisely evaluate the system. Another perspective is to integrate an endogenous coreference solver (Giguet & Lucas, 2004).

References

Altenberg B. & Granger, S. 2002. *Recent trends in cross-linguistic lexical studies.* In *Lexis in Conrast,* Altenberg & Granger (eds.).

Boutsis, S., & Piperidis, S. 1998. *Aligning clauses in parallel texts.* In *Third Conference on Empirical Methods in Natural Language Processing,* 2 June, Granada, Spain, p. 17-26.

Brown P., Lai J. & Mercer R. 1991. *Aligning sentences in parallel corpora*. In *Proc. 29th Annual Meeting of the Association for Computational Linguistics*, p. 169-176, 18-21 June, Berkley, California.

Déjean H. 1998. *Morphemes as Necessary Concept for Structures Discovery from Untagged Corpora.* In Workshop on Paradigms and Grounding in Natural Language Learning, pages 295-299, PaGNLL Adelaide.

Gale W.A. & K.W. Church. 1991a. *Identifying word correspondences in parallel texts*. In *Fourth DARPA Speech and Natural Language Workshop*, p. 152-157. San Mateo, California: Morgan Kaufmann.

Gale W.A. & Church K. W. 1991b. *A Program for Aligning Sentences in Bilingual Corpora.* In *Proc. 29th Annual Meeting of the Association for Computational Linguistics*, p. 177-184, 18-21 June, Berkley, California.

Giguet E. & Apidianaki M. 2005. *Alignement d'unités textuelles de taille variable*. Journée Internationales de la Linguistique de Corpus. Lorient.

Giguet E. 2005. *Multi-grained alignment of parallel texts with endogenous resources*. RANLP'2005 Workshop "Crossing Barriers in Text Summarization Research". Borovets, Bulgaria.

Giguet E. & Lucas N. 2004. *La détection automatique des citations et des locuteurs dans les textes informatifs*. In *Le discours rapporté dans tous ses états : Question de frontières*, J. M. López-Muñoz S. Marnette, L. Rosier, (eds.). Paris, l'Harmattan, pp. 410-418.

Harris B. *Bi-text, a New Concept in Translation Theory,* Language Monthly (54), p. 8-10, 1998.

Isabelle P. & Warwick-Armstrong S. 1993. *Les corpus bilingues: une nouvelle ressource pour le traducteur.* In Bouillon, P. & Clas A. (eds.), *La Traductique : études et recherches de traduction par ordinateur.* Montréal : Les Presses de l'Université de Montréal, p. 288-306.

Kay M. & Röscheisen M. 1993. *Text-translation alignment*. Computational Linguistics, p.121-142, March.

Kitamura M. & Matsumoto Y. 1996. *Automatic extraction of word sequence correspondences in parallel corpora*. In *Proc. 4th Workshop on Very Large Corpora*, p. 79-87. Copenhagen, Denmark, 4 August.

Kupiec J. 1993. *An algorithm for Finding Noun Phrase Correspondences in Bilingual Corpora*, *Proceedings of the 31st Annual Meeting of the Association of Computational Linguistics*, p. 23-30.

Papageorgiou H., Cranias L. & Piperidis S. 1994. *Automatic alignment in parallel corpora*. In *Proceed. 32nd Annual Meeting of the Association for Computational Linguistics*, p. 334-336, 27-30 June, Las Cruses, New Mexico.

Salkie R. 2002. *How can linguists profit from parallel corpora?,* In *Parallel Corpora, Parallel Worlds: selected papers from a symposium on parallel and comparable corpora at Uppsala University, Sweden, 22-23 April, 1999,* Lars Borin (ed.), Amsterdam, New York: Rodopi, p. 93-109.

Simard M., Foster G., & Isabelle P. , 1992*Using cognates to align sentences in bilingual corpora*. In *Proceedings of TMI-92*, Montréal, Québec.

Simard M. 2003. *Mémoires de Traduction sous-phrastiques.* Thèse de l'Université de Montréal.

Smadja F. 1992. *How to compile a bilingual collocational lexicon automatically.* In *Proceedings of the AAAI-92 Workshop on Statistically -based NLP Techniques*.

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

Ralf Steinberger, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaž Erjavec, Dan Tufiş, Alexander Ceausu & Dániel Varga. *The JRC-Acquis: A multilingual aligned parallel corpus with 20+ Languages*. Proceedings of LREC'2006.

Tiedemann J. 1993. *Combining clues for word alignment*. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, p. 339-346, Budapest, Hungary, April2003.

ANNEX A: Some alignments on 20 English-French documents

source	ndoc	freq	target
and	12	[336]	et|
Member	10	[206]	membre~|
Member State~	10	[201]	membre~|
Member States	13	[143]	**États membres**|
the	4	[392]	d~|
of	5	[313]	de~|
EEC	9	[118]	CEE|
3	8	[41]	3|
Annex	7	[42]	l'annexe|
State	4	[71]	membre|
Whereas	10	[28]	**considérant que**|
Member State	4	[63]	membre|
EEC pattern approval	4	[35]	CEE de modèle|
verification	4	[34]	vérification|
Council Directive	9	[15]	Conseil|
EEC initial verification	5	[27]	vérification primitive CEE|
Having regard to the Opinion of the	8	[16]	**vu l'avis**|
THE	8	[16]	DES|
certain	3	[11]	certain~|
marks	3	[11]	marques|
mark	4	[8]	la marque|
directive	2	[16]	directive particulière|
trade	2	[16]	échanges|
pattern approval	1	[31]	de modèle|
pattern approval~	1	[31]	de modèle|
4~	5	[6]	4|
12	3	[10]	12|
approximat~	3	[10]	rapprochement|
certificate	3	[10]	certificat|
device~	3	[10]	dispositif~|
other	3	[10]	autres que|
for liquid~	2	[15]	de liquides|
July	3	[9]	juillet|
competent	2	[13]	compétent~|
this Directive	2	[13]	la présente directive|
relat~	3	[8]	relativ~|
26 July 1971	4	[6]	du 26 juillet 1971|
procedure	2	[12]	procédure|
on	1	[23]	la commercialisation des|
fresh poultrymeat	1	[23]	**viandes fraîches de volaille**|
into force	3	[7]	en vigueur|
symbol~	3	[7]	marque~|
the word~	1	[21]	mot~|
p~	1	[21]	masse|
subject to	3	[7]	font l'objet|
initial verification	1	[20]	vérification primitive CEE|
Directive~	1	[20]	directiv~|
two	4	[5]	deux|
material	1	[19]	de multiplication|
mass~	1	[19]	à l'hectolitre|
type-approv~	1	[19]	CEE|
than	2	[9]	autres que|
weight	1	[18]	poids|
amendments to	2	[9]	les modifications|

ANNEX B: Some alignments on 250 English-Spanish documents

source	ndoc	freq	target
and	174	[4462]	y|
article	162	[3008]	artículo|
.	134	[5482]	.|
3	118	[982]	3|
whereas	114	[714]	considerando que|
regulation	97	[1623]	reglamento|
the commission	94	[919]	la comisión|
or	92	[2018]	o|
having regard to the opinion of the	90	[180]	visto el dictamen del|
directive	88	[1087]	directiva|
this directive	86	[576]	la presente directiva|
annex	63	[380]	anexo|
member states	59	[1002]	estados miembros|
5	56	[296]	5|
article 1	56	[166]	artículo 1|
the treaty	54	[354]	tratado|
this regulation	54	[191]	el presente reglamento|
of the european communities	54	[189]	de las comunidades europeas|
member state	40	[1006]	estado miembro|
(a)	38	[334]	a)|
this	37	[256]	la presente directiva|
having regard to	37	[98]	visto el|
votes	19	[40]	votos|
"	18	[309]	"|

months	18	[95]	meses|
ii	18	[92]	ii|
b	17	[299]	b|
conditions	17	[169]	condiciones|
market	17	[126]	mercado|
(d)	17	[74]	d)|
1970	17	[63]	de 1970|
, and in particular	17	[37]	y , en particular ,|
agreement	16	[149]	acuerdo|
(e)	16	[64]	e)|
council directive	16	[57]	del consejo|
article 7	16	[46]	artículo 7|
in order	16	[32]	de ello|
no	15	[141]	n º|
eec	15	[140]	cee|
vehicle	15	[115]	vehículo|
a member state	15	[87]	un estado miembro|
14	15	[75]	14|
a	14	[104]	un|
each	14	[91]	cada|
two	14	[83]	dos|
methods	14	[80]	métodos|
if	14	[72]	si|
june	14	[71]	de junio de|
: (a)	14	[66]	a)|

ANNEX C: Some alignments on 250 English-German documents

source	ndoc	freq	target
artikel	106	[1536]	article|
2	98	[1184]	2|
und	93	[2265]	and|
kommission	91	[848]	the commission|
europäischen	89	[331]	the european|
oder	76	[1722]	or|
nach stellungnahme des	73	[146]	having regard to the opinion of the|
der europäischen	65	[303]	the european|
verordnung	59	[871]	regulation|
mitgliedstaaten	58	[888]	member states|
richtlinie	57	[682]	directive|
artikel 1	51	[170]	article 1|
der europäischen gemeinschaften	44	[147]	of the european communities|
der	41	[1679]	the|
6	41	[197]	6|
verordnung (ewg) nr .	40	[231]	regulation (eec) no|
artikel 2	38	[122]	article 2|
gestützt auf	35	[78]	having regard to|
insbesondere	29	[136]	in particular|
artikel 4	29	[99]	article 4|
artikel 3	27	[80]	article 3|
:	26	[251]	:|
auf vorschlag der kommission	26	[104]	proposal from the commission|
rat	25	[205]	the council|
der europäischen wirtschaftsgemeinschaft	25	[81]	the european economic community|
maßnahmen	20	[160]	measures|
7	20	[85]	7|
technischen	19	[64]	technical|
artikel 5	19	[61]	article 5|
hat	19	[51]	has|
.	17	[826]	.|
(3)	17	[122]	3 .|
8	16	[78]	8|
d)	16	[74]	(d)|
des vertrages	15	[122]	of the treaty|
ii	15	[92]	ii|
stellungnahme	15	[70]	opinion|
, s .	15	[62]	, p .|
. "	14	[124]	. "|
. juni	14	[81]	june|
anhang	14	[76]	annex|
nur	14	[75]	only|
nicht	14	[65]	not|
11	14	[46]	11|
, daß	14	[40]	that|
artikel 7	14	[39]	article 7|
zwischen	13	[69]	between|
geändert	11	[44]	amended|
auf	11	[36]	having regard to the|
, insbesondere	11	[28]	in particular|
, insbesondere auf	11	[23]	thereof ;|
gemeinsamen	11	[22]	a single|
behörden	10	[91]	authorities|
verordnung nr .	10	[53]	regulation no|
1970	10	[49]	1970|
der gemeinschaft	10	[47]	the community|

Factoring Synchronous Grammars By Sorting

Daniel Gildea
Computer Science Dept.
University of Rochester
Rochester, NY 14627

Giorgio Satta
Dept. of Information Eng'g
University of Padua
I-35131 Padua, Italy

Hao Zhang
Computer Science Dept.
University of Rochester
Rochester, NY 14627

Abstract

Synchronous Context-Free Grammars (SCFGs) have been successfully exploited as translation models in machine translation applications. When parsing with an SCFG, computational complexity grows exponentially with the length of the rules, in the worst case. In this paper we examine the problem of factorizing each rule of an input SCFG to a generatively equivalent set of rules, each having the smallest possible length. Our algorithm works in time $O(n \log n)$, for each rule of length n. This improves upon previous results and solves an open problem about recognizing permutations that can be factored.

1 Introduction

Synchronous Context-Free Grammars (SCFGs) are a generalization of the Context-Free Grammar (CFG) formalism to simultaneously produce strings in two languages. SCFGs have a wide range of applications, including machine translation, word and phrase alignments, and automatic dictionary construction. Variations of SCFGs go back to Aho and Ullman (1972)'s Syntax-Directed Translation Schemata, but also include the Inversion Transduction Grammars in Wu (1997), which restrict grammar rules to be binary, the synchronous grammars in Chiang (2005), which use only a single nonterminal symbol, and the Multitext Grammars in Melamed (2003), which allow independent rewriting, as well as other tree-based models such as Yamada and Knight (2001) and Galley et al. (2004).

When viewed as a rewriting system, an SCFG generates a set of string pairs, representing some translation relation. We are concerned here with the time complexity of parsing such a pair, according to the grammar. Assume then a pair with each string having a maximum length of N, and consider an SCFG G of size $|G|$, with a bound of n nonterminals in the right-hand side of each rule in a single dimension, which we call below the **rank** of G. As an upper bound, parsing can be carried out in time $O(|G|\, N^{n+4})$ by a dynamic programming algorithm maintaining continuous spans in one dimension. As a lower bound, parsing strategies with discontinuous spans in both dimensions can take time $\Omega(|G|\, N^{c\sqrt{n}})$ for unfriendly permutations (Satta and Peserico, 2005). A natural question to ask then is: What if we could reduce the rank of G, preserving the generated translation? As in the case of CFGs, one way of doing this would be to factorize each single rule into several rules of rank strictly smaller than n. It is not difficult to see that this would result in a new grammar of size at most $2 \cdot |G|$. In the time complexities reported above, we see that such a size increase would be more than compensated by the reduction in the degree of the polynomial in N. We thus conclude that a reduction in the rank of an SCFG would result in more efficient parsing algorithms, for most common parsing strategies.

In the general case, normal forms with bounded rank are not admitted by SCFGs, as shown in (Aho and Ullman, 1972). Nonetheless, an SCFG with a rank of n may not necessarily meet the worst case of Aho and Ullman (1972). It is then reasonable to ask if our SCFG G can be factorized, and what is the smallest rank $k < n$ that can be obtained in this way. This paper answers these two questions, by providing an algorithm that factorizes the rules of an input SCFG, resulting in a new, generatively equivalent, SCFG with rank k as low as possible. The algorithm works in time $O(n \log n)$ for each rule, regardless of the rank k of the factorized rules. As discussed above, in this way we achieve an improvement of the parsing time for SCFGs, obtaining an upper bound of $O(|G|\, N^{k+4})$ by using a parsing strategy that maintains continuous

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 279–286,
Sydney, July 2006.

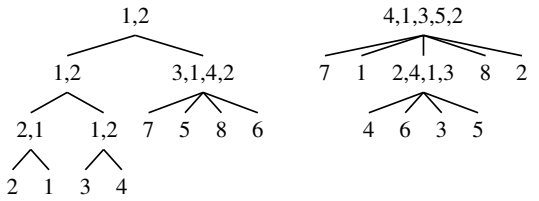


Figure 1: Two permutation trees. The permutations associated with the leaves can be produced by composing the permutations at the internal nodes.

spans in one dimension.

Previous work on this problem has been presented in Zhang et al. (2006), where a method is provided for casting an SCFG to a form with rank $k = 2$. If generalized to any value of k, that algorithm would run in time $O(n^2)$. We thus improve existing factorization methods by almost a factor of n. We also solve an open problem mentioned by Albert et al. (2003), who pose the question of whether irreducible, or *simple*, permutations can be recognized in time less than $\Theta(n^2)$.

2 Synchronous CFGs and permutation trees

We begin by describing the synchronous CFG formalism, which is more rigorously defined by Aho and Ullman (1972) and Satta and Peserico (2005). Let us consider strings defined over some set of nonterminal and terminal symbols, as defined for CFGs. We say that two such strings are **synchronous** if some bijective relation is given between the occurrences of the nonterminals in the two strings. A **synchronous context-free grammar** (SCFG) is defined as a CFG, with the difference that it uses synchronous rules of the form $[A_1 \rightarrow \alpha_1,\ A_2 \rightarrow \alpha_2]$, with A_1, A_2 nonterminals and α_1, α_2 synchronous strings. We can use production $[A_1 \rightarrow \alpha_1,\ A_2 \rightarrow \alpha_2]$ to rewrite any synchronous strings $[\gamma_{11} A_1 \gamma_{12},\ \ \gamma_{21} A_2 \gamma_{22}]$ into the synchronous strings $[\gamma_{11}\alpha_1\gamma_{12},\ \ \gamma_{21}\alpha_2\gamma_{22}]$, under the condition that the indicated occurrences of A_1 and A_2 be related by the bijection associated with the source synchronous strings. Furthermore, the bijective relation associated with the target synchronous strings is obtained by composing the relation associated with the source synchronous strings and the relation associated with synchronous pair $[\alpha_1, \alpha_2]$, in the most obvious way.

As in standard constructions that reduce the rank of a CFG, in this paper we focus on each single synchronous rule and factorize it into synchronous rules of lower rank. If we view the bijective relation associated with a synchronous rule as a permutation, we can further reduce our factorization problem to the problem of factorizing a permutation of arity n into the composition of several permutations of arity $k < n$. Such factorization can be represented as a tree of composed permutations, called in what follows a **permutation tree**. A permutation tree can be converted into a set of k-ary SCFG rules equivalent to the input rule. For example, the input rule:

$$[\, X \rightarrow A^{(1)} B^{(2)} C^{(3)} D^{(4)} E^{(5)} F^{(6)} G^{(7)} H^{(8)},\ X \rightarrow B^{(2)} A^{(1)} C^{(3)} D^{(4)} G^{(7)} E^{(5)} H^{(8)} F^{(6)} \,]$$

yields the permutation tree of Figure 1(left). Introducing a new grammar nonterminal X_i for each internal node of the tree yields an equivalent set of smaller rules:

$$[\, X \rightarrow X_1^{(1)} X_2^{(2)},\ X \rightarrow X_1^{(1)} X_2^{(2)} \,]$$
$$[\, X_1 \rightarrow X_3^{(1)} X_4^{(2)},\ X_1 \rightarrow X_3^{(1)} X_4^{(2)} \,]$$
$$[\, X_3 \rightarrow A^{(1)} B^{(2)},\ X_3 \rightarrow B^{(2)} A^{(1)} \,]$$
$$[\, X_4 \rightarrow C^{(1)} D^{(2)},\ X_4 \rightarrow C^{(1)} D^{(2)} \,]$$
$$[\, X_2 \rightarrow E^{(1)} F^{(2)} G^{(3)} H^{(4)},\ X_2 \rightarrow G^{(3)} E^{(1)} H^{(4)} F^{(2)} \,]$$

In the case of stochastic grammars, the rule corresponding to the root of the permutation tree is assigned the original rule's probability, while all other rules, associated with new grammar nonterminals, are assigned probability 1. We process each rule of an input SCFG independently, producing an equivalent grammar with the smallest possible arity.

3 Factorization Algorithm

In this section we specify and discuss our factorization algorithm. The algorithm takes as input a permutation defined on the set $\{1, \cdots, n\}$, representing a rule of some SCFG, and provides a permutation tree of arity $k \leq n$ for that permutation, with k as small as possible.

Permutation trees covering a given input permutation are unambiguous with the exception of sequences of binary rules of the same type (either inverted or straight) (Albert et al., 2003). Thus, when factorizing a permutation into a permutation

tree, it is safe to greedily reduce a subsequence into a new subtree as soon as a subsequence is found which represents a continuous span in both dimensions of the permutation matrix[1] associated with the input permutation. For space reasons, we omit the proof, but emphasize that any greedy reduction turns out to be either necessary, or equivalent to the other alternatives.

Any sequences of binary rules can be rearranged into a normalized form (e.g. always left-branching) as a postprocessing step, if desired.

The top-level structure of the algorithm exploits a divide-and-conquer approach, and is the same as that of the well-known mergesort algorithm (Cormen et al., 1990). We work on subsequences of the original permutation, and 'merge' neighboring subsequences into successively longer subsequences, combining two subsequences of length 2^i into a subsequence of length 2^{i+1} until we have built one subsequence spanning the entire permutation. If each combination of subsequences can be performed in linear time, then the entire permutation can be processed in time $O(n \log n)$. As in the case of mergesort, this is an application of the so-called master theorem (Cormen et al., 1990).

As the algorithm operates, we will maintain the invariant that we must have built all subtrees of the target permutation tree that are entirely within a given subsequence that has been processed. This is analogous to the invariant in mergesort that all processed subsequences are in sorted order. When we combine two subsequences, we need only build nodes in the tree that cover parts of both subsequences, but are entirely within the combined subsequence. Thus, we are looking for subtrees that span the midpoint of the combined subsequence, but have left and right boundaries within the boundaries of the combined subsequence. In what follows, this midpoint is called the **split point**.

From this invariant, we will be guaranteed to have a complete, correct permutation tree at the end of last subsequence combination. An example of the operation of the general algorithm is shown in Figure 2. The top-level structure of the algorithm is presented in function KARIZE of Figure 3.

There may be more than one reduction necessary spanning a given split point when combining two subsequences. Function MERGE in Figure 3 initializes certain data structures described below, and then checks for reductions repeatedly until no further reduction is possible. It looks first for the smallest reduction crossing the split point of the subsequences being combined. If SCAN, described below, finds a valid reduction, it is committed by calling REDUCE. If a reduction is found, we look for further reductions crossing either the left or right boundary of the new reduction, repeating until no further reductions are possible. Because we only need to find reductions spanning the original split point at a given combination step, this process is guaranteed to find all reductions needed.

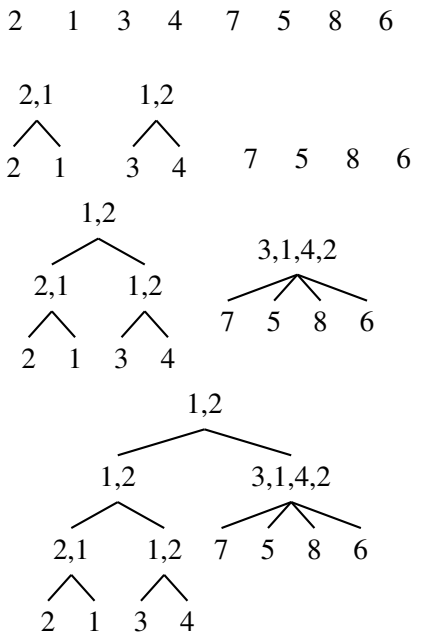


Figure 2: Recursive combination of permutation trees. Top row, the input permutation. Second row, after combination into sequences of length two, binary nodes have been built where possible. Third row, after combination into sequences of length four; bottom row, the entire output tree.

We now turn to the problem of identifying a specific reduction to be made across a split point, which involves identifying the reduction's left and right boundaries. Given a subsequence and candidate left and right boundaries for that subsequence, the validity of making a reduction over this span can be tested by verifying whether the span constitutes a **permuted sequence**, that is, a permutation of a contiguous sequence of integers. Since the starting permutation is defined on a set $\{1, 2, \cdots, n\}$, we have no repeated integers in our subsequences, and the above condi-

[1] A permutation matrix is a way of representing a permutation, and is obtained by rearranging the row (or the columns) of an identity matrix, according to the permutation itself.

```
function KARIZE(π)
                ▷ initialize with identity mapping
    h ← hmin ← hmax ← (0..|π|);
                                ▷ mergesort core
    for size ← 1; size ≤ |π|; size ← size * 2 do
        for min ← 0;
                  min < |π|-size+1;
                  min ← min + 2 * size do
            div = min + size - 1;
            max ← min(|π|, min + 2*size - 1);
            MERGE(min, div, max);
```

```
function MERGE(min, div, max)
                                    ▷ initialize h
    sort h[min..max] according to π[i];
    sort hmin[min..max] according to π[i];
    sort hmax[min..max] according to π[i];
        ▷ merging sorted list takes linear time
                                    ▷ initialize v
    for i ← min; i ≤ max; i ← i + 1 do
        v [ h[i] ] ← i;
          ▷ check if start of new reduced block
        if i = min or
                          hmin[i] ≠ hmin[i-1] then
            vmin ← i;
        vmin[ h[i] ] ← vmin;
    for i ← max; i ≥ min; i ← i - 1 do
          ▷ check if start of new reduced block
        if i = max or
                        hmax[i] ≠ hmax[i+1] then
            vmax ← i ;
        vmax[ h[i] ] ← vmax;
                              ▷ look for reductions
    if SCAN(div) then
        REDUCE(scanned reduction);
        while SCAN(left) or SCAN(right) do
            REDUCE(smaller reduction);
```

```
function REDUCE(left, right, bot, top)
    for i ← bot..top do
        hmin[i] ← left;
        hmax[i] ← right;
    for i ← left..right do
        vmin[i] ← bot;
        vmax[i] ← top;
    print "reduce:" left..right ;
```

Figure 3: KARIZE: Top level of algorithm, identical to that of mergesort. MERGE: combines two subsequences of size 2^i into new subsequence of size 2^{i+1}. REDUCE: commits reduction by updating *min* and *max* arrays.

tion can be tested by scanning the span in question, finding the minimum and maximum integers in the span, and checking whether their difference is equal to the length of the span minus one. Below we call this condition the **reduction test**. As an example of the reduction test, consider the subsequence $(7, 5, 8, 6)$, and take the last three elements, $(5, 8, 6)$, as a candidate span. We see that 5 and 8 are the minimum and maximum integers in the corresponding span, respectively. We then compute $8 - 5 = 3$, while the length of the span minus one is 2, implying that no reduction is possible. However, examining the entire subsequence, the minimum is 5 and the maximum is 8, and $8 - 5 = 3$, which is the length of the span minus one. We therefore conclude that we can reduce that span by means of some permutation, that is, parse the span by means of a node in the permutation tree. This reduction constitutes the 4-ary node in the permutation tree of Figure 2.

A trivial implementation of the reduction test would be to tests all combinations of left and right boundaries for the new reduction. Unfortunately, this would take time $\Omega(n^2)$ for a single subsequence combination step, whereas to achieve the overall $O(n \log n)$ complexity we need linear time for each combination step.

It turns out that the boundaries of the next reduction, covering a given split point, can be computed in linear time with the technique shown in function SCAN of Figure 5. We start with left and right candidate boundaries at the two points immediately to the left and right of the split point, and then repeatedly check whether the current left and right boundaries identify a permuted sequence by applying the reduction test, and move the left and right boundaries outward as necessary, as soon as 'missing' integers are identified outside the current boundaries, as explained below. We will show that, as we move outward, the number of possible configurations achieved for the positions of the left and the right boundaries is linearly bounded in the length of the combined subsequence (as opposed to quadratically bounded).

In order to efficiently implement the above idea, we will in fact maintain four boundaries for the candidate reduction, which can be visualized as the left, right, top and bottom boundaries in the permutation matrix. No explicit representation of the permutation matrix itself is constructed, as that would require quadratic time. Rather, we

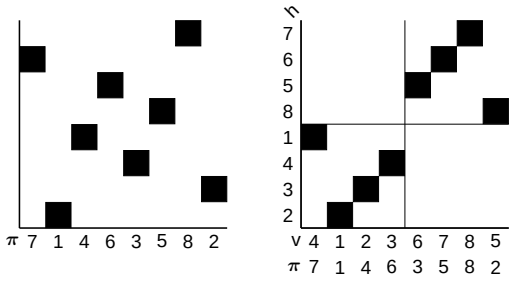


Figure 4: Permutation matrix for input permutation π (left) and within-subsequence permutation v (right) for subsequences of size four.

maintain two arrays: h, which maps from vertical to horizontal positions within the current subsequence, and v which maps from horizontal to vertical positions. These arrays represent the within-subsequence permutation obtained by sorting the elements of each subsequence according to the input permutation, while keeping each element within its block, as shown in Figure 4.

Within each subsequence, we alternate between scanning horizontally from left to right, possibly extending the top and bottom boundaries (Figure 5 lines 9 to 14), and scanning vertically from bottom to top, possibly extending the left and right boundaries (lines 20 to 26). Each extension is forced when, looking at the within-subsequence permutation, we find that some element is within the current boundaries in one dimension but outside the boundaries in the other. If the distance between vertical boundaries is larger in the input permutation than in the subsequence permutation, necessary elements are missing from the current subsequence and no reduction is possible at this step (line 18). When all necessary elements are present in the current subsequence and no further extensions are necessary to the boundaries (line 30), we have satisfied the reduction test on the input permutation, and make a reduction.

The trick used to keep the iterative scanning linear is that we *skip* the subsequence scanned on the previous iteration on each scan, in both the horizontal and vertical directions. Lines 13 and 25 of Figure 5 perform this skip by advancing the x and y counters past previously scanned regions. Considering the horizontal scan of lines 9 to 14, in a given iteration of the while loop, we scan only the items between *newleft* and *left* and between *right* and *newright*. On the next iteration of the while loop, the *newleft* boundary has moved further to the left,

```
1:  function SCAN (div)
2:      left ← −∞;
3:      right ← −∞;
4:      newleft ← div;
5:      newright ← div + 1 ;
6:      newtop ← −∞;
7:      newbot ← ∞;
8:      while 1 do
                                        ▷ horizontal scan
9:          for x ← newleft; x ≤ newright ; do
10:             newtop ← max(newtop, vmax[x]);
11:             newbot ← min(newbot, vmin[x]);
                    ▷ skip to end of reduced block
12:             x ← hmax[vmin[x]] + 1;
                ▷ skip section scanned on last iter
13:             if x = left then
14:                 x ← right + 1;
15:         right ← newright;
16:         left ← newleft;
                                ▷ the reduction test
17:         if newtop - newbot <
18:            π[h[newtop]] - π[h[newbot]] then
19:             return (0);
                                      ▷ vertical scan
20:         for y ← newbot; y ≤ newtop ; do
21:             newright ←
22:                         max(newright, hmax[y]);
23:             newleft ← min(newleft, hmin[y]);
                    ▷ skip to end of reduced block
24:             y ← vmax[hmin[y]] + 1;
                ▷ skip section scanned on last iter
25:             if y = bot then
26:                 y ← top + 1;
27:         top ← newtop;
28:         bot ← newbot;
            ▷ if no change to boundaries, reduce
29:         if newright = right
30:                         and newleft = left then
31:             return (1, left, right, bot, top);
```

Figure 5: Linear time function to check for a single reduction at split point *div*.

while the variable *left* takes the previous value of *newleft*, ensuring that the items scanned on this iteration are distinct from those already processed. Similarly, on the right edge we scan new items, between *right* and *newright*. The same analysis applies to the vertical scan. Because each item in the permutation is scanned only once in the vertical direction and once in the horizontal direction, the entire call to SCAN takes linear time, regardless of the number of iterations of the while loop that are required.

We must further show that each call to MERGE takes only linear time, despite that fact that it may involve many calls to SCAN. We accomplish this by introducing a second type of skipping in the scans, which advances past any previously reduced block in a single step. In order to skip past previous reductions, we maintain (in function REDUCE) auxiliary arrays with the minimum and maximum positions of the largest block each point has been reduced to, in both the horizontal and vertical dimensions. We use these data structures (*hmin*, *hmax*, *vmin*, *vmax*) when advancing to the next position of the scan in lines 12 and 24 of Figure 5. Because each call to SCAN skips items scanned by previous calls, each item is scanned at most twice across an entire call to MERGE, once when scanning across a new reduction's left boundary and once when scanning across the right boundary, guaranteeing that MERGE completes in linear time.

4 An Example

In this section we examine the operation of the algorithm on a permutation of length eight, resulting in the permutation tree of Figure 1(right). We will build up our analysis of the permutation by starting with individual items of the input permutation and building up subsequences of length 2, 4, and finally 8. In our example permutation, $(7, 1, 4, 6, 3, 5, 8, 2)$, no reductions can be made until the final combination step, in which one permutation of size 4 is used, and one of size 5.

We begin with the input permutation along the bottom of Figure 6a. We represent the intermediate data structures *h*, *hmin*, and *hmax* along the vertical axis of the figure; these three arrays are all initialized to be the sequence $(1, 2, \cdots, 8)$.

Figure 6b shows the combination of individual items into subsequences of length two. Each new subsequence of the *h* array is sorted according to

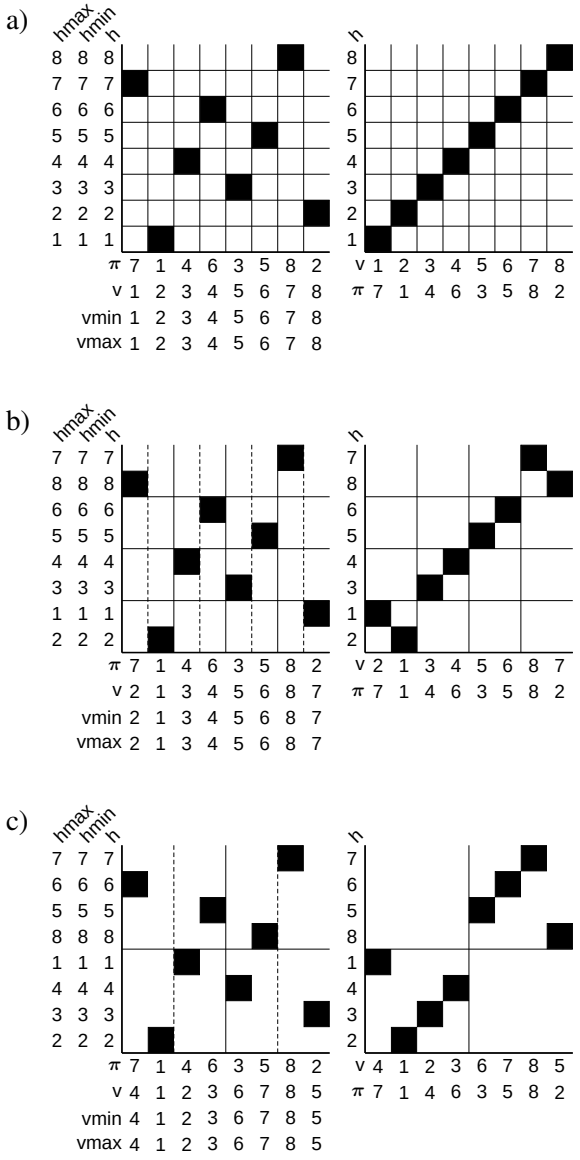
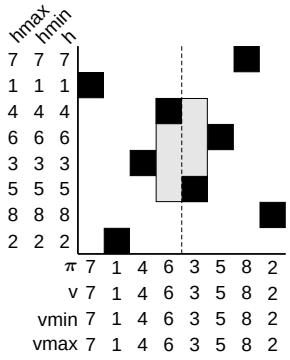


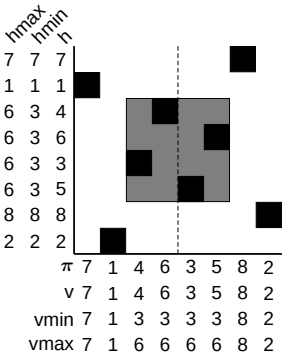
Figure 6: Steps in an example computation, with input permutation π on left and within-subsequence permutation described by *v* array on right. Panel (a) shows initial blocks of unit size, (b) shows combination of unit blocks into blocks of size two, and (c) size two into size four. No reductions are possible in these stages; example continued in next figure.

a)



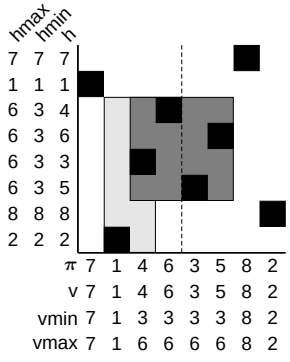
Left and right boundaries are initialized to be adjacent to horizontal split point.

b)



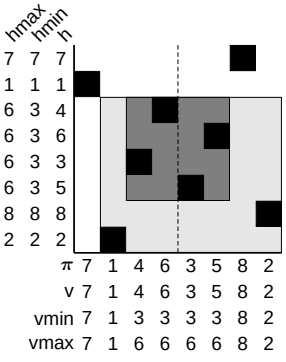
Vertical scan shows left and right boundaries must be extended. Permutation of size four is reduced.

c)



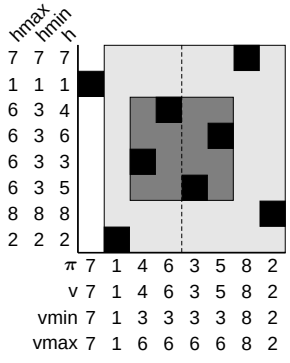
Search for next reduction: left and right boundaries initialized to be adjacent to left edge of previous reduction.

d)



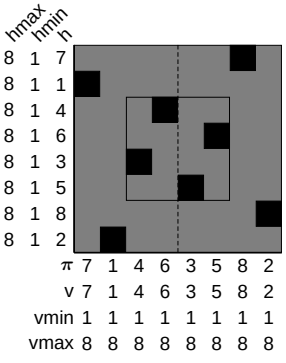
Vertical scan shows right boundary must be extended.

e)



Horizontal scan shows top boundary must be extended.

f)



Vertical scan shows left boundary must be extended. Permutation of size five is reduced.

Figure 7: Steps in scanning for final combination of subsequences, where $\nu = \pi$. Area within current left, right, top and bottom boundaries is shaded; darker shading indicates a reduction. In each scan, the span scanned in the previous panel is skipped over.

the vertical position of the dots in the corresponding columns. Thus, because $\pi[7] = 8 > \pi[8] = 2$, we swap 7 and 8 in the *h* array. The algorithm checks whether any reductions can be made at this step by computing the difference between the integers on each side of each split point. Because none of the pairs of integers in are consecutive, no reductions are made at this step.

Figure 6c shows the combination the pairs into subsequences of length four. The two split points to be examined are between the second and third position, and the sixth and seventh position. Again, no reductions are possible.

Finally we combine the two subsequences of length four to complete the analysis of the entire permutation. The split point is between the fourth and fifth positions of the input permutation, and in the first horizontal scan of these two positions, we see that $\pi[4] = 6$ and $\pi[5] = 3$, meaning our top boundary will be 6 and our bottom boundary 3, shown in Figure 7a. Scanning vertically from position 3 to 6, we see horizontal positions 5, 3, 6, and 4, giving the minimum, 3, as the new left boundary and the maximum, 6, as the new right boundary, shown in Figure 7b. We now perform another horizontal scan starting at position 3, but then jumping directly to position 6, as horizontal positions 4 and 5 were scanned previously. After this scan, the minimum vertical position seen remains 3, and the maximum vertical position is still 6. At this point, because we have the same boundaries as on the previous scan, we can stop and verify whether the region determined by our current boundaries has the same length in the vertical and horizontal dimensions. Both dimensions have length four, meaning that we have found a subsequence that is continuous in both dimensions and can safely be reduced, as shown in Figure 6d.

After making this reduction, we update the *hmin* array to have all 3's for the newly reduced span, and update *hmax* to have all sixes. We then check whether further reductions are possible covering this split point. We repeat the process of scanning horizontally and vertically in Figure 7c-f, this time skipping the span just reduced. One further reduction is possible, covering the entire input permutation, as shown in Figure 7f.

5 Conclusion

The algorithm above not only identifies whether a permutation can be factored into a composition of permutations, but also returns the factorization that minimizes the largest rule size, in time $O(n \log n)$. The factored SCFG with rules of size at most k can be used to synchronously parse in time $O(N^{k+4})$ by dynamic programming with continuous spans in one dimension.

As mentioned in the introduction, the optimal parsing strategy for SCFG rules with a given permutation may involve dynamic programming states with discontinuous spans in both dimensions. Whether these optimal parsing strategies can be found efficiently remains an interesting open problem.

Acknowledgments This work was partially supported by NSF ITR IIS-09325646 and NSF ITR IIS-0428020.

References

Albert V. Aho and Jeffery D. Ullman. 1972. *The Theory of Parsing, Translation, and Compiling*, volume 1. Prentice-Hall, Englewood Cliffs, NJ.

M. H. Albert, M. D. Atkinson, and M. Klazar. 2003. The enumeration of simple permutations. *Journal of Integer Sequences*, 6(03.4.4):18 pages.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL-05*, pages 263–270.

Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. 1990. *Introduction to algorithms*. MIT Press, Cambridge, MA.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *Proceedings of HLT/NAACL*.

I. Dan Melamed. 2003. Multitext grammars and synchronous parsers. In *Proceedings of HLT/NAACL*.

Giorgio Satta and Enoch Peserico. 2005. Some computational complexity results for synchronous context-free grammars. In *Proceedings of HLT/EMNLP*, pages 803–810.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of ACL-01*.

Hao Zhang, Liang Huang, Daniel Gildea, and Kevin Knight. 2006. Synchronous binarization for machine translation. In *Proceedings of HLT/NAACL*.

Low-cost Enrichment of Spanish WordNet with Automatically Translated Glosses: Combining General and Specialized Models

Jesús Giménez and **Lluís Màrquez**
TALP Research Center, LSI Department
Universitat Politècnica de Catalunya
Jordi Girona Salgado 1–3, E-08034, Barcelona
{jgimenez,lluism}@lsi.upc.edu

Abstract

This paper studies the enrichment of Spanish WordNet with synset glosses automatically obtained from the English WordNet glosses using a phrase-based Statistical Machine Translation system. We construct the English-Spanish translation system from a parallel corpus of proceedings of the European Parliament, and study how to adapt statistical models to the domain of dictionary definitions. We build specialized language and translation models from a small set of parallel definitions and experiment with robust manners to combine them. A statistically significant increase in performance is obtained. The best system is finally used to generate a definition for all Spanish synsets, which are currently ready for a manual revision. As a complementary issue, we analyze the impact of the amount of in-domain data needed to improve a system trained entirely on out-of-domain data.

1 Introduction

Statistical Machine Translation (SMT) is today a very promising approach. It allows to build very quickly and fully automatically Machine Translation (MT) systems, exhibiting very competitive results, only from a parallel corpus aligning sentences from the two languages involved.

In this work we approach the task of enriching Spanish WordNet with automatically translated glosses[1]. The source glosses for these translations are taken from the English WordNet (Fellbaum, 1998), which is linked, at the synset level, to Spanish WordNet. This resource is available, among other sources, through the Multilingual Central Repository (MCR) developed by the MEANING project (Atserias et al., 2004).

We start by empirically testing the performance of a previously developed English–Spanish SMT system, built from the large Europarl corpus[2] (Koehn, 2003). The first observation is that this system completely fails to translate the specific WordNet glosses, due to the large language variations in both domains (vocabulary, style, grammar, etc.). Actually, this is confirming one of the main criticisms against SMT, which is its strong domain dependence. Since parameters are estimated from a corpus in a concrete domain, the performance of the system on a different domain is often much worse. This flaw of statistical and machine learning approaches is well known and has been largely described in the NLP literature, for a variety of tasks (e.g., parsing, word sense disambiguation, and semantic role labeling).

Fortunately, we count on a small set of Spanish hand-developed glosses in MCR[3]. Thus, we move to a working scenario in which we introduce a small corpus of aligned translations from the concrete domain of WordNet glosses. This in-domain corpus could be itself used as a source for constructing a specialized SMT system. Again, experiments show that this small corpus alone does not suffice, since it does not allow to estimate good translation parameters. However, it is well suited for combination with the Europarl corpus, to generate combined Language and Translation

[1] Glosses are short dictionary definitions that accompany WordNet synsets. See examples in Tables 5 and 6.

[2] The Europarl Corpus is available at: http://-people.csail.mit.edu/people/koehn/publications/europarl

[3] About 10% of the 68,000 Spanish synsets contain a definition, generated without considering its English counterpart.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 287–294,
Sydney, July 2006.

Models. A substantial increase in performance is achieved, according to several standard MT evaluation metrics. Although moderate, this boost in performance is statistically significant according to the bootstrap resampling test described by Koehn (2004b) and applied to the BLEU metric.

The main reason behind this improvement is that the large out-of-domain corpus contributes mainly with coverage and recall and the in-domain corpus provides more precise translations. We present a qualitative error analysis to support these claims. Finally, we also address the important question of how much in-domain data is needed to be able to improve the baseline results.

Apart from the experimental findings, our study has generated a very valuable resource. Currently, we have the complete Spanish WordNet enriched with one gloss per synset, which, far from being perfect, constitutes an axcellent starting point for a posterior manual revision.

Finally, we note that the construction of a SMT system when few domain-specific data are available has been also investigated by other authors. For instance, Vogel and Tribble (2002) studied whether an SMT system for speech-to-speech translation built on top of a small parallel corpus can be improved by adding knowledge sources which are not domain specific. In this work, we look at the same problem the other way around. We study how to adapt an out-of-domain SMT system using in-domain data.

The rest of the paper is organized as follows. In Section 2 the fundamentals of SMT and the components of our MT architecture are described. The experimental setting is described in Section 3. Evaluation is carried out in Section 4. Finally, Section 5 contains error analysis and Section 6 concludes and outlines future work.

2 Background

Current state-of-the-art SMT systems are based on ideas borrowed from the Communication Theory field. Brown et al. (1988) suggested that MT can be statistically approximated to the transmission of information through a *noisy channel*. Given a sentence $f = f_1..f_n$ (distorted signal), it is possible to approximate the sentence $e = e_1..e_m$ (original signal) which produced f. We need to estimate $P(e|f)$, the probability that a translator produces f as a translation of e. By applying Bayes' rule it is decomposed into: $P(e|f) = \frac{P(f|e)*P(e)}{P(f)}$.

To obtain the string e which maximizes the translation probability for f, a search in the probability space must be performed. Because the denominator is independent of e, we can ignore it for the purpose of the search: $e = argmax_e P(f|e) * P(e)$. This last equation devises three components in a SMT system. First, a *language model* that estimates $P(e)$. Second, a *translation model* representing $P(f|e)$. Last, a *decoder* responsible for performing the arg-max search. Language models are typically estimated from large monolingual corpora, translation models are built out from parallel corpora, and decoders usually perform approximate search, e.g., by using dynamic programming and beam search.

However, in word-based models the modeling of the context in which the words occur is very weak. This problem is significantly alleviated by phrase-based models (Och, 2002), which represent nowadays the state-of-the-art in SMT.

2.1 System Construction

Fortunately, there is a number of freely available tools to build a phrase-based SMT system. We used only standard components and techniques for our basic system, which are all described below.

The *SRI Language Modeling Toolkit* (SRILM) (Stolcke, 2002) supports creation and evaluation of a variety of language models. We build trigram language models applying linear interpolation and Kneser-Ney discounting for smoothing.

In order to build phrase-based translation models, a phrase extraction must be performed on a word-aligned parallel corpus. We used the GIZA++ SMT Toolkit[4] (Och and Ney, 2003) to generate word alignments We applied the phrase-extract algorithm, as described by Och (2002), on the Viterbi alignments output by GIZA++. We work with the union of source-to-target and target-to-source alignments, with no heuristic refinement. Phrases up to length five are considered. Also, phrase pairs appearing only once are discarded, and phrase pairs in which the source/target phrase was more than three times longer than the target/source phrase are ignored. Finally, phrase pairs are scored by relative frequency. Note that no smoothing is performed.

Regarding the arg-max search, we used the *Pharaoh* beam search decoder (Koehn, 2004a), which naturally fits with the previous tools.

[4] http://www.fjoch.com/GIZA++.html

3 Data Sets and Evaluation Metrics

As a general source of English–Spanish parallel text, we used a collection of 730,740 parallel sentences extracted from the Europarl corpus. These correspond exactly to the training data from the Shared Task 2: *Exploiting Parallel Texts for Statistical Machine Translation* from the ACL-2005 Workshop on *Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond*[5].

To be used as specialized source, we extracted, from the MCR , the set of 6,519 English–Spanish parallel glosses corresponding to the already defined synsets in Spanish WordNet. These definitions corresponded to 5,698 nouns, 87 verbs, and 734 adjectives. Examples and parenthesized texts were removed. Parallel glosses were tokenized and case lowered. We discarded some of these parallel glosses based on the difference in length between the source and the target. The gloss average length for the resulting 5,843 glosses was 8.25 words for English and 8.13 for Spanish. Finally, gloss pairs were randomly split into training (4,843), development (500) and test (500) sets.

Additionally, we counted on two large monolingual Spanish electronic dictionaries, consisting of 142,892 definitions (2,112,592 tokens) ('D1') (Martí, 1996) and 168,779 definitons (1,553,674 tokens) ('D2') (Vox, 1990), respectively.

Regarding evaluation, we used up to four different metrics with the aim of showing whether the improvements attained are consistent or not. We have computed the BLEU score (accumulated up to 4-grams) (Papineni et al., 2001), the NIST score (accumulated up to 5-grams) (Doddington, 2002), the General Text Matching (GTM) F-measure ($e = 1, 2$) (Melamed et al., 2003), and the METEOR measure (Banerjee and Lavie, 2005). These metrics work at the lexical level by rewarding n-gram matches between the candidate translation and a set of human references. Additionally, METEOR considers stemming, and allows for WordNet synonymy lookup.

The discussion of the significance of the results will be based on the BLEU score, for which we computed a bootstrap resampling test of significance (Koehn, 2004b).

[5] http://www.statmt.org/wpt05/.

4 Experimental Evaluation

4.1 Baseline Systems

As explained in the introduction we built two individual baseline systems. The first baseline ('EU') system is entirely based on the training data from the Europarl corpus. The second baseline system ('WNG') is entirely based on the training set from of the in-domain corpus of parallel glosses. In the second case phrase pairs occurring only once in the training corpus are not discarded due to the extremely small size of the corpus.

Table 1 shows results of the two baseline systems, both for the development and test sets. We compare the performance of the 'EU' baseline on these data sets with respect to the (in-domain) Europarl test set provided by the organizers of the ACL-2005 MT workshop. As expected, there is a very significant decrease in performance (e.g., from 0.24 to 0.08 according to BLEU) when the 'EU' baseline system is applied to the new domain. Some of this decrement is also due to a certain degree of free translation exhibited by the set of available 'quasi-parallel' glosses. We further discuss this issue in Section 5.

The results obtained by 'WNG' are also very low, though slightly better than those of 'EU'. This is a very interesting fact. Although the amount of data utilized to construct the 'WNG' baseline is 150 times smaller than the amount utilized to construct the 'EU' baseline, its performance is higher consistently according to all metrics. We interpret this result as an indicator that models estimated from in-domain data provide higher precision.

We also compare the results to those of a commercial system such as the on-line version 5.0 of SYSTRAN[6], a general-purpose MT system based on manually-defined lexical and syntactic transfer rules. The performance of the baseline systems is significantly worse than SYSTRAN's on both development and test sets. This means that a rule-based system like SYSTRAN is more robust than the SMT-based systems. The difference against the specialized 'WNG' also suggests that the amount of data used to train the 'WNG' baseline is clearly insufficient.

4.2 Combining Sources: Language Models

In order to improve results, in first place we turned our eyes to language modeling. In addition to

[6] http://www.systransoft.com/.

system	BLEU.n4	NIST.n5	GTM.e1	GTM.e2	METEOR
development					
EU-baseline	0.0737	2.8832	0.3131	0.2216	0.2881
WNG-baseline	0.1149	3.3492	0.3604	0.2605	0.3288
SYSTRAN	0.1625	3.9467	0.4257	0.2971	0.4394
test					
EU-baseline	0.0790	2.8896	0.3131	0.2262	0.2920
WNG-baseline	0.0951	3.1307	0.3471	0.2510	0.3219
SYSTRAN	0.1463	3.7873	0.4085	0.2921	0.4295
acl05-test					
EU-baseline	0.2381	6.5848	0.5699	0.2429	0.5153

Table 1: MT Results on development and test sets, for the two baseline systems compared to SYSTRAN and to the 'EU' baseline system on the ACL-2005 SMT workshop test set extracted from the Europarl Corpus. BLEU.n4 shows the accumulated BLEU score for 4-grams. NIST.n5 shows the accumulated NIST score for 5-grams. GTM.e1 and GTM.e2 show the GTM F_1-measure for different values of the e parameter ($e = 1$, $e = 2$, respectively). METEOR reflects the METEOR score.

the language model built from the Europarl corpus ('EU') and the specialized language model based on the small training set of parallel glosses ('WNG'), two specialized language models, based on the two large monolingual Spanish electronic dictionaries ('D1' and 'D2') were used. We tried several configurations. In all cases, language models are combined with equal probability. See results, for the development set, in Table 2.

As expected, the closer the language model is to the target domain, the better results. Observe how results using language models 'D1' and 'D2' outperform results using 'EU'. Note also that best results are in all cases consistently attained by using the 'WNG' language model. This means that language models estimated from small sets of in-domain data are helpful. A second conclusion is that a significant gain is obtained by incrementally adding (in-domain) specialized language models to the baselines, according to all metrics but BLEU for which no combination seems to significantly outperform the 'WNG' baseline alone. Observe that best results are obtained, except in the case of BLEU, by the system using 'EU' as translation model and 'WNG' as language model. We interpret this result as an indicator that translation models estimated from out-of-domain data are helpful because they provide recall. A third interesting point is that adding an out-of-domain language model ('EU') does not seem to help, at least combined with equal probability than in-domain models. Same conclusions hold for the test set, too.

4.3 Tuning the System

Adjusting the *Pharaoh* parameters that control the importance of the different probabilities that govern the search may yield significant improvements. In our case, it is specially important to properly adjust the contribution of the language models. We adjusted parameters by means of a software based on the *Downhill Simplex Method in Multidimensions* (William H. Press and Flannery, 2002). The tuning was based on the improvement attained in BLEU score over the development set. We tuned 6 parameters: 4 language models (λ_{lmEU}, λ_{lmD1}, λ_{lmD2}, λ_{lmWNG}), the translation model (λ_ϕ), and the word penalty (λ_w)[7].

Results improve substantially. See Table 3. Best results are still attained using the 'EU' translation model. Interestingly, as suggested by Table 2, the weight of language models is concentrated on the 'WNG' language model ($\lambda_{lmWNG} = 0.95$).

4.4 Combining Sources: Translation Models

In this section we study the possibility of combining out-of-domain and in-domain translation models aiming at achieving a good balance between precision and recall that yields better MT results.

Two different strategies have been tried. In a first stragegy we simply concatenate the out-of-domain corpus ('EU') and the in-domain corpus ('WNG'). Then, we construct the translatation model ('EUWNG') as detailed in Section 2.1. A second manner to proceed is to linearly combine the two different translation models into a single translation model ('EU+WNG'). In this case, we can assign different weights (ω) to the contribution of the different models to the search. We can also determine a certain threshold θ which allows us

[7] Final values when using the 'EU' translation model are $\lambda_{lmEU} = 0.22$, $\lambda_{lmD1} = 0$, $\lambda_{lmD2} = 0.01$, $\lambda_{lmWNG} = 0.95$, $\lambda_\phi = 1$, and $\lambda_w = -2.97$, while when using the 'WNG' translation model final values are $\lambda_{lmEU} = 0.17$, $\lambda_{lmD1} = 0.07$, $\lambda_{lmD2} = 0.13$, $\lambda_{lmWNG} = 1$, $\lambda_\phi = 0.95$, and $\lambda_w = -2.64$.

Translation Model	Language Model	BLEU.n4	NIST.n5	GTM.e1	GTM.e2	METEOR
EU	EU	0.0737	2.8832	0.3131	0.2216	0.2881
EU	WNG	0.1062	3.4831	0.3714	0.2631	0.3377
EU	D1	0.0959	3.2570	0.3461	0.2503	0.3158
EU	D2	0.0896	3.2518	0.3497	0.2482	0.3163
EU	D1 + D2	0.0993	3.3773	0.3585	0.2579	0.3244
EU	EU + D1 + D2	0.0960	3.2851	0.3472	0.2499	0.3160
EU	D1 + D2 + WNG	0.1094	3.4954	0.3690	0.2662	0.3372
EU	EU + D1 + D2 + WNG	0.1080	3.4248	0.3638	0.2614	0.3321
WNG	EU	0.0743	2.8864	0.3128	0.2202	0.2689
WNG	WNG	0.1149	3.3492	0.3604	0.2605	0.3288
WNG	D1	0.0926	3.1544	0.3404	0.2418	0.3050
WNG	D2	0.0845	3.0295	0.3256	0.2326	0.2883
WNG	D1 + D2	0.0917	3.1185	0.3331	0.2394	0.2995
WNG	EU + D1 + D2	0.0856	3.0361	0.3221	0.2312	0.2847
WNG	D1 + D2 + WNG	0.0980	3.2238	0.3462	0.2479	0.3117
WNG	EU + D1 + D2 + WNG	0.0890	3.0974	0.3309	0.2373	0.2941

Table 2: MT Results on development set, for several translation/language model configurations. 'EU' and 'WNG' refer to the models estimated from the Europarl corpus and the training set of parallel WordNet glosses, respectively. 'D1', and 'D2' denote the specialized language models estimated from the two dictionaries.

Translation Model	Language Model	BLEU.n4	NIST.n5	GTM.e1	GTM.e2	METEOR
development						
EU	EU + D1 + D2 + WNG	0.1272	3.6094	0.3856	0.2727	0.3695
WNG	EU + D1 + D2 + WNG	0.1269	3.3740	0.3688	0.2676	0.3452
test						
EU	EU + D1 + D2 + WNG	0.1133	3.4180	0.3720	0.2650	0.3644
WNG	EU + D1 + D2 + WNG	0.1015	3.1084	0.3525	0.2552	0.3343

Table 3: MT Results on development and test sets after tuning for the 'EU + D1 + D2 + WNG' language model configuration for the two translation models, 'EU' and 'WNG'.

to discard phrase pairs under a certain probability. These weights and thresholds were adjusted[8] as detailed in Subsection 4.3. Interestingly, at combination time the importance of the 'WNG' translation model ($\omega_{tmWNG} = 0.9$) is much higher than that of the 'EU' translation model ($\omega_{tmEU} = 0.1$).

Table 4 shows results for the two strategies. As expected, the 'EU+WNG' strategy consistently obtains the best results according to all metrics both on the development and test sets, since it allows to better adjust the relative importance of each translation model. However, both techniques achieve a very competitive performance. Results improve, according to BLEU, from 0.13 to 0.16, and from 0.11 to 0.14, for the development and test sets, respectively.

We measured the statistical signficance of the overall improvement in BLEU.n4 attained with respect to the baseline results by applying the bootstrap resampling technique described by Koehn (2004b). The 95% confidence intervals extracted from the test set after 10,000 samples are the following: $I_{\mathrm{EU-base}} = [0.0642, 0.0939]$, $I_{\mathrm{WNG-base}} = [0.0788, 0.1112]$, $I_{\mathrm{EU+WNG-best}} = [0.1221, 0.1572]$. Since the intervals are not ovelapping, we can conclude that the performance of the best combined method is statistically higher than the ones of the two baseline systems.

[8]We used values $\omega_{tmEU} = 0.1$, $\omega_{tmWNG} = 0.9$, $\theta_{tmEU} = 0.1$, and $\theta_{tmWNG} = 0.01$

4.5 How much in-domain data is needed?

In principle, the more in-domain data we have the better, but these may be difficult or expensive to collect. Thus, a very interesting issue in the context of our work is how much in-domain data is needed in order to improve results attained using out-of-domain data alone. To answer this question we focus on the 'EU+WNG' strategy and analyze the impact on performance (BLEU.n4) of specialized models extracted from an incrementally bigger number of example glosses. The results are presented in the plot of Figure 1. We compute three variants separately, by considering the use of the in-domain data: only for the translation model (TM), only for the language model (LM), and simultaneously in both models (TM+LM). In order

Translation Model	Language Model	BLEU.n4	NIST.n5	GTM.e1	GTM.e2	METEOR
development						
EUWNG	WNG	0.1288	3.7677	0.3949	0.2832	0.3711
EUWNG	EU + D1 + D2 + WNG	0.1182	3.6034	0.3835	0.2759	0.3552
EUWNG	EU + D1 + D2 + WNG (TUNED)	0.1554	3.8925	0.4081	0.2944	0.3998
EU+WNG	WNG	0.1384	3.9743	0.4096	0.2936	0.3804
EU+WNG	EU + D1 + D2 + WNG	0.1235	3.7652	0.3911	0.2801	0.3606
EU+WNG	EU + D1 + D2 + WNG (TUNED)	0.1618	4.1415	0.4234	0.3029	0.4130
test						
EUWNG	WNG	0.1123	3.6777	0.3829	0.2771	0.3595
EUWNG	EU + D1 + D2 + WNG	0.1183	3.5819	0.3737	0.2772	0.3518
EUWNG	EU + D1 + D2 + WNG (TUNED)	0.1290	3.6478	0.3920	0.2810	0.3885
EU+WNG	WNG	0.1227	3.8970	0.3997	0.2872	0.3723
EU+WNG	EU + D1 + D2 + WNG	0.1199	3.7353	0.3846	0.2812	0.3583
EU+WNG	EU + D1 + D2 + WNG (TUNED)	0.1400	3.8930	0.4084	0.2907	0.3963

Table 4: MT Results on development and test sets for the two strategies for combining translations models.

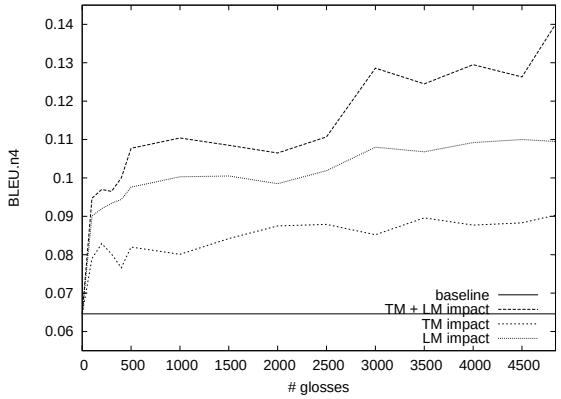


Figure 1: Impact of the size of in-domain data on MT system performance for the test set.

to avoid the possible effect of over-fitting we focus on the behavior on the test set. Note that the optimization of parameters is performed at each point in the x-axis using only the development set.

A significant initial gain of around 0.3 BLEU points is observed when adding as few as 100 glosses. In all cases, it is not until around 1,000 glosses are added that the 'EU+WNG' system stabilizes. After that, results continue improving as more in-domain data are added. We observe a very significant increase by just adding around 3,000 glosses. Another interesting observation is the boosting effect of the combination of TM and LM specialized models. While individual curves for TM and LM tend to be more stable with more than 4,000 added examples, the TM+LM curve still shows a steep increase in this last part.

5 Error Analysis

We inspected results at the sentence level based on the GTM F-measure ($e = 1$) for the best configuration of the 'EU+WNG' system. 196 sentences out from the 500 obtain an F-measure equal to or higher than 0.5 on the development set (181 sentences in the case of test set), whereas only 54 sentences obtain a score lower than 0.1. These numbers give a first idea of the relative usefulness of our system. Table 5 shows some translation cases selected for discussion. For instance, Case 1 is a clear example of unfair low score. The problem is that source and reference are not parallel but 'quasi-parallel'. Both glosses define the same concept but in a different way. Thus, metrics based on rewarding lexical similarities are not well suited for these cases. Cases 2, 3, 4 are examples of proper cooperation between 'EU' and 'WNG' models. 'EU' models provides recall, for instance by suggesting translation candidates for 'bombs' or 'price below'. 'WNG' models provide precision, for instance by choosing the right translation for 'an attack' or 'the act of'.

We also compared the 'EU+WNG' system to SYSTRAN. In the case of SYSTRAN 167 sentences obtain a score equal to or higher than 0.5 whereas 79 sentences obtain a score lower than 0.1. These numbers are slightly under the performance of the 'EU+WNG' system. Table 6 shows some translation cases selected for discussion. Case 1 is again an example of both systems obtaining very low scores because of 'quasi-parallelism'. Cases 2 and 3 are examples of SYSTRAN outperforming our system. In case 2 SYSTRAN exhibits higher precision in the translation of 'accompanying' and 'illustration', whereas in case 3 it shows higher recall by suggesting appropriate translation candidates for 'fibers', 'silkworm', 'cocoon', 'threads', and 'knitting'. Cases

F_E	F_W	F_{EW}	Source	Out_E	Out_W	Out_{EW}	Reference
0.0000	0.1333	0.1111	of the younger of two boys with the same family name	de acuerdo con el más joven de dos boys con la misma familia fama	de la younger de dos boys tiene el mismo nombre familia	de acuerdo con el más joven de dos muchachos tiene el mismo nombre familia	**que tiene menos edad**
0.2857	0.2500	0.5000	an attack by dropping bombs	atacar por cayendo **bombas**	**ataque** realizado por dropping bombs	**ataque** realizado por cayendo **bombas**	ataque con bombas
0.1250	0.7059	0.5882	the act of informing by verbal report	acto de la información por verbales ponencia	**acción y efecto** de informing por verbal **explicación**	**acción y efecto** de informaba por verbales **explicación**	acción y efecto de informar con una explicación verbal
0.5000	0.0000	0.5000	a price below the standard price	un **precio por debajo de la** norma precio	una price below número estándar price	un **precio por debajo de** la estándar precio	precio que está por debajo de lo normal

Table 5: MT output analysis of the 'EU', 'WNG' and 'EU+WNG' systems. F_E, F_W and F_{EW} refer to the GTM ($e = 1$) F-measure attained by the 'EU', 'WNG' and 'EU+WNG' systems, respectively. 'Source', Out_E, Out_W and Out_{EW} refer to the input and the output of the systems. 'Reference' corresponds to the expected output.

4 and 5 are examples where our system outperforms SYSTRAN. In case 4, our system provides higher recall by suggesting an adequate translation for 'top of something'. In case 5, our system shows higher precision by selecting a better translation for 'rate'. However, we observed that SYSTRAN tends in most cases to construct sentences exhibiting a higher degree of grammaticality.

6 Conclusions

In this work, we have enriched every synset in Spanish WordNet with a preliminary gloss, which can be later updated in a lighter process of manual revision. Though imperfect, this material constitutes a very valuable resource. For instance, WordNet glosses have been used in the past to generate sense tagged corpora (Mihalcea and Moldovan, 1999), or as external knowledge for Question Answering systems (Hovy et al., 2001).

We have also shown the importance of using a small set of in-domain parallel sentences in order to adapt a phrase-based general SMT system to a new domain. In particular, we have worked on specialized language and translation models and on their combination with general models in order to achieve a proper balance between precision (specialized in-domain models) and recall (general out-of-domain models). A substantial increase is consistently obtained according to standard MT evaluation metrics, which has been shown to be statistically significant in the case of BLEU. Broadly speaking, we have shown that around 3,000 glosses (very short sentence fragments) suffice in this domain to obtain a significant improvement. Besides, all the methods used are language independent, assumed the availability of the required in-domain additional resources.

In the future we plan to work on domain independent translation models built from WordNet itself. We may use the WordNet topology to provide translation candidates weighted according to the given domain. Moreover, we are experimenting the applicability of current Word Sense Disambiguation (WSD) technology to MT. We could favor those translation candidates showing a closer semantic relation to the source. We believe that coarse-grained is sufficient for the purpose of MT.

Acknowledgements

This research has been funded by the Spanish Ministry of Science and Technology (ALIADO TIC2002-04447-C02) and the Spanish Ministry of Education and Science (TRANGRAM, TIN2004-07925-C03-02). Our research group, TALP Research Center, is recognized as a Quality Research Group (2001 SGR 00254) by DURSI, the Research Department of the Catalan Government. Authors are grateful to Patrik Lambert for providing us with the implementation of the Simplex Method, and specially to German Rigau for motivating in its origin all this work.

References

Jordi Atserias, Luis Villarejo, German Rigau, Eneko Agirre, John Carroll, Bernardo Magnini, and Piek

F_{EW}	F_S	Source	Out_{EW}	Out_S	Reference
0.0000	0.0000	a newspaper that is published every day	periódico que se publica diario	un periódico que se publica cada día	publicación periódica monotemática
0.1818	0.8333	brief description accompanying an illustration	breve descripción adjuntas un aclaración	breve descripción **que acompaña una ilustración**	pequeña descripción que acompaña una ilustración
0.1905	0.7333	fibers from silkworm cocoons provide threads for knitting	fibers desde silkworm cocoons proporcionan threads para knitting	las **fibras** de los **capullos** del **gusano de seda proporcionan** los **hilos** de rosca para **hacer punto**	fibras de los capullos de gusano de seda que proporcionan hilos para tejer
1.0000	0.0000	the top of something	parte superior de una cosa	la tapa algo	parte superior de una cosa
0.6667	0.3077	a rate at which something happens	un **ritmo** al que sucede algo	una tarifa en la cual algo sucede	ritmo al que sucede una cosa

Table 6: MT output analysis of the 'EU+WNG' and SYSTRAN systems. F_{EW} and F_S refer to the GTM ($e = 1$) F-measure attained by the 'EU+WNG' and SYSTRAN systems, respectively. 'Source', Out_{EW} and Out_S refer to the input and the output of the systems. 'Reference' corresponds to the expected output.

Vossen. 2004. The MEANING Multilingual Central Repository. In *Proceedings of 2nd GWC*.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *Proceedings of ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, Robert L. Mercer, , and Paul S. Roossin. 1988. A statistical approach to language translation. In *Proceedings of COLING'88*.

George Doddington. 2002. Automatic Evaluation of Machine Translation Quality Using N-gram Co-Occurrence Statistics. In *Proceedings of the 2nd Internation Conference on Human Language Technology*, pages 138–145.

C. Fellbaum, editor. 1998. *WordNet. An Electronic Lexical Database*. The MIT Press.

Eduard Hovy, Ulf Hermjakob, and Chin-Yew Lin. 2001. The Use of External Knowledge of Factoid QA. In *Proceedings of TREC*.

Philipp Koehn. 2003. Europarl: A Multilingual Corpus for Evaluation of Machine Translation. Technical report, http://people.csail.mit.edu/-people/koehn/publications/europarl/.

Philipp Koehn. 2004a. Pharaoh: a Beam Search Decoder for Phrase-Based Statistical Machine Translation Models. In *Proceedings of AMTA'04*.

Philipp Koehn. 2004b. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of EMNLP'04*.

María Antonia Martí, editor. 1996. *Gran diccionario de la Lengua Española*. Larousse Planeta, Barcelona.

I. Dan Melamed, Ryan Green, and Joseph P. Turian. 2003. Precision and Recall of Machine Translation. In *Proceedings of HLT/NAACL'03*.

Rada Mihalcea and Dan Moldovan. 1999. An Automatic Method for Generating Sense Tagged Corpora. In *Proceedings of AAAI*.

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.

Franz Josef Och. 2002. *Statistical Machine Translation: From Single-Word Models to Alignment Templates*. Ph.D. thesis, RWTH Aachen.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. Bleu: a method for automatic evaluation of machine translation, IBM Research Report, RC22176. Technical report, IBM T.J. Watson Research Center.

Andreas Stolcke. 2002. SRILM - An Extensible Language Modeling Toolkit. In *Proceedings of ICSLP'02*.

Stephan Vogel and Alicia Tribble. 2002. Improving Statistical Machine Translation for a Speech-to-Speech Translation Task. In *Proceedings of ICSLP-2002 Workshop on Speech-to-Speech Translation*.

Vox, editor. 1990. *Diccionario Actual de la Lengua Española*. Bibliograf, Barcelona.

William T. Vetterling William H. Press, Saul A. Teukolsky and Brian P. Flannery. 2002. *Numerical Recipes in C++: the Art of Scientific Computing*. Cambridge University Press.

Speeding Up Full Syntactic Parsing by Leveraging Partial Parsing Decisions

Elliot Glaysher and **Dan Moldovan**
Language Computer Corporation
1701 N. Collins Blvd. Suite 2000
Richardson, TX 75080
{eglaysher,moldovan}@languagecomputer.com

Abstract

Parsing is a computationally intensive task due to the combinatorial explosion seen in chart parsing algorithms that explore possible parse trees. In this paper, we propose a method to limit the combinatorial explosion by restricting the CYK chart parsing algorithm based on the output of a chunk parser. When tested on the three parsers presented in (Collins, 1999), we observed an approximate three–fold speedup with only an average decrease of 0.17% in both precision and recall.

1 Introduction

1.1 Motivation

Syntactic parsing is a computationally intensive and slow task. The cost of parsing quickly becomes prohibitively expensive as the amount of text to parse grows. Even worse, syntactic parsing is a prerequisite for many natural language processing tasks. These costs make it impossible to work with large collections of documents in any reasonable amount of time.

We started looking into methods and improvements that would speed up syntactic parsing. These are divided into simple software engineering solutions, which are only touched on briefly, and an optimization to the CYK parsing algorithm, which is the main topic of this paper.

While we made large speed gains through simple software engineering improvements, such as internal symbolization, optimizing critical areas, optimization of the training data format, et cetera, the largest *individual* gain in speed was made by modifying the CYK parsing algorithm to leverage the decisions of a syntactic chunk parser so that it avoided combinations that conflicted with the output of the chunk parser.

1.2 Previous Work

Chart parsing is a method of building a parse tree that systematically explores combinations based on a set of grammatical rules, while using a chart to store partial results. The general CYK algorithm is a bottom-up parsing algorithm that will generate all possible parse trees that are accepted by a formal grammar. Michael Collins, first in (1996), and then in his PhD thesis (1999), describes a modification to the standard CYK chart parse for natural languages which uses probabilities instead of simple context free grammars.

The CYK algorithm considers all possible combinations. In Figure 1, we present a CYK chart graph for the sentence "The red balloon flew away." The algorithm will search the pyramid, from left to right, from the bottom to the top. Each box contains a pair of numbers that we will refer to as the span, which represent the sequence of words currently being considered. Calculating each "box" in the chart means trying all combinations of the lower parts of the box's sub-pyramid to form possible sub-parse trees. For example, one calculates the results for the span $(1, 4)$ by trying to combine the results in $(1, 1)$ and $(2, 4)$, $(1, 2)$ and $(3, 4)$, and $(1, 3)$ and $(4, 4)$.

In (Collins, 1999), Collins describes three new parsers. The Model 2 gives the best output, parsing section 23 at 88.26% precision and 88.05% recall in 40 minutes. The Model 1 is by far the fastest of the three, parsing section 23 of Treebank (Marcus et al., 1994) at 87.75% precision and 87.45% recall in 26 minutes.

Syntactic Chunking is the partial parsing process of segmenting a sentence into non-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 295–300,
Sydney, July 2006.

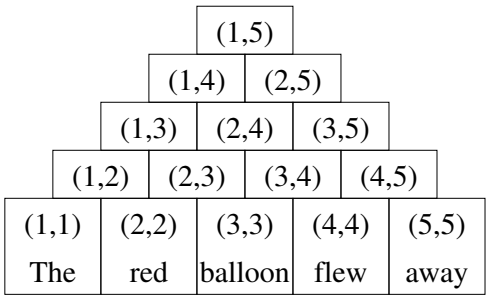


Figure 1: The CYK parse visualized as a pyramid. CYK will search from the left to right, bottom to top.

overlapping "chunks" of syntactically connected words. (Tjong Kim Sang and Buchholz, 2000) Unlike a parse tree, a set of syntactic chunks has no hierarchical information on how sequences of words relate to each other. The only information given is an additional label describing the chunk.

We use the YamCha (Kudo and Matsumoto, 2003; Kudo and Matsumoto, 2001) chunker for our text chunking. When trained on all of Penn Treebank , except for section 23 and tested on section 23, the model had a precision of 95.96% and a recall of 96.08%. YamCha parses section 23 of Treebank in 36 seconds.

Clause Identification is the partial parsing process of annotating the hierarchical structure of clauses—groupings of words that contain a subject and a predicate (Tjong Kim Sang and Déjean, 2001). Our clause identifier is an implementation of (Carreras et al., 2002), except that we use C5.0 as the machine learning method instead of Carreras' own TreeBoost algorithm (Carreras and Márquez, 2001). When trained and scored on the CoNLL 2001 shared task data[1] with the results of our chunker, our clause identifier performs at 90.73% precision, 73.72% recall on the development set and 88.85% precision, 70.22% recall on the test set.

In this paper, we describe modifications to the version of the CYK algorithm described in (Collins, 1999) and experiment with the modifications to both our proprietary parser and the (Collins, 1999) parser.

[1] http://www.cnts.ua.ac.be/conll2001/clauses/clauses.tgz

2 Methods

2.1 Software Optimizations

While each of the following optimizations, individually, had a smaller effect on our parser's speed than the CYK restrictions, collectively, simple software engineering improvements resulted in the largest speed increase to our syntactic parser. In the experiments section, we will refer to this as the "Optimized" version.

Optimization of the training data and internal symbolization: We discovered that our parser was bound by the number of probability hash-table lookups. We changed the format for our training data/hash keys so that they were as short as possible, eliminating deliminators and using integers to represent a closed set of POS tags that were seen in the training data, reducing the two to four byte POS tags such as "VP" or "ADJP" down to single byte integers. In the most extreme cases, this reduces the length of non-word characters in a hash from 28 characters to 6. The training data takes up less space, hashes faster, and many string comparisons are reduced to simple integer comparisons.

Optimization of our hash-table implementation: The majority of look ups in the hash-table at runtime were for non-existent keys. We put a bloomfilter on each hash bucket so that such lookups would often be trivially rejected, instead of having to compare the lookup key with every key in the bucket. We also switched to the Fowler/Noll/Vo (Noll, 2005) hash function, which is faster and has less collisions then our previous hash function.

Optimization of critical areas: There were several areas in our code that were optimized after profiling our parser.

Rules based pre/post-processing: We were able to get very minor increases in precision, recall and speed by adding hard coded rules to our parser that handle things that are handled poorly, specifically parenthetical phrases and quotations.

2.2 CYK restrictions

In this section, we describe modifications that restrict the chart search based on the output of a partial parser (in this case, a chunker) that marks groups of constituents.

First, we define a span to be a pair $c = (s, t)$, where s is the index of the first word in the span and t is the index of the last word in the span. We then define a set S, where S is the set of spans

$c_1, \ldots, c_n$ that represent the restrictions placed on the CYK parse. We say that c_1 and c_2 overlap iff $s_1 < s_2 \leq t_1 < t_2$ or $s_2 < s_1 \leq t_2 < t_1$, and we note it as $c_1 \sim c_2$.[2]

When using the output of a chunker, S is the set of spans that describe the non-VP, non-PP chunks where $t_i - s_i > 0$.

During the CYK parse, after a span's start and end points are selected, but before iterating across all splits of that span and their generative rules, we propose that the span in question be checked to make sure that it does not overlap with any span in set S. We give the pseudocode in Algorithm 1, which is a modification of the parse() function given in Appendix B of (Collins, 1999).

Algorithm 1 The modified parse() function

```
initialize()
for span = 2 to n do
  for start = 1 to n − span + 1 do
    end ← start + span − 1
    if ∀x ∈ S(x ≁ (start, end)) then
      complete(start, end)
    end if
  end for
end for
X ← edge in chart[1,n,TOP] with highest
probability
return X
```

For example, given the chunk parse:

$[\text{The red balloon}]_{NP}$ $[\text{flew}]_{VP}$ $[\text{away}]_{ADVP}$,

$S = \{(1,3)\}$ because there is only one chunk with a length greater than 1.

Suppose we are analyzing the span $(3,4)$ on the example sentence above. This span will be rejected, as it overlaps with the chunk $(1,3)$; the leaf nodes "balloon" and "flew" are not going to be children of the same parsetree parent node. Thus, this method does not compute the generative rules for all the splits of the spans $\{(2,4),(2,5),(3,4),(3,5)\}$. This will also reduce the number of calculations done when calculating higher spans. When computing $(1,4)$ in this example, time will be saved since the spans $(2,4)$ and $(3,4)$ were not considered. This example is visualized in Figure 2.

A more complex, real–world example from section 23 of Treebank is visualized in Figure 3, using the sentence "Under an agreement signed by the Big Board and the Chicago Mercantile Exchange, trading was temporarily halted in Chicago." This sentence has three usable chunks, $[\text{an agreement}]_{NP}$, $[\text{the Big Board}]_{NP}$, and $[\text{the Chicago Mercantile Exchange}]_{NP}$. This example shows the effects of the above algorithm on a longer sentence with multiple chunks.

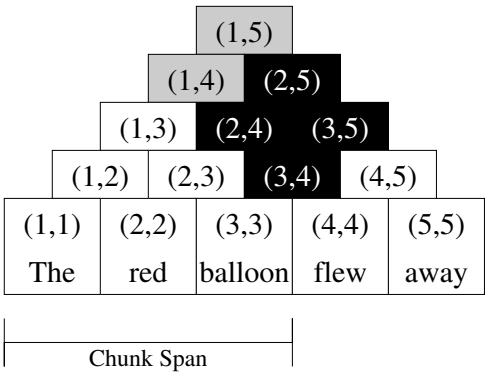


Figure 2: The same CYK example as in Figure 1. Blacked out box spans will not be calculated, while half toned box spans do not have to calculate as many possibilities because they depend on an uncalculated span.

3 Experiments & Results

3.1 Our parser with chunks

Our parser uses a simplified version of the model presented in (Collins, 1996). For this experiment,we tested four versions of our internal parser:

- Our original parser. No optimizations or chunking information.
- Our original parser with chunking information.
- Our optimized parser without chunking information.
- Our optimized parser with chunking information.

For parsers that use chunking information, the runtime of the chunk parsing is included in the parser's runtime, to show that total gains in runtime offset the cost of running the chunker.

We trained the chunk parser on all of Treebank except for section 23, which will be used as the test set. We trained our parser on all of Treebank except for section 23. Scoring of the parse trees

[2]This notation was originally used in (Carreras et al., 2002).

Figure 3: The CYK chart for the chunk parsed sentence "[Under]$_{PP}$ [an agreement]$_{NP}$ [signed]$_{VP}$ [by]$_{PP}$ [the Big Board]$_{NP}$ [and]$_{NP}$ [the Chicago Mercantile Exchange]$_{NP}$, [trading]$_{NP}$ [was temporarily halted]$_{VP}$ [in]$_{PP}$ [Chicago]$_{NP}$." The color coding scheme is the same as in Figure 2.

	Precision	Recall	Time
Original	82.79%	83.19%	25'45"
With chunks	84.40%	83.74%	7'37"
Optimized	83.86%	83.24%	4'28"
With chunks	84.42%	84.06%	1'35"

Table 1: Results from our parser on Section 23

	Precision	Recall	Time
Model 1	87.75%	87.45%	26'18"
With chunks	87.63%	87.27%	8'54"
Model 2	88.26%	88.05%	40'00"
With chunks	88.04%	87.87%	13'47"
Model 3	88.25%	88.04%	42'24"
With chunks	88.10%	87.89%	14'58"

Table 2: Results from the Collins parsers on Section 23 with chunking information

was done using the EVALB package that was used to score the (Collins, 1999) parser. The numbers represent the labeled bracketing of all sentences; not just those with 40 words or less.

The experiment was run on a dual Pentium 4, 3.20Ghz machine with two gigabytes of memory.

The results are presented in Table 1.

The most notable result is the greatly reduced time to parse when chunking information was added. Both versions of our parser saw an average three–fold increase in speed by leveraging chunking decisions. We also saw small increases in both precision and recall.

3.2 Collins Parsers with chunks

To show that this method is general and does not exploit weaknesses in the lexical model of our parser, we repeated the previous experiments with the three models of parsers presented in the (Collins, 1999). We made sure to use the exact same chunk post-processing rules in the Collins parser code to make sure that the same chunk information was being used. We used Collins' training data. We did not retrain the parser in any way to optimize for chunked input. We only modified the parsing algorithm.

Once again, the chunk parser was trained on all of Treebank except for section 23, the trees are evaluated with EVALB, and these experiments were run on the same dual Pentium 4 machine.

These results are presented in Table 2.

Like our parser, each Collins parser saw a slightly under three fold increase in speed. But unlike our parser, all three models of the Collins parser saw slight decreases in accuracy, averaging at -0.17% for both precision and recall. We theorize that this is because the errors in our lexical model are more severe than the errors in the chunks, but the Collins parser models make fewer errors in word grouping at the leaf node level than the chunker does. We theorize that a more accurate chunker would result in an increase in the precision and recall of the Collins parsers, while preserving the substantial speed gains.

	Precision	Recall	Time
Optimized	*83.86%*	*83.24%*	*4'28"*
With chunks	*84.42%*	*84.06%*	*1'35"*
With clauses	83.66%	83.06%	5'02"
With both	84.20%	83.84%	2'26"

Table 3: Results from our parser on Section 23 with clause identification information. Data copied from the first experiment has been italicized for comparison.

3.3 Clause Identification

Encouraged by the improvement brought by using chunking as a source of restrictions, we used the data from our clause identifier.

Again, our clause identifier was derived from (Carreras et al., 2002), using boosted C5.0 decision trees instead of their boosted binary decision tree method, which performs below their numbers: 88.85% precision, 70.22% recall on the CoNLL 2001 shared task test set.

These results are presented in Table 3.

Adding clause detection information hurt performance in every category. The increases in runtime are caused by the clause identifier's runtime complexity of over $O(n^3)$. The time to identify clauses is greater then the speed increases gained by using the output as restrictions.

In terms of the drop in precision and recall, we believe that errors from the clause detector are grouping words together that are not all constituents of the same parent node. While errors in a chunk parse are relatively localized, errors in the hierarchical structure of clauses can affect the entire parse tree, preventing the parser from exploring the correct high-level structure of the sentence.

4 Future Work

While the modification given in section 2.2 is specific to CYK parsing, we believe that placing restrictions based on the output of a chunk parser is general enough to be applied to any generative, statistical parser, such as the Charniak parser (2000), or a Lexical Tree Adjoining Grammar based parser (Sarkar, 2000). Restrictions can be placed where the parser would explore possible trees that would violate the boundaries determined by the chunk parser, pruning paths that will not yield the correct parse tree.

5 Conclusion

Using decisions from partial parsing greatly reduces the time to perform full syntactic parses, and we have presented a method to apply the information from partial parsing to full syntactic parsers that use a variant of the CYK algorithm. We have shown that this method is not specific to the implementation of our parser and causes a negligible effect on precision and recall, while decreasing the time to parse by an approximate factor of three.

References

Xavier Carreras and Lluís Márquez. 2001. Boosting trees for anti-spam email filtering. In *Proceedings of RANLP-01, 4th International Conference on Recent Advances in Natural Language Processing*, Tzigov Chark, BG.

Xavier Carreras, Lluís Màrquez, Vasin Punyakanok, and Dan Roth. 2002. Learning and inference for clause identification. In *ECML '02: Proceedings of the 13th European Conference on Machine Learning*, pages 35–47, London, UK. Springer-Verlag.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the first conference on North American chapter of the Association for Computational Linguistics*, pages 132–139, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Michael John Collins. 1996. A new statistical parser based on bigram lexical dependencies. In Arivind Joshi and Martha Palmer, editors, *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, pages 184–191, San Francisco. Morgan Kaufmann Publishers.

Michael John Collins. 1999. *Head-driven statistical models for natural language parsing*. Ph.D. thesis, University of Pennsylvania. Supervisor-Mitchell P. Marcus.

Taku Kudo and Yuji Matsumoto. 2001. Chunking with support vector machines. In *NAACL '01: Second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies 2001*, pages 1–8, Morristown, NJ, USA. Association for Computational Linguistics.

Taku Kudo and Yuji Matsumoto. 2003. Fast methods for kernel-based text analysis. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 24–31, Morristown, NJ, USA. Association for Computational Linguistics.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1994. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330.

Landon C. Noll. 2005. Fnv hash. http://www.isthe.com/chongo/tech/comp/fnv/.

Anoop Sarkar. 2000. Practical experiments in parsing using tree adjoining grammars. In *Proceedings of the Fifth International Workshop on Tree Adjoining Grammars*, Paris, France.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the conll-2000 shared task: Chunking. In Claire Cardie, Walter Daelemans, Claire Nedellec, and Erik Tjong Kim Sang, editors, *Proceedings of CoNLL-2000 and LLL-2000*, pages 127–132. Lisbon, Portugal.

Erik F. Tjong Kim Sang and Hervé Déjean. 2001. Introduction to the conll-2001 shared task: Clause identification. In Walter Daelemans and Rémi Zajac, editors, *Proceedings of CoNLL-2001*, pages 53–57. Toulouse, France.

Parsing Aligned Parallel Corpus by Projecting Syntactic Relations from Annotated Source Corpus

Shailly Goyal **Niladri Chatterjee**
Department of Mathematics
Indian Institute of Technology Delhi
Hauz Khas, New Delhi - 110 016, India
{shailly_goyal, niladri_iitd}@yahoo.com

Abstract

Example-based parsing has already been proposed in literature. In particular, attempts are being made to develop techniques for language pairs where the source and target languages are different, e.g. Direct Projection Algorithm (Hwa et al., 2005). This enables one to develop parsed corpus for target languages having fewer linguistic tools with the help of a resource-rich source language. The DPA algorithm works on the assumption of *Direct Correspondence* which simply means that the relation between two words of the source language sentence can be projected directly between the corresponding words of the parallel target language sentence. However, we find that this assumption does not hold good all the time. This leads to wrong parsed structure of the target language sentence. As a solution we propose an algorithm called pseudo DPA (pDPA) that can work even if Direct Correspondence assumption is not guaranteed. The proposed algorithm works in a recursive manner by considering the embedded phrase structures from outermost level to the innermost. The present work discusses the pDPA algorithm, and illustrates it with respect to English-Hindi language pair. Link Grammar based parsing has been considered as the underlying parsing scheme for this work.

1 Introduction

Example-based approaches for developing parsers have already been proposed in literature. These approaches either use examples from the same language, e.g., (Bod et al., 2003; Streiter, 2002), or they try to imitate the parse of a given sentence using the parse of the corresponding sentence in some other language (Hwa et al., 2005; Yarowsky and Ngai, 2001). In particular, Hwa et al. (2005) have proposed a scheme called *direct projection algorithm* (DPA) which assumes that the relation between two words in the source language sentence is preserved across the corresponding words in the parallel target language. This is called Direct Correspondence Assumption (DCA).

However, with respect to Indian languages we observed that the DCA does not hold good all the time. In order to overcome the difficulty, in this work, we propose an algorithm based on a variation of the DCA, which we call *pseudo Direct Correspondence Assumption* (pDCA). Through pDCA the syntactic knowledge can be transferred even if not all syntactic relations may be projected directly from the source language to the target language in toto. Further, the proposed algorithm projects the relations between phrases instead of projecting relations between words. Keeping in line with (Hwa et al., 2005), we call this algorithm as *pseudo Direct Projection Algorithm* (pDPA).

The present work discusses the proposed parsing scheme for a new (target) language with the help of a parser that is already available for a language (source) and using word-aligned parallel corpus of the two languages under consideration. We propose that the syntactic relationships between the chunks of the input sentence T (of the target language) are given depending upon the relationships of the corresponding chunks in the translation S of T. Along with the parsed structure of the input, the system also outputs the constituent structure (phrases) of the given input sen-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 301–308,
Sydney, July 2006.

tence.

In this work, we first discuss the proposed scheme in a general framework. We illustrate the scheme with respect to parsing of Hindi sentences using the Link Grammar (LG) based parser for English and the experimental results are discussed. Before that in the following section we discuss Link Grammar briefly.

2 Link Grammar and Phrases

Link grammar (LG) is a theory of syntax which builds simple relations between pairs of words, rather than constructing constituents in tree-like hierarchy. For example, in an SVO language like English, the verb forms a subject link (`S-`) to some word on its left, and an object link (`O+`) with some word on its right. Nouns make the subject link (`S+`) to some word (verb) on its right, or object link (`O-`) to some word on its left.

The English Link Grammar Parser (Sleator and Temperley, 1991) is a syntactic parser of English based on LG. Given a sentence, the system assigns to it a syntactic structure, which consists of a set of labeled links connecting pairs of words. The parser also produces a "constituent" representation of a sentence (showing noun phrases, verb phrases, etc.). It is a dictionary-based system in which each word in the dictionary is associated with a set of links. Most of the links have some associated suffixes to provide various information (e.g., gender (`m/f`), number (`s/p`)), describing some properties of the underlying word. The English link parser lists total of 107 links. Table 1 gives a list of some important links of English LG along with the information about the words on their left/right and some suffixes.

Link	Word in Left	Word in Right	Suffixes
A	Premodifier	Noun	-
D	Determiners	Nouns	s/m,c/u
J	Preposition	Object of the preposition	s/p
M	Noun	Post-nominal Modifier	p/v/g/a
MV	Verbs/adjectives	Modifying phrase	p/a/i/l/x
O	Transitive verb	Direct or indirect object	s/p
P	Forms of "be"	Complement of "be"	p/v/g/a
PP	Forms of "have"	Past participle	-
S	Subject	Finite verb	s/p, i, g

Table 1: Some English Links and Their Suffixes

As an example, consider the syntactic structure and constituent representation of the sentence given below.

```
        +--------Ss--------+
        |    +----Jp---+   |
 +--Ds-+-Mp-+   +-Dmc-+   +-Pa-+
 |     |    |   |     |   |    |
the teacher of the  boys is   good

    (S (NP (NP The teacher)
           (PP of (NP the boys)))
       (VP is)
       (ADJP good).)
```

It may be noted that in the phrase structure of the above sentence, verb phrase as obtained from the phrase parser has been modified to some extent. The algorithm discussed in this work assumes verb phrases as *the main verb along with all the auxiliary verbs*.

For ease of presentation and understanding, we classify phrase relations as *Inter-Phrase* and *Intra-phrase* relations. Since the phrases are often embedded, different levels of phrase relations are obtained. From the outermost level to the innermost, we call them as "first level", "second level" of relations and so on. One should note that an i^{th} level Intra-phrase relation may become Inter-phrase relation at a higher level.

As an example, consider the parsing and phrase structure of the English sentence given above. In the first level the Inter-phrase relations (corresponding to the phrases "`the teacher of the boys`", "`is`" and "`good`") are `Ss` and `Pa` and the remaining links are Intra-phrase relations. In the second level the only Inter-phrase relationship is `Mp` (connecting "`the teacher`" and "`the boys`"), and the Intra-phrase relations are `Ds`, `Jp` and `Dmc`. In third and the last level, `Jp` is the Inter-phrase relationship and `Dmc` is the Intra-phrase relation (corresponding to "`of`" and "`the boys`").

The algorithm proposed in Section 4 uses pDCA to first establish the relations of the target language corresponding to the first-level Inter-phrase relations of the source language sentence. Then recursively it assigns the relations corresponding to the inner level relations.

3 DCA vis-à-vis pDCA

Direct Correspondence Assumption (DCA) states that the relation between words in source language sentence can be projected as the relations between corresponding words in the (literal) translation in the target language. Direct Projection Algorithm

(DPA), which is based on DCA, is a straightforward projection procedure in which the dependencies in an English sentence are projected to the sentence's translation, using the word-level alignments as a bridge. DPA also uses some monolingual knowledge specific to the projected-to language. This knowledge is applied in the form of Post-Projection transformation.

However with respect to many language pairs syntactic relationships between the words *cannot* always be imitated to project a parse structure from source language to target language. For illustration consider the sentence given in Figure 1. We try to project the links from English to Hindi in Figure 1(a) and Hindi to Bangla in Figure 1(b). For Hindi sentence, links are given as discussed by Goyal and Chatterjee (2005a; 2005b).

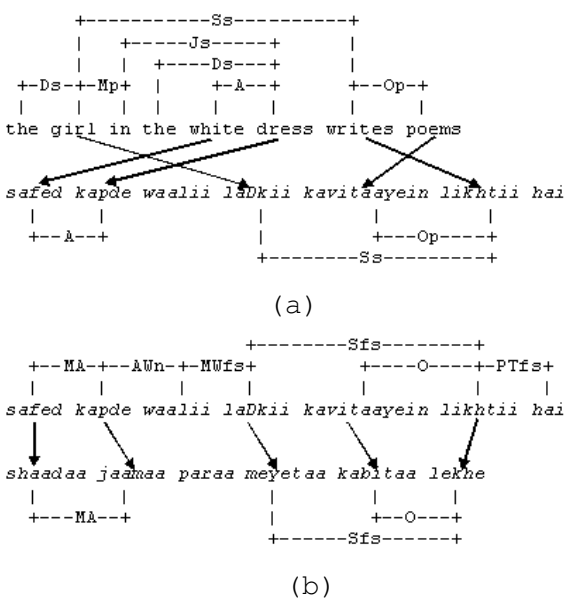


Figure 1: Failure of DCA

We observe that in the parse structure of the target language sentences, neither all relations are correct nor the parse tree is complete. Thus, we observe that DPA leads to, if not wrong, a very shallow parse structure. Further, Figure 1(b) suggests that DCA fails not only for languages belonging to different families (English-Hindi), but also for languages belonging to the same family (Hindi-Bangla).

Hence it is necessary that the parsing algorithm should be able to differentiate between the links which can be projected directly and the links which cannot. Further it needs to identify the chunks of the target language sentence that cannot be linked even after projecting the links from the source language sentence. Thus we propose pseudo Direct Correspondence Assumption (pDCA) where not all relations can be projected directly. The projection algorithm needs to take care of the following three categories of links:

Category 1: Relationship between two chunks in the source language can be projected to the target language with minor or no changes (for example, subject-verb, object-verb relationships in the above illustration). It may be noted that since except for some suffix differences (due to morphological variations), the relation is same in the source and the target language.

Category 2: Relationship between two chunks in the source language can be projected to the target language with major changes. For example, in the English sentence given in Figure 2(a), the relationship between `the girl` and `in the white dress` is `Mp`, i.e. "nominal modifier (preposition phrase)". In the corresponding phrases *ladkii* and *safed kapde waalii* of Hindi, although the relationship is same, i.e., "nominal modifier", the type of nominal modifier is changing to *waalaa/waale/waalii*-adjective. If the distinction between the types of nominal modifiers is not maintained, the parsing will be very shallow. Hence the modification in the link is necessary.

Category 3: Relationship between two chunks in the target language is either entirely different or can not be captured from the relationship between the corresponding chunk(s) in the source language. For example, the relationship between the main verb and the auxiliary verb of the Hindi sentence in Figure 2(a) can not be defined using the English parsing. Such phrases should be parsed independently.

The proposed algorithm is based on the above-described concept of pDCA which gives the parse structure of the sentences given in Fig. 2.

While working with Indian languages, we found that outermost Inter-phrase relations usually belong to Category 1, and remaining relations belong to Category 2. Generally an innermost Intra-phrase relation (like verb phrase) belongs to Category 3. Thus, outermost Inter-phrase relations can usually be projected to target language directly, innermost Intra-phrase relations for the target language which are independent of the source language should be decided on the basis of language specific study and remaining relationship should

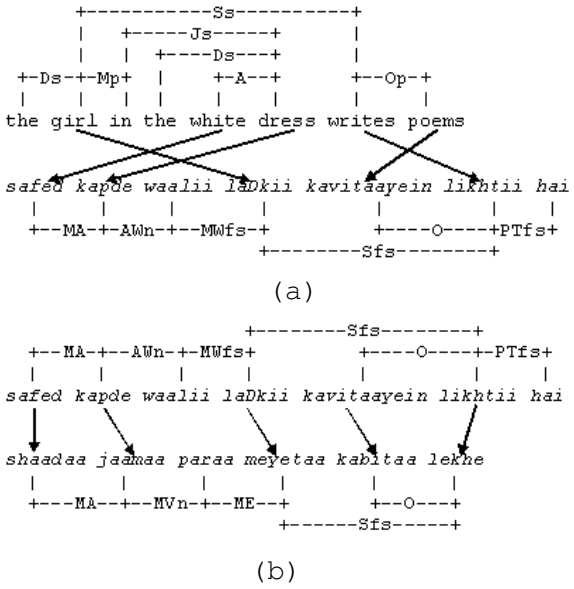


Figure 2: Parsing Using pDCA

be modified before projection from source to target language.

4 The Proposed Algorithm

DPA (Hwa et al., 2005) discusses projection procedure for five different cases of word alignment of source-target language: one-to-one, one-to-none, one-to-many, many-to-one and many-to-many. As discussed earlier, DPA is not sufficient for many cases. For example, in case of one-to-many alignment, the proposed solution is to first create a new empty word that is set as head of all multiply aligned words in target language sentence, and then the relation is projected accordingly. But, in such cases, relations between these multiply-aligned words can not be given, and thus the resulting parsing becomes shallow. The proposed algorithm (pDPA) overcomes these shortcomings as well.

The pDPA works in the following way. It recursively identifies the phrases of the target language sentence, and assigns the links between the two phrases/words of the target language sentence by using the links between the corresponding phrases/words in the source language sentence. It may be noted that link between phrases means link between the head words of the corresponding phrases. Assignment of links starts from the outermost level phrases. Syntactic relations between the constituents of the target language phrase(s) for which the syntactic structure does not correspond with the corresponding phrase(s) in the target language are given independently. A list of link rules is maintained which keeps the information about modification(s) required in a link while projecting from the source language to the target language. These rules are limited to closed category words, to parts of speech projected from source language, or to easily enumerated lexical categories.

Figure 3 describes the algorithm. The algorithm takes an input sentence (T) and the parsing and the constituent structure of its parallel sentence (S). Further S and T are assumed to be word-aligned. Initially, S and T are passed to the module Project-From(), which identifies the constituent phrases of S and the relations between them. Then each set of phrases and relations is passed to the module ParseFrom(). ParseFrom() module takes as input two source phrases/words, relation between them, and corresponding target phrases. It projects the corresponding relations in the target language sentence T. ParseFromSpecial() module is required if the relation between phrases of source language can not be projected so directly to the target language. Module Parse() assigns links between the constituent words of the target language phrases $\in \mathscr{P}$. Notations used in the algorithm are as follows:

- By $T' \sim S'$ we mean that T' is aligned with S', T' and S' being some text in the target and source language, respectively.
- Given a language, the head of a phrase is usually defined as the keyword of the phrase. For example, for a verb phrase, the head word is the main verb.
- $\mathscr{P}$ is the exhaustive set of target language phrases for which Intra-phrase relations are independent of the corresponding source language phrases.
- Rule list $\mathscr{R}$ is the list of source-target language specific rules which specifies the modifications in the source language relations to be projected appropriately in the target language.
- Given the parse and constituent structure of a text S, $\Psi_{ij} = \langle S_i,\ S_j,\ L \rangle$, where L is the relation between the constituent phrases/words S_i and S_j of S. $\Psi'_{ij} = \langle T_i,\ T_j \rangle$, $T_i \sim S_i$ and $T_j \sim S_j$. Further, $\Phi_{ij} = \langle \Psi_{ij}, \Psi'_{ij} \rangle$.

ProjectFrom(S', T'): // S' is a source
// language sentence or phrase, $T' \sim S'$
{
IF $T' \in \mathscr{P}$
THEN Parse(T');
ELSE
$S' = \{S_1, S_2, \ldots, S_n\}$; // S_is are
//constituent phrases/words of S'
$T' = \{T_1, T_2, \ldots, T_n\}$ // $T_i \sim S_i$
Find all $\Psi_{ij} = \langle S_i, S_j, L \rangle$ from S' and
corresponding $\Psi'_{ij} = \{T_i, T_j\}$ from T';
$\Phi_{ij} = \langle \Psi_{ij}, \Psi'_{ij} \rangle$
For all i, j, push ($\mathscr{S}, \Phi_{ij}$);
While !empty($\mathscr{S}$)
Φ = pop($\mathscr{S}$);
IF $L \notin \mathscr{L}$
THEN ParseFrom(Φ);
ELSE ParseFromSpecial(Φ);
}
Parse(T'): // T' is a target language phrase
{
Assign links between constituent words of T'
using target language specific rules;
}
ParseFrom(Φ): // $\Phi = \langle \Psi, \Psi' \rangle$;
// $\Psi = \langle S_1, S_2, L \rangle; \Psi' = \langle T_1, T_2 \rangle$;
{
IF $T_1 \neq \{\phi\}$ & $T_2 \neq \{\phi\}$ THEN
Find head words $t_1 \in T_1$ and $t_2 \in T_2$;
Assign relation L' between t_1 and t_2; // L'
//is target language link corresponding
//to L identified using $\mathscr{R}$
IF T_1 is a phrase and not already parsed
THEN ProjectFrom(S_1, T_1);
IF T_2 is a phrase and not already parsed
THEN ProjectFrom(S_2, T_2);
}
ParseFromSpecial(Φ): // $\Phi = \langle \Psi, \Psi' \rangle$;
// $\Psi = \langle S_1, S_2, L \rangle; \Psi' = \langle T_1, T_2 \rangle$;
{
Use target language specific rules to identify if
the relation between T_1 and T_2 is given by L';
IF true THEN ParseFrom(Φ);
ELSE
Assign required relations using rules;
IF T_1 is a phrase and not already parsed
THEN ProjectFrom(S_1, T_1);
IF T_2 is a phrase and not already parsed
THEN ProjectFrom(S_2, T_2);
}

Figure 3: pseudo Direct Projection Algorithm

- $\mathscr{S}$ is a stack of Φ_{ij}s.
- $\mathscr{L}$ is the set of source language relations whose occurrence in parse of some S' may lead to different structure of T', where $T' \sim S'$.

In the following sections we discuss in detail the scheme for parsing Hindi sentences using parse structure of the corresponding English sentence. Along with the parse structure of the input, the phrase structure is also obtained.

5 Case study: English to Hindi

Prior requirements for developing a parsing scheme for the target language using the proposed algorithm are: development of target language links, word alignment technique, phrase identification procedure, creation of rule set $\mathscr{R}$, morphological analysis, development of ParseFromSpecial() module. In this section we discuss these details for adapting a parser for Hindi using English LG based parser.

Hindi Links. Goyal and Chatterjee (2005a; 2005b) have developed links for Hindi Link Grammar along with their suffixes. Some of the Hindi links are briefly discussed in the Table 2. It may be noted that due to the free word order of Hindi, direction can not be specified for some links, i.e., for such links "Word in Left" and "Word in Right" (second and third column of Table 2) shall be read as "Word on one side" and "Word on the other side", respectively.

Link	Word in Left	Word in Right	Directed
S	Subject	Main verb	NO
SN	*ne*	Main verb	NO
O	Object	Main verb	NO
J	noun/pronoun	postposition	YES
MV	verb modifier	Main verb	NO
MA	Adjective	Noun	YES
ME	*aa-e-ii* form of verb	Noun	YES
MW	*waalaa/waale/ waalii*	Noun	YES
PT	*taa-te-tii* form of verb	declension of verb *honaa*	YES
D	Determiner	Head noun	YES

Table 2: Some Hindi Links

Word Alignment. The algorithm requires that the source and target language sentences are word aligned. Some English-Hindi word alignment algorithms have already been developed, e.g.

(Aswani and Gaizauskas, 2005). However, for the current implementation alignment has been done manually with the help of an online English-Hindi dictionary[1].

Identification of Phrases and Head Words.

Verb Phrases. Corresponding to any main verb v_i present in the Hindi sentence, a verb phrase is formed by considering all the auxiliary verbs following it. A list of Hindi auxiliary verbs, along with the linkage requirements has been maintained. This list is used to identify and link verb phrases. Main verb of the verb phrase is considered to be the head word.

Noun and Postposition[2] Phrases. English NP is translated in Hindi as either NP or PP[3]. Also, English PP can be translated as either NP or PP. If the Hindi noun is followed by any postposition, then that postposition is attached with the noun to get a PP. In this case the postposition is considered as the head. Hindi NP corresponding to some English NP is the *maximal span* of the words (in Hindi sentence) aligned with the words in the corresponding English NP. The Hindi noun whose English translation is involved in establishing the Inter-phrase link is the head word. Note that if the last word (noun) in this Hindi NP is followed by any postposition (resulting in some PP), then that postposition is also included in the NP concerned . In this case the postposition is the head of the NP. The system maintains a list of Hindi postpositions to identify Hindi PPs.

For example, consider the translation pair `the lady in the room had cooked the food`$\sim$ *kamre* (`room`) *mein* (`in`) *baiThii huii* (`-`) *aurat* (`lady`) *ne* (`-`) *khaanaa* (`food`) *banaayaa* (`cooked`) *thaa* (`-`).

The phrase structure of the English sentence is (`(`NP_1 (`(`NP_2 `the lady)` (`(`PP_1 `in (`NP_3 `the room)))` (`(`VP_1 `had cooked)` (`$NP_4$ `the food)`.

Here, some of the Hindi phrases are as follows:

kamre mein and *aurat ne* are identified as Hindi PP corresponding to English PP_1 and NP_2. The words *mein* and *ne* are considered as their head words, respectively. Since the maximal span of translation of words of English NP_1 is *kamre mein baiThii huii aurat* which is followed by postposition *ne*, the Hindi phrase corresponding to NP_1 is *kamre mein baiThii huii aurat ne* with *ne* as the head word. As *huii* and *thii*, which follow the verbs *baiThii*[4] and *banaayaa* respectively, are present in the auxiliary verb list, Hindi VPs are obtained as *baiThii huii* and *banaayaa thaa* (corresponding to VP_1).

Phrase Set $\mathscr{P}$. Hindi verb phrase and postposition phrases are linked independent of the corresponding phrases in the English sentence. Thus, $\mathscr{P} = \{VP, PP\}$.

Rule List $\mathscr{R}$. Below we enlist some of the rules defined for parsing Hindi sentences using the English links (E-links) of the parallel English sentences. Note that these rules are dependent on the target language.

Corresponding to E-link S: If the Hindi subject is followed by *ne*, then the subject makes a `Jn` link with *ne*, and *ne* makes an `SN` link with the verb.

Corresponding to E-link O: If the Hindi object is followed by *ko*, then the object makes a `Jk` link with *ko*, and *ko* makes an `OK` link with the verb.

Corresponding to E-links M, MX: English NPs may have preposition phrase, present participle, past participle or adjective as postnominal modifiers which are translated as prenominal modifiers, or as relative clause in Hindi. The structure of postnominal modifier, however, may not be preserved in the Hindi sentence. If the sentence is not complex, then the corresponding Hindi link may be one of `MA` (adjective), `MP` (postposition phrase), `MT` (present participle), `ME` (past participle), or `MW` (*waalaa/waale/waalii*-adjective). An appropriate link is to be assigned in Hindi sentence after identification of the structure of the nominal modifier. These cases are handled in the module ParseFromSpecial(). The segment of the module that handles English `Mp` link is given in Figure 4.

Further, since morphological information of Hindi words can not be always extracted using corresponding English sentence, a morphological analyzer is required to extract the information[5]. For the current implementation, morphological infor-

[1] www.sanskrit.gde.to/hindi/dict/eng-hin-itrans.html

[2] In Hindi prepositions are used immediately after the noun. Thus, we refer to them as "postposition".

[3] PP for English is preposition phrase and for Hindi it stands for postposition phrase.

[4] We observe that English PP as postnominal modifier may be translated as verbal prenominal modifier in Hindi and in such cases some unaligned word is effectively a verb.

[5] For Hindi, some work is being carried out in this direction, e.g., http://ccat.sas.upenn.edu/plc/ tamilweb/hindi.html

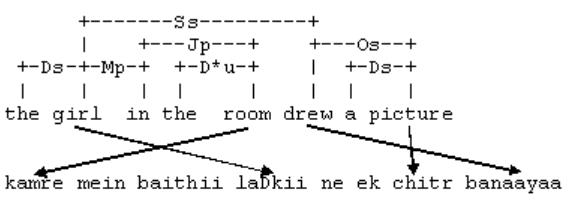
ParseFromSpecial(Φ): // $\Phi = \langle \Psi, \Psi' \rangle$;
// $\Psi = \langle S_1, S_2, L \rangle$; $\Psi' = \langle T_1, T_2 \rangle$;
{
IF $L =$ Mp THEN //S_1 and S_2 are NP and PP, resp.
IF T_2 is followed by some verb, v, not aligned with any word in S THEN
$T_3 =$ VP corresponding to v;
Parse(T_3);
Find head word $t_1 \in T_1$;
Assign MT/ME link between v and t_1;
Assign MVp link between postposition (in T_2) and v;
ProjectFrom(S_1, T_1); ProjectFrom(S_2, T_2);
ELSE
ParseFrom(Φ);
ELSE
Check for other cases of L;
}

Figure 4: ParseFromSpecial() for 'Mp' Link

mation is being extracted using some rules in simpler cases, and manually for more complex cases.

5.1 Illustration with an Example

Consider the English sentence (S) the girl in the room drew a picture, its parsed and constituent structure as given in Figure 5. Further, the corresponding Hindi sentence (T), and the word-alignment is also given.



(S (NP (NP The girl) (PP in (NP the room)))
(VP drew) (NP a picture).)

Figure 5: An Example

The step-by-step parsing of the sentence as per the pDPA is given below.

ProjectFrom(S, T):

$S = \{S_1, S_2, S_3\}$, where S_1, S_2, S_3 are the phrases the girl in the room, drew and a picture, respectively. From the definition of Hindi phrases, corresponding T_i's are identified as "*kamre mein baithii laDkii ne*", "*banaayaa*" and "*ek chitr*". From the parse structure of S, Φ's are obtained as $\Phi_{12} = \langle\langle S_1, S_2, \text{Ss}\rangle, \langle T_1, T_2\rangle\rangle$ and $\Phi_{23} = \langle\langle S_2, S_3, \text{Os}\rangle, \langle T_2, T_3\rangle\rangle$. These Φ's are pushed in the stack $\mathscr{S}$ and further processing is done one-by-one for each of them. We show the further process for the Φ_{12}.

Since Ss $\notin \mathscr{L}$, ParseFrom(Φ_{12}) is executed.

ParseFrom(Φ_{12}):

The algorithm identifies t_1 = *ne*, t_2 = *banaayaa*. The Hindi link corresponding to Ss will be SN. The module ProjectFrom(S_1, T_1) is then called.

ProjectFrom(S_1, T_1):

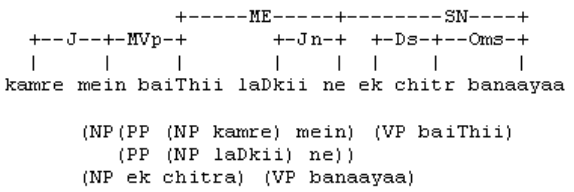
$S_1 = \{S_{11}, S_{12}\}$, where S_{11} and S_{12} are the girl and in the room, respectively. Corresponding T_{11} and T_{12} are *ladkii ne* and *kamre mein*. Thus, $\Phi = \langle\langle S_{11}, S_{12}, \text{Mp}\rangle, \langle T_{11}, T_{12}\rangle\rangle$. Since $L =$ Mp $\in \mathscr{L}$, ParseFromSpecial(Φ) is called.

ParseFromSpecial(Φ): (Refer to Figure 4)

Since T_2 is followed by an unaligned verb *baithii*, the algorithm finds T_3 as *baithii*, and t_1 as *ne*. It assigns ME link between *baithii* and *ne*. Further, MVp link is assigned between *mein* and *baithii*. Then ProjectFrom(S_{11}, T_{11}) and ProjectFrom(S_{12}, T_{12}) are called. Since both T_{11} and $T_{12} \in \mathscr{S}$, J and Jn links are assigned between constituent words of T_{11} and T_{12}, respectively, using Hindi-specific rules.

Similarly, Φ_{23} is parsed.

The final parse and phrase structure of the sentence are obtained as given in Figure 6.



(NP(PP (NP kamre) mein) (VP baiThii)
(PP (NP laDkii) ne))
(NP ek chitra) (VP banaayaa)

Figure 6: Parsing of Example Sentence

6 Experimental Results

Currently the system can handle the following types of phrases in different simple sentences.

Noun Phrase. There can be four basic elements of an English NP[6]: determiner, pre-modifier, noun (essential), post-modifier. The system can handle any combination of the following: adjective, noun, present participle or past participle as pre-modifier, and adjective, present participle, past participle or preposition phrase as post-modifier. Note that some of these cases may be translated as complex sentence in Hindi (e.g., book on the table $\sim$ *jo kitaab mej par rakhii hai*). We are working upon such cases.

[6]Pronouns as NPs are simple.

Verb Phrase. The system can handle all the four aspects (indefinite, continuous, perfect and perfect continuous) for all three tenses. Other cases of VPs (e.g., modals, passives, compound verbs) can be handled easily by just identifying and putting the corresponding auxiliary verbs and their linking requirements in the auxiliary verb list.

Since the system is not fully automated yet, we could not test our system on a large corpus. The system has been tested on about 200 sentences following the specific phrase structures mentioned above. These sentences have been taken randomly from translation books, stories books and advertisement materials. These sentences were manually parsed and a total of 1347 links were obtained. These links were compared with the system's output. Table 3 summarizes the findings.

Correct Links	: 1254
Links with wrong suffix	: 47
Wrong links	: 22
Links missing	: 31

Table 3: Experimental Results

After analyzing the results, we found that

- For some links, suffixes were wrong. This was due to insufficiency of rules identifying morphological information.
- Due to incompleteness of some cases of ParseFromSpecial() module, some wrong links were assigned. Also, some links which should not have been projected, were projected in the Hindi sentence. We are working towards exploring these cases in detail.
- Some links were found missing in the parsing since corresponding sentence structures are yet to be considered in the scheme.

7 Concluding Remarks

The present work focuses on development of Example based parsing scheme for a pair of languages in general, and for English to Hindi in particular.

Although the current work is motivated by (Hwa et al., 2005), the algorithm proposed herein provides a more generalized version of the projection algorithm by making use of some target language specific rules while projecting links. This provide more flexibility in the projection algorithm. The flexibility comes from the fact that unlike DPA the algorithm can project links from the source language to the target language even if the translations are not literal. Use of rules at the projection level gives more robust parsing and reduces the need of post-editing. The proposed scheme should work for other target languages also provided the relevant rules can be identified. Further, since LG can be converted to Dependency Grammar (DG) (Sleator and Temperley, 1991), this work can be easily extended for languages for which DG implementation is available.

At present, we have focused on developing parsing scheme for simple sentences. Work has to be done to parse complex sentences. Once a sizeable parsed corpus is generated, it can be used for developing the parser for a target language using bootstrapping. We are currently working on these lines for developing a Hindi parser.

References

Niraj Aswani and Robert Gaizauskas. 2005. A hybrid approach to aligning sentences and words in English-Hindi parallel corpora. In *ACL 2005 Workshop on Building and Using Parallel Texts: Data-driven machine translation and Beyond*.

Rens Bod, Remko Scha, and Khalil Sima'an, editors. 2003. *Data-Oriented Parsing*. Stanford: CSLI Publications.

Shailly Goyal and Niladri Chatterjee. 2005a. Study of Hindi noun phrase morphology for developing a link grammar based parser. *Language in India*, 5.

Shailly Goyal and Niladri Chatterjee. 2005b. Towards developing a link grammar based parser for Hindi. In *Proc. of Workshop on Morphology*, Bombay.

Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. 2005. Bootstrapping parsers via syntactic projection across parallel texts. *Natural Language Engineering*, 11(3):311–325, September.

Daniel Sleator and Davy Temperley. 1991. Parsing English with a link grammar. Computer Science technical report CMU-CS-91-196, Carnegie Mellon University, October.

Oliver Streiter. 2002. Abduction, induction and memorizing in corpus-based parsing. In *ESSLLI-2002 Workshop on Machine Learning Approaches in Computational Linguistics,*, Trento, Italy.

David Yarowsky and Grace Ngai. 2001. Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora. In *NAACL-2001*, pages 200–207.

Reduced *n*-gram models for English and Chinese corpora

Le Q Ha, P Hanna, D W Stewart and F J Smith
School of Electronics, Electrical Engineering and Computer Science,
Queen's University Belfast
Belfast BT7 1NN, Northern Ireland, United Kingdom
lequanha@lequanha.com

Abstract

Statistical language models should improve as the size of the *n*-grams increases from 3 to 5 or higher. However, the number of parameters and calculations, and the storage requirement increase very rapidly if we attempt to store all possible combinations of *n*-grams. To avoid these problems, the reduced *n*-grams' approach previously developed by O'Boyle (1993) can be applied. A reduced *n*-gram language model can store an entire corpus's phrase-history length within feasible storage limits. Another theoretical advantage of reduced *n*-grams is that they are closer to being semantically complete than traditional models, which include all *n*-grams. In our experiments, the reduced *n*-gram Zipf curves are first presented, and compared with previously obtained conventional *n*-grams for both English and Chinese. The reduced *n*-gram model is then applied to large English and Chinese corpora. For English, we can reduce the model sizes, compared to 7-gram traditional model sizes, with factors of 14.6 for a 40-million-word corpus and 11.0 for a 500-million-word corpus while obtaining 5.8% and 4.2% improvements in perplexities. For Chinese, we gain a 16.9% perplexity reductions and we reduce the model size by a factor larger than 11.2. This paper is a step towards the modeling of English and Chinese using semantically complete phrases in an *n*-gram model.

1 Introduction to the Reduced *N*-Gram Approach

Shortly after this laboratory first published a variable *n*-gram algorithm (Smith and O'Boyle, 1992), O'Boyle (1993) proposed a statistical method to improve language models based on the removal of overlapping phrases.

A distortion in the use of phrase frequencies had been observed in the small railway timetable Vodis Corpus when the bigram "RAIL ENQUIRIES" and its super-phrase "BRITISH RAIL ENQUIRIES" were examined. Both occur 73 times, which is a large number for such a small corpus. "ENQUIRIES" follows "RAIL" with a very high probability when it is preceded by "BRITISH." However, when "RAIL" is preceded by words other than "BRITISH," "ENQUIRIES" does not occur, but words like "TICKET" or "JOURNEY" may. Thus, the bigram "RAIL ENQUIRIES" gives a misleading probability that "RAIL" is followed by "ENQUIRIES" irrespective of what precedes it. At the time of their research, O'Boyle reduced the frequencies of "RAIL ENQUIRIES" by subtracting the frequency of the larger trigram, which gave a probability of zero for "ENQUIRIES" following "RAIL" if it was not preceded by "BRITISH." The phrase with a new reduced frequency is called a reduced phrase.

Therefore, a phrase can occur in a corpus as a reduced *n*-gram in some places and as part of a larger reduced *n*-gram in other places. In a reduced model, the occurrence of an *n*-gram is not counted when it is a part of a larger reduced *n*-gram. One algorithm to detect/identify/extract reduced *n*-grams from a corpus is the so-called reduced *n*-gram algorithm. In 1992, O'Boyle was able to use it to analyse the Brown corpus of American English (Francis and Kucera, 1964) (of one million word tokens, whose longest phrase-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 309–315,
Sydney, July 2006.

length is 30), which was a considerable improvement at the time. The results were used in an *n*-gram language model by O'Boyle, but with poor results, due to lack of statistics from such a small corpus. We have developed and present here a modification of his method, and we discuss its usefulness for reducing *n*-gram perplexity.

2 Similar Approaches and Capability

Recent progress in variable *n*-gram language modeling has provided an efficient representation of *n*-gram models and made the training of higher order *n*-grams possible. Compared to variable *n*-grams, class-based language models are more often used to reduce the size of a language model, but this typically leads to recognition performance degradation. Classes can alternatively be used to smooth a language model or provide back-off estimates, which have led to small performance gains. For the LOB corpus, the varigram model obtained 11.3% higher perplexity in comparison with the word-trigram model (Niesler and Woodland, 1996.)

Kneser (1996) built up variable-context length language models based on the North American Business News (NAB-240 million words) and the German Verbmobil (300,000 words with a vocabulary of 5,000 types.) His results show that the variable-length model outperforms conventional models of the same size, and if a moderate loss in performance is acceptable, that the size of a language model can be reduced drastically by using his pruning algorithm. Kneser's results improve with longer contexts and a same number of parameters. For example, reducing the size of the standard NAB trigram model by a factor of 3 results in a loss of only 7% in perplexity and 3% in the word error rate. The improvement obtained by Kneser's method depended on the length of the fixed context and on the amount of available training data. In the case of the NAB corpus, the improvement was 10% in perplexity.

Siu and Ostendorf (2000) developed Kneser's basic ideas further and applied the variable 4-gram, thus improving the perplexity and word error rate results compared to a fixed trigram model. They obtained word error reductions of 0.1 and 0.5% (absolute) in development and evaluation test sets, respectively. However, the number of parameters was reduced by 60%. By using the variable 4-gram, they were able to model a longer history while reducing the size of the model by more than 50%, compared to a regular trigram model, and at the same time improve both the test-set perplexity and recognition performance. They also reduced the size of the model by an additional 8%.

Other related work are those of Seymore and Rosenfeld (1996); Hu, Turin and Brown (1997); Blasig (1999); and Goodman and Gao (2000.)

In order to obtain an overview of variable *n*-grams, Table 1 combines all of their results.

COMBINATION OF LANGUAGE MODEL TYPES								
Basic *n*-gram	**Variable *n*-grams**	**Category**	**Skipping distance**	**Classes**	**#params**	**Perplexity**	**Size**	**Source**
Trigram√					987k	474	1M	LOB
		Bigram√			-	603.2		
		Trigram√			-	544.1		
	√	√			-	534.1		
Trigram√					743k	81.5	2M	Switch board Corpus
	Trigram√				379k	78.1		
	Trigram√		√		363k	78.0		
	Trigram√		√	√	338k	77.7		
	4-gram√				580k	108		
	4-gram√		√		577k	108		
	4-gram√		√	√	536k	107		
	5-gram√				383k	77.5		
	5-gram√		√		381k	77.4		
	5-gram√		√	√	359k	77.2		

Table 1. Comparison of combinations of variable *n*-grams and other Language Models

3 Reduced *N*-Gram Algorithm

The main goal of this algorithm (Ha, 2005) is to produce three main files from the training text:

- The file that contains all the complete *n*-grams appearing at least *m* times is called the PHR file ($m \geq 2$.)
- The file that contains all the *n*-grams appearing as sub-phrases, following the removal of the first word from any other complete *n*-gram in the PHR file, is called the SUB file.
- The file that contains any overlapping *n*-grams that occur at least *m* times in the SUB file is called the LOS file.

The final list of reduced phrases is called the FIN file, where

$$FIN := PHR + LOS - SUB \quad (1)$$

Before O'Boyle's work, a student (Craig) in an unpublished project used a loop algorithm that was equivalent to FIN:=PHR–SUB. This yields negative frequencies for some resulting *n*-grams with overlapping, hence the need for the LOS file.

There are 2 additional files

- To create the PHR file, a SOR file is needed that contains all the complete *n*-grams regardless of *m* (the SOR file is the PHR file in the special case where $m = 1$.) To create the PHR file, words are removed from the right-hand side of each SOR phrase in the SOR file until the resultant phrase appears at least *m* times (if the phrase already occurs more than *m* times, no words will be removed.)
- To create the LOS file, O'Boyle applied a POS file: for any SUB phrase, if one word can be added back on the right-hand side (previously removed when the PHR file was created from the SOR file), then one POS phrase will exist as the added phrase. Thus, if any POS phrase appears at least *m* times, its original SUB phrase will be an overlapping *n*-gram in the LOS file.

The application scope of O'Boyle's reduced *n*-gram algorithm is limited to small corpora, such as the Brown corpus (American English) of 1 million words (1992), in which the longest phrase has 30 words. Now their algorithm, re-checked by us, still works for medium size and large corpora. In order to work well for very large corpora, it has been implemented by file distribution and sort processes.

Ha, Seymour, Hanna and Smith (2005) have investigated a reduced *n*-gram model for the Chinese TREC corpus of the Linguistic Data Consortium (LDC) (http://www.ldc.upenn.edu/), catalog no. LDC2000T52.

4 Reduced *N*-Grams and Zipf's Law

By re-applying O'Boyle and Smith's algorithm, we obtained reduced *n*-grams from two English large corpora and a Chinese large corpus.

The two English corpora used in our experiments are the full text of articles appearing in the Wall Street Journal (WSJ) (Paul and Baker, 1992) for 1987, 1988, 1989, with sizes approximately 19 million, 16 million and 6 million tokens respectively; and the North American News Text (NANT) corpus from the LDC, sizing 500 million tokens, including Los Angeles Times & Washington Post for May 1994-August 1997, New York Times News Syndicate for July 1994-December 1996, Reuters News Service (General & Financial) for April 1994-December 1996 and Wall Street Journal for July 1994-December 1996. Therefore, the WSJ parts from the two English corpora are not overlapping together.

The Mandarin News corpus from the LDC, catalog no. LDC95T13 was obtained from the People's Daily Newspaper from 1991 to 1996 (125 million syllables); from the Xinhua News Agency from 1994 to 1996 (25 million syllables); and from transcripts of China Radio International broadcast from 1994 to 1996 (100 million syllables), altogether over 250 million syllables. The number of syllable types (i.e. unigrams) in the Mandarin News corpus is 6,800. Ha, Sicilia-Garcia, Ming and Smith (2003) produced a compound word version of the Mandarin News corpus with 50,000 types; this version was employed in our study for reduced *n*-grams.

We next present the Zipf curves (Zipf, 1949) for the English and Chinese reduced *n*-grams.

4.1 Wall Street Journal

The WSJ reduced *n*-grams can be created by the original O'Boyle-Smith algorithm implemented on a Pentium II 586 of 512MByte RAM for over 40 hours, the disk storage requirement being only 5GBytes.

The conventional 10-highest frequency WSJ words have been published by Ha et al. (2002) and the most common WSJ reduced unigrams, bigrams and trigrams are shown in Table 2. It illustrates that the most common reduced word is not THE; even OF is not in the top ten. These words are now mainly part of longer *n*-grams with large *n*.

The Zipf curves are plotted for reduced unigrams and *n*-grams in Figure 1 showing all the curves have slopes within [-0.6, -0.5]. The WSJ reduced bigram, trigram, 4-gram and 5-gram curves become almost parallel and straight, with a small observed noise between the reduced 4-gram and 5-gram curves when they cut each other at the beginning. Note that information theory tells us that an ideal information channel would be made of symbols with the same probability. So having a slope of –0.5 is closer than –1 to this ideal.

Rank	Unigrams		Bigrams		Trigrams	
	Freq	Token	Freq	Token	Freq	Token
1	4,273	Mr.	2,268	he said	1,231	terms weren't disclosed
2	2,469	but	2,052	he says	709	the company said
3	2,422	and	1,945	but the	664	as previously reported
4	2,144	the	1,503	but Mr.	538	he said the
5	1,918	says	1,332	and the	524	a spokesman for
6	1,660	or	950	says Mr.	523	the spokesman said
7	1,249	said	856	in addition	488	as a result
8	1,101	however	855	and Mr.	484	earlier this year
9	1,007	while	832	last year	469	in addition to
10	997	meanwhile	754	for example	466	according to Mr.

Table 2. Most common WSJ reduced *n*-grams

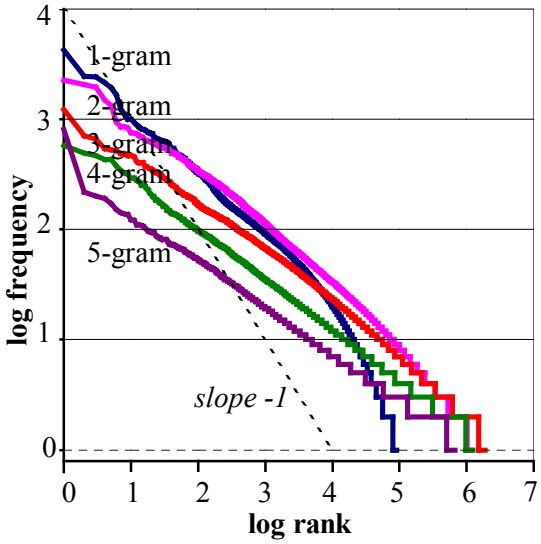


Figure 1. The WSJ reduced *n*-gram Zipf curves

4.2 North American News Text corpus

The NANT reduced *n*-grams are created by the improved algorithm after over 300 hours processing, needing a storage requirement of 100GBytes on a Pentium II 586 of 512MByte RAM.

Their Zipf curves are plotted for reduced unigrams and *n*-grams in Figure 2 showing all the curves are just sloped around [-0.54, -0.5]. The reduced unigrams of NANT still show the 2-slope behavior when it starts with slope –0.54 and then drop with slope nearly –2 at the end of the curve. We have found that the traditional *n*-grams also show this behaviour, with an initial slope of –1 changing to –2 for large ranks (Ha and Smith, 2004.)

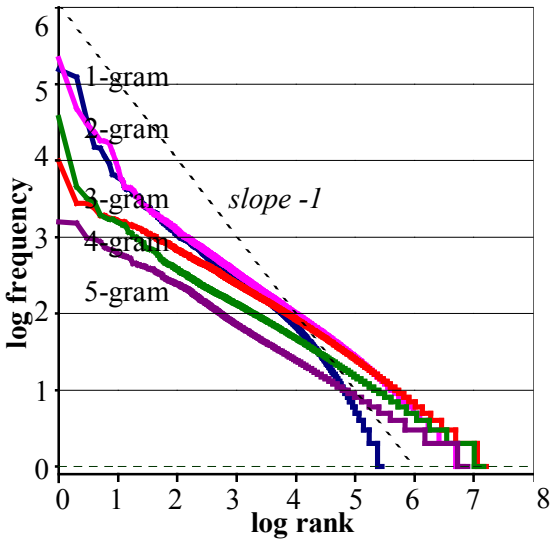


Figure 2. The NANT reduced *n*-gram Zipf curves

4.3 Mandarin News compound words

The Zipf curves are plotted for the smaller Chinese TREC reduced unigrams and *n*-grams were shown by Ha et al. (2005.)

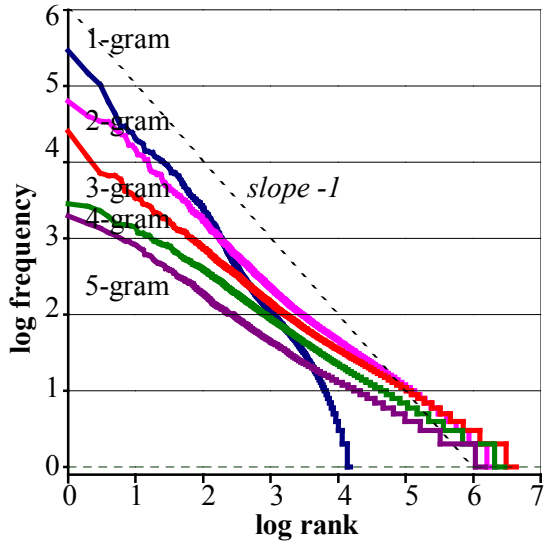


Figure 3. Mandarin reduced *n*-gram Zipf curves

The Mandarin News reduced word *n*-grams were created in 120 hours, using 20GB of disk space. The Zipf curves are plotted in Figure 3 showing that the unigram curve now has a larger slope than –1, it is around –1.2. All the *n*-gram curves are now straighter and more parallel than the traditional *n*-gram curves, have slopes within [-0.67, -0.5].

Usually, Zipf's rank-frequency law with a slope –1 is confirmed by empirical data, but the reduced *n*-grams for English and Chinese shown in Figure 1, Figure 2 and Figure 3 do not confirm it. In fact, various more sophisticated models for frequency distributions have been proposed by Baayen (2001) and Evert (2004.)

5 Perplexity for Reduced *N*-Grams

The reduced *n*-gram approach was used to build a statistical language model based on the weighted average model of O'Boyle, Owens and Smith (1994.) We rewrite this model in formulae (2) and (3)

$$wgt\left(w_j^i\right) = \log\left(f\left(w_j^{i-1}\right)\right) \times 2^{i-j+1} \tag{2}$$

$$P_{WA}\left(w_i \middle| w_{i-N+1}^{i-1}\right) = \frac{wgt(w_i) \times P(w_i) + \sum_{l=1}^{N-1} wgt\left(w_{i-l}^{i}\right) \times P\left(w_i \middle| w_{i-l}^{i-1}\right)}{\sum_{l=0}^{N-1} wgt\left(w_{i-l}^{i}\right)} \tag{3}$$

This averages the probabilities of a word w_i following the previous one word, two words, three words, etc. (i.e. making the last word of an *n*-gram.) The averaging uses weights that increase slowly with their frequency and rapidly with the length of *n*-gram. This weighted average model is a variable length model that gives results comparable to the Katz back-off method (1987), but is quicker to use.

The probabilities of all of the sentences w_1^m in a test text are then calculated by the weighted average model

$$P\left(w_1^m\right) = P_{WA}\left(w_1\right) P_{WA}\left(w_2 \middle| w_1\right) \ldots P_{WA}\left(w_m \middle| w_1^{m-1}\right) \tag{4}$$

and an average perplexity of each sentence is evaluated using Equation (5)

$$PP\left(w_1^m\right) = \exp\left(-\frac{1}{L}\sum_{i=1}^{L} Ln\left(P_{WA}\left(w_i \middle| w_1 w_2 \ldots w_{i-1}\right)\right)\right) \tag{5}$$

Ha et al. (2005) already investigated and analysed the main difficulties arising from perplexity calculations for our reduced model: a statistical model problem, an unseen word problem and an unknown word problem. Their solutions are applied in this paper also. Similar problems have been found by other authors, e.g. Martin, Liermann and Ney (1997); Kneser and Ney (1995.)

The perplexity calculations for both the English and Chinese reduced *n*-grams includes statistics on phrase lengths starting with unigrams, bigrams, trigrams…and on up to the longest phrase which occur in the reduced model. The perplexities of the WSJ reduced model are shown in Table 3, North American News Text corpus in Table 4 and Mandarin News words in Table 5.

The nature of the reduced model makes the reporting of results for limited sizes of *n*-grams to be inappropriate, although these are valid for a traditional *n*-gram model. Therefore we show results for several *n*-gram sizes for the traditional model, but only one perplexity for the reduced model.

Unknowns	Tokens	**0**
	Types	**0**
Traditional Model	Unigrams	762.69
	Bigrams	144.33
	Trigrams	**75.36**
	4-grams	60.73
	5-grams	56.85
	6-grams	55.66
	7-grams	55.29
Reduced Model		**70.98**
%Improvement of Reduced Model on baseline Trigrams		**5.81%**
Model size reduction		**14.56**

Table 3. Reduced perplexities for English WSJ

Unknowns	Tokens	**24**
	Types	**23**
Traditional Model	Unigrams	1,442.99
	Bigrams	399.61
	Trigrams	**240.52**
	4-grams	202.59
	5-grams	194.06
	6-grams	191.91
	7-grams	191.23
Reduced Model		**230.46**
%Improvement of Reduced Model on baseline Trigrams		**4.18%**
Model size reduction		**11.01**

Table 4. Reduced perplexities for English NANT

Unknowns	Tokens	**84**
	Types	**26**
Traditional Model	Unigrams	1,620.56
	Bigrams	377.43
	Trigrams	**179.07**
	4-grams	135.69
	5-grams	121.53
	6-grams	114.96
	7-grams	111.69
Reduced Model		**148.71**
%Improvement of Reduced Model on baseline Trigrams		**16.95%**
Model size reduction		**11.28**

Table 5. Reduced perplexities for Mandarin News words

In all three cases the reduced model produces a modest improvement over the traditional 3-gram model, but is not as good as the traditional 4-gram or higher models. However in all three cases the result is obtained with a significant reduction in model size, from a factor of 11 to almost 15 compared to the traditional 7-gram model size.

We did expect a greater improvement in perplexity than we obtained and we believe that a further look at the methods used to solve the difficult problems listed by Ha et al. (2005) (mentioned above) and others mentioned by Ha (2005) might lead to an improvement. Missing word tests are also needed.

6 Conclusions

The conventional *n*-gram language model is limited in terms of its ability to represent extended phrase histories because of the exponential growth in the number of parameters. To overcome this limitation, we have re-investigated the approach of O'Boyle (1993) and created reduced *n*-gram models. Our aim was to try to create an *n*-gram model that used semantically more complete *n*-grams than traditional *n*-grams in the expectation that this might lead to an improvement in language modeling. The improvement in perplexity is modest, but there is a large decrease in model size. So this represents an encouraging step forward, although still very far from the final step in language modelling.

Acknowledgements

The authors would like to thank Dr Ji Ming for his support and the reviewers for their valuable comments.

References

Douglas B. Paul and Janet B. Baker. 1992. The Design for the Wall Street Journal based CSR Corpus. In *Proc. of the DARPA SLS Workshop*, pages 357-361.

Francis J. Smith and Peter O'Boyle. 1992. The *N*-Gram Language Model. *The Cognitive Science of Natural Language Processing Workshop*, pages 51-58. Dublin City University.

George K. Zipf. 1949. *Human Behaviour and the Principle of Least Effort*. Reading, MA: Addison-Wesley Publishing Co.

Harald R. Baayen. 2001. *Word Frequency Distributions*. Kluwer Academic Publishers.

Jianying Hu, William Turin and Michael K. Brown. 1997. Language Modeling using Stochastic Automata with Variable Length Contexts. *Computer Speech and Language*, volume 11, pages 1-16.

Joshua Goodman and Jianfeng Gao. 2000. Language Model Size Reduction By Pruning And Clustering. *ICSLP'00*. Beijing, China.

Kristie Seymore and Ronald Rosenfeld. 1996. Scalable Backoff Language Models. *ICSLP'96*, pages 232-235.

Le Q. Ha and Francis. J. Smith. 2004. Zipf and Type-Token rules for the English and Irish languages. *MIDL workshop*. Paris.

Le Q. Ha, Elvira I. Sicilia-Garcia, Ji Ming and Francis J. Smith. 2002. Extension of Zipf's Law to Words and Phrases. *COLING'02*, volume 1, pages 315-320.

Le Q. Ha, Elvira I. Sicilia-Garcia, Ji Ming and Francis J. Smith. 2003. Extension of Zipf's Law to Word and Character *N*-Grams for English and Chinese. *CLCLP*, 8(1):77-102.

Le Q. Ha, Rowan Seymour, Philip Hanna and Francis J. Smith. 2005. Reduced *N*-Grams for Chinese Evaluation. *CLCLP*, 10(1):19-34.

Manhung Siu and Mari Ostendorf. 2000. Integrating a Context-Dependent Phrase Grammar in the Variable *N*-Gram framework. *ICASSP'00*, volume 3, pages 1643-1646.

Manhung Siu and Mari Ostendorf. 2000. Variable *N*-Grams and Extensions for Conversational Speech Language Modelling. *IEEE Transactions on Speech and Audio Processing*, 8(1):63-75.

Nelson Francis and Henry Kucera. 1964. *Manual of Information to Accompany A Standard Corpus of Present-Day Edited American English, for use with Digital Computers*. Department of Linguistics, Brown University, Providence, Rhode Island.

Peter L. O'Boyle. 1993. A study of an *N*-Gram Language Model for Speech Recognition. *PhD thesis.* Queen's University Belfast.

Peter O'Boyle, John McMahon and Francis J. Smith. 1995. Combining a Multi-Level Class Hierarchy with Weighted-Average Function-Based Smoothing. *IEEE Automatic Speech Recognition Workshop*. Snowbird, Utah.

Peter O'Boyle, Marie Owens and Francis J. Smith. 1994. A weighted average *N*-Gram model of natural language. *Computer Speech and Language*, volume 8, pages 337-349.

Ramon Ferrer I. Cancho and Ricard V. Solé. 2002. Two Regimes in the Frequency of Words and the Origin of Complex Lexicons. *Journal of Quantitative Linguistics*, 8(3):165-173.

Reinhard Blasig. 1999. Combination of Words and Word Categories in Varigram Histories. *ICASSP'99*, volume 1, pages 529-532.

Reinhard Kneser and Hermann Ney. 1995. Improved Backing-off for *M*-Gram Language Modeling. *ICASSP'95*, volume 1, pages 181-184. Detroit.

Reinhard Kneser. 1996. Statistical Language Modeling Using a Variable Context Length. *ICSLP'96*, volume 1, pages 494-497.

Slava M. Katz. 1987. Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. In *IEEE Transactions on Acoustics, Speech and Signal Processing*, volume ASSP-35, pages 400-401.

Stefan Evert. 2004. A Simple LNRE Model for Random Character Sequences. In *Proc. of the 7èmes Journées Internationales d'Analyse Statistique des Données Textuelles*, pages 411-422.

Sven C. Martin, Jörg Liermann and Hermann Ney. 1997. Adaptive Topic-Dependent Language Modelling Using Word-Based Varigrams. *EuroSpeech'97*, volume 3, pages 1447-1450. Rhodes.

Thomas R. Niesler and Phil C. Woodland. 1996. A Variable-Length Category-Based *N*-Gram Language Model. *ICASSP'96*, volume 1, pages 164-167.

Thomas R. Niesler. 1997. *Category-based statistical language models*. St. John's College, University of Cambridge.

Discriminative Classifiers for Deterministic Dependency Parsing

Johan Hall
Växjö University
jni@msi.vxu.se

Joakim Nivre
Växjö University and
Uppsala University
nivre@msi.vxu.se

Jens Nilsson
Växjö University
jha@msi.vxu.se

Abstract

Deterministic parsing guided by treebank-induced classifiers has emerged as a simple and efficient alternative to more complex models for data-driven parsing. We present a systematic comparison of memory-based learning (MBL) and support vector machines (SVM) for inducing classifiers for deterministic dependency parsing, using data from Chinese, English and Swedish, together with a variety of different feature models. The comparison shows that SVM gives higher accuracy for richly articulated feature models across all languages, albeit with considerably longer training times. The results also confirm that classifier-based deterministic parsing can achieve parsing accuracy very close to the best results reported for more complex parsing models.

1 Introduction

Mainstream approaches in statistical parsing are based on nondeterministic parsing techniques, usually employing some kind of dynamic programming, in combination with generative probabilistic models that provide an n-best ranking of the set of candidate analyses derived by the parser (Collins, 1997; Collins, 1999; Charniak, 2000). These parsers can be enhanced by using a discriminative model, which reranks the analyses output by the parser (Johnson et al., 1999; Collins and Duffy, 2005; Charniak and Johnson, 2005). Alternatively, discriminative models can be used to search the complete space of possible parses (Taskar et al., 2004; McDonald et al., 2005).

A radically different approach is to perform disambiguation deterministically, using a greedy parsing algorithm that approximates a globally optimal solution by making a sequence of locally optimal choices, guided by a classifier trained on gold standard derivations from a treebank. This methodology has emerged as an alternative to more complex models, especially in dependency-based parsing. It was first used for unlabeled dependency parsing by Kudo and Matsumoto (2002) (for Japanese) and Yamada and Matsumoto (2003) (for English). It was extended to labeled dependency parsing by Nivre et al. (2004) (for Swedish) and Nivre and Scholz (2004) (for English). More recently, it has been applied with good results to lexicalized phrase structure parsing by Sagae and Lavie (2005).

The machine learning methods used to induce classifiers for deterministic parsing are dominated by two approaches. Support vector machines (SVM), which combine the maximum margin strategy introduced by Vapnik (1995) with the use of kernel functions to map the original feature space to a higher-dimensional space, have been used by Kudo and Matsumoto (2002), Yamada and Matsumoto (2003), and Sagae and Lavie (2005), among others. Memory-based learning (MBL), which is based on the idea that learning is the simple storage of experiences in memory and that solving a new problem is achieved by reusing solutions from similar previously solved problems (Daelemans and Van den Bosch, 2005), has been used primarily by Nivre et al. (2004), Nivre and Scholz (2004), and Sagae and Lavie (2005).

Comparative studies of learning algorithms are relatively rare. Cheng et al. (2005b) report that SVM outperforms MaxEnt models in Chinese dependency parsing, using the algorithms of Yamada and Matsumoto (2003) and Nivre (2003), while Sagae and Lavie (2005) find that SVM gives better

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 316–323,
Sydney, July 2006.

performance than MBL in a constituency-based shift-reduce parser for English.

In this paper, we present a detailed comparison of SVM and MBL for dependency parsing using the deterministic algorithm of Nivre (2003). The comparison is based on data from three different languages – Chinese, English, and Swedish – and on five different feature models of varying complexity, with a separate optimization of learning algorithm parameters for each combination of language and feature model. The central importance of feature selection and parameter optimization in machine learning research has been shown very clearly in recent research (Daelemans and Hoste, 2002; Daelemans et al., 2003).

The rest of the paper is structured as follows. Section 2 presents the parsing framework, including the deterministic parsing algorithm and the history-based feature models. Section 3 discusses the two learning algorithms used in the experiments, and section 4 describes the experimental setup, including data sets, feature models, learning algorithm parameters, and evaluation metrics. Experimental results are presented and discussed in section 5, and conclusions in section 6.

2 Inductive Dependency Parsing

The system we use for the experiments uses no grammar but relies completely on inductive learning from treebank data. The methodology is based on three essential components:

1. Deterministic parsing algorithms for building dependency graphs (Kudo and Matsumoto, 2002; Yamada and Matsumoto, 2003; Nivre, 2003)
2. History-based models for predicting the next parser action (Black et al., 1992; Magerman, 1995; Ratnaparkhi, 1997; Collins, 1999)
3. Discriminative learning to map histories to parser actions (Kudo and Matsumoto, 2002; Yamada and Matsumoto, 2003; Nivre et al., 2004)

In this section we will define dependency graphs, describe the parsing algorithm used in the experiments and finally explain the extraction of features for the history-based models.

2.1 Dependency Graphs

A dependency graph is a labeled directed graph, the nodes of which are indices corresponding to the tokens of a sentence. Formally:

Definition 1 Given a set R of dependency types (arc labels), a *dependency graph* for a sentence $x = (w_1, \ldots, w_n)$ is a labeled directed graph $G = (V, E, L)$, where:

1. $V = \mathbf{Z}_{n+1}$
2. $E \subseteq V \times V$
3. $L : E \rightarrow R$

The set V of *nodes* (or *vertices*) is the set $\mathbf{Z}_{n+1} = \{0, 1, 2, \ldots, n\}$ ($n \in \mathbf{Z}^+$), i.e., the set of non-negative integers up to and including n. This means that every token index i of the sentence is a node ($1 \leq i \leq n$) and that there is a special node 0, which does not correspond to any token of the sentence and which will always be a root of the dependency graph (normally the only root). We use V^+ to denote the set of nodes corresponding to tokens (i.e., $V^+ = V - \{0\}$), and we use the term *token node* for members of V^+.

The set E of *arcs* (or *edges*) is a set of ordered pairs (i, j), where i and j are nodes. Since arcs are used to represent dependency relations, we will say that i is the *head* and j is the *dependent* of the arc (i, j). As usual, we will use the notation $i \rightarrow j$ to mean that there is an arc connecting i and j (i.e., $(i, j) \in E$) and we will use the notation $i \rightarrow^* j$ for the reflexive and transitive closure of the arc relation E (i.e., $i \rightarrow^* j$ if and only if $i = j$ or there is a path of arcs connecting i to j).

The function L assigns a dependency type (arc label) $r \in R$ to every arc $e \in E$.

Definition 2 A dependency graph G is *well-formed* if and only if:

1. The node 0 is a root.
2. Every node has in-degree at most 1.
3. G is connected.[1]
4. G is acyclic.
5. G is projective.[2]

Conditions 1–4, which are more or less standard in dependency parsing, together entail that the graph is a rooted tree. The condition of projectivity, by contrast, is somewhat controversial, since the analysis of certain linguistic constructions appears to

[1] To be more exact, we require G to be *weakly connected*, which entails that the corresponding undirected graph is connected, whereas a *strongly connected* graph has a *directed* path between any pair of nodes.

[2] An arc (i, j) is projective iff there is a path from i to every node k such that $i < j < k$ or $i > j > k$. A graph G is projective if all its arcs are projective.

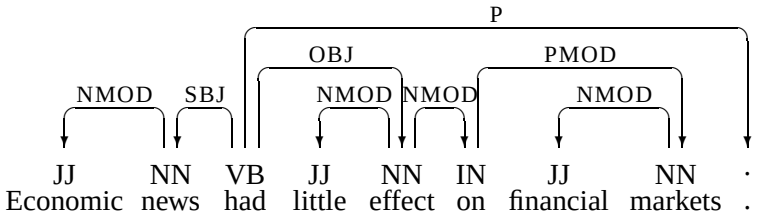


Figure 1: Dependency graph for an English sentence from the WSJ section of the Penn Treebank

require non-projective dependency arcs. For the purpose of this paper, however, this assumption is unproblematic, given that all the treebanks used in the experiments are restricted to projective dependency graphs.

Figure 1 shows a well-formed dependency graph for an English sentence, where each word of the sentence is tagged with its part-of-speech and each arc labeled with a dependency type.

2.2 Parsing Algorithm

We begin by defining parser configurations and the abstract data structures needed for the definition of history-based feature models.

Definition 3 Given a set $R = \{r_0, r_1, \ldots r_m\}$ of dependency types and a sentence $x = (w_1, \ldots, w_n)$, a *parser configuration* for x is a quadruple $c = (\sigma, \tau, h, d)$, where:

1. σ is a stack of tokens nodes.
2. τ is a sequence of token nodes.
3. $h : V_x^+ \rightarrow V$ is a function from token nodes to nodes.
4. $d : V_x^+ \rightarrow R$ is a function from token nodes to dependency types.
5. For every token node $i \in V_x^+$, $h(i) = 0$ if and only if $d(i) = r_0$.

The idea is that the sequence τ represents the remaining input tokens in a left-to-right pass over the input sentence x; the stack σ contains partially processed nodes that are still candidates for dependency arcs, either as heads or dependents; and the functions h and d represent a (dynamically defined) dependency graph for the input sentence x. We refer to the token node on top of the stack as the *top token* and the first token node of the input sequence as the *next token*.

When parsing a sentence $x = (w_1, \ldots, w_n)$, the parser is initialized to a configuration $c_0 = (\epsilon, (1, \ldots, n), h_0, d_0)$ with an empty stack, with all the token nodes in the input sequence, and with all token nodes attached to the special root node 0 with a special dependency type r_0. The parser terminates in any configuration $c_m = (\sigma, \epsilon, h, d)$ where the input sequence is empty, which happens after one left-to-right pass over the input.

There are four possible parser transitions, two of which are parameterized for a dependency type $r \in R$.

1. LEFT-ARC(r) makes the top token i a (left) dependent of the next token j with dependency type r, i.e., $j \xrightarrow{r} i$, and immediately pops the stack.
2. RIGHT-ARC(r) makes the next token j a (right) dependent of the top token i with dependency type r, i.e., $i \xrightarrow{r} j$, and immediately pushes j onto the stack.
3. REDUCE pops the stack.
4. SHIFT pushes the next token i onto the stack.

The choice between different transitions is nondeterministic in the general case and is resolved by a classifier induced from a treebank, using features extracted from the parser configuration.

2.3 Feature Models

The task of the classifier is to predict the next transition given the current parser configuration, where the configuration is represented by a feature vector $\Phi_{(1,p)} = (\phi_1, \ldots, \phi_p)$. Each feature ϕ_i is a function of the current configuration, defined in terms of an *address function* a_{ϕ_i}, which identifies a specific token in the current parser configuration, and an *attribute function* f_{ϕ_i}, which picks out a specific attribute of the token.

Definition 4 Let $c = (\sigma, \tau, h, d)$ be the current parser configuration.

1. For every i ($i \geq 0$), σ_i and τ_i are address functions identifying the ith token of σ and τ, respectively (with indexing starting at 0).

2. If α is an address function, then $h(\alpha)$, $l(\alpha)$, and $r(\alpha)$ are address functions, identifying the head (h), the leftmost child (l), and the rightmost child (r), of the token identified by α (according to the function h).

3. If α is an address function, then $p(\alpha)$, $w(\alpha)$ and $d(\alpha)$ are feature functions, identifying the part-of-speech (p), word form (w) and dependency type (d) of the token identified by α. We call p, w and d attribute functions.

A feature model is defined by specifying a vector of feature functions. In section 4.2 we will define the feature models used in the experiments.

3 Learning Algorithms

The learning problem for inductive dependency parsing, defined in the preceding section, is a pure classification problem, where the input instances are parser configurations, represented by feature vectors, and the output classes are parser transitions. In this section, we introduce the two machine learning methods used to solve this problem in the experiments.

3.1 MBL

MBL is a lazy learning method, based on the idea that learning is the simple storage of experiences in memory and that solving a new problem is achieved by reusing solutions from similar previously solved problems (Daelemans and Van den Bosch, 2005). In essence, this is a k nearest neighbor approach to classification, although a variety of sophisticated techniques, including different distance metrics and feature weighting schemes can be used to improve classification accuracy.

For the experiments reported in this paper we use the TiMBL software package for memory-based learning and classification (Daelemans and Van den Bosch, 2005), which directly handles multi-valued symbolic features. Based on results from previous optimization experiments (Nivre et al., 2004), we use the modified value difference metric (MVDM) to determine distances between instances, and distance-weighted class voting for determining the class of a new instance. The parameters varied during experiments are the number k of nearest neighbors and the frequency threshold l below which MVDM is replaced by the simple Overlap metric.

3.2 SVM

SVM in its simplest form is a binary classifier that tries to separate positive and negative cases in training data by a hyperplane using a linear kernel function. The goal is to find the hyperplane that separates the training data into two classes with the largest margin. By using other kernel functions, such as polynomial or radial basis function (RBF), feature vectors are mapped into a higher dimensional space (Vapnik, 1998; Kudo and Matsumoto, 2001). Multi-class classification with n classes can be handled by the one-versus-all method, with n classifiers that each separate one class from the rest, or the one-versus-one method, with $n(n-1)/2$ classifiers, one for each pair of classes (Vural and Dy, 2004). SVM requires all features to be numerical, which means that symbolic features have to be converted, normally by introducing one binary feature for each value of the symbolic feature.

For the experiments reported in this paper we use the LIBSVM library (Wu et al., 2004; Chang and Lin, 2005) with the polynomial kernel $K(x_i, x_j) = (\gamma x_i^T x_j + r)^d, \gamma > 0$, where d, γ and r are kernel parameters. Other parameters that are varied in experiments are the penalty parameter C, which defines the tradeoff between training error and the magnitude of the margin, and the termination criterion ϵ, which determines the tolerance of training errors.

We adopt the standard method for converting symbolic features to numerical features by binarization, and we use the one-versus-one strategy for multi-class classification. However, to reduce training times, we divide the training data into smaller sets, according to the part-of-speech of the next token in the current parser configuration, and train one set of classifiers for each smaller set. Similar techniques have previously been used by Yamada and Matsumoto (2003), among others, without significant loss of accuracy. In order to avoid too small training sets, we pool together all parts-of-speech that have a frequency below a certain threshold t (set to 1000 in all the experiments).

4 Experimental Setup

In this section, we describe the experimental setup, including data sets, feature models, parameter optimization, and evaluation metrics. Experimental results are presented in section 5.

4.1 Data Sets

The data set used for Swedish comes from Talbanken (Einarsson, 1976), which contains both written and spoken Swedish. In the experiments, the professional prose section is used, consisting of about 100k words taken from newspapers, textbooks and information brochures. The data has been manually annotated with a combination of constituent structure, dependency structure, and topological fields (Teleman, 1974). This annotation has been converted to dependency graphs and the original fine-grained classification of grammatical functions has been reduced to 17 dependency types. We use a pseudo-randomized data split, dividing the data into 10 sections by allocating sentence i to section $i \bmod 10$. Sections 1–9 are used for 9-fold cross-validation during development and section 0 for final evaluation.

The English data are from the Wall Street Journal section of the Penn Treebank II (Marcus et al., 1994). We use sections 2–21 for training, section 0 for development, and section 23 for the final evaluation. The head percolation table of Yamada and Matsumoto (2003) has been used to convert constituent structures to dependency graphs, and a variation of the scheme employed by Collins (1999) has been used to construct arc labels that can be mapped to a set of 12 dependency types.

The Chinese data are taken from the Penn Chinese Treebank (CTB) version 5.1 (Xue et al., 2005), consisting of about 500k words mostly from Xinhua newswire, Sinorama news magazine and Hong Kong News. CTB is annotated with a combination of constituent structure and grammatical functions in the Penn Treebank style, and has been converted to dependency graphs using essentially the same method as for the English data, although with a different head percolation table and mapping scheme. We use the same kind of pseudo-randomized data split as for Swedish, but we use section 9 as the development test set (training on section 1–8) and section 0 as the final test set (training on section 1–9).

A standard HMM part-of-speech tagger with suffix smoothing has been used to tag the test data with an accuracy of 96.5% for English and 95.1% for Swedish. For the Chinese experiments we have used the original (gold standard) tags from the treebank, to facilitate comparison with results previously reported in the literature.

Feature	Φ_1	Φ_2	Φ_3	Φ_4	Φ_5
$p(\sigma_0)$	+	+	+	+	+
$p(\tau_0)$	+	+	+	+	+
$p(\tau_1)$	+	+	+	+	+
$p(\tau_2)$				+	+
$p(\tau_3)$				+	+
$p(\sigma_1)$					+
$d(\sigma_0)$		+	+	+	+
$d(l(\sigma_0))$		+	+	+	+
$d(r(\sigma_0))$		+	+	+	+
$d(l(\tau_0))$		+	+	+	+
$w(\sigma_0)$			+	+	+
$w(\tau_0)$			+	+	+
$w(\tau_1)$					+
$w(h(\sigma_0))$					+

Table 1: Feature models

4.2 Feature Models

Table 1 describes the five feature models Φ_1–Φ_5 used in the experiments, with features specified in column 1 using the functional notation defined in section 2.3. Thus, $p(\sigma_0)$ refers to the part-of-speech of the top token, while $d(l(\tau_0))$ picks out the dependency type of the leftmost child of the next token. It is worth noting that models Φ_1–Φ_2 are unlexicalized, since they do not contain any features of the form $w(\alpha)$, while models Φ_3–Φ_5 are all lexicalized to different degrees.

4.3 Optimization

As already noted, optimization of learning algorithm parameters is a prerequisite for meaningful comparison of different algorithms, although an exhaustive search of the parameter space is usually impossible in practice.

For MBL we have used the modified value difference metric (MVDM) and class voting weighted by inverse distance (ID) in all experiments, and performed a grid search for the optimal values of the number k of nearest neighbors and the frequency threshold l for switching from MVDM to the simple Overlap metric (cf. section 3.1). The best values are different for different combinations of data sets and models but are generally found in the range 3–10 for k and in the range 1–8 for l.

The polynomial kernel of degree 2 has been used for all the SVM experiments, but the kernel parameters γ and r have been optimized together with the penalty parameter C and the termination

FM	LM	Swedish				English				Chinese			
		AS		EM		AS		EM		AS		EM	
		U	L	U	L	U	L	U	L	U	L	U	L
Φ_1	MBL	75.3	68.7	16.0	11.4	*76.5	73.7	9.8	7.7	66.4	63.6	14.3	12.1
	SVM	75.4	68.9	16.3	12.1	76.4	73.6	9.8	7.7	66.4	63.6	14.2	12.1
Φ_2	MBL	81.9	74.4	31.4	19.8	81.2	78.2	19.8	14.9	73.0	70.7	22.6	18.8
	SVM	*83.1	*76.3	*34.3	*24.0	81.3	78.3	19.4	14.9	*73.2	*71.0	22.1	18.6
Φ_3	MBL	85.9	81.4	37.9	28.9	85.5	83.7	26.5	23.7	77.9	76.3	26.3	23.4
	SVM	86.2	*82.6	38.7	*32.5	*86.4	*84.8	*28.5	*25.9	*79.7	*78.3	*30.1	*25.9
Φ_4	MBL	86.1	82.1	37.6	30.1	87.0	85.2	29.8	26.0	79.4	77.7	28.0	24.7
	SVM	86.0	82.2	37.9	31.2	*88.4	*86.8	*33.2	*30.3	*81.7	*80.1	*31.0	*27.0
Φ_5	MBL	86.6	82.3	39.9	29.9	88.0	86.2	32.8	28.4	81.1	79.2	30.2	25.9
	SVM	86.9	*83.2	40.7	*33.7	*89.4	*87.9	*36.4	*33.1	*84.3	*82.7	*34.5	*30.5

Table 2: Parsing accuracy; FM: feature model; LM: learning method; AS: attachment score, EM: exact match; U: unlabeled, L: labeled

criterion e. The intervals for the parameters are: γ: 0.16–0.40; r: 0–0.6; C: 0.5–1.0; e: 0.1–1.0.

4.4 Evaluation Metrics

The evaluation metrics used for parsing accuracy are the *unlabeled attachment score* AS_U, which is the proportion of tokens that are assigned the correct head (regardless of dependency type), and the *labeled attachment score* AS_L, which is the proportion of tokens that are assigned the correct head and the correct dependency type. We also consider the *unlabeled exact match* EM_U, which is the proportion of sentences that are assigned a completely correct dependency graph without considering dependency type labels, and the *labeled exact match* EM_L, which also takes dependency type labels into account. Attachment scores are presented as mean scores per token, and punctuation tokens are excluded from all counts. For all experiments we have performed a McNemar test of significance at $\alpha = 0.01$ for differences between the two learning methods. We also compare learning and parsing times, as measured on an AMD 64-bit processor running Linux.

5 Results and Discussion

Table 2 shows the parsing accuracy for the combination of three languages (Swedish, English and Chinese), two learning methods (MBL and SVM) and five feature models (Φ_1–Φ_5), with algorithm parameters optimized as described in section 4.3. For each combination, we measure the *attachment score* (AS) and the *exact match* (EM). A significant improvement for one learning method over the other is marked by an asterisk (*).

Independently of language and learning method, the most complex feature model Φ_5 gives the highest accuracy across all metrics. Not surprisingly, the lowest accuracy is obtained with the simplest feature model Φ_1. By and large, more complex feature models give higher accuracy, with one exception for Swedish and the feature models Φ_3 and Φ_4. It is significant in this context that the Swedish data set is the smallest of the three (about 20% of the Chinese data set and about 10% of the English one).

If we compare MBL and SVM, we see that SVM outperforms MBL for the three most complex models Φ_3, Φ_4 and Φ_5, both for English and Chinese. The results for Swedish are less clear, although the labeled accuracy for Φ_3 and Φ_5 are significantly better. For the Φ_1 model there is no significant improvement using SVM. In fact, the small differences found in the AS_U scores are to the advantage of MBL. By contrast, there is a large gap between MBL and SVM for the model Φ_5 and the languages Chinese and English. For Swedish, the differences are much smaller (except for the EM_L score), which may be due to the smaller size of the Swedish data set in combination with the technique of dividing the training data for SVM (cf. section 3.2).

Another important factor when comparing two learning methods is the efficiency in terms of time. Table 3 reports learning and parsing time for the three languages and the five feature models. The learning time correlates very well with the complexity of the feature model and MBL, being a lazy learning method, is much faster than SVM. For the unlexicalized feature models Φ_1 and Φ_2, the parsing time is also considerably lower for MBL, especially for the large data sets (English and Chinese). But as model complexity grows, especially with the addition of lexical features, SVM gradually gains an advantage over MBL with respect to parsing time. This is especially striking for Swedish,

Method	Model	Swedish		English		Chinese	
		LT	PT	LT	PT	LT	PT
Φ_1	MBL	1 s	2 s	16 s	26 s	7 s	8 s
	SVM	40 s	14 s	1.5 h	14 min	1.5 h	17 min
Φ_2	MBL	3 s	5 s	35 s	32 s	13 s	14 s
	SVM	40 s	13 s	1 h	11 min	1.5 h	15 min
Φ_3	MBL	6 s	1 min	1.5 min	9.5 min	46 s	10 min
	SVM	1 min	15 s	1 h	9 min	2 h	16 min
Φ_4	MBL	8 s	2 min	1.5 min	9 min	45 s	12 min
	SVM	2 min	18 s	2 h	12 min	2.5 h	14 min
Φ_5	MBL	10 s	7 min	3 min	41 min	1.5 min	46 min
	SVM	2 min	25 s	1.5 h	10 min	6 h	24 min

Table 3: Time efficiency; LT: learning time, PT: parsing time

where the training data set is considerably smaller than for the other languages.

Compared to the state of the art in dependency parsing, the unlabeled attachment scores obtained for Swedish with model Φ_5, for both MBL and SVM, are about 1 percentage point higher than the results reported for MBL by Nivre et al. (2004). For the English data, the result for SVM with model Φ_5 is about 3 percentage points below the results obtained with the parser of Charniak (2000) and reported by Yamada and Matsumoto (2003). For Chinese, finally, the accuracy for SVM with model Φ_5 is about one percentage point lower than the best reported results, achieved with a deterministic classifier-based approach using SVM and preprocessing to detect root nodes (Cheng et al., 2005a), although these results are not based on exactly the same dependency conversion and data split as ours.

6 Conclusion

We have performed an empirical comparison of MBL (TiMBL) and SVM (LIBSVM) as learning methods for classifier-based deterministic dependency parsing, using data from three languages and feature models of varying complexity. The evaluation shows that SVM gives higher parsing accuracy and comparable or better parsing efficiency for complex, lexicalized feature models across all languages, whereas MBL is superior with respect to training efficiency, even if training data is divided into smaller sets for SVM. The best accuracy obtained for SVM is close to the state of the art for all languages involved.

Acknowledgements

The work presented in this paper was partially supported by the Swedish Research Council. We are grateful to Hiroyasu Yamada and Yuan Ding for sharing their head percolation tables for English and Chinese, respectively, and to three anonymous reviewers for helpful comments and suggestions.

References

Ezra Black, Frederick Jelinek, John D. Lafferty, David M. Magerman, Robert L. Mercer, and Salim Roukos. 1992. Towards history-based grammars: Using richer models for probabilistic parsing. In *Proceedings of the 5th DARPA Speech and Natural Language Workshop*, pages 31–37.

Chih-Chung Chang and Chih-Jen Lin. 2005. LIBSVM: A library for support vector machines.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 173–180.

Eugene Charniak. 2000. A Maximum-Entropy-Inspired Parser. In *Proceedings of the First Annual Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 132–139.

Yuchang Cheng, Masayuki Asahara, and Yuji Matsumoto. 2005a. Chinese deterministic dependency analyzer: Examining effects of global features and root node finder. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, pages 17–24.

Yuchang Cheng, Masayuki Asahara, and Yuji Matsumoto. 2005b. Machine learning-based dependency analyzer for Chinese. In *Proceedings of the International Conference on Chinese Computing (ICCC)*.

Michael Collins and Nigel Duffy. 2005. Discriminative reranking for natural language parsing. *Computational Linguistics*, 31:25–70.

Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 16–23.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

Walter Daelemans and Veronique Hoste. 2002. Evaluation of machine learning methods for natural language processing tasks. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC)*, pages 755–760.

Walter Daelemans and Antal Van den Bosch. 2005. *Memory-Based Language Processing*. Cambridge University Press.

Walter Daelemans, Veronique Hoste, Fien De Meulder, and Bart Naudts. 2003. Combined optimization of feature selection and algorithm parameter interaction in machine learning of language. In *Proceedings of the 14th European Conference on Machine Learning (ECML)*, pages 84–95.

Jan Einarsson. 1976. *Talbankens skriftspråkskonkordans*. Lund University, Department of Scandinavian Languages.

Mark Johnson, Stuart Geman, Steven Canon, Zhiyi Chi, and Stefan Riezler. 1999. Estimators for stochastic "unification-based" grammars. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 535–541.

Taku Kudo and Yuji Matsumoto. 2001. Chunking with support vector machines. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Taku Kudo and Yuji Matsumoto. 2002. Japanese dependency analysis using cascaded chunking. In *Proceedings of the Sixth Workshop on Computational Language Learning (CoNLL)*, pages 63–69.

David M. Magerman. 1995. Statistical decision-tree models for parsing. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 276–283.

Mitchell P. Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The Penn Treebank: Annotating predicate-argument structure. In *Proceedings of the ARPA Human Language Technology Workshop*, pages 114–119.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 91–98.

Joakim Nivre and Mario Scholz. 2004. Deterministic dependency parsing of English text. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, pages 64–70.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2004. Memory-based dependency parsing. In *Proceedings of the 8th Conference on Computational Natural Language Learning (CoNLL)*, pages 49–56.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*, pages 149–160.

Adwait Ratnaparkhi. 1997. A linear observed time statistical parser based on maximum entropy models. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1–10.

Kenji Sagae and Alon Lavie. 2005. A classifier-based parser with linear run-time complexity. In *Proceedings of the 9th International Workshop on Parsing Technologies (IWPT)*, pages 125–132.

Ben Taskar, Dan Klein, Michael Collins, Daphne Koller, and Christopher Manning. 2004. Max-margin parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1–8.

Ulf Teleman. 1974. *Manual för grammatisk beskrivning av talad och skriven svenska*. Studentlitteratur.

Vladimir Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer.

Vladimir Vapnik. 1998. *Statistical Learning Theory*. John Wiley and Sons, New York.

Volkan Vural and Jennifer G. Dy. 2004. A hierarchical method for multi-class support vector machines. *ACM International Conference Proceeding Series*, 69:105–113.

Ting-Fan Wu, Chih-Jen Lin, and Ruby C. Weng. 2004. Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research*, 5:975–1005.

Nianwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The Penn Chinese Treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(2):207–238.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*, pages 195–206.

Detection of Quotations and Inserted Clauses and its Application to Dependency Structure Analysis in Spontaneous Japanese

Ryoji Hamabe[†] **Kiyotaka Uchimoto**[‡] **Tatsuya Kawahara**[†] **Hitoshi Isahara**[‡]

[†]School of Informatics,
Kyoto University
Yoshida-honmachi, Sakyo-ku,
Kyoto 606-8501, Japan

[‡]National Institute of Information
and Communications Technology
3-5 Hikari-dai, Seika-cho, Soraku-gun,
Kyoto 619-0289, Japan

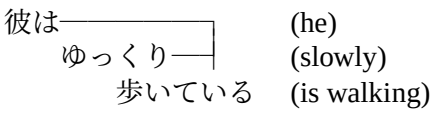
Abstract

Japanese dependency structure is usually represented by relationships between phrasal units called *bunsetsu*s. One of the biggest problems with dependency structure analysis in spontaneous speech is that clause boundaries are ambiguous. This paper describes a method for detecting the boundaries of quotations and inserted clauses and that for improving the dependency accuracy by applying the detected boundaries to dependency structure analysis. The quotations and inserted clauses are determined by using an SVM-based text chunking method that considers information on morphemes, pauses, fillers, etc. The information on automatically analyzed dependency structure is also used to detect the beginning of the clauses. Our evaluation experiment using *Corpus of Spontaneous Japanese (CSJ)* showed that the automatically estimated boundaries of quotations and inserted clauses helped to improve the accuracy of dependency structure analysis.

1 Introduction

The "Spontaneous Speech: Corpus and Processing Technology" project sponsored the construction of the *Corpus of Spontaneous Japanese (CSJ)* (Maekawa et al., 2000). The CSJ is the biggest spontaneous speech corpus in the world, consisting of roughly 7M words with the total speech length of 700 hours, and is a collection of monologues such as academic presentations and simulated public speeches. The CSJ includes transcriptions of the speeches as well as audio recordings of them. Approximately one tenth of the speeches in the CSJ were manually annotated with various kinds of information such as morphemes, sentence boundaries, dependency structures, and discourse structures.

In Japanese sentences, word order is rather free, and subjects or objects are often omitted. In Japanese, therefore, the syntactic structure of a sentence is generally represented by the relationships between phrasal units, or *bunsetsu*s in Japanese, based on a dependency grammar, as represented in the Kyoto University text corpus (Kurohashi and Nagao, 1997). In the same way, the syntactic structure of a sentence is represented by dependency relationships between *bunsetsu*s in the CSJ. For example, the sentence "彼はゆっくり歩いている" (He is walking slowly) can be divided into three *bunsetsu*s, "彼は, *kare-wa*" (he), "ゆっくり, *yukkuri*" (slowly), and "歩いている, *arui-te-iru*" (is walking). In this sentence, the first and second *bunsetsu*s depend on the third one. The dependency structure is described as follows.



In this paper, we first describe the problems with dependency structure analysis of spontaneous speech. We focus on ambiguous clause boundaries as the biggest problem and present a solution.

2 Problems with Dependency Structure Analysis in Spontaneous Japanese

There are many differences between written text and spontaneous speech, and consequently, problems peculiar to spontaneous speech arise in de-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 324–330,
Sydney, July 2006.

pendency structure analysis, such as ambiguous clause boundaries, independent *bunsetsu*s, crossed dependencies, self-corrections, and inversions. In this study, we address the problem of ambiguous clause boundaries in dependency structure analysis in spontaneous speech. We treated the other problems in the same way as Shitaoka et al. (Shitaoka et al., 2004). For example, inversions are represented as dependency relationships going in the direction from right to left in the CSJ, and their direction was changed to that from left to right in our experiments. In this paper, therefore, all the dependency relationships were assumed to go in the direction from left to right (Uchimoto et al., 2006).

There are several types of clause boundaries such as sentence boundaries, boundaries of quotations and inserted clauses. In the CSJ, clause boundaries were automatically detected by using surface information (Maruyama et al., 2003), and sentence boundaries were manually selected from them (Takanashi et al., 2003). Boundaries of quotations and inserted clauses were also defined and detected manually. Dependency relationships between *bunsetsu*s were annotated within sentences (Uchimoto et al., 2006). Our definition of clause boundaries follows the definition used in the CSJ.

Shitaoka et al. worked on automatic sentence boundary detection by using SVM-based text chunking. However, quotations and inserted clauses were not considered. In this paper, we focus on these problems in a context of ambiguous clause boundaries.

Quotations

In written text, quotations are often bracketed by 「」(angle brackets), but no brackets are inserted in spontaneous speech.
ex) “一度でもいいから行ってみたい” (I want to go there at any rate) is a quotation. In the CSJ, quotations were manually annotated as follows.

ここは┐	(here)
昔から┤	(since early times)
{一度でも┐	(once)
いいから┤	(at any rate)
行ってみたい} と┤	(want to go)
思っていたところです	(is the place I think)

Inserted Clauses

In spontaneous speech, speakers insert clauses in the middle of other clauses. This occurs when speakers change their speech plans while producing utterances, which results in supplements, annotations, or paraphrases of main clauses.
ex) “夜着いたんですけども” (where I arrived at night) is an inserted clause.

ホテルの┐	(hotel)
部屋の┐	(room)
中も┐	(inside)
早速┤	(without delay)
（夜┐	(at night)
着いたんですけども）┤	(arrived)
チェックしました	(I checked)

Dependency relationships are closed within a quotation or an inserted clause. Therefore, dependencies except the rightmost *bunsetsu* in each clause do not cross boundaries of the same clause, meaning no dependency exists between the *bunsetsu* inside a clause and that outside the clause. However, automatically detected dependencies often cross clause boundaries erroneously because sentences including quotations or inserted clauses can have complicated clause structures. This is one of the reasons dependency structure analysis of spontaneous speech has more errors than that of written texts. We propose a method for improving dependency structure analysis based on automatic detection of quotations and inserted clauses.

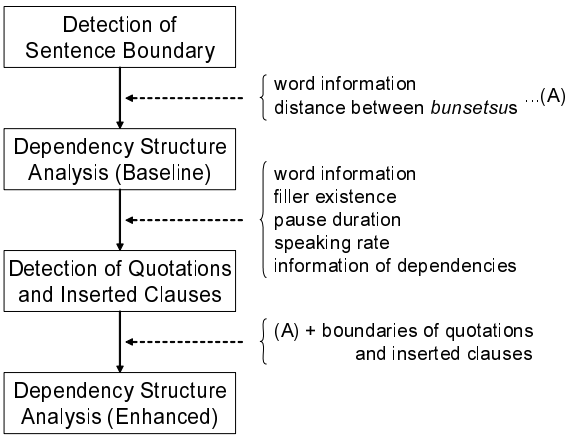


Figure 1: Outline of proposed processes

3 Dependency Structure Analysis and Detection of Quotations and Inserted Clauses

The outline of the proposed processes is shown in Figure 1. Here, we use “clause” to describe a quotation and an inserted clause.

3.1 Dependency Structure Analysis

In this research, we use the method proposed by Uchimoto et al. (Uchimoto et al., 2000) to ana-

lyze dependency structures. This method is a two-step procedure, and the first step is preparation of a dependency matrix in which each element represents the likelihood that one *bunsetsu* depends on another. The second step of the analysis is finding an optimal set of dependencies for the entire sentence. The likelihood of dependency is represented by a probability, using a dependency probability model. The model in this study (Uchimoto et al., 2000) takes into account not only the relationship between two *bunsetsu*s but also the relationship between the left *bunsetsu* and all the *bunsetsu* to its right.

We implemented this model within a maximum entropy modeling framework. The features used in the model were basically attributes related to the target two *bunsetsu*s: attributes of a *bunsetsu* itself, such as character strings, parts of speech, and inflection types of a *bunsetsu* together with attributes between *bunsetsu*s, such as the distance between *bunsetsu*s, etc. Combinations of these features were also used. In this work, we added to the features whether there is a boundary of quotations or inserted clauses between the target *bunsetsu*s. If there is, the probability that the left *bunsetsu* depends on the right *bunsetsu* is estimated to be low.

In the CSJ, some *bunsetsu*s are defined to have no modifiee. In our experiments, we defined their dependencies as follows.

- The rightmost *bunsetsu* in a quotation or an inserted clause depends on the rightmost one in the sentence.
- If a sentence boundary is included in a quotation or an inserted clause, the *bunsetsu* to the immediate left of the boundary depends on the rightmost *bunsetsu* in the quotation or the inserted clause.
- Other *bunsetsu*s that have no modifiee depend on the next one.

3.2 Detection of Quotations and Inserted Clauses

We regard the problem of clause boundary detection as a text chunking task. We used YamCha (Kudo and Matsumoto, 2001) as a text chunker, which is based on Support Vector Machine (SVM). We used the chunk labels consisting of three tags which correspond to sentence boundaries, boundaries of quotations, and boundaries of inserted clauses, respectively. The tag for sentence boundaries can be either E (the rightmost *bunsetsu* in a sentence) or I (the others). The tags for the boundaries of quotations and inserted clauses are shown in Table 1. An example of chunk labels assigned to each *bunsetsu* in a sentence is as follows.

Table 1: Tag categories used for chunking

Tag	Explanation of tag
B	Beginning of a clause
E	End of a clause
I	Interior of a clause (except B and E)
O	Exterior of a clause
S	Clause consisting of one *bunsetsu*

ex) “予算の関係だ” (It is because of the budget) is a quotation, and “予算の関係だと思いますが” (which I think is because of the budget) is an inserted clause. For a chunk label, for example, the *bunsetsu* that the chunk label (I, B, B) is assigned to means that it is not related to a sentence boundary but is related to the beginning of a quotation and an inserted clause.

```
(I,O,O) 今は┌──────────────────┐ (now)
(I,B,B) ({予算の┐               │ (budget)
(I,E,I)    関係だ} と┐          │ (because of)
(I,O,E)      思いますが)         │ (I think)
(I,O,O)        一夏に┐           │ (in summer)
(I,O,O)          三回ぐらいしか┘ (three times)
(E,O,O)                  やりません (they do it)
```

The three tags of each chunk label are simultaneously estimated. Therefore, the relationships between sentence boundaries, quotations, and inserted clauses are considered in this model. For instance, quotations and inserted clauses should not cross the sentence boundaries, and the chunk label such as (E,I,O) is never estimated because this label means that a sentence boundary exists within a quotation.

We used the following parameters for YamCha.

- Degree of polynomial kernel: 3rd
- Analysis direction: Right to left
- Dynamic features: Following three chunk labels
- Multi-class method: Pairwise

The chunk label is estimated for each *bunsetsu*, The features used to estimate the chunk labels are as follows.

(1) word information We used word information such as character strings, pronunciation, part of speech, inflection type, and inflection form. Specific expressions are often used at the ends of quotations and inserted clauses.

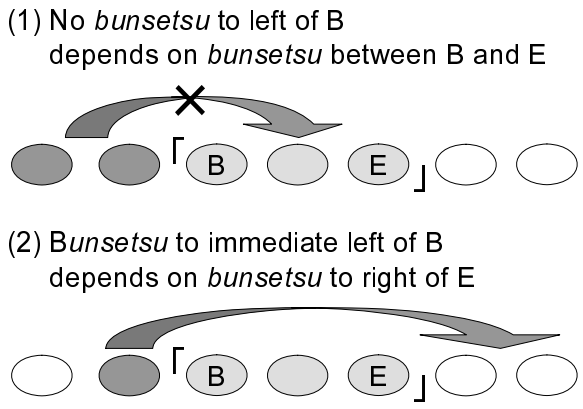


Figure 2: Dependency structures of *bunsetsu*s to left of beginning of quotations or inserted clauses

For instance, "と思う, *to-omou*" (think) and "って言う, *tte-iu*" (say) are used at the ends of quotations. Expressions such as "ですが, *desu-ga*" and "けれども, *keredo-mo*" are used at the ends of inserted clauses.

(2) fillers and pauses Fillers and pauses are often inserted just before or after quotations and inserted clauses. Pause duration is normalized in a talk with its mean and variance.

(3) speaking rate Inside inserted clauses, speakers tend to speak fast. The speaking rate is also normalized in a talk.

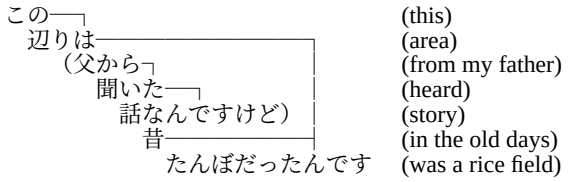
Detecting the ends of clauses appears easy because specific expressions are frequently used at the ends of clauses as previously mentioned. However, determining the beginnings of clauses is difficult in a single process because all features mentioned above are local information. Therefore, the global information is also used to detect the beginning of the clauses. If the end of a clause is given, the *bunsetsu*s to the left of the clause should satisfy the two conditions described in Figure 2. Our method uses the constraint as global information. They are considered as additional features based on dependency probabilities estimated for the *bunsetsu*s to the left of the clause. Thus, our chunking method has two steps. First, clause boundaries are detected based on the three types of features itemized above. Second, the beginnings of clauses are determined after adding to the features the following probabilities obtained by automatic dependency structure analysis.

(4) probability that *bunsetsu* to left of target depends on *bunsetsu* inside clause

(5) probability that *bunsetsu* to immediate left of target depends on *bunsetsu* to right of clause

Figure 2 shows that the target *bunsetsu* is likely to be the beginning of the clause if probability (4) is low and probability (5) is high. For instance, the following example sentence has an inserted clause. In the first chunking step, the *bunsetsu* "話なんですけど" (which is a story) is found to be the end of the inserted clause.

ex) "父から聞いた話なんですけど" (which is a story that I heard from my father) is an inserted clause.



The three *bunsetsu*s "辺りは, *atari-wa*", "聞いた, *kii-ta*", and "話なんですけど, *hanashi-na-ndesu-kedo*" are less likely to be the beginning of the inserted clause because in the three cases the *bunsetsu* to the immediate left depends on the target *bunsetsu*. On the other hand, the *bunsetsu* "父から, *chichi-kara*" is the most likely to be the beginning since the *bunsetsu* to its immediate left "辺りは, *atari-wa*" depends on the *bunsetsu* to the right of the inserted clause "田んぼだったんです, *tanbo-datta-ndesu*".

4 Experiments and Discussion

For experimental evaluation, we used the transcriptions of 188 talks in the CSJ, which contain 6,255 quotations and 818 inserted clauses. We used 20 talks for testing. The test data included 643 quotations and 76 inserted clauses. For training, we used 168 talks excluding the test data to conduct the open test and all the 188 talks to conduct the closed test.

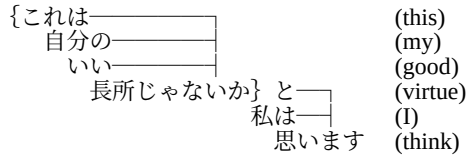
First, we detected sentence boundaries by using the method (Shitaoka et al., 2004) and analyzed the dependency structure of each sentence by the method described in Section 3.1 without using information on quotations and inserted clauses. We obtained an F-measure of 85.9 for the sentence boundary detection, and the baseline accuracy of the dependency structure analysis was 77.7% for the open test and 86.5% for the closed test.

(a) Results of clause boundary detection

The results obtained by the method described in Section 3.2 are shown in Table 2. The table shows five kinds of results:

- results obtained without dependency structure (in the first chunking step)
- results obtained with dependency structure analyzed for the open test (in the second chunking step)
- results obtained with dependency structure analyzed for the closed test (in the second chunking step)
- results obtained with manually annotated dependency structure (in the second chunking step)
- the rate that the ends of clauses are detected correctly

These results indicate that around 90% of quotations were detected correctly, and the boundary detection accuracy of quotations was improved by using automatically analyzed dependency structure. We found that features (4) and (5) in Section 3.2 obtained from automatically analyzed dependency structure contributed to the improvement. In the following example, a part of the quotation "自分のいい長所じゃないか" (my good virtue) was erroneously detected as a quotation in the first chunking step. But, in the second chunking step, automatically analyzed dependency structure contributed to detection of the correct part "これは自分のいい長所じゃないか" (this is my good virtue) as a quotation.



We also found that the boundary detection accuracy of quotations was significantly improved by using manually annotated dependency structure. This indicates that the boundary detection accuracy of quotations improves as the accuracy of dependency structure analysis improves.

By contrast, only a few inserted clauses were detected even if dependency structures were used. Most of the ends of the inserted clauses were detected incorrectly as sentence boundaries. The main reason for this is our method could not distinguish between the ends of the inserted clauses and those of the sentences, since the same words often appeared at the ends of both, and it was difficult to learn the difference between them even though our method used features based on acoustic information.

Table 2: Clause boundary detection results (sentence boundaries automatically detected)

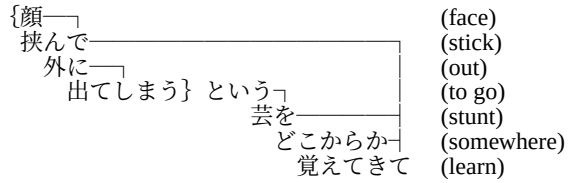
Quotations			Inserted clauses		
recall	precision	F	recall	precision	F
Without dependency information					
41.1% (264/643)	44.3% (264/596)	42.6	1.3% (1/76)	20.0% (1/5)	2.5
With dependency information (open)					
42.1% (271/643)	45.5% (271/596)	43.7	2.6% (2/76)	40.0% (2/5)	4.9
With dependency information (closed)					
50.9% (327/643)	54.9% (327/596)	52.8	2.6% (2/76)	40.0% (2/5)	4.9
With dependency information (correct)					
74.2% (477/643)	80.0% (477/596)	77.0	2.6% (2/76)	33.3% (2/6)	4.9
Correct end of clauses					
89.1% (573/643)	96.1% (573/596)	92.5	2.6% (2/76)	40.0% (2/5)	4.9

Table 3: Dependency structure analysis results obtained with clause boundaries (sentence boundaries automatically detected)

Without boundaries of quotations and inserted clauses	open	77.7%
	closed	86.5%
With boundaries of quotations and inserted clauses (automatically detected)	open	78.5%
	closed	86.6%
With boundaries of quotations and inserted clauses (correct)	open	79.4%
	closed	87.4%

(b) Dependency structure analysis results

We investigated the accuracies of dependency structure analysis obtained when the automatically or manually detected boundaries of quotations and inserted clauses were used. The results are shown in Table 3. Although the accuracy of detecting the boundaries of quotations and inserted clauses using automatically analyzed dependency structure was not high, the accuracy of dependency structure analysis was improved by 0.7% absolute for the open test. This shows that the model for dependency structure analysis could robustly learn useful information on clause boundaries even if errors were included in the results of clause boundary detection. In the following example, for instance, "顔挟んで外に出てしまう" (to go out with its face stuck) was correctly detected as a quotation in the first chunking step. Then, the initial inappropriate modifiee "覚えてきて, *oboe-te-ki-te*" (learn) of the *bunsetsu* inside the quotation '挟んで, *hasan-de*" (stick) was correctly modified to the *bunsetsu* inside the quotation "出てしまうという, *de-te-shimau-to-iu*" (to go) by using the automatically detected boundary of the quotation.



(c) Results obtained when correct sentence boundaries are given

We investigated the clause boundary detection accuracy of quotations and inserted clauses and the dependency accuracy when correct sentence boundaries were given manually. The results are shown in Tables 4 and 5, respectively.

When correct sentence boundaries were given, the accuracy of clause detection and dependency structure analysis was improved significantly. Table 4 shows that the boundary detection accuracy of inserted clauses as well as that of quotations was significantly improved by using information of dependencies. Table 5 indicates that when using automatically detected clause boundaries, the accuracy of dependency structure analysis was improved by 0.7% for the open test, and it was further improved by using correct clause boundaries.

These experimental results show that detecting the boundaries of quotations and inserted clauses as well as sentence boundaries is sensitive to the accuracy of dependency structure analysis and the improvements of the boundary detection of quotations and inserted clauses contribute to improvement of dependency structure analysis. Especially, the difference between Table3 and 5 shows that the sentence boundary detection accuracy is more sensitive to the accuracy of dependency structure analysis than the boundary detection accuracy of quotations and inserted clauses. This indicates that sentence boundaries rather than quotations and inserted clauses should be manually examined first to effectively improve the accuracy of dependency structure analysis in a semi-automatic way.

5 Conclusion

This paper described the method for detecting the boundaries of quotations and inserted clauses and that for applying it to dependency structure analysis. The experiment results showed that the automatically estimated boundaries of quotations and inserted clauses contributed to improvement of dependency structure analysis. In the future, we plan to solve the problems found in the experiments and investigate the robustness of our method when the results of automatic speech recognition are given as the inputs. We will also study use of information on quotations and inserted clauses to text formatting, such as text summarization.

Table 4: Clause boundary detection results (sentence boundaries given)

Quotations			Inserted clauses		
recall	precision	F	recall	precision	F
Without dependency information					
46.0% (296/643)	50.8% (296/583)	48.3	22.4% (17/76)	23.6% (17/72)	23.0
With dependency information (open)					
46.7% (300/643)	53.3% (300/563)	49.8	30.3% (23/76)	38.3% (23/60)	33.8
With dependency information (closed)					
55.1% (354/643)	62.9% (354/563)	58.7	30.3% (23/76)	39.0% (23/59)	34.1
With dependency information (correct)					
75.3% (484/643)	86.0% (484/563)	80.3	46.1% (35/76)	60.3% (35/58)	52.2
Correct end of clauses					
86.5% (556/643)	95.4% (556/583)	90.7	64.5% (49/76)	68.1% (49/72)	66.2

Table 5: Dependency structure analysis results obtained with clause boundaries (sentence boundaries given)

Without boundaries of quotations and inserted clauses	open	81.0%
	closed	90.3%
With boundaries of quotations and inserted clauses (automatically detected)	open	81.7%
	closed	90.3%
With boundaries of quotations and inserted clauses (correct)	open	82.8%
	closed	91.3%

References

Taku Kudo and Yuji Matsumoto. 2001. Chunking with support vector machines. In *Proceedings of the NAACL*.

Sadao Kurohashi and Makoto Nagao. 1997. Building a Japanese Parsed Corpus while Improving the Parsing System. In *Proceedings of the NLPRS*, pages 451–456.

Kikuo Maekawa, Hanae Koiso, Sadaoki Furui, and Hitoshi Isahara. 2000. Spontaneous Speech Corpus of Japanese. In *Proceedings of the LREC2000*, pages 947–952.

Takehiko Maruyama, Hideki Kashioka, Tadashi Kumano, and Hideki Tanaka. 2003. Rules for Automatic Clause Boundary Detection and Their Evaluation. In *Proceedings of the Nineth Annual Meeting of the Association for Natural Language proceeding*, pages 517–520. (in Japanese).

Katsuya Takanashi, Takehiko Maruyama, Kiyotaka Uchimoto, and Hitoshi Isahara. 2003. Identification of "Sentences" in Spontaneous

Japanese — Detection and Modification of Clause Boundaries —. In *Proceedings of the ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition*, pages 183–186.

Kiyotaka Uchimoto, Masaki Murata, Satoshi Sekine, and Hitoshi Isahara. 2000. Dependency Model Using Posterior Context. In *Proceedings of the IWPT*, pages 321–322.

Kiyotaka Uchimoto, Ryoji Hamabe, Takehiko Maruyama, Katsuya Takanashi, Tatsuya Kawahara, and Hitoshi Isahara. 2006. Dependency-structure Annotation to Corpus of Spontaneous Japanese. In *Proceedings of the LREC2006*, pages 635-638.

Kazuya Shitaoka, Kiyotaka Uchimoto, Tatsuya Kawahara, and Hitoshi Isahara. 2004. Dependency Structure Analysis and Sentence Boundary Detection in Spontaneous Japanese. In *Proceedings of the COLING2004*, pages 1107–1113.

Improving English Subcategorization Acquisition with Diathesis Alternations as Heuristic Information

Xiwu Han
Institute of Computational Linguistics
Heilongjiang University
Harbin City 150080 China
hxw@hlju.edu.cn

Tiejun Zhao
School of Computer Science and Technology
Harbin Institute of Technology
Harbin City 150001 China
tjzhao@mtlab.hit.edu.cn

Xingshang Fu
Institute of Computational Linguistics
Heilongjiang University
Harbin City 150080 China
fxs@hlju.edu.cn

Abstract

Automatically acquired lexicons with subcategorization information have already proved accurate and useful enough for some purposes but their accuracy still shows room for improvement. By means of diathesis alternation, this paper proposes a new filtering method, which improved the performance of Korhonen's acquisition system remarkably, with the precision increased to 91.18% and recall unchanged, making the acquired lexicon much more practical for further manual proofreading and other NLP uses.

1 Introduction

Subcategorization is the process that further classifies a syntactic category into its subsets. Chomsky (1965) defines the function of strict subcategorization features as appointing a set of constraints that dominate the selection of verbs and other arguments in deep structure. Large subcategorized verbal lexicons have proved to be crucially important for many tasks of natural language processing, such as probabilistic parsers (Korhonen, 2001, 2002) and verb classifications (Schulte im Walde, 2002; Korhonen, 2003). Since Brent (1993) a considerable amount of research focusing on large-scaled automatic acquisition of subcategorization frames (SCF) has met with some success not only in English but also in many other languages, including German (Schulte im Walde, 2002), Spanish (Chrupala, 2003), Czech (Sarkar and Zeman, 2000), Portuguese (Gamallo et. al, 2002), and Chinese (Han et al, 2004). The general objective of this research is to acquire from a given corpus the SCF types and numbers for predicate verbs. Two typical steps during the process of automatic acquisition are hypothesis generation and selection. Usually based on heuristic rules, the first step generates SCF hypotheses for involved verbs; and the second selects reliable ones via statistical methods, such as BHT (binomial hypothesis testing), LLR (log likelihood ratio) and MLE (maximum likelihood estimation). This second step is also called statistical filtering and has been widely regarded as problematic. English researchers have proposed some methods adjusting the corpus hypothesis frequencies before or while filtering. These methods are often called backoff techniques for SCF acquisition. Some of them represent a remarkable improvement in the acquisition performance, for example diathesis alternation and semantic motivation (Korhonen, 1998, 2001, 2002).

For the convenience of comparison between performances of different SCF acquisition methods, we define absolute and relative recall in this paper. By absolute recall, we mean the figure computed against the background of input corpus, while relative recall is against the set of generated hypotheses.

At present, automatically acquired verb lexicons with SCF information have already proved accurate and useful enough for some NLP purposes (Korhonen, 2001; Han et al, 2004). As for English, Korhonen (2002) reported that semantically motivated SCF acquisition achieved a precision of 87.1%, an absolute recall of 71.2% and a relative recall of 85.27%, thus making the acquired lexicon much more accurate and useful. However, the accuracy still shows room for improvement, especially for those SCF hypotheses with low frequencies. Detailed analysis on the acquisition system and some resulting data shows that three main causes should account for the comparatively unsatisfactory performance: a. the imperfect hypothesis generator, b. the Zipfian

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 331–336,
Sydney, July 2006.

distribution of syntactic patterns, c. the incomplete partition over SCF types of a given verb. The first problem mainly comes from the inadequate parsing performance and noises existing in the corpus, while the other two problems are inherent to natural languages and should be solved in terms of acquisition techniques particularly during the process of hypothesis selection.

2 Related Work

The empirical background of this paper is the public resource for subcategorization acquisition of English verbs, provided by Anna Korhonen (2005) in her personal home page. The data include 30 verbs, as shown in Table 1, and their unfiltered SCF hypotheses, which were automatically generated via Briscoe and Carroll's (1997) SCF acquisition system, and the manually established standard.

Table 1. English Verbs in Use.

add	agree	attach
bring	carry	carve
chop	cling	clip
fly	cut	travel
drag	communicate	give
lend	lock	marry
meet	mix	move
offer	provide	visit
push	sail	send
slice	supply	swing

For each verb, there is a corpus of 1000 sentences extracted from the BNC, and all together 42 SCF types are involved in the corpus. The framework of Briscoe and Carroll's system consists of six overall components, which are applied in sequence to sentences containing a specific predicate in order to retrieve a set of SCFs for that verb:

- **A tagger**, a first-order Hidden Markov Model POS and punctuation tag disambiguator.
- **A lemmatizer**, an enhanced version of the General Architecture for Text Engineering project stemmer.
- **A probabilistic LR parser**, trained on a tree-bank derived semi-automatically from the SUSANNE corpus, returns ranked analyses using a feature-based unification grammar.
- **A pattern extractor**, which extracts subcategorization patterns, i.e. local syntactic frames, including the syntactic frames, including the syntactic categories and head lemmas.
- **A pattern classifier**, which assigns patterns to SCFs or rejects them as unclassifiable.
- **A SCF filter**, which evaluates sets of SCFs gathered for a predicate verb.

Nowadays, in most related researches, the performances of subcategorization acquisition systems are often evaluated in terms of precision, recall and F measure of SCF types (Korhonen, 2001, 2002). Generally, precision is the percentage of SCFs that the system proposes correctly, while recall is the percentage of SCFs in the gold standard that the system proposes:

$$\text{Precision} = \frac{|\text{True positives}|}{|\text{True positives}|+|\text{False positives}|}$$

$$\text{Recall} = \frac{|\text{True positives}|}{|\text{True positives}|+|\text{False negatives}|}$$

$$\text{F-measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

Here, true positives are correct SCF types proposed by the system, false positives are incorrect SCF types proposed by system, and false negatives are correct SCF types not proposed by the system.

3 The MLE Filtering Method

The present SCF acquisition system for English verbs employs a MLE filter to test the automatically generated SCF hypotheses. Due to noises accumulated while tagging, lemmatizing and parsing the corpus, even though correction is implemented for some typical errors when classifying the extracted patterns, the hypothesis generator does not perform as efficiently as hoped. Sampling analysis on the unfiltered hypotheses in Korhonen's evaluation corpus indicates that about 74% incorrectly proposed and rejected SCF types come from the defects of the MLE filtering method.

Performance of the MLE filter is closely related to the actual distributions $p(scf_i|v)$ over predicates and SCF types in the input corpus. First, from the overall corpus a training set is drawn randomly; it must be large enough to ensure a similar distribution. Then, the frequency of a subcategorization frame scf_i occurring with a verb v is recorded and used to estimate the probability $p(scf_i|v)$. Thirdly, an empirical threshold θ is determined, which ensures that a maximum

value of the F-measure will result for the training set. Finally, the threshold is used to filter out from the total set those SCF hypotheses with frequencies lower than θ.

Therefore, the statistical foundation of this filtering method is the assumption of independence among the SCFs that a verb enters, which can be probabilistically expressed in two formulas as follows:

$$\forall i, \forall j, i \neq j, p(scf_i \mid scf_j, v) = 0 \quad \ldots (1)$$

$$\sum_{i=1}^{n} p(scf_i \mid v) = 1 \quad \ldots (2)$$

Here, i and j are natural numbers, scf_i and scf_j are two SCF types that verb v enters, and variables in formulas henceforth will hold the same meanings. In actual application, the probability $p(scf_i|v)$ is estimated from the observed frequency $f(scf_i, v)$, and the conditional probability $p(scf_i|scf_j, v)$ is assumed to be zero. This means any two SCF types entered by a given verb are taken for granted to be probabilistically independent from each other. However, this assumption can sometimes be far from appropriate.

4 Diathesis Alternations and Filtering

Much linguistic research focusing on child language acquisition has revealed that many children are able to produce new grammatical sentences from what they have learned (Peters, 1983; Ellis, 1985). This implies that the widely-used independence assumption in the field of NLP may not be very appropriate, at least for syntactic patterns. If this assumption should be removed, a possible heuristic could be the information of diathesis alternations, which is also another convincing counterargument. Diathesis alternations are generally regarded as alternative ways in which verbs express their arguments. Examples are as follows:

a. He broke the glass.
b. The glass broke.
c. Ta1 chi1 le0 pin2guo3.
(他 吃 了 苹果。)
d. Ta1 ba3 pin2guo3 chi1 le0[1].
(他 把 苹果 吃 了。)

In the above examples, the English verb *break* takes the causative-inchoative alternation as shown in sentences a and b, while sentences c and d indicate that the Chinese verb *chi1* (吃, eat) may enter the *ba*-object-raising alternation where the object is shifted forward by the preposition *ba3* (把) to the location between the subject and the predicate, as illustrated in Figure 1.

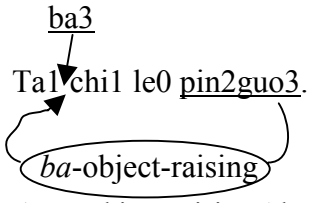


Figure 1. Ba-object-raising Alternation.

Subcategorization of verbs has much to do with diathesis alternations, and most SCF researchers regard information of diathesis alternation as an indispensable part of subcategorization (Korhonen, 2001; McCarthy, 2001). Therefore, one may conclude that, for subcategorization acquisition, the independence assumption supporting the MLE filter is not as appropriate as previously thought.

For a given verb, the assumption will be appropriate and sufficient if and only if there is no diathesis alternation between all the SCFs it enters, and formula (1) and (2) in Section 3 are efficient enough to serve as a foundation for the MLE filtering method. Otherwise, if there are diathesis alternations between some of the SCFs that a verb enters, then formula (1) and (2) must be modified as illustrated in formula (3) and (4). In either case, for the sake of convenience, it would be better to combine the formulas as shown in (5) and (6).

$$\exists i, \exists j, i \neq j, p(scf_i \mid scf_j, v) > 0 \quad \ldots (3)$$

$$\sum_{i=1}^{n} p(scf_i \mid v) > 1 \quad \ldots (4)$$

$$\forall i, \forall j, i \neq j, p(scf_i \mid scf_j, v) \geq 0 \quad \ldots (5)$$

$$\sum_{i=1}^{n} p(scf_i \mid v) \geq 1 \quad \ldots (6)$$

For English verbs, previous research has achieved great progress in diathesis alternation and relative applications, such as the work of Levin (1993) and McCarthy (2001). Besides, Korhonen (1998) has proved that diathesis alternation could be used as heuristic information for backoff estimates to improve the general performance of subcategorization acquisition. However, determining where and how to seed the heuristic remains difficult.

Korhonen (1998) employed diathesis alternations in Briscoe and Carroll's system to improve the performance of their BHT filter. Although the precision rate increased from 61.22% to

[1] The numbers in sentences c and d, which are pinyin notations, show tones of the Chinese syllables, and the two sentences, in English, generally mean *He ate an apple*.

69.42% and the recall rate from 44.70% to 50.81%, the results were still not accurate enough for possible practical NLP uses.

Korhonen obtained her one-way diathesis alternations from the ANLT dictionary (Boguraev and Briscoe, 1987), calculated the alternating probability $p(scf_j|scf_i)$ according to the number of common verbs that took the alternation ($scf_i \rightarrow scf_j$), and used formula (7) and (8), where w is an empirical weight, to adjust the previously estimated $p(scf_i|v)$:

$$\text{If } p(scf_i|scf_j, v)>0,$$
$$p(scf_i|v) = p(scf_i|v) - w(p(scf_i|v) \bullet p(scf_j|scf_i)) \quad \ldots(7)$$
$$\text{If } p(scf_i|v)>0 \ \&\ p(scf_j|v)=0,$$
$$p(scf_i|v) = p(scf_i|v) + w(p(scf_i|v) \bullet p(scf_j|scf_i)) \quad \ldots(8)^2$$

Following the adjustment, a BHT filter with a confidence rate of 95% was used to check the SCF hypotheses.

This method removes the assumption of independence among SCF types but establishes another assumption of independence between $p(scf_j|scf_i)$ and certain verbs, which assumes that all verbs take each diathesis alternation with the same probability. Nevertheless, linguistic knowledge tells us that verbs often enter different diathesis alternations and can be classified accordingly. Consider the following examples:

e. He broke the glass. / The glass broke.
f. The police dispersed the crowd. / The crowd dispersed.
g. Mum cut the bread. / *The bread cut.

Both of the English verbs "break" and "disperse" can take the causative-inchoative alternation and, hence, may be classified together, while the verb "cut" does not take this alternation. Therefore, the newly established assumption doesn't fit the actual situation either, and the probability sums $\Sigma_i p(scf_i|v)$ and $\Sigma_{i,j} p(scf_i|scf_j, v)$ neither need or can be normalized.

Based on the above methodology, we formed a new filtering method with diathesis alternations as heuristic information, which is, in fact, derived from the simple MLE filter and based on formula (5) and (6). The algorithm can be briefly expressed as shown in Table 2.

Table 2. The New Filtering Method.

For hypotheses of a given verb v,
1. if $p(scf_i|v) > \theta_1$,
   accept scf_i into the output set $\boldsymbol{S}$;
2. else
   if $p(scf_i|v) > \theta_2$,
   & $p(scf_i|scf_j, v) > 0$,
   & $scf_j \in \boldsymbol{S}$,
   accept scf_i into set $\boldsymbol{S}$;
3. Go to step 1 until $\boldsymbol{S}$ doesn't increase.

In our method, two filters are employed. For each verb involved, first a common MLE filter is used, but it employs a threshold θ_1 that is much higher than usual, and those SCF hypotheses that satisfy the requirement are accepted. Then, all of the remainder of the hypotheses are checked by another MLE filter seeded with diathesis alternations as heuristic information and equipped with a much lower threshold θ_2. Any hypothesis scf_i left out by the first filter will be accepted if its probability exceeds θ_2 and it is an alternative of an SCF type scf_j that has been accepted by the first filter, which means that $p(scf_i|scf_j, v)>0$ and $scf_j \in \boldsymbol{S}$. The filtering process will be performed repeatedly for those unaccepted hypotheses until no more hypotheses can be accepted for the verb.

5 Experimental Evaluation

We implemented an acquisition experiment on Korhonen's evaluation resources with the above-mentioned filtering method.

The diathesis alternations in use are also those provided by Korhonen, except that we used them in a two-way manner ($scf_i \leftrightarrow scf_j$) instead of one-way ($scf_i \rightarrow scf_j$), because the two involved SCF types are usually alternative pragmatic formats of the concerned verb, as shown in examples in Section 3 and 4.

In the experiment we empirically set $\theta_1 = 0.2$, which is ten times of Korhonen's threshold for her MLE filter; $\theta_2 = 0.002$, which is one tenth of Korhonen's. Thus, in a token set of hypotheses no more than 1000, an SCF type scf_i will be accepted if it occurs two times or more and has a diathesis alternative type scf_j already accepted for the verb.

The gold standard was the manually analysed results by Korhonen. Precision, recall and F-measure were calculated via expressions given in Section 2.

Table 3 lists the performances of the baseline method of non-filtering (No_f), MLE filtering with $\theta = 0.02$, and our filtering method on the

[2] For the sake of consistency in this paper and for the convenience of understanding, formulae formats here are modified. They may look different from those of Korhonen (1998), but they are actually the same.

evaluation corpus, and also gives the best results of Korhonen's method that is using extra semantic information (Kor) to make a comparison. Here, Ab_R is the absolute recall ratio, Re_R the relative recall ratio, Ab_F the absolute F-measure that is calculated from Precision and Ab_R, and Re_F the relative F-measure that is from Precision and Re_R.

Table 3. Performance Comparison.

Methods	No-f	MLE	ours	Kor
P(%)	47.85	67.89	91.18	87.1
Ab_R(%)	34.62	32.52	32.52	71.2
Re_R(%)	100	93.93	93.93	85.27
Ab_F	40.17	43.98	47.94	78.35
Re_F	64.73	78.81	92.53	86.18

The evaluation shows that our new filtering method improved the acquisition performance remarkably: a. Compared with MLE, precision increased by 23.29%, recall ratio remained unchanged, absolute F-measure increased by 3.96, and relative F-measure increased by 13.72; b. Compared with Korhonen's best results, precision, Re_R and Re_F also increased respectively[3]. Thus, the general performance of our filtering method makes the acquired lexicon much more practical for further manual proofreading and other NLP uses.

What's more, the data shown in Table 3 implies that there is little room left for improvement of the statistical filter, since the absolute recall ratio is only 2.1% lower than that of the non-filtering method. Whereas, detailed analysis of the evaluation corpus shows that the hypothesis generator accounts for about 95% of those unrecalled and wrongly recalled SCF types, which indicates, for the present time, more improvement efforts need to be made on the first step of subcategorization acquisition, i.e. hypothesis generation.

6 Conclusion

Our new filtering method removed the inappropriate assumptions and takes much more advantage of what can be observed in the corpus by drawing on the alternative relationship between SCF hypotheses with higher and lower frequencies. Unlike the semantically motivated method (Korhonen, 2001, 2002), which is dependent on verb classifications that linguistic resources are able to provide, our filter needs no prior knowledge other than reasonable diathesis alternation information and may work well for most verbs in other languages with sufficient predicative tokens.

Our experimental results suggest that the proposed technique improves the general performance of the English subcategorization acquisition system, and leaves only a little room for further improvement in statistical filtering methods. However, approaches that are more complicated still exist theoretically, for instance, some SCF types unseen by the hypothesis generator may be recalled by integrating semantic verb-classification information into the system.

More essential aspects of our future work, however, will focus on improving the performance of the hypothesis generator, and testing and applying the acquired subcategorization lexicons in some concrete NLP tasks.

Acknowledgement This research has been jointly sponsored by the NSFC project No. 60373101 and the post-doctor scholarship of foreign linguistics and literature in Heilongjiang University. And at the same time, our great thanks go to Dr. Anna Korhonen for her public evaluation resources, and Dr. Chrys Chrystello for his helpful advice on the English writing of this paper.

References

Boguraev B. K., E. J. Briscoe. Large lexicons for natural language processing utilizing the grammar coding system of the Longman Dictionary of Contemporary English. *Computational Linguistics*, 1987: 219-240

Brent, M., From Grammar to Lexicon: unsupervised learning of lexical syntax, *Computational Linguistics* 19(3) 1993: 243-262.

Briscoe, Ted and John Carroll, Automatic extraction of subcategorization from corpora, *Proceedings of the 5th ACL Conference on Applied Natural Language Processing*, Washington, DC, 1997: 356-363.

Chomsky, Noam, *Aspects of the Theory of Syntax*, MIT Press, Cambridge, 1965.

Chrupala, Grzegorz, Acquiring Verb Subcategorization from Spanish Corpora, *PhD program "Cogni-*

[3] Korhonen (2002) reported the non-filtering absolute recall ratio of her experiment was about 83.5%. She didn't give any explanation with her evaluation resources why here non-filtering Ab_R was so much lower. Therefore, the Ab_R and Ab_F figures are not comparable here.

tive Science and Language", Universitat de Barcelona, 2003: 67-68.

Ellis, R. *Understanding Second language Acquisition*, Oxford University Press.1985

Gamallo, P., Agustini, A. and Lopes Gabriel P., Using Co-Composition for Acquiring Syntactic and Semantic Subcategorisation, *Proceedings of the Workshop of the ACL Special Interest Group on the Lexicon (SIGLEX)*, Philadelphia, 2002: 34-41.

Han, Xiwu, Tiejun Zhao, Haoliang Qi, and Hao Yu, Subcategorization Acquisition and Evaluation for Chinese Verbs, *Proceedings of the COLING 2004*, 2004: 723-728.

Korhonen, Anna, Automatic Extraction of Subcategorization Frames from Corpora –Improving Filtering with Diathesis Alternations, 1998. Please refer to http://www.folli.uva.nl/CD/1998/pdf/keller/korhonen.pdf

Korhonen, Anna, *Subcategorization Acquisition*, Dissertation for PhD, Trinity Hall University of Cambridge, 2001.

Korhonen, Anna, *Subcategorization Acquisition*, Technical Report Number 530, Trinity Hall University of Cambridge, 2002.

Korhonen, Anna, Yuval Krymolowski, Zvika Marx, Clustering Polysemic Subcategorization Frame Distributions Semantically, *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, 2003: 64-71.

Korhonen, Anna. Subcategorization Evaluation Resources. http://www.cl.cam.ac.uk/users/alk23/subcat/subcat.html. 2005

Levin, B., *English Verb Classes and Alternations*, Chicago University Press, Chicago, 1993.

McCarthy, D., *Lexical Acquisition at the Syntax-Semantics Interface: Diathesis Alternations, Subcategorization Frames and Selectional Preferences*, PhD thesis, University of Sussex, 2001.

Peters, A. The Unit of Language Acquisition, Cambridge University Press. 1983.

Sarkar, A. and Zeman, D., Automatic Extraction of Subcategorization Frames for Czech, *Proceedings of the 19th International Conference on Computational Linguistics*, Saarbrucken, Germany, 2000. Please refer to http://www.sfu.ca/~anoop/papers/pdf/coling0final.pdf

Shulte im Walde, Sabine, Inducing German Semantic Verb Classes from Purely Syntactic Subcategorization Information, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 2002: 223-230.

Local constraints on sentence markers and focus in Somali

Katherine Hargreaves
School of Informatics
University of Manchester
Manchester M60 1QD, UK
kat@evilution.co.uk

Allan Ramsay
School of Informatics
University of Manchester
Manchester M60 1QD, UK
Allan.Ramsay@manchester.ac.uk

Abstract

We present a computationally tractable account of the interactions between sentence markers and focus marking in Somali. Somali, as a Cushitic language, has a basic pattern wherein a small 'core' clause is preceded, and in some cases followed by, a set of 'topics', which provide scene-seting information against which the core is interpreted. Some topics appear to carry a 'focus marker', indicating that they are particularly salient. We will outline a computationally tractable grammar for Somali in which focus marking emerges naturally from a consideration of the use of a range of sentence markers.

1 Introduction

This paper presents a computationally tractable account of a number of phenomena in Somali. Somali displays a number of properties which distinguish it from most languages for which computational treatments are available, and which are potentially problematic. We therefore start with a brief introduction to the major properties of the language, together with a description of how we cover the key phenomena within a general purpose NLP framework.

2 Morphology

Somali has a fairly standard set of inflectional affixes for nouns and verbs, as outlined below. In addition, there are a substantial set of 'spelling rules' which insert and delete graphemes at the boundaries between roots and suffixes (and clitics). There is not that much to be said about the spelling rules – Fig. 1 shows the format of a typical rule, which we compile into an FST to be used during the process of lexical lookup.

```
[q/x/c/h,↑,v0] ==> [+, k, v0]
```

Figure 1: Insert *'k'* and a morpheme boundary between *'q/x/c/h'* and a following vowel

The rule in Fig. 1 would, for instance, say that the surface form *'saca'* might correspond to the underlying form *'sac+ka'*, with a morpheme boundary and a *'k'* inserted after the *'c'*. These rules, of which we currently employ about 30, can be efficiently implemented using the standard machinery of cascaded FSTs (Koskiennemi, 1985) interwoven with the general lookup process.

2.1 Noun morphology

In general, a noun consists of a root and a single affix, which provides a combination of gender and number marking. The main complication is that there are several declension classes, with specific singular and plural suffixes for groups of classes (e.g. the plural ending for declensions 1 and 3 is *'o'*) (Saeed, 1999; Lecarme, 2002). Some plural forms involve reduplication of some part of the word ending, e.g. declension 4 nouns form their plural by adding *'aC'* where *'C'* is the final consonant of the root, but this can easily be handled by using spelling rules.

2.2 Verb morphology

Verb morphology is slightly more complex. Again, a typical verb consists of a root plus a number of affixes. These include derivational affixes (Somali includes a passivising form which can only be applied to verbs which have a 'causative' argument, and a causative affix which adds such

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 337–344,
Sydney, July 2006.

an argument) and a set of inflectional affixes which mark aspect, tense and agreement (Andrzejewski, 1968).

The forms of the tense and agreement markers vary depending on whether the clause containing the verb is the main clause or is a subordinate clause (either a relative clause or a sentential complement), marked by ±`main` and on whether it is in a context where the subject is required to be a zero item, marked by ±`fullForm`. Note that the situation here is fairly complicated: `-fullForm` versions are required in situations where the subject is forced by local syntactic constraints to be a zero. There are also situations where the subject is omitted for discourse reasons, and here the `+fullForm` version is used ((Lecarme, 1995) uses the terms 'restrictive' and 'extensive' for `-fullForm` and `+fullForm` respectively).

2.3 Cliticisation

There are a number of Somali morphemes which can appear either bound to an adjacent word (usually the preceding word) or as free-standing lexical items. The sentence marker *'waa'* and the pronoun *'uu'*, for instance, combine to produce the form *'wuu'* when they are adjacent to one another. In several cases, there are quite dramatic morphophonemic alterations at the boundary, so that it is extremely important to ensure that the processes of applying spelling rules and inspecting the lexicon are appropriately interwoven. The definite articles, in particular, require considerable care. There are a number of forms of the definite article, as in Fig. 2:

	masculine	feminine
the-acc	ka	ta
the-nom	ku	tu
'remote' (nom or acc)	kii	tii
this-acc	tan	kan
this-nom	tanu	kanu
that-acc	taas	kaas
that-nom	taasu	kaasu

Figure 2: Definite articles

We deal with this by assuming that determiners have the form gender-root-case, where the gender markers are *'k-'* (masculine) and *'t-'* (feminine), and the case markers are *'-'* (accusative) and *'-u'* (nominative), with spelling rules that collapse *'kau'* to *'ku'* and *'kiiu'* to *'kii'*.

The definite articles, however, cliticise onto the preceding word, with consequential spelling changes. It is again important to ensure that the spelling changes are applied at the right time to ensure that we can recognise *'barahu'* as *'bare'* plus *'k+a+u'*, with appropriate changes to the *'e'* at the end of the root *'bare'* and the *'k'* at the start of the determiner *'ku'*.

3 Syntax

3.1 Framework

The syntactic description is couched in a framework which provides a skeletal version of the HPSG schemas, supplemented by a variant on the well-known distinction between internal and external syntax.

3.1.1 Lexical heads and their arguments

We assume that lexical items specify a (possibly empty) list of required arguments, together with a description of whether these arguments are normally expected to appear to the left or right. The direction in which the arguments are expected is language dependent, as shown in Fig. 3. Note that the description of where the arguments are to be found specifies the order of combination, very much like categorial descriptions. The description of an English transitive verb, for instance, is like the categorial description `(S\NP)/NP`, which corresponds to an SVO surface order.

```
English transitive verb(SOV)
{syn(nonfoot(head(cat(xbar(+v, -n)))),
     subcat(args(["NP"(obj)→, "NP"(subj)←])))}
Persian transitive verb (SOV)
{syn(nonfoot(head(cat(xbar(+v, -n)))),
     subcat(args(["NP"(obj)←, "NP"(subj)←])))}
Arabic transitive verb (VSO)
{syn(nonfoot(head(cat(xbar(+v, -n)))),
     subcat(args(["NP"(subj)→, "NP"(obj)→])))}
```

Figure 3: Subcat frames

3.1.2 Adjuncts and modifiers

Items such as adjectival phrases, PPs and relative clauses which add information about some target item combine via a principle captured in Fig. 4

```
R ⟹ {syntax(target=T→, result=R)}, T
R ⟹ T, {syntax(target=T←, result=R)}
```

Figure 4: Modifiers and targets

Then if we said that an English adjective was of type {`syntax(target="`$\overrightarrow{\texttt{NN}}$`", result="NN")` the first rule in Fig. 4 would allow it to combine with an NN to its right to form an NN, and likewise saying that a PP was of type {`syntax(target="`$\overleftarrow{\texttt{VP}}$`", result="VP")` would allow it to combine with a VP to its left to form a VP.

3.1.3 Non-canonical order

The patterns and principles outlined in §3.1.1 and §3.1.2 specify the unmarked orders for the relevant phenomena. Other orders are often permitted, sometimes for discourse reasons (particularly in free word order languages such as Arabic and Persian) and sometimes for structural reasons (e.g. the left shifting of the WH-pronoun in *'I distrust the man who$_i$ she wants to marry $\emptyset_i$.'*).

We take the view that rather than introducing explicit rules to allow for various non-canonical orders, we will simply allow all possible orders subject to the application of penalties. This approach has affinities with optimality theory (Grimshaw, 1997), save that our penalties are treated cumulatively rather than being applied once and for all to competing local analyses. The algorithm we use for parsing withing this framework is very similar to the algorithm described by (Foth et al., 2005), though we use the scores associated with partial analyses to guide the search for a complete analysis, whereas Foth et al. use them to choose a complete but flawed analysis to be reconstructed. We have described the application of this algorithm to a variety of languages (including Greek, Spanish, German, Persian and Arabic) elsewhere (Ramsay and Schäler, 1997; Ramsay and Mansour, 2003; Ramsay et al., 2005): space precludes a detailed discussion here.

3.1.4 Internal:external syntax

In certain circumstances a phrase that looks as though it belongs to category A is used in circumstances where you would normally expect an item belonging to category B. The phrase *'eating the owl'* in *'He concluded the banquet by eating the owl.'*, for instance, has the internal structure of a VP, but is being used as the complement of the preposition *'by'* where you would normally expect an NP. This notion has been around for too long for its origin to be easily traced, but has been used more recently in (Malouf, 1996)'s addition of 'lexical rules' to HPSG for treating English nominal gerunds, and in (Sadler, 1996)'s description of the possibility of allowing a single c-structure to map to multiple f-structures in LFG. We write 'equivalence rules' of the kind given in Fig. 5 to deal with such phenomena:

```
{syn(head(cat(xbar(-v, +n))),
     +specified)}
<==>{syn(head(cat(xbar(+v, -n)),
               vform(participle,present)),
          subcat(args([{struct(B)}])))}
```

Figure 5: External and internal views of English verbal gerund

The rule in Fig. 5 says that if you have a present participle VP (something of type `+v, -n` which has vform `participle, present` and which needs one more argument) then you can use whereever you need an NP (type `-v, +n` with a specifier `+specified`).

3.2 Somali syntax

As noted earlier, the framework outlined in §3.1 has been used to provide accounts of a number of languages. In the current section we will sketch some of the major properties of Somali syntax and show how they can be captured within this framework.

3.2.1 The 'core & topic' structure

Every Somali sentence has a 'core', or 'verbal complex' (Svolacchia et al., 1995), consisting of the verb and a number of pronominal elements. The structure of the core can be fairly easily described by the rule in Fig. 6:

```
CORE ==> SUBJ,(OBJ1),(ADP*),(OBJ2),VERB
```

Figure 6: The structure of the core

The situation is not, in fact, quite as simple as suggested by Fig. 6. The major complications are outlined below:

1. the third person object pronouns are never actually written, so that in many cases what you see has the form `SUBJ, VERB`, as in (1a), rather than the full form given in (1b) (we will write *'(him)'* to denote zero pronouns):

   (1) a. uu (him) sugay
   he (him) waited for

   b. uu i sugay
   he me waited for

2. The second complication arises with ditransitive verbs. The distinction between OBJ1

and OBJ2 in Fig. 6 simply corresponds to the surface order of the two pronouns, and has very little connection with their semantic roles (Saeed, 1999). Thus each of the sentences in (2a) could mean *'He gave me to you'*, and neither of the sentences in (2b) is grammatical.

(2)

a.	i.	uu	i	kaa	siiyey
		he	me1	you2	gave
	ii.	uu	ku	kay	siiyey
		he	you1	me2	gave
b.	i.	uu	kay	ku	siiyey
		he	me2	you1	gave
	ii.	uu	kaa	i	siiyey
		he	you2	me1	gave

3. The next problem is that subject pronouns are also sometimes omitted. There are two cases: (i) in certain circumstances, the subject pronoun *must* be omitted, and when this happens the verb takes a form which indicates that this has happened. (ii) in situations where the subject is normally present, and hence where the verb has its standard form, the subject may nonetheless be omitted (usually for discourse reasons) (Gebert, 1986).
4. There are a small number of preposition-like items, referred to as adpositions in Fig. 6, which can occur between the two objects, and which cliticise onto the preceding pronoun if there is one. The major complication here is that just like prepositions, these require an NP as a complement: but unlike prepositions, they can combine either with the preceding pronoun or the following one, or with a zero pronoun. Thus a core like (3) has two analyses, as shown in Fig. 7:

(3)

	uu	ika		sugaa
	uu	i ka		sugaa
	he	me at	(it)	waits-for

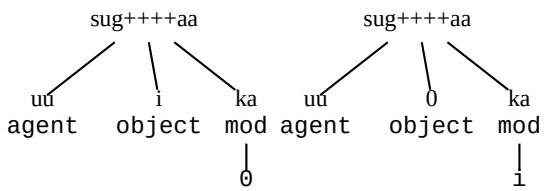


Figure 7: Analyses for (3) (0.010 secs)

The second analysis in Fig. 7, *'he waits for it at me'*, doesn't make much sense, but it is nonetheless perfectly grammatical.

5. Finally, there are a number of other minor elements that can occur in the core. We do not have space to discuss these here, and their presence or absence does not affect the discussion in §3.3 and §3.4.

To capture these phenomena within the framework outlined in §3.1, we assign Somali transitive verbs a subcat frame like the one in Fig. 8 (the patterns for intransitive and ditransitive verbs differ from this in the obvious ways).

```
{syn(nonfoot(head(cat(xbar(+v, -n)))),
     subcat(args(["NP"(obj, +clitic),
                  "NP"(subj, +clitic)])),
     foot(...))}
```

Figure 8: Somali transitive verb

Fig. 8 says that the core of a Somali sentence is a clause of the form S-O-V, where S and O are both clitic pronouns.

The canonical position of S and O is as given. They can appear further to the left than that to allow for clitic modifiers: exactly where they can go is specified by requiring the clitic modifiers to appear adjacent to the verb (subject to further local constraints on their positions relative to one another), and requiring S and O to fall inside the scope of the 'sentence markers'.

3.3 Sentence markers

A core by itself cannot be uttered as a free standing sentence. At the very least, it has to include a 'sentence marker'. The simplest of these is the word *'waa'*. (4), for instance, is a well-formed sentence, with the structure shown in Fig. 9.

(4)

		wuu		sugaa.
	waa	uu		sugaa.
	s-marker	he	(it)	waits-for

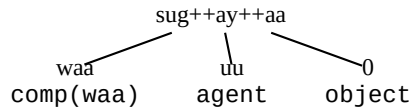


Figure 9: Analysis for (4) (0.01 secs)

Note that the pronoun *'uu'* cliticises onto the end of the sentence marker *'waa'*, producing the written form *'wuu'*, as discussed above.

In general, however, the situation is not quite as simple as in (4). Most sentences contain NPs other than the pronouns in the core. The first such examples involve introducing 'topics' in front of the sentence marker.

Topics are normally definite NPs or PPs which set the scene for the interpretation of the core. A typical example is given in (5):

(5)

ninka	wuu	sugaa.
nim ka	waa uu	sugaa.
man the	S-marker he (him)	waits-for

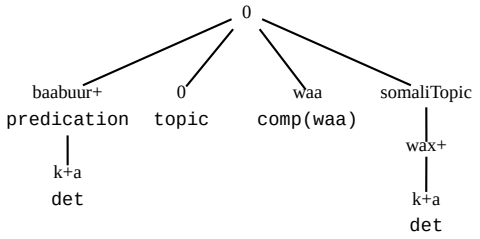


Figure 10: Sentence with topic

The analysis in Fig. 10 was obtained by exploiting an equivalence rule which says that an item which has the internal properties of a `-clitic` NP can be used as a 'topic', which we take to be a sentence modifier.

Topics set the scene for the interpretation of the core by providing potential referents for the pronominal elements in the core. There are no very strong syntactic links between the topics and the clitic pronouns – if a topic is `+nom` then it will provide the referent for the subject, but in some (focused) contexts subject referents are not explicitly marked as `+nom`. The situation is rather like saying *'You know that man we were talking about, and you know the girl we were talking about. Well, she's waiting for him.'*.

$$
\begin{aligned}
&topical(ref(\lambda B(NIM(B))))\\
&\&claim(\exists C:\{aspect(now,simple,C)\}\\
&\qquad \theta(C,agent,ref(\lambda D female(D)))\\
&\qquad \&\theta(C,object,ref(\lambda F thing(F)))\\
&\qquad \&SUG(C))
\end{aligned}
$$

Figure 11: Interpretation of (5)

The logical form given in Fig. 11, which was constructed using using standard compositional techniques (Dowty et al., 1981), says the speaker is marking some known man $ref(\lambda B(NIM(B)))$ as being topical, and is then making a claim about the existence of a waiting event $SUG(C)$ involving some known female as its agent and some other known entity as its object. Note that we include discourse related information – that the speaker is first marking something as being topical and then making a claim – in the logical form. This seems like a sensible thing to do, since this information is encoded by lexical and syntactic choices in the same way as the propositional content itself, and hence it makes sense to extract it compositionally at the same time and in the same way as we do the propositional content.

Somali provides a number of such sentence markers. *'in'* is used for marking sentential complements, in much the same way as the English complementiser *'that'* is used to mark the start of a sentential clause in *'I know that she like strawberry icecream.*' (Lecarme, 1984). There is, however, an alternative form for main clauses, where one of the topics is marked as being particularly interesting by the sentence markers *'baa'* or *'ayaa'* (*'baa'* and *'ayaa'* seem to be virtually equivalent, with the choice between them being driven by stylistic/phonological considerations):

(6) baraha baa ninka sugaa.

'baa'/*'ayaa'* and *'waa'* are in complementary distribution: every main clause has to have a sentence marker, which is nearly always one of these two, and they never occur in the same sentence. The key difference is that *'baa'* marks the item to its left as being particularly significant. Ordinary topics introduce an item into the context, to be picked up by one of the core pronouns, without marking any of them as being more prominent than the others. The item to the left of *'baa'* is indeed available as an anchor for a core pronoun, but it is also marked as being more important than the other topics.

We deal with this by assuming that *'baa'* subcategorises for an NP to its left, and then forms a sentence marker looking to modify a sentence to its right. The resulting parse tree for (6) is given in Fig. 12, with the interpretation that arises from this tree in Fig. 13.

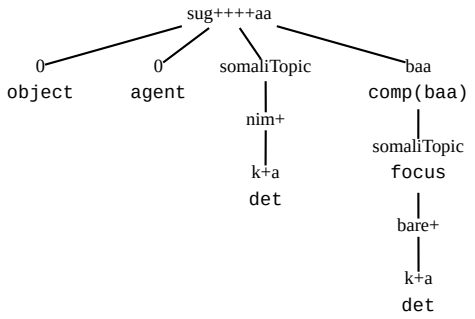


Figure 12: Parse tree for (6)

$$
\begin{aligned}
&topical(ref(\lambda C(NIM(C))))\\
&\&focus(ref(\lambda D(BARE(D))))\\
&\&claim(\exists B:\{aspect(now,simple,B)\}\\
&\qquad \theta(B,object,ref(\lambda Ething(E)))\\
&\qquad \&\theta(B,agent,ref(\lambda Gspeaker(G)))\\
&\qquad \&SUG(B))
\end{aligned}
$$

Figure 13: Interpretation for (6)

Treating *'baa'* as an item which looks first to its left for an NP and then acts as a sentence modifier gives us a fairly simple analysis of (6), ensuring that when we have *'baa'* we do indeed have a

focused item, and also accounting for its complementary distribution with *'waa'*. The fact that the combination of *'baa'* and the focussed NP can be either preceded or followed by other topics means that we have to put very careful constraints on where it can appear. This is made more complex by the fact that the subject of the core sentence can cliticise onto *'baa'*, despite the fact that there may be a subsequent topic, as in (7).

(7) baraha buu ninku sugaa.

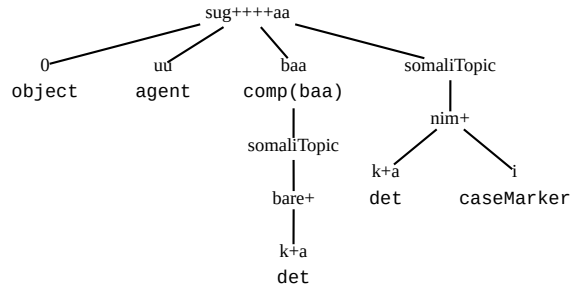


Figure 14: Parse tree for (7)

To ensure that we get the right analyses, we have to put the following constraints on *'baa'* and *'waa'*:

1. if the subject of the core is realised as an explicit short pronoun, it cliticises onto the sentence marker
2. the sentence marker attaches to the sentence before any topics (note that this is a constraint on the order of combination, not on the left←right surface order: the tree in Fig. 14 shows that *'baraha baa'* was attached to the tree before *'ninka'*, despite the fact that *'ninka'* is nearer to the core than *'baraha baa'*.

Between them, these two ensure that we get unique analyses for sentences involving a sentence marker and a number of topics, despite the wide range of potential surface orders.

3.4 Relative clauses & *'waxa'*-clefts

We noted above that in general Somali clauses contain a sentence marker – generally one of *'waa'*, *'baa'* and *'ayaa'* for main clauses, or one of *'in'* for subordinate clauses. There are two linked exceptions to this rule: relative clauses, and *'waxa'*-clefts.

Somali does not possess distinct WH-pronouns (Saeed, 1999). Instead, the clitic pronouns (including the zero third-person pronoun) can act as WH-markers.

This is a bit awkward for any parsing algorithm which depends propagating the WH-marker up the parse tree until a complete clause has been analysed, and then using it to decide whether that clause is a relative clause or not. We do not want to introduce two versions of each pronoun, one with a WH-marker and the other without, and then produce alternative analyses for each. Doing this would produce very large numbers of alternative analyses, since each core item is can be viewed either way, so that a simple clause involving a transitive clause would produce three analyses (one with the subject WH-marked, one with the object WH-marked, and one with neither).

We therefore leave the WH-marking on the clitic pronouns open until we have an analysis of the clause containing them. If we need to consider using this clause in a context where a relative clause is required, we inspect the clitic pronouns and decide which ones, if any are suitable for use as the pivot (i.e. the WH-pronoun which links to the modified analysis).

Relative clauses do *not* require a sentence marker. We thus get analyses of relative clauses as shown in Fig. 15 for (8).

(8)

ninka	wadaya	wuu	shaqeeyayaa
nim ka	wadaya	waa uu	shaqeeyaa
man the	is-driving	s-marker he	is-working

The man who is driving it: he's working

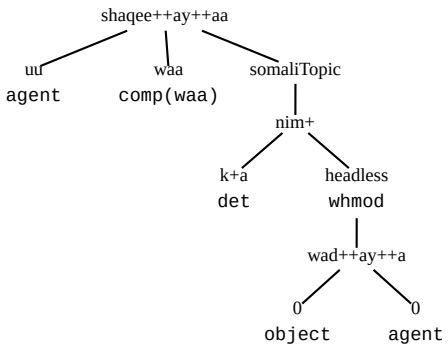


Figure 15: Parse tree for (8)

Note the reduced form of *'wadaya'* in (8). The key here is that the subject of *'wadaya'* is the *'pivot'* of the relative clause (the item linking the clause to the modified nominal). When the subject plays this role it is forced to be a zero item, and it is this that makes the verb take the `-fullForm` versions of the agreement and tense markers.

Apart from the fact that you can't tell whether a clitic pronoun is acting as a WH-marker or not until you see the context, and the requirement for

reduced form verbs with zero subjects, Somali relative clauses are not all that different from relative clauses in other languages. They are, however, related to a phenomenon which is rather less common.

We start by considering nominal sentences. Somali allows for scarenominal sentences consisting of just a pair of NPs. This is a fairly common phenomenon, where the overall semantic effect is as though there were an invisible copula linking them (see Arabic, malay, English 'small clauses', ...). We deal with this by assuming that any accusative NP could be the predication in a zero sentence. The only complication is that in ordinary Somali sentences the only items which follow the sentence marker are clitic pronouns and modifiers. For nominal sentences, the predicative NP, and nothing else, follows the sentence marker.

For uniformity we assume that there is in fact a zero subject, with the `+nom` NP that appears before the sentence marker acting as a topic.

	waxu	waa	baabuurka.
(9)	wax ka I	waa	baabuur ka
	thing the `+NOM`	s-marker	truck the

Any normal NP can appear as the topic of such a sentence. In particular, the noun *'wax'*, which means *'thing'*, can appear in this position:

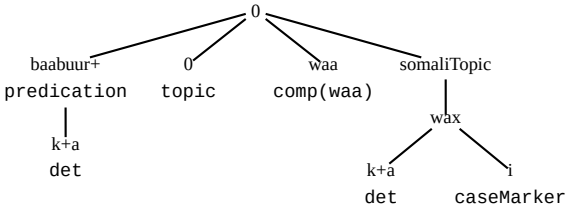


Figure 16: *'the thing: it's the truck'*

The analysis in Fig. 16 corresponds to an interpretation something like *'The thing we were talking about, well it's the truck'*. Note the analysis of *'waxu'* here as the noun *'wax'* followed by the definite article *'ka'* and the nominative case marker *'I'*.

There is no reason why the topic in such a sentence should not contain a relative clause. In (10), for instance, the topic is *'waxaan doonayo I'* – *'the thing which I want'*.

	waxaan	doonayaa	waa	lacag.
(10)	wax ka aan	doonayo I	waa	lacag
	thing the I	want +NOM	s-marker	money

Note that *'doonayaa'* here is being read as the {`+fullForm,-main`} version of the verb *'doonayo'* followed by a cliticised nominative marker *'I'*. The choice of `+fullForm` this time arises because the subject pronoun is not WH-marked, which means that it is not *forced* to be zero: remember that `-fullForm` is used if the local constraints require the subject to be zero, not just if it happens to be omitted for discourse or stylistic reasons. Then in the analysis in Fig. 17 *'wax ka aan doonayo I'* is a `+nom` NP functioning as the topic of a nominal sentence.

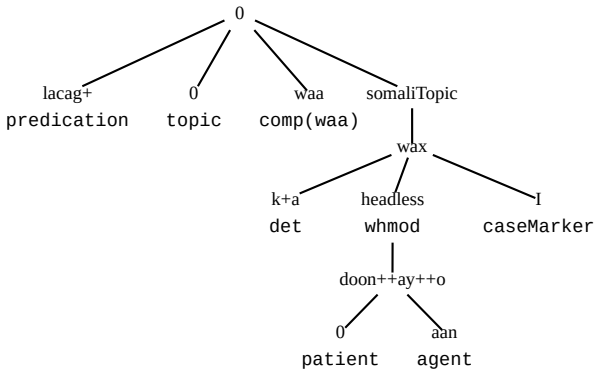


Figure 17: (10) the thing I want: it's some money

So far so simple. *'waxa'*, however, also takes part in a rather more complex construction.

In general, the items that occur as topics in Somali are *definite* NPs (Saeed, 1984). In all the examples above, we have used definite NPs in the topic positions, because that it is what normally happens. If you want to introduce something into the conversation it is more usual to use a *'waxa*-cleft', or 'heralding sentence' (Andrzejewski, 1975).

The typical surface form of such a construction is shown in (11):

	waxaan	doonayaa	lacag.
(11)	waxa aan	doonayaa	lacag
	waxa I	want	money

The key things to note about (11) are as follow:

- There is no sentence marker. Or at any rate, the standard sentence markers *'waa'* and *'baa'* are missing.
- The subject pronoun *'aan'* has cliticised onto the word *'waxa'* to form *'waxaan'*.
- The verb *'doonayaa'* is `+fullForm`
- The noun *'lacag' follows* the verb. This is unusual, since generally NPs are used as topics preceding the core and, generally, the sentence marker.

These facts are very suggestive: (i) the lack of any other item acting as sentence marker suggests that *'waxa'* is playing this role. (ii) the fact that *'uu'* has cliticised onto this item supports this claim, since subject pronouns typically cliticise onto sentence markers rather than onto topic NPs.

We therefore suggest that *'waxa'* here is functioning as sentence marker. Like *'baa'*, it focuses attention on some particular NP, but in this case the NP *follows* the core.

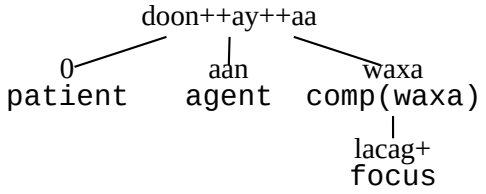


Figure 18: Parse tree for (11)

Thus *'waxa'*, as a sentence marker, is just like *'baa'* except that *'baa'* expects its focused NP to follow it immediately, with the core following that, whereas the order is reversed for *'waxa'* (Andrzejewski, 1975).

It seems extremely likely that *'waxa'*-clefts are historically related to sentences like (10). The subtle differences in the surface forms (presence or absence of *'waa'* and form of the verb), however, lead to radically different analyses. How simple nominal sentences with topics including *'waxa'* and a relative clause turned into *'waxa'*-clefts is beyond the scope of this paper. The key observation here is that *'waxa'*-clefts can be given a straightforward analysis by assuming that *'waxa'* can function as a sentence-marker that focuses attention on a topical NP that *'follows'* the core of the sentence.

4 Conclusions

We have outlined a computational treatment of Somali that runs right through from morphology and morphographemics to logical forms. The construction of logical forms is a fairly routine activity, given that we have carried out this work within a framework that has already been used for a number of other languages, and hence the machinery for deriving logical forms from semantically annotated parse trees is already available. The most notable point about Somali semantics within this framework is the inclusion of the basic illocutionary force within the logic form, which allows us to also treat topic and focus as discourse phenomena within the logical form.

References

B W Andrzejewski. 1968. Inflectional characteristics of the so-called weak verbs in Somali. *African Language Studies*, 9:1–51.

B W Andrzejewski. 1975. The role of indicator particles in Somali. *Afroasiatic Linguistics*, 1(6):123–191.

D R Dowty, R E Wall, and S Peters. 1981. *Introduction to Montague Semantics*. D. Reidel, Dordrecht.

Killian Foth, Wolfgang Menzel, and Ingo Schröder. 2005. Robust parsing with weighted constraints. *Natural Language Engineering*, 11(1):1–25.

L Gebert. 1986. Focus and word order in Somali. *Afrikanistische Arbeitspapiere*, 5:43–69.

J Grimshaw. 1997. Projection, heads, and optimality. *Linguistic Inquiry*, 28:373–422.

K Koskiennemi. 1985. A general two-level computational model for word-form recognition and production. In *COLING-84*, pages 178–181.

J Lecarme. 1984. On Somali complement constructions. In T Labahn, editor, *Proceedings of the Second International Congress of Somali Studies, 1: Linguistics and Literature*, pages 37–54, Hamburg. Helmut Buske.

J Lecarme. 1995. L'accord restrictif en Somali. *Langues Orientals Anciennes Philologie et Linguistique*, 5-6:133–152.

J Lecarme. 2002. Gender polarity: Theoretical aspects of Somali nominal morphology. In P Boucher, editor, *Many Morphologies*, pages 109–141, Somerville. Cascadilla Press.

Robert Malouf. 1996. A constructional approach to english verbal gerunds. In *Proceedings of the Twenty-second Annual Meeting of the Berkeley Linguistics Society*, Marseille.

A M Ramsay and H Mansour. 2003. Arabic morphosyntax for text-to-speech. In *Recent advance in natural language processing*, Sofi a.

A M Ramsay and R Schäler. 1997. Case and word order in English and German. In R Mitkov and N Nicolo, editors, *Recent Advances in Natural Language Processing*. John Benjamin.

A M Ramsay, Najmeh Ahmed, and Vahid Mirzaiean. 2005. Persian word-order is free but not (quite) discontinuous. In *5th International Conference on Recent Advances in Natural Language Processing (RANLP-05)*, pages 412–418, Borovets, Bulgaria.

Louisa Sadler. 1996. New developments in LFG. In Keith Brown and Jim Miller, editors, *Concise Encyclopedia of Syntactic Theories*. Elsevier Science, Oxford.

J I Saeed. 1984. *The Syntax of Focus and Topic in Somali*. Helmut Buske Verlag, Hamburg.

J I Saeed. 1999. *Somali*. John Benjamins Publishing Co, Amsterdam.

M Svolacchia, L Mereu, and A. Puglielli. 1995. Aspects of discourse configurationality in Somali. In K E Kiss, editor, *Discourse Configurational Languages*, pages 65–98, New York. Oxford University Press.

A Collaborative Framework for Collecting Thai Unknown Words from the Web

Choochart Haruechaiyasak, Chatchawal Sangkeettrakarn, Pornpimon Palingoon
Sarawoot Kongyoung and **Chaianun Damrongrat**
Information Research and Development Division (RDI)
National Electronics and Computer Technology Center (NECTEC)
Thailand Science Park, Klong Luang, Pathumthani 12120, Thailand
rdi5@nnet.nectec.or.th

Abstract

We propose a collaborative framework for collecting Thai unknown words found on Web pages over the Internet. Our main goal is to design and construct a Web-based system which allows a group of interested users to participate in constructing a Thai unknown-word open dictionary. The proposed framework provides supporting algorithms and tools for automatically identifying and extracting unknown words from Web pages of given URLs. The system yields the result of unknown-word candidates which are presented to the users for verification. The approved unknown words could be combined with the set of existing words in the lexicon to improve the performance of many NLP tasks such as word segmentation, information retrieval and machine translation. Our framework includes word segmentation and morphological analysis modules for handling the non-segmenting characteristic of Thai written language. To take advantage of large available text resource on the Web, our unknown-word boundary identification approach is based on the statistical string pattern-matching algorithm.

Keywords: Unknown words, open dictionary, word segmentation, morphological analysis, word-boundary detection.

1 Introduction

The advent of the Internet and the increasing popularity of the Web have altered many aspects of natural language usage. As more people turn to the Internet as a new communicating channel, the textual information has increased tremendously and is also widely accessible. More importantly, the available information is varied largely in terms of topic difference and multi-language characteristic. It is not uncommon to find a Web page written in Thai lies adjacent to a Web page written in English via a hyperlink, or a Web page containing both Thai and English languages. In order to perform well in this versatile environment, an NLP system must be adaptive enough to handle the variation in language usage. One of the problems which requires special attention is unknown words.

As with most other languages, unknown words also play an extremely important role in Thai-language NLP. Unknown words are viewed as one of the problematic sources of degrading the performance of traditional NLP applications such as MT (Machine Translation), IR (Information Retrieval) and TTS (Text-To-Speech). Reduction in the amount of unknown words or being able to correctly identify unknown words in these systems would help increase the overall system performance.

The problem of unknown words in Thai language is perhaps more severe than in English or other latin-based languages. As a result of the information technology revolution, Thai people have become more familiar with other foreign languages especially English. It is not uncommon to hear a few English words over a course of conversation between two Thai people. The foreign words along with other Thai named entities are among the new words which are continuously created and widely circulated. To write a foreign word, the transliterated form of Thai alphabets is often used. The Royal Institute of Thailand is the official organization in Thailand who has respon-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 345–352,
Sydney, July 2006.

sibility and authority in defining and approving the use of new words. The process of defining a new word is manual and time-consuming as each word must be approved by a working group of linguists. Therefore, this traditional approach of constructing the lexicon is not a suitable solution, especially for systems running on the Web environment.

Due to the inefficiency of using linguists in defining new lexicon, there must be a way to automatically or at least semi-automatically collect new unknown words. In this paper, we propose a collaborative framework for collecting unknown words from Web pages over the Internet. Our main purpose is to design and construct a system which automatically identifies and extracts unknown words found on Web pages of given URLs. The compiled list of unknown-word candidates is to be verified by a group of participants. The approved unknown words are then added to the existing lexicon along with the other related information such as meaning and POS (part of speech).

This paper focuses on the underlying algorithms for supporting the process of identifying and extracting unknown words. The overall process is composed of two steps: unknown-word detection and unknown-word boundary identification. The first step is to detect the locations of unknown-word occurrences from a given text. Since Thai language belongs to the class of non-segmenting language group in which words are written continuously without using any explicit delimiting character, detection of unknown words could be accomplished mainly by using a word-segmentation algorithm with a morphological analysis. By using a dictionary-based word-segmentation algorithm, locations of words which are not previously included in the dictionary will be easily detected. These unknown words belong to the class of *explicit* unknown words and often represent the transliteration of foreign words.

The other class of unknown words is *hidden* unknown words. This class includes new words which are created through the combination of some existing words in the lexicon. The *hidden* unknown words are usually named entities such as a person's name and an organization's name. The *hidden* unknown words could be identified using the approaches such as n-gram generation and phrase chunking. The scope of this paper focuses only on the extraction of the *explicit* unknown words. However, the design of our framework also includes the extraction of *hidden* unknown words. We will continue to explore this issue in our future works.

Once the location of an unknown word is detected, the second step involves the identification of its boundary. Since we use the Web as our main resource, we could take advantage of its large availability of textual contents. We are interested in collecting unknown words which occur more than once throughout the corpus. Unknown words which occur only once in the large corpus are not considered as being significant. These words may be unusual words which are not widely accepted, or could be misspelling words. Using this assumption, our approach for identifying the unknown-word boundary is based on a statistical pattern-matching algorithm. The basic idea is that the same unknown word which occurs more than once would likely to appear in different surrounding contexts. Therefore, a group of characters which form the unknown word could be extracted by analyzing the string matching patterns.

To evaluate the effectiveness of our proposed framework, experiments using a real data set collected from the Web are performed. The experiments are designed to test each of the two main steps of the framework. Variation of morphological analysis are tested for the unknown-word detection. The detection rate of unknown words were found to be as high as approximately *96%*. Three variations of string pattern-matching techniques were tested for unknown-word boundary identification. The identification accuracy was found to be as high as approximately *36%*. The relatively low accuracy is not the major concern since the unknown-word candidates are to be verified and corrected by users before they are actually added to the dictionary. The system is implemented via the Web-browser environment which provides user-friendly interface for verification process.

The rest of this paper is organized as follows. The next section presents and discusses related works previously done in the unknown-word problem. Section 3 provides an overview of unknown-word problem in the relation to the word-segmentation process. Section 4 presents the proposed framework with underlying algorithms in details. Experiments are performed in Section 5 with results and discussion. The conclusion is given in Section 6.

2 Previous Works

The research and study in unknown-word problem have been extensively done over the past decades. Unknown words are viewed as problematic source in the NLP systems. Techniques in identifying and extracting unknown words are somewhat language-dependent. However, these techniques could be classified into two major categories, one for segmenting languages and another for non-segmenting languages. Segmenting languages, such as latin-based languages, use delimiting characters to separate written words. Therefore, once the unknown words are detected, their boundaries could be identified relatively easily when compared to those for non-segmenting languages.

Some examples of techniques involving segmenting languages are listed as follows. Toole (2000) used multiple decision trees to identify names and misspellings in English texts. Features used in constructing the decision trees are, for example, POS (Part-Of-Speech), word length, edit distance and character sequence frequency. Similarly, a decision-tree approach was used to solve the POS disambiguation and unknown word guessing in (Orphanos and Christodoulakis, 1999). The research in the unknown-word problem for segmenting languages is also closely related to the extraction of named entities. The difference of these techniques to those in non-segmenting languages is that the approach needs to parse the written text in word-level as opposed to character-level.

The research in unknown-word problem for non-segmenting languages is highly active for Chinese and Japanese. Many approaches have been proposed and experimented with. Asahara and Matsumoto (2004) proposed a technique of SVM-based chunking to identify unknown words from Japanese texts. Their approach used a statistical morphological analyzer to segment texts into segments. The SVM was trained by using POS tags to identify the unknown-word boundary. Chen and Ma (2002) proposed a practical unknown word extraction system by considering both morphological and statistical rule sets for word segmentation. Chang and Su (1997) proposed an unsupervised iterative method for extracting unknown lexicons from Chinese text corpus. Their idea is to include the potential unknown words to the augmented dictionary in order to improve the word segmentation process. Their proposed approach also includes both contextual constraints and the joint character association metric to filter the unlikely unknown words. Other approaches to identify unknown words include statistical or corpus-based (Chen and Bai, 1998), and the use of heuristic knowledge (Nie et al. , 1995) and contextual information (Khoo and Loh, 2002). Some extensions to unknown-word identification have been done. An example include the determination of POS for unknown words (Nakagawa et al. , 2001).

The research in unknown words for Thai language has not been widely done as in other languages. Kawtrakul et al. (1997) used the combination of a statistical model and a set of context sensitive rules to detect unknown words. Our framework has a different goal from previous works. We consider unknown-word problem as collaborative task among a group of interested users. As more textual content is provided to the system, new unknown words could be extracted with more accuracy. Thus, our framework can be viewed as collaborative and statistical or corpus-based.

3 Unknown-Word Problem in Word Segmentation Algorithms

Similar to Chinese, Japanese and Korea, Thai language belongs to the class of non-segmenting languages in which words are written continuously without using any explicit delimiting character. To handle non-segmenting languages, the first required step is to perform word segmentation. Most word segmentation algorithms use a lexicon or dictionary to parse texts at the character-level. A typical word segmentation algorithm yields three types of results: known words, ambiguous segments, and unknown segments. Known words are existing words in the lexicon. Ambiguous segments are caused by the overlapping of two known words. Unknown segments are the combination of characters which are not defined in the lexicon.

In this paper, we are interested in extracting the unknown words with high precision and recall results. Three types of unknown words are *hidden*, *explicit* and *mixed* (Kawtrakul et al. , 1997). Hidden unknown words are composed by different words existing in the lexicon. To illustrate the idea, let us consider an unknown word *ABCD* where *A*, *B*, *C*, and *D* represents individual characters. Suppose that *AB* and *CD* both ex-

ist in a dictionary, then *ABCD* is considered as a hidden unknown word. The explicit unknown words are newly created words by using different characters. Let us again consider an unknown word ABCD. Suppose that there is no substring of *ABCD* (i.e., *AB*, *BC*, *CD*, *ABC*, *BCD*) exists in the dictionary, then *ABCD* is considered as explicit unknown words. The mixed unknown words are composed of both existing words in a dictionary and non-existing substrings. From the example of unknown string *ABCD*, if there is at least one substring of *ABCD* (i.e., *AB*, *BC*, *CD*, *ABC*, *BCD*) exists in the dictionary, then *ABCD* is considered as a mixed unknown word.

It can be immediately seen that the detection of the *hidden* unknown words are not trivial since the parser would mistakenly assume that all the fragments of the words are valid, i.e., previously defined in the dictionary. In this paper, we limit ourself to the extraction of the *explicit* and *mixed* unknown words. This type of unknown words usually represent the transliteration of foreign words. Detection of these unknown words could be accomplished mainly by using a word-segmentation algorithm with a morphological analysis. By using a dictionary-based word-segmentation algorithm, locations of words which are not previously defined in the lexicon could be easily detected.

4 The Proposed Framework

The overall framework is shown in Figure 1. Two major components are information agent and unknown-word analyzer. The details of each component are given as follows.

- **Information agent**: This module is composed of a Web crawler and an HTML parser. It is responsible for collecting HTML sources from the given URLs and extracting the textual data from the pages. Our framework is designed to support multi-user and collaborative environment. The advantage of this design approach is that unknown words could be collected and verified more efficiently. More importantly, it allows users to select the Web pages which suit their interests.

- **Unknown-word analyzer**: This module is composed of many components for analyzing and extracting unknown words. Word segmentation module receives text strings from the information agent and segments them into a list of words. N-gram generation module is responsible for generating hidden unknown-word candidates. Morphological analysis module is used to form initial explicit unknown-word segments. String pattern matching unit performs unknown-word boundary identification task. It takes the intermediate unknown segments and identifies their boundaries by analyzing string matching patterns The results are processed unknown-word candidates which are presented to linguists for final post-processing and verification. New unknown words are combined with the dictionary to iteratively improve the performance of the word segmentation module. Details of each component are given in the following subsections.

4.1 Unknown-Word Detection

As previously mentioned in Section 3, applying a word-segmentation algorithm on a text string yields three different segmented outputs: known, ambiguous, and unknown segments. Since our goal is to simply detect the unknown segments without solving or analyzing other related issues in word segmentation, using the *longest-matching* word segmentation algorithm previously proposed by Poowarawan (1986) is sufficient. An example to illustrate the word-segmentation process is given as follows.

Let the following string denotes a text string written in Thai language: $\{a_1a_2...a_ib_1b_2...b_jc_1c_2...c_k\}$. Suppose that $\{a_1a_2...a_i\}$ and $\{c_1c_2...c_k\}$ are known words from the dictionary, and $\{b_1b_2...b_j\}$ be an unknown word. For the explicit unknown-word case, applying the word-segmentation algorithm would yield the following segments: $\{a_1a_2...a_i\}\{b_1\}\{b_2\}...\{b_j\}\{c_1c_2...c_k\}$. It can be observed that the detected unknown positions for a single unknown word are individual characters in the unknown word itself. Based on the initial statistical analysis of a Thai lexicon, it was found that the averaged number of characters in a word is equal to *7*. This characteristic is quite different from other non-segmenting languages such as Chinese and Japanese in which a word could be a character or a combination of only a few characters. Therefore, to reduce the complexity in unknown-word boundary identification task, the unknown segments could be merged to form multiple-character segments. For exam-

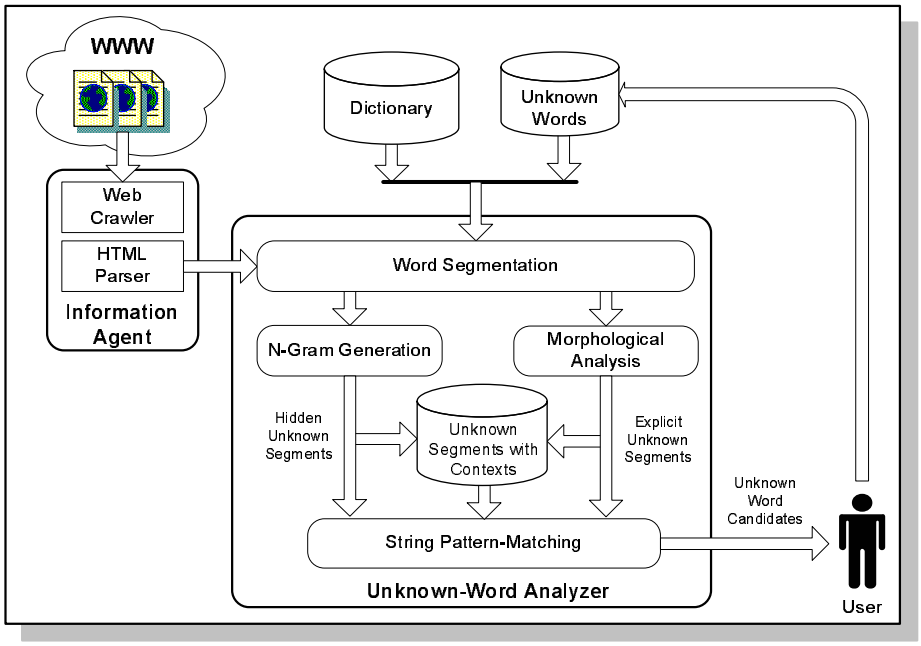


Figure 1: The proposed framework for collecting Thai unknown words.

ple, a merging of two characters per segment would give the following unknown segments: $\{b_1b_2\}\{b_3b_4\}...\{b_{j-1}b_j\}$. In the following experiment section, the merging of two to five characters per segment including the merging of all unknown segments without limitation will be compared.

Morphological analysis is applied to guarantee grammatically correct word boundaries. Simple morphological rules are used in the framework. The rule set is based on two types of characters, front-dependent characters and rear-dependent characters. Front-dependent characters are characters which must be merged to the segment leading them. Rear-dependent characters are characters which must be merged to the segment following them. In Thai written language, these dependent characters are some vowels and tonal characters which have specific grammatical constraints. Applying morphological analysis will help making the unknown segments more reliable.

4.2 Unknown-Word Boundary Identification

Once the unknown segments are detected, they are stored into a hashtable along with their contextual information. Our unknown-word boundary identification approach is based on a string pattern-matching algorithm previously proposed by Boyer and Moore (1977). Consider the unknown-word boundary identification as a string pattern-matching problem, there are two possible strategies: considering the longest matching pattern and considering the most frequent matching pattern as the unknown-word candidates. Both strategies could be explained more formally as follows.

Given a set of N text strings, $\{S_1S_2...S_N\}$, where S_i, is a series of len_i characters denoted by $\{c_{i,1}c_{i,2}...c_{i,len_i}\}$ and each is marked with an unknown-segment position, pos_i, where $1 \leq pos_i \leq len_i$. Given a new string, S_j, with an unknown-segment position, pos_j, the longest pattern-matching strategy iterates through each string, S_1 to S_N and records the *longest* string pattern which occur in both S_j and the other string in the set. On the other hand, the *most frequent* pattern-matching strategy iterates through each string, S_1 to S_N, but records the matching pattern which occur most frequently.

The results from the unknown-word boundary identification are unknown-word candidates. These candidates are presented to the users for verification. Our framework is implemented via a Web-browser interface which provides a user-friendly environment. Figure 2 shows a screen snapshot of our system. Each unknown word is listed within a *text field* box which allows a user to edit and correct its boundary. The contexts could be used as some editing guidelines and are also stored into the database.

Unknown Word	Context
อีเมล์ Add Discard	• 548 : ยังกบดานเงียบ แจ็กพอต 47 ล้าน **อีเมล์**ผู้ส่ง อีเมล์เพื่อน ข้อความถึงเพื่อน • เงียบ แจ็กพอต 47 ล้าน อีเมล์ผู้ส่ง **อีเมล์**เพื่อน ข้อความถึงเพื่อน ข่าวหน้าใน ข่
แจ็กพอต Add Discard	• เปิดเผยอีกว่า นับตั้งแต่มีการออกรางวัล**แจ็กพอต**มาฝ่ายจ่ายรางวัลยังไม่เคยพบว่ามีผู้ถู • บเงินล่าข้าขนาดนี้ที่ผ่านมาเคยมีผู้ถูก**แจ็กพอต**รายหนึ่ง มารับเงินล่าช้า แต่หลังออกรา • อยู่ 47 ล้านบาทนั้น จะนำมารวมกับรางวัล**แจ็กพอต**งวดวันที่1 ต.ค. นี้หรือไม่ ผอ.สำนักงา • ลากกล่าวว่า กรณีที่จะยกยอดเงินมารวมกับ**แจ็กพอต**งวดใหม่ก็ต่อเมืองวดดังกล่าวไม่มีผู้ถ
กสท Add Discard	• ล ที่สำนักสลากรูปแบบใหม่ ชั้น 22 อาคาร **กสท** โทรคมนาคม บางรักเจ้าหน้าที่จ่ายเงินรางว

Figure 2: Example of Web-Based Interface

5 Experiments and Results

In this section, we evaluate the performance of our proposed framework. The corpus used in the experiments is composed of *8,137* newspaper articles collected from a top-selling Thai newspaper's Web site (Thairath, 2003) during 2003. The corpus contains a total of *78,529* unknown words of which *14,943* are unique. This corpus was focused on unknown words which are transliterated from foreign languages, e.g., English, Spanish, Japanese and Chinese. We use the publicly available Thai dictionary *LEXiTRON*, which contains approximately *30,000* words, in our framework (Lexitron, 2006).

We first analyze the unknown-word set to observe its characteristics. Figure 3 shows the plot of unknown-word frequency distribution. Not surprisingly, the frequency of unknown-word usage follows a Zipf-like distribution. This means there are a group of unknown words which are used very often, while some unknown words are used only a few times over a time period. Based on the frequency statistics of unknown words, only about *3%* (*2,375* words out of *78,529*) occur only once in the corpus. Therefore, this finding supports the use of statistical pattern-matching algorithm described in previous section.

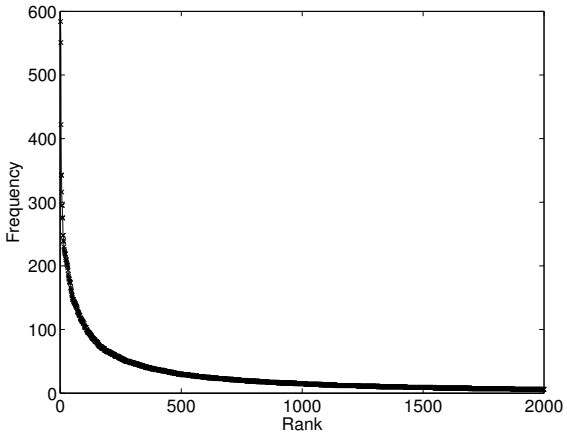


Figure 3: Unknown-word frequency distribution.

5.1 Evaluation of Unknown-Word Detection Approaches

As discussed in Section 4, multiple unknown segments could be merged to form a representative unknown segment. The merging will help reduce the complexity in the unknown-word boundary identification as fewer segments will be checked for the same set of unknown words.

The following variations of merging approach are compared.

- No merging (*none*): No merging process is applied.
- N-character Merging (*N-char*): Allow the maximum of *N* characters per segment.
- Merging all segments (*all*): No limit on number of characters per segment.

We measure the performance of unknown-word detection task by using two metrics. The first is the detection rate (or recall) which is equal to the number of detected unknown words divided by the total number of previously tagged unknown words in the corpus. The second is the averaged detected positions per word. The second metric directly represents the overhead or the complexity to the unknown-word boundary identification process. This is because all detected positions from a single unknown word must be checked by the process. The comparison results are shown in Figure 4. As expected, the approach *none* gives the maximum detection rate of *96.6%*, while the approach *all* yields the lowest detection rate. Another interesting observation is that the approach *2-char* yields comparable detection rate to the ap-

	Unknown-Segment Merging Approach					
	none	2-char	3-char	4-char	5-char	all
Detection Rate (%)	96.6	95.3	93.9	92.9	91.6	85.4
Averaged Detected Positions Per Word	5.9	1.93	1.63	1.46	1.32	0.9

Figure 4: Unknown-word detection results

proach *none*, however, its averaged detected positions per word is about three times lower. Therefore to reduce the complexity during the unknown-word boundary identification process, one might want to consider using the merging approach of *2-char*.

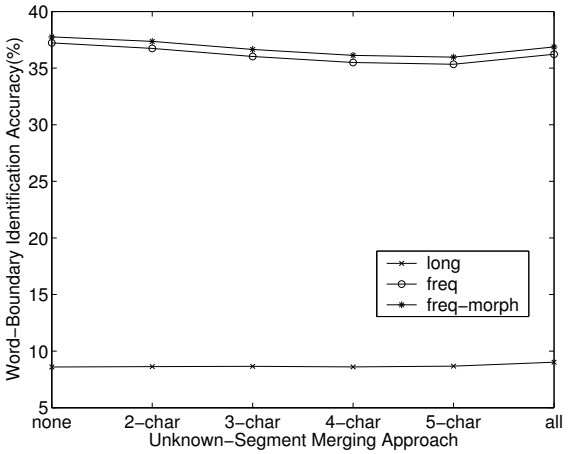


Figure 5: Comparison between different unknown-word boundary detection approaches.

5.2 Evaluation of Unknown-Word Boundary Identification

The unknown-word boundary identification is based on string pattern-matching algorithm. The following variations of string pattern-matching technique are compared.

- Longest matching pattern (*long*): Select the longest-matching unknown-word candidate
- Most-frequent matching pattern (*freq*): Select the most-frequent-matching unknown-word candidate
- Most-frequent matching pattern with morphological analysis (*freq-morph*): Similar the the approach *freq* but with additional morphological analysis to guarantee that the word boundaries are grammatically correct.

The comparison among all variations of string pattern-matching approaches are performed across all unknown-segment merging approach. The results are shown in Figure 5. The performance metric is the word-boundary identification accuracy which is equal to the number of unknown words correctly extracted divided by the total number of tested unknown segments. It can be observed that the selection of different merging approaches does not really effect the accuracy of the unknown-word boundary identification process. But since the approach *none* generates approximately *6* positions per unknown segment on average, it would be more efficient to perform a merging approach which could reduce the number of positions down by at least 3 times.

The plot also shows the comparison among three approaches of string pattern-matching. Figure 6 summarizes the accuracy results of each string pattern-matching approach by taking the average on all different merging approaches. The approach *long* performed poorly with the averaged accuracy of *8.68%*. This is not surprising because selection of the longest matching pattern does not mean that its boundary will be identified correctly. The approaches *freq* and *freq-morph* yield similar accuracy of about *36%*. The *freq-morph* improves the performance of the approach *freq* by less than *1%*. The little improvement is due to the fact that the matching strings are mostly grammatically correct. However, the error is caused by the matching collocations of the unknown-word context. If an unknown word occurs together adjacent to another word very frequently, they will likely be extracted by the algorithm. Our solution to this problem is by providing the users with a user-friendly interface so unknown-word candidates could be easily filtered and corrected.

6 Conclusion

We proposed a framework for collecting Thai unknown words from the Web. Our framework

	Unknown-Word Boundary Identification Approach		
	long	freq	freq-morph
Averaged Accuracy (%)	8.68	36.18	36.79

Figure 6: Unknown-word boundary identification results

is composed of an information agent and an unknown-word analyzer. The task of the information agent is to collect and extract textual data from Web pages of given URLs. The unknown-word analyzer involves two processes: unknown-word detection and unknown-word boundary identification. Due to the non-segmenting characteristic of Thai written language, the unknown-word detection is based on a word-segmentation algorithm with a morphological analysis. To take advantage of large available text resource from the Web, the unknown-word boundary identification is based on the statistical pattern-matching algorithm.

We evaluate our proposed framework on a collection of Web Pages obtained from a Thai newspaper's Web site. The evaluation is divided to test each of the two processes underlying the framework. For the unknown-word detection, the detection rate is found to be as high as *96%*. In addition, by merging a few characters into a segment, the number of required unknown-word extraction is reduced by at least 3 times, while the detection rate is relatively maintained. For the unknown-word boundary identification, considering the highest frequent occurrence of string pattern is found to be the most effective approach. The identification accuracy was found to be as high as approximately *36%*. The relatively low accuracy is not the major concern since the unknown-word candidates are to be verified and corrected by users before they are actually added to the dictionary.

References

Masayuki Asahara and Yuji Matsumoto. 2004. Japanese unknown word identification by character-based chunking. *Proceedings of the 20th International Conference on Computational Linguistics (COLING-2004)*, 459–465.

R. Boyer and S. Moore. 1977. A fast string searching algorithm. *Communications of the ACM*, 20:762–772.

Jing-Shin Chang and Keh-Yih Su. 1997. An Unsupervised Iterative Method for Chinese New Lexicon Extraction. *International Journal of Computational Linguistics & Chinese Language Processing*, 2(2).

Keh-Jianne Chen and Ming-Hong Bai. 1998. Unknown Word Detection for Chinese by a Corpus-based Learning Method. *Computational Linguistics and Chinese Language Processing*, 3(1):27–44.

Keh-Jianne Chen and Wei-Yun Ma. 2002. Unknown Word Extraction for Chinese Documents. *Proceedings of the 19th International Conference on Computational Linguistics (COLING-2002)*, 169–175.

Asanee Kawtrakul, Chalatip Thumkanon, Yuen Poovorawan, Patcharee Varasrai, and Mukda Suktarachan. 1997. Automatic Thai Unknown Word Recognition. *Proceedings of the Natural Language Processing Pacific Rim Symposium*, 341–348.

Christopher S.G. Khoo and Teck Ee Loh. 2002. Using statistical and contextual information to identify two-and three-character words in Chinese text. *Journal of the American Society for Information Science and Technology*, 53(5):365–377.

Lexitron Version 2.1, Thai-English Dictionary. Source available: http://lexitron.nectec.or.th, February 2006.

Tetsuji Nakagawa, Taku Kudoh and Yuji Matsumoto. 2001. Unknown Word Guessing and Part-of-Speech Tagging Using Support Vector Machines. *Proceedings of the Sixth Natural Language Processing Pacific Rim Symposium (NLPRS 2001)*, 325–331.

Jian-Yun Nie, Marie-Louise Hannan and Wanying Jin. 1995. Unknown Word Detection and Segmentation of Chinese Using Statistical and Heuristic Knowledge. *Communications of COLIPS*, 5(1&2):47–57.

Giorgos S. Orphanos and Dimitris N. Christodoulakis. 1999. POS Disambiguation and Unknown Word Guessing with Decision Trees. *Proceedings of the EACL*, 134–141.

Yuen Poowarawan. 1986. Dictionary-based Thai Syllable Separation. *Proceedings of the Ninth Electronics Engineering Conference*.

Thairath Newspaper. Source available: http://www.thairath.com.

Janine Toole. 2000. Categorizing Unknown Words: Using Decision Trees to Identify Names and Misspellings. *Proceeding of the 6th Applied Natural Language Processing Conference (ANLP 2000)*, 173–179.

Japanese Idiom Recognition: Drawing a Line between Literal and Idiomatic Meanings

Chikara Hashimoto[*] **Satoshi Sato**[†] **Takehito Utsuro**[‡]

[*] Graduate School of Informatics
Kyoto University
Kyoto, 606-8501, Japan

[†] Graduate School of Engineering
Nagoya University
Nagoya, 464-8603, Japan

[‡] Graduate School of Systems and Information Engineering
University of Tsukuba
Tsukuba, 305-8573, Japan

Abstract

Recognizing idioms in a sentence is important to sentence understanding. This paper discusses the lexical knowledge of idioms for idiom recognition. The challenges are that idioms can be ambiguous between literal and idiomatic meanings, and that they can be "transformed" when expressed in a sentence. However, there has been little research on Japanese idiom recognition with its ambiguity and transformations taken into account. We propose a set of lexical knowledge for idiom recognition. We evaluated the knowledge by measuring the performance of an idiom recognizer that exploits the knowledge. As a result, more than 90% of the idioms in a corpus are recognized with 90% accuracy.

1 Introduction

Recognizing idioms in a sentence is important to sentence understanding. Failure of recognizing idioms leads to, for example, mistranslation.

In the case of the translation service of Excite[1], it sometimes mistranslates sentences that contain idioms such as (1a), due to the recognition failure.

(1) a. Kare-wa mondai-no kaiketu-ni
he-TOP problem-GEN solving-DAT
hone-o *o*-tta.
bone-ACC *break*-PAST
"He *made an effort* to solve the problem."

b. "He broke his bone to the resolution of a question."

[1] http://www.excite.co.jp/world/

(1a) contains an idiom, *hone-o oru* (bone-ACC break) "make an effort." (1b) is the mistranslation of (1a), in which the idiom is interpreted literally.

In this paper, we discuss lexical knowledge for idiom recognition. The lexical knowledge is implemented in an idiom dictionary that is used by an idiom recognizer we implemented. Note that the idiom recognition we define includes distinguishing literal and idiomatic meanings.[2] Though there has been a growing interest in MWEs (Sag et al., 2002), few proposals on idiom recognition take into account ambiguity and transformations. Note also that we tentatively define an idiom as a phrase that is semantically non-compositional. A precise characterization of the notion "idiom" is beyond the scope of the paper.[3]

Section 2 defines what makes idiom recognition difficult. Section 3 discusses the classification of Japanese idioms, the requisite lexical knowledge, and implementation of an idiom recognizer. Section 4 evaluates the recognizer that exploits the knowledge. After the overview of related works in Section 5, we conclude the paper in Section 6.

2 Two Challenges of Idiom Recognition

Two factors make idiom recognition difficult: **ambiguity** between literal and idiomatic meanings and "**transformations**" that idioms could undergo.[4] In fact, the mistranslation in (1) is caused by the inability of disambiguation between the two meanings. "Transformation" also causes mistrans-

[2] Some idioms represent two or three idiomatic meanings. But those meanings in an idiom are not distinguished. We concerned only whether a phrase is used as an idiom or not.

[3] For a detailed discussion of what constitutes the notion of (Japanese) idiom, see Miyaji (1982), which details usages of commonly used Japanese idioms.

[4] The term "transformation" in the paper is not relevant to the Chomskyan term in Generative Grammar.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 353–360,
Sydney, July 2006.

lation. Sentences in (2) and (3a) contain an idiom, *yaku-ni tatu* (part-DAT stand) "serve the purpose."

(2) Kare-wa *yaku-ni tatu*.
he-TOP *part*-DAT *stand*
"He *serves the purpose*."

(3) a. Kare-wa *yaku-ni* sugoku *tatu*.
he-TOP *part*-DAT very *stand*
"He really *serves the purpose*."

b. "He stands enormously in part."

Google's translation system[5] mistranslates (3a) as in (3b), which does not make sense,[6] though it successfully translates (2). The only difference between (2) and (3a) is that bunsetu[7] constituents of the idiom are detached from each other.

3 Knowledge for Idiom Recognition

3.1 Classification of Japanese Idioms

Requisite lexical knowledge to recognize an idiom depends on how difficult it is to recognize it. Thus, we first classify idioms based on **recognition difficulty**. The recognition difficulty is determined by the two factors: ambiguity and transformability.

Consequently, we identify three classes (Figure 1).[8] **Class A** is not transformable nor ambiguous. **Class B** is transformable but not ambiguous.[9] **Class C** is transformable and ambiguous. Class A amounts to unambiguous single words, which are easy to recognize, while Class C is the most difficult to recognize. Only Class C needs further classifications, since only Class C needs disambiguation and lexical knowledge for disambiguation depends on its **part-of-speech** (POS) and **internal structure**. The POS of Class C is either verbal or adjectival, as in Figure 1. Internal structure represents constituent words' POS and a dependency between bunsetus. The internal structure

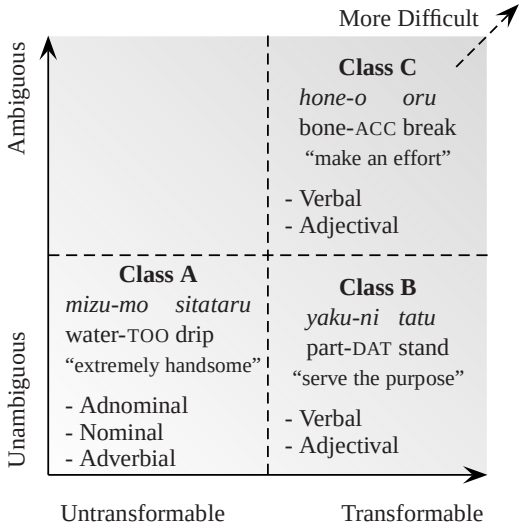


Figure 1: Idiom Classification based on the Recognition Difficulty

of *hone-o oru* (bone-ACC bone), for instance, is "(Noun/Particle Verb)," abbreviated as "(N/P V)."

Then, let us give a full account of the further classification of Class C. We exploit grammatical differences between literal and idiomatic usages for disambiguation. We will call the knowledge of the differences the **disambiguation knowledge**. For instance, a phrase, *hone-o oru*, does not allow passivization when used as an idiom, though it does when used literally. Thus, (4), in which the phrase is passivized, cannot be an idiom.

(4) *hone*-ga *o*-rareru
bone-NOM *break*-PASS
"A bone is broken."

In this case, passivizability can be used as a disambiguation knowledge. Also, detachability of the two bunsetu constituents can serve for disambiguating the idiom; they cannot be separated. In general, usages applicable to idioms are also applicable to literal phrases, but the reverse is not always true (Figure 2). Then, finding the disambiguation knowledge amounts to finding usages applicable to only literal phrases.

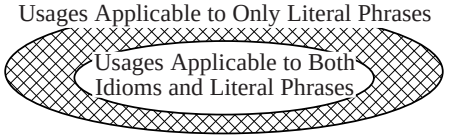


Figure 2: Difference of Applicable Usages

Naturally, the disambiguation knowledge for an idiom depends on its POS and internal structure.

[5] http://www.google.co.jp/language_tools

[6] In fact, the idiom has no literal interpretation.

[7] A bunsetu is a syntactic unit in Japanese, consisting of one independent word and more than zero ancillary words. The sentence in (3a) consists of four bunsetu constituents.

[8] The blank space at the upper left in the figure implies that there is no idiom that does not undergo any transformation and yet is ambiguous. Actually, we have not come up with such an example that should fill in the blank space.

[9] Anonymous reviewers pointed out that Class A and B could also be ambiguous. In fact, one can devise a context that makes the literal interpretation of those Classes possible. However, virtually no phrase of Class A or B is interpreted literally in real texts, and we think our generalization safely captures the reality of idioms.

As for **POS**, disambiguation of verbal idioms can be performed by the knowledge of passivizability, while that of adjectival idioms cannot. Regarding **internal structure**, detachability should be annotated on every boundary of bunsetus. Thus, the number of annotations of detachability depends on the number of bunsetus of an idiom.

There is no need for further classification of Class A and B, since lexical knowledge for them is invariable. The next section mentions their invariableness. After all, Japanese idioms are classified as in Figure 3. The whole picture of the subclasses of Class C remains to be seen.

3.2 Knowledge for Each Class

What lexical knowledge is needed for each class?

Class A needs only a string information; idioms of the class amount to unambiguous single words.

A string information is undoubtedly invariable across all kinds of POS and internal structure.

Class B requires not only a string but also knowledge that normalizes transformations idioms could undergo, such as passivization and detachment of bunsetus. We identify three types of transformations that are relevant to idioms: **1)** Detachment of Bunsetu Constituents, **2)** Predicate's Change, and **3)** Particle's Change. Predicate's change includes inflection, attachment of a negative morpheme, a passive morpheme or modal verbs, and so on. Particle's change represents attachment of topic or restrictive particles. (5b) is an example of predicate's change from (5a) by adding a negative morpheme to a verb. (5c) is an example of particle's change from (5a) by adding a topic particle to the preexsistent particle of an idiom.

(5) a. Kare-wa *yaku-ni tatu*.
he-TOP *part*-DAT *stand*
"He *serves the purpose*."

b. Kare-wa *yaku-ni tat*-**anai**.
he-TOP *part*-DAT *stand*-**NEG**
"He does not *serve the purpose*."

c. Kare-wa *yaku-ni*-**wa** *tatu*.
he-TOP *part*-DAT-**TOP** *stand*
"He *serves the purpose*."

To normalize the transformations, we utilize a dependency relation between constituent words, and we call it the **dependency knowledge**. This amounts to checking the presence of all the constituent words of an idiom. Note that we ignore, among constituent words, endings of a predicate and case particles, *ga* (NOM) and *o* (ACC), since they could change their forms or disappear.

The dependency knowledge is also invariable across all kinds of POS and internal structure.

Class C requires the disambiguation knowledge, as well as all the knowledge for Class B.

As a result, all the requisite knowledge for idiom recognition is summarized as in Table 1.

	String	Dependency	Disambiguation
Class A	✔		
Class B	✔	✔	
Class C	✔	✔	✔

Table 1: Requisite Knowledge for each Class

As discussed in §3.1, the disambiguation knowledge for an idiom depends on which subclass it belongs to. A comprehensive idiom recognizer calls for all the disambiguation knowledge for all the subclasses, but we have not figured out all of them. Then, we decided to blaze a trail to discover the disambiguation knowledge by investigating the most commonly used idioms.

3.3 Disambiguation Knowledge for the Verbal (N/P V) Idioms

What type of idiom is used most commonly? The answer is the **verbal (N/P V)** type like *hone-o oru* (bone-ACC break); it is the most abundant in terms of both type and token. Actually, 1,834 out of 4,581 idioms (≃40%) in Kindaichi and Ikeda (1989), which is a Japanese dictionary with more than 100,000 words, are this type.[10] Also, 167,268 out of 220,684 idiom tokens in Mainichi newspaper of 10 years ('91–'00) (≃76%) are this type.[11]

Then we discuss what can be used to disambiguate the verbal (N/P V) type. First, we examined literature of linguistics (Miyaji, 1982; Morita, 1985; Ishida, 2000) that observed characteristics of Japanese idioms. Then, among the characteristics, we picked those that could help with the disambiguation of the type. (6) summarizes them.

[10] Counting was performed automatically by means of the morphological analyzer ChaSen (Matsumoto et al., 2000) with no human intervention. Note that Kindaichi and Ikeda (1989) consists of 4,802 idioms, but 221 of them were ignored since they contained unknown words for ChaSen.

[11] We counted idiom tokens by string matching with inflection taken into account. And we referred to Kindaichi and Ikeda (1989) for a comprehensive idiom list. Note that counting was performed totally automatically.

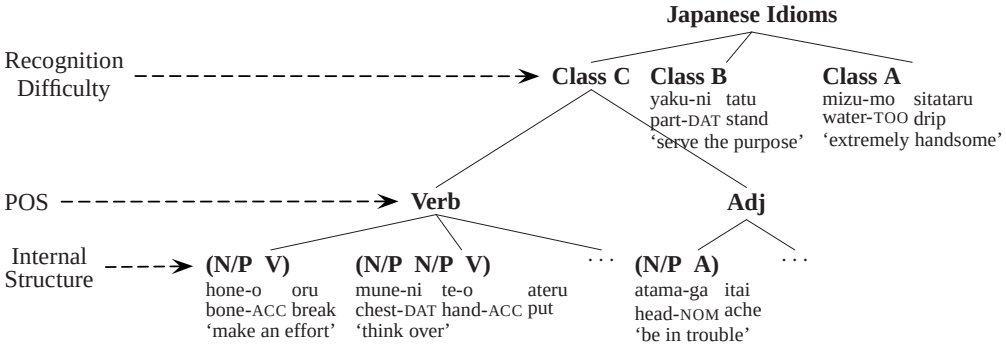


Figure 3: Classification of Japanese Idioms for the Recognition Task

(6) **Disambiguation Knowledge for the Verbal (N/P V) Idioms**

a. Adnominal Modification Constraints

I. Relative Clause Prohibition
II. Genitive Phrase Prohibition
III. Adnominal Word Prohibition

b. Topic/Restrictive Particle Constraints

c. Voice Constraints

I. Passivization Prohibition
II. Causativization Prohibition

d. Modality Constraints

I. Negation Prohibition
II. Volitional Modality Prohibition[12]

e. Detachment Constraint

f. Selectional Restriction

For example, the idiom, *hone-o oru*, does not allow adnominal modification by a genitive phrase. Thus, (7) can be interpreted only literally.

(7) **kare-no** *hone-o oru*
he-GEN *bone-*ACC *break*
"(Someone) breaks **his** bone."

That is, the Genitive Phrase Prohibition, (6aII), is in effect for the idiom. Likewise, the idiom does not allow its case particle *o* (ACC) to be substituted with restrictive particles such as *dake* (only). Thus, (8) represents only a literal meaning.

(8) *hone-***dake** *oru*
*bone-***ONLY** *break*
"(Someone) breaks **only** some bones."

This means the Restrictive Particle Constraint, (6b), is also in effect. Also, (4) shows that the Passivization Prohibition, (6cI), is in effect, too.

Note that the constraints in (6) are not always in effect for an idiom. For instance, the Causativization Prohibition, (6cII), is invalid for the idiom, *hone-o oru*. In fact, (9a) can be interpreted both literally and idiomatically.

(9) a. kare-ni *hone-o or-***aseru**
he-DAT *bone-*ACC *break-***CAUS**

b. "(Someone) makes him break a bone."

c. "(Someone) makes him *make an effort*."

3.4 Implementation

We implemented an idiom dictionary based on the outcome above and a recognizer that exploits the dictionary. This section illustrates how they work, and we focus on Class B and C hereafter.

The idiom recognizer looks up **dependency patterns** in the dictionary that match a part of the dependency structure of a sentence (Figure 4). A dependency pattern is equipped with all the requisite knowledge for idiom recognition. Rough sketch of the recognition algorithm is as follows:

1. Analyze the morphology and dependency structures of an input sentence.
2. Look up dependency patterns in the dictionary that match a part of the dependency structure of the input sentence.
3. Mark constituents of an idiom in the sentence if any.[13] Constituents that are marked are constituent words and bunsetu constituents that include one of those constituent words.

[12] "Volitional Modality" represents those verbal expressions of order, request, permission, prohibition, and volition.

[13] As a constituent marker, we use an ID that is assigned to each idiom in the dictionary.

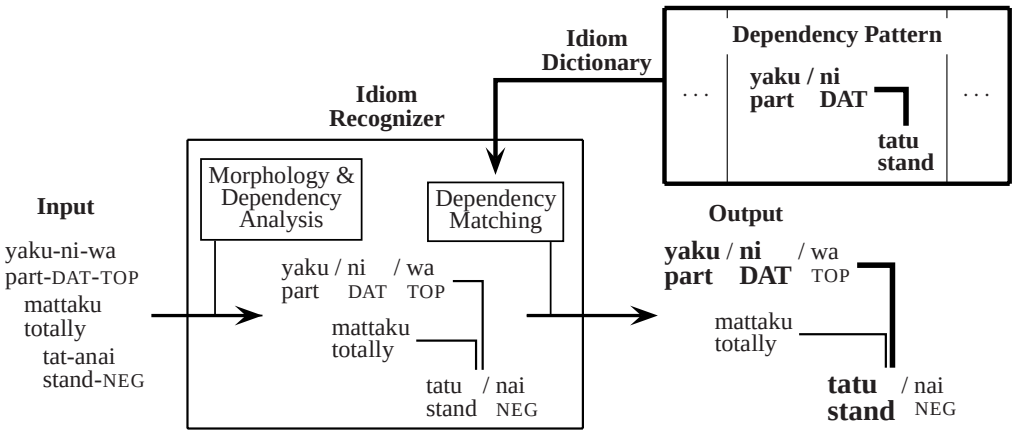


Figure 4: Internal Working of the Idiom Recognizer

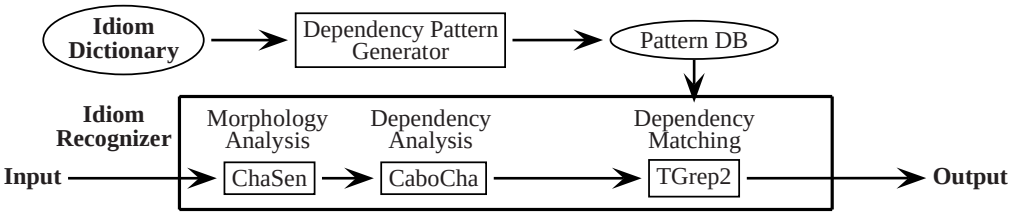


Figure 5: Organization of the System

As in Figure 5, we use ChaSen as a morphology analyzer and CaboCha (Kudo and Matsumoto, 2002) as a dependency analyzer. Dependency matching is performed by TGrep2 (Rohde, 2005), which finds syntactic patterns in a sentence or treebank. The dependency pattern is usually getting complicated since it is tailored to the specification of TGrep2. Thus, we developed the Dependency Pattern Generator that compiles the pattern database from a human-readable idiom dictionary.

Only the difference in treatments of Class B and C lies in their dependency patterns. The dependency pattern of Class B consists of only its dependency knowledge, while that of Class C consists of not only its dependency knowledge but also its disambiguation knowledge (Figure 6).

The idiom dictionary consists of 100 idioms, which are all verbal (N/P V) and belong to either Class B or C. Among the knowledge in (6), the Selectional Restriction has not been implemented yet. The 100 idioms are those that are used most frequently. To be precise, 50 idioms in Kindaichi and Ikeda (1989) and 50 in Miyaji (1982) were extracted by the following steps:[14]

1. From Miyaji (1982), 50 idioms that were used most frequently in Mainichi newspaper of 10 years ('91–'00) were extracted.
2. From Kindaichi and Ikeda (1989), 50 idioms that were used most frequently in the newspaper of 10 years but were not included in the 50 idioms from Miyaji (1982) were extracted.

As a result, 66 out of the 100 idioms were Class B, and the other 34 idioms were Class C.[15]

4 Evaluation

4.1 Experiment Condition

We conducted an experiment to see the effectiveness of the lexical knowledge we proposed.

As an **evaluation corpus**, we collected 300 example sentences of the 100 idioms from Mainichi newspaper of '95: three sentences for each idiom. Then we added another nine sentences for three idioms that are orthographic variants of one of the 100 idioms. Among the three idioms, one belonged to Class B and the other two belonged to Class C. Thus, 67 out of the 103 idioms were Class B and the other 36 were Class C. After all, 309

[14]We counted idiom tokens by string matching with inflection taken into account. Note that counting was performed automatically without human intervention.

[15]We found that the most frequently used 100 idioms in Kindaichi and Ikeda (1989) cover as many as 53.49% of all tokens in Mainichi newspaper of 10 years. This implies that our dictionary accounts for approximately half of all idiom tokens in a corpus.

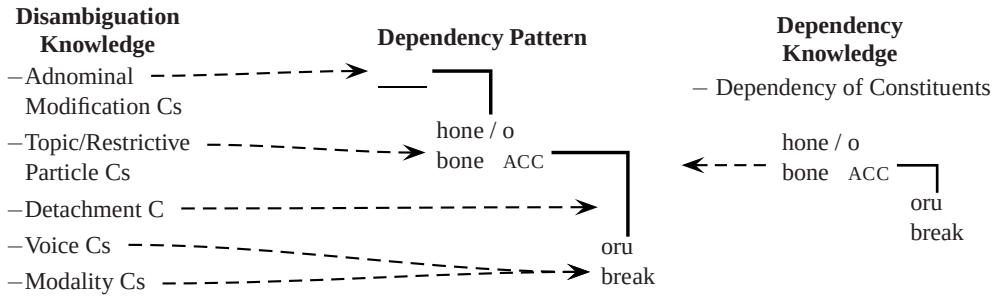


Figure 6: Dependency Pattern of Class C

sentences were prepared. Table 2 shows the breakdown of them. "Positive" indicates sentences including a true idiom, while "Negative" indicates those including a literal-usage "idiom."

	Class B	Class C	Total
Positive	200	66	266
Negative	1	42	43
Total	201	108	309

Table 2: Breakdown of the Evaluation Corpus

A **baseline system** was prepared to see the effect of the disambiguation knowledge. The baseline system was the same as the recognizer except that it exploited no disambiguation knowledge.

4.2 Result

The result is shown in Table 3. The left side shows the performances of the recognizer, while the right side shows that of the baseline. Differences of performances between the two systems are marked with **bold**. Recall, Precision, and F-Measure, are calculated using the following equations.

$$Recall = \frac{|Correct\ Outputs|}{|Positive|}$$

$$Precision = \frac{|Correct\ Outputs|}{|All\ Outputs|}$$

$$F\text{-}Measure = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

As a result, more than 90% of the idioms can be recognized with 90% accuracy. Note that the recognizer made fewer errors due to the employment of the disambiguation knowledge.

The result shows the high performances. However, there turns out to be a long way to go to solve the most difficult problem of idiom recognition: drawing a line between literal and idiomatic meanings. In fact, the precision of recognizing idioms of Class C remains less than 70% as in Table 3. Besides, the recognizer successfully rejected only 15 out of 42 negative sentences. That is, its success rate of rejecting negative ones is only 35.71%

4.3 Discussion of the Disambiguation Knowledge

First of all, positive sentences, i.e., sentences containing true idioms, are in the blank region of Figure 2, while negative ones, i.e., those containing literal phrases, are in both regions. Accordingly, the disambiguation amounts to **i)** rejecting negative ones in the shaded region, **ii)** rejecting negative ones in the blank region, or **iii)** accepting positive ones in the blank region. i) is relatively easy since there are visible evidences in a sentence that tell us that it is NOT an idiom. However, ii) and iii) are difficult due to the absence of visible evidences. Our method is intended to perform i), and thus has an obvious limitation.

Next, we look cloosely at cases of success or failure of rejecting negative sentences. There were 15 cases where rejection succeeded, which correspond to i). The disambiguation knowledge that contributed to rejection and the number of sentences it rejects are as follows.[16]

1. Genitive Phrase Prohibition (6aII) 6
2. Relative Clause Prohibition (6aI) 5
3. Detachment Constraint (6e) 2
4. Negation Prohibition (6dI) 1

This shows that the Adnominal Modification Constraints, 1. and 2. above, are the most effective.

There were 27 cases where rejection failed. These are classified into two types:

[16]There was one case where rejection succeeded due to the dependency analysis error.

	Class B	Class C	All	Class B	Class C	All
Recall	0.975 ($\frac{195}{200}$)	0.939 ($\frac{62}{66}$)	0.966 ($\frac{257}{266}$)	0.975 ($\frac{195}{200}$)	0.939 ($\frac{62}{66}$)	0.966 ($\frac{257}{266}$)
Precision	1.000 ($\frac{195}{195}$)	**0.697** ($\frac{62}{89}$)	**0.905** ($\frac{257}{284}$)	1.000 ($\frac{195}{195}$)	0.602 ($\frac{62}{103}$)	0.862 ($\frac{257}{298}$)
F-Measure	0.987	**0.800**	**0.935**	0.987	0.734	0.911

Table 3: Performances of the Recognizer (left side) and the Baseline System (right side)

1. Those that could have been rejected by the Selectional Restriction (6f) 5
2. Those that might be beyond the current technology 22

1. and 2. correspond to i) and ii), respectively. We see that the Selectional Restriction would have been as effective as the Adnominal Modification Constraints. A part of a sentence that the knowledge could have rejected is below.

(10) basu-ga *tyuu-ni* *ui*-ta
bus-NOM *midair*-DAT *float*-PAST
"The bus floated in midair."

An idiom, *tyuu-ni uku* (midair-DAT float) "remain to be decided," takes as its argument something that can be decided, i.e., ⟨`1000:abstract`⟩ rather than ⟨`2:concrete`⟩ in the sense of the *Goi-Taikei* ontology (Ikehara et al., 1997). Thus, (10) has no idiomatic sense.

A simplified example of 2. is illustrated in (11).

(11) ase-o nagasi-te huku-o
sweat-ACC shed-and clothes-ACC
kiru-yorimo, hadaka-ga gouriteki-da
wear-rather.than, nudity-NOM rational-DECL
"It makes more sense to be naked than wearing clothes in a sweat."

The phrase *ase-o nagasu* (sweat-ACC shed) could have been an idiom meaning "work hard." It is contextual knowledge that prevented it from being the idiom. Clearly, our technique is unable to handle such a case, which belongs to ii), since no visible evidence is available. Dealing with that might require some sort of machine learning technique that exploits contextual information. Exploring that possibility is one of our future works.

Finally, the 42 negative sentences consist of 15 sentences, which we could disambiguate, 5 sentences, which Selectional Restriction could have disambiguated, and 22, which belong to ii) and are beyond the current technique. Thus, the real challenge lies in 7% ($\frac{22}{309}$) of all idiom occurrences.

4.4 Discussion of the Dependency Knowledge

The dependency knowledge failed in only five cases. Three of them were due to the defect of dealing with case particles' change like omission. The other two cases were due to the noun constituent's incorporation into a compound noun. (12) is a part of such a case.

(12) kaihuku-*kidou-ni* *nori*-hajimeru
recovery-*orbit*-DAT *ride*-begin
"(Economics) *get* back *on* a recovery *track*."

The idiom, *kidou-ni noru* (orbit-DAT ride) "get on track," has a constituent, *kidou*, which is incorporated into a compound noun *kaihuku-kidou* "recovery track." This is unexpected and cannot be handled by the current machinery.

5 Related Work

There has been a growing awareness of Japanese MWE problems (Baldwin and Bond, 2002). However, few attempts have been made to recognize idioms in a sentence with their ambiguity and transformations taken into account. In fact, most of them only create catalogs of Japanese idiom: collecting idioms as many as possible and classifying them based on some general linguistic properties (Tanaka, 1997; Shudo et al., 2004).

A notable exception is Oku (1990); his idiom recognizer takes the ambiguity and transformations into account. However, he only uses the Genitive Phrase Prohibition, the Detachment Constraint, and the Selectional Restriction, which would be too few to disambiguate idioms.[17] As well, his classification does not take the recognition difficulty into account. This makes his idiom dictionary get bloated, since disambiguation knowledge is given to unambiguous idioms, too.

Uchiyama et al. (2005) deals with disambiguating some Japanese verbal compounds. Though verbal compounds are not counted as idioms, their study is in line with this study.

[17] We cannot compare his recognizer with ours numerically since no disambiguation success rate is presented in Oku (1990); only the overall performance is presented.

Our classification of idioms correlates loosely with that of MWEs by Sag et al. (2002). Japanese idioms that we define correspond to *lexicalized phrases*. Among lexicalized phrases, *fixed expressions* are equal to Class A. Class B and C roughly correspond to *semi-fixed* or *syntactically-flexible expressions*. Note that, though the three subtypes of lexicalized phrases are distinguished based on what we call **transformability**, no distinction is made based on the **ambiguity**.[18]

6 Conclusion

Aiming at Japanese idiom recognition with ambiguity and transformations taken into accout, we proposed a set of lexical knowledge for idioms and implemented a recognizer that exploits the knowledge. We maintain that requisite knowledge depends on its transformability and ambiguity; transformable idioms require the dependency knowledge, while ambiguous ones require the disambiguation knowledge as well as the dependency knowledge. As the disambiguation knowledge, we proposed a set of constraints applicable to a phrase when it is used as an idiom. The experiment showed that more than 90% idioms could be recognized with 90% accuracy but the success rate of rejecting negative sentences remained 35.71%. The experiment also revealed that, among the disambiguation knowledge, the Adnominal Modification Constraints and the Selectional Restriction are the most effective.

What remains to be done is two things; one is to reveal all the subclasses of Class C and all the disambiguation knowledge, and the other is to apply a machine learning technique to disambiguating those cases that the current technique is unable to handle, i.e., cases without visible evidence.

In conclusion, there is still a long way to go to draw a perfect line between literal and idiomatic meanings, but we believe we broke new ground in Japanese idiom recognition.

Acknowledgment A special thank goes to Gakushu Kenkyu-sha, who permitted us to use Gakken's Dictionary for our research.

[18]The notion of *decomposability* of Sag et al. (2002) and Nunberg et al. (1994) is independent of **ambiguity**. In fact, ambiguous idioms are either decomposable (*hara-ga kuroi* (belly-NOM black) "black-hearted") or non-decomposable (*hiza-o utu* (knee-ACC hit) "have a brainwave"). Also, unambiguous idioms are either decomposable (*hara-o yomu* (belly-ACC read) "fathom someone's thinking") or non-decomposable (*saba-o yomu* (chub.mackerel-ACC read) "cheat in counting").

References

Timothy Baldwin and Francis Bond. 2002. Multiword Expressions: Some Problems for Japanese NLP. In *Proceedings of the 8th Annual Meeting of the Association of Natural Language Processing, Japan*, pages 379–382, Keihanna, Japan.

Satoru Ikehara, Masahiro Miyazaki, Satoshi Shirai, Akio Yokoo, Hiromi Nakaiwa, Kentaro Ogura, Yoshifumi Ooyama, and Yoshihiko Hayashi. 1997. *Goi-Taikei — A Japanese Lexicon*. Iwanami Shoten.

Priscilla Ishida. 2000. Doushi Kanyouku-ni taisuru Tougoteki Sousa-no Kaisou Kankei (On the Hierarchy of Syntactic Operations Applicable to Verb Idioms). *Nihongo Kagaku (Japanese Linguistics)*, 7:24–43, April.

Haruhiko Kindaichi and Yasaburo Ikeda, editors. 1989. *Gakken Kokugo Daijiten (Gakken's Dictionary)*. Gakushu Kenkyu-sha.

Taku Kudo and Yuji Matsumoto. 2002. Japanese Dependency Analyisis using Cascaded Chunking. In *Proceedings of the 6th Conference on Natural Language Learning (CoNLL-2002)*, pages 63–69.

Yuji Matsumoto, Akira Kitauchi, Tatsuo Yamashita, Yoshitaka Hirano, Hiroshi Matsuda, Kazuma Takaoka, and Masayuki Asahara, 2000. *Morphological Analysis System ChaSen version 2.2.1 Manual*. Nara Institute of Science and Technology, Dec.

Yutaka Miyaji. 1982. *Kanyouku-no Imi-to Youhou (Usage and Semantics of Idioms)*. Meiji Shoin.

Yoshiyuki Morita. 1985. Doushikanyouku (Verb Idioms). *Nihongogaku (Japanese Linguistics)*, 4(1):37–44.

Geoffrey Nunberg, Ivan A. Sag, and Thomas Wasow. 1994. Idioms. *Language*, 70:491–538.

Masahiro Oku. 1990. Nihongo-bun Kaiseki-ni-okeru Jutsugo Soutou-no Kanyouteki Hyougen-no Atsukai (Treatments of Predicative Idiomatic Expressions in Parsing Japanese). *Journal of Information Processing Society of Japan*, 31(12):1727–1734.

Douglas L. T. Rohde, 2005. *TGrep2 User Manual version 1.15*. Massachusetts Institute of Technology. http://tedlab.mit.edu/~dr/Tgrep2/.

Ivan A. Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. 2002. Multiword expressions: A pain in the neck for nlp. In *Computational Linguistics and Intelligent Text Processing: Third International Conference*, pages 1–15.

Kosho Shudo, Toshifumi Tanabe, Masahito Takahashi, and Kenji Yoshimura. 2004. MWEs as Non-propositional Content Indicators. In *the 2nd ACL Workshop on Multiword Expressions: Integrating Processing*, pages 32–39.

Yasuhito Tanaka. 1997. Collecting idioms and their equivalents. In *IPSJ SIGNL 1997-NL-121*.

Kiyoko Uchiyama, Timothy Baldwin, and Shun Ishizaki. 2005. Disambiguating Japanese Compound Verbs. *Computer Speech and Language, Special Issue on Multiword Expressions*, 19, Issue 4:497–512.

Graph Branch Algorithm: An Optimum Tree Search Method for Scored Dependency Graph with Arc Co-occurrence Constraints

Hideki Hirakawa
Toshiba R&D Center
1 Komukai Toshiba-cho, Saiwai-ku,
Kawasaki 210, JAPAN
hideki.hirakawa@toshiba.co.jp

Abstract

Various kinds of scored dependency graphs are proposed as packed shared data structures in combination with optimum dependency tree search algorithms. This paper classifies the scored dependency graphs and discusses the specific features of the "Dependency Forest" (DF) which is the packed shared data structure adopted in the "Preference Dependency Grammar" (PDG), and proposes the "Graph Branch Algorithm" for computing the optimum dependency tree from a DF. This paper also reports the experiment showing the computational amount and behavior of the graph branch algorithm.

1 Introduction

The dependency graph (DG) is a packed shared data structure which consists of the nodes corresponding to the words in a sentence and the arcs showing dependency relations between the nodes. The scored DG has preference scores attached to the arcs and is widely used as a basis of the optimum tree search method. For example, the scored DG is used in Japanese Kakari-uke analysis[1] to represent all possible kakari-uke(dependency) trees(Ozeki, 1994),(Hirakawa, 2001). (McDonald et al., 2005) proposed a dependency analysis method using a scored DG and some maximum spanning tree search algorithms. In this method, scores on arcs are computed from a set of features obtained from the dependency trees based on the optimum parameters for scoring dependency arcs obtained by the discriminative learning method.

There are various kinds of dependency analysis methods based on the scored DGs. This paper classifies these methods based on the types of the DGs and the basic well-formed constraints and explains the features of the DF adopted in PDG(Hirakawa, 2006). This paper proposes the graph branch algorithm which searches the optimum dependency tree from a DF based on the branch and bound (B&B) method(Ibaraki, 1978) and reports the experiment showing the computational amount and behavior of the graph branch algorithm. As shown below, the combination of the DF and the graph branch algorithm enables the treatment of non-projective dependency analysis and optimum solution search satisfying the single valence occupation constraint, which are out of the scope of most of the DP(dynamic programming)-based parsing methods.

2 Optimum Tree Search in a Scored DG

2.1 Basic Framework

Figure 1 shows the basic framework of the optimum dependency tree search in a scored DG. In general, nodes in a DG correspond to words in the sentence and the arcs show some kind of dependency relations between nodes. Each arc has

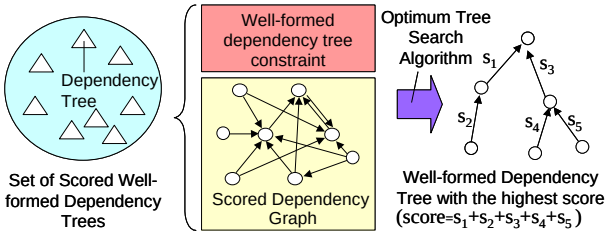


Figure 1: The optimum tree search in a scored DG

[1]Kakari-uke relation, widely adopted in Japanese sentence analysis, is projective dependency relation with a constraint such that the dependent word is located at the left-hand side of its governor word.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 361–368,
Sydney, July 2006.

a preference score representing plausibility of the relation. The well-formed dependency tree constraint is a set of well-formed constraints which should be satisfied by all dependency trees representing sentence interpretations. A DG and a well-formed dependency tree constraint prescribe a set of well-formed dependency trees. The score of a dependency tree is the sum total of arc scores. The optimum tree is a dependency tree with the highest score in the set of dependency trees.

2.2 Dependency Graph

DGs are classified into some classes based on the types of nodes and arcs. This paper assumes three types of nodes, i.e. word-type, WPP-type[2] and concept-type[3]. The types of DGs are called a word DG, a WPP DG and a concept DG, respectively. DGs are also classified into non-labeled and labeled DGs. There are some types of arc labels such as syntactic label (ex. "subject","object") and semantic label (ex. "agent","target"). Various types of DGs are used in existing systems according to these classifications, such as non-label word DG(Lee and Choi, 1997; Eisner, 1996; McDonald et al., 2005)[4], syntactic-label word DG (Maruyama, 1990), semantic-label word DG(Hirakawa, 2001), non-label WPP DG(Ozeki, 1994; Katoh and Ehara, 1989), syntactic-label WPP DG(Wang and Harper, 2004), semantic-label concept DG(Harada and Mizuno, 2001).

2.3 Well-formedness Constraints and Graph Search Algorithms

There can be a variety of well-formedness constraints from very basic and language-independent constraints to specific and language-dependent constraints. This paper focuses on the following four basic and language-independent constraints which may be embedded in data structure and/or the optimum tree search algorithm.

(C1) Coverage constraint: Every input word has a corresponding node in the tree

(C2) Single role constraint(SRC): No two nodes in a dependency tree occupy the same input position

(C3) Projectivity constraint(PJC): No arc crosses another arc[5]

(C4) Single valence occupation constraint(SVOC): No two arcs in a tree occupy the same valence of a predicate

(C1) and (C2), collectively referred to as "covering constraint", are basic constraints adopted by almost all dependency parsers. (C3) is adopted by the majority of dependency parsers which are called projective dependency parsers. A projective dependency parser fails to analyze non-projective sentences. (C4) is a basic constraint for valency but is not adopted by the majority of dependency parsers.

Graph search algorithms, such as the Chu-Liu-Edmonds maximum spanning tree algorithm (Chu and Liu, 1965; Edmonds, 1967), algorithms based on the dynamic programming (DP) principle (Ozeki, 1994; Eisner, 1996) and the algorithm based on the B&B method (Hirakawa, 2001), are used for the optimum tree search in scored DGs. The applicability of these algorithms is closely related to the types of DGs and/or well-formedness constraints. The Chu-Liu-Edmonds algorithm is very fast ($O(n^2)$ for sentence length n), but it works correctly only on word DGs. DP-based algorithms can satisfy (C1)-(C3) and run efficiently, but seems not to satisfy (C4) as shown in 2.4.

(C2)-(C4) can be described as a set of co-occurrence constraints between two arcs in a DG. As described in Section 2.6, the DF can represent (C2)-(C4) and more precise constraints because it can handle co-occurrence constraints between two arbitrary arcs in a DG. The graph branch algorithm described in Section 3 can find the optimum tree from the DF.

2.4 SVOC and DP

(Ozeki and Zhang, 1999) proposed the minimum cost partitioning method (MCPM) which is a partitioning computation based on the recurrence equation where the cost of joining two partitions is the cost of these partitions plus the cost of combining these partitions. MCPM is a generalization of (Ozeki, 1994) and (Katoh and Ehara, 1989) which compute the optimum dependency tree in a scored DG. MCPM is also a generalization of the probabilistic CKY algorithm and the Viterbi algo-

[2]WPP is a pair of a word and a part of speech (POS). The word "time" has WPPs such as "time/n" and "time/v".

[3]One WPP (ex. "time/n") can be categorized into one or more concepts semantically (ex. "time/n/period_time" and "time/n/clock_time").

[4]This does not mean that these algorithms can not handle labeled DGs.

[5]Another condition for projectivity, i.e. "no arc covers top node" is equivalent to the crossing arc constraint if special root node , which is a governor of top node, is introduced at the top (or end) of a sentence.

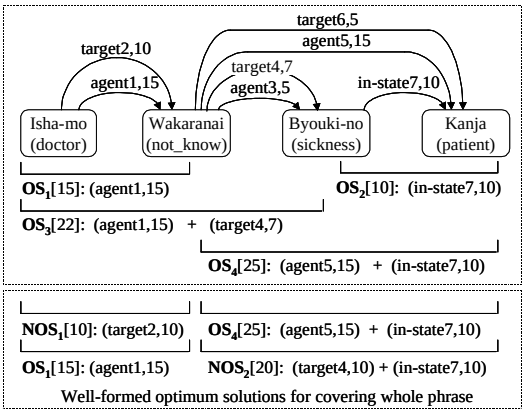


Figure 2: Optimum tree search satisfying SVOC

rithm[6]. The minimum cost partition of the whole sentence is calculated very efficiently by the DP principle. The optimum partitioning obtained by MCPM constitutes a tree covering the whole sentence satisfying the SRC and PJC. However, it is not assured that the SVOC is satisfied by MCPM.

Figure 2 shows a DG for the Japanese phrase "Isha-mo Wakaranai Byouki-no Kanja" encompassing dependency trees corresponding to "a patient suffering from a disease that the doctor doesn't know", "a sick patient who does not know the doctor", and so on. OS_1-OS_4 represent the optimum solutions for the phrases specified by their brackets computed based on MCPM. For example, OS_3 gives an optimum tree with a score of 22 (consisting of $agent1$ and $target4$) for the phrase "Isha-mo Wakaranai Byouki-no". The optimum solution for the whole phrase is either $OS_1 + OS_4$ or $OS_3 + OS_2$ due to MCPM. The former has the highest score $40(= 15 + 25)$ but does not satisfy the SVOC because it has $agent1$ and $agent5$ simultaneously. The optimum solutions satisfying the SVOC are $NOS_1 + OS_4$ and $OS_1 + NOS_2$ shown at the bottom of Figure 2. NOS_1 and NOS_2 are not optimum solutions for their word coverages. This shows that it is not assured that MCPM will obtain the optimum solution satisfying the SVOC.

On the contrary, it is assured that the graph branch algorithm computes the optimum solution(s) satisfying the SVOC because it computes the optimum solution(s) satisfying any co-occurrence constraints in the constraint matrix. It is an open problem whether an algorithm based on the DP framework exists which can handle the SVOC and arbitrary arc co-occurrence constraints.

[6]Specifically, MTCM corresponds to probabilistic CKY and the Viterbi algorithm because it computes both the optimum tree score and its structure.

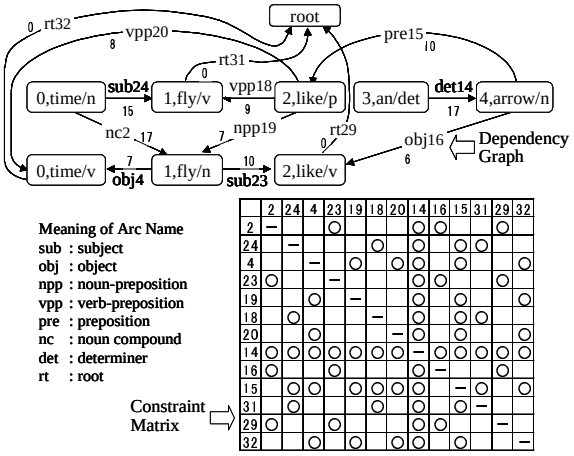


Figure 3: Scored dependency forest

2.5 Semantic Dependency Graph (SDG)

The SDG is a semantic-label word DG designed for Japanese sentence analysis. The optimum tree search algorithm searches for the optimum tree satisfying the well-formed constraints (C1)-(C4) in a SDG(Hirakawa, 2001). This method is lacking in terms of generality in that it cannot handle backward dependency and multiple WPP because it depends on some linguistic features peculiar to Japanese. Therefore, this method is inherently inapplicable to languages like English that require backward dependency and multiple POS analysis.

The DF described below can be seen as the extension of the SDG. Since the DF has none of the language-dependent premises that the SDG has, it is applicable to English and other languages.

2.6 Dependency Forest (DF)

The DF is a packed shared data structure encompassing all possible dependency trees for a sentence adopted in PDG. The DF consists of a dependency graph (DG) and a constraint matrix (CM). Figure 3 shows a DF for the example sentence "Time flies like an arrow." The DG consists of nodes and directed arcs. A node represents a WPP and an arc shows the dependency relation between nodes. An arc has its ID and preference score. CM is a matrix whose rows and columns are a set of arcs in DG and prescribes the co-occurrence constraint between arcs. Only when CM(i,j) is ○, arc_i and arc_j are co-occurrable in one dependency tree.

The DF is generated by using a phrase structure parser in PDG. PDG grammar rule is an extended CFG rule, which defines the mapping between a sequence of constituents (the body of a CFG rule) and a set of arcs (a partial dependency tree).

The generated CM assures that the parse trees in the parse forest and the dependency trees in the DF have mutual correspondence(Hirakawa, 2006). CM can represent (C2)-(C4) in 2.3 and more precise constraints. For example, PDG can generate a DF encompassing non-projective dependency trees by introducing the grammar rules defining non-projective constructions. This is called the controlled non-projectivity in this paper. Treatment of non-projectivity as described in (Kanahe et al., 1998; Nivre and Nilsson, 2005) is an important topic out of the scope of this paper.

3 The Optimum Tree Search in DF

This section shows the graph branch algorithm based on the B&B principle, which searches for the optimum well-formed tree in a DF by applying problem expansions called graph branching.

3.1 Outline of B&B Method

The B&B method(Ibaraki, 1978) is a principle for solving computationally hard problems such as NP-complete problems. The basic strategy is that the original problem is decomposed into easier partial-problems (branching) and the original problem is solved by solving them. Pruning called a bound operation is applied if it turns out that the optimum solution to a partial-problem is inferior to the solution obtained from some other partial-problem (dominance test)[7], or if it turns out that a partial-problem gives no optimum solutions to the original problem (maximum value test). Usually, the B&B algorithm is constructed to minimize the value of the solution. The graph branch algorithm in this paper is constructed to maximize the score of the solution because the best solution is the maximum tree in the DF.

3.2 Graph Branch Algorithm

The graph branch algorithm is obtained by defining the components of the original B&B skeleton algorithm, i.e. the partial-problem, the feasible solution, the lower bound value, the upper bound value, the branch operation, and so on(Ibaraki, 1978). Figure 4 shows the graph branch algorithm which has been extended from the original B&B skeleton algorithm to search for all the optimum trees in a DF. The following sections explain the B&B components of the graph branch algorithm.

[7]The dominance test is not used in the graph branch algorithm.

```
P0 : Initial problem, Pi : Partial problem,
AP: Active partial problem list,
O : Set of incumbent solutions, z : Incumbent value

start: /* S1(initial value setup) */
  AP := {P0}; z = -1; O := {};
  UB = get_ub(P0); /* Upper bound of P0 */

search_top: /* S2(search) */
  if(AP == {}) { goto exit; }
  else{ Pi := select_problem(AP); }
  /* Compute the feasible solution FS and the lower */
  /* bound LB (= the score of FS) for Pi.          */
  (FS,LB) := get_fs(Pi);
  /* If no feasible solution found, terminate the problem. */
  if(FS == no_solution) { goto terminate_problem; }
  /* S3(incumbent value update): If LB is better than z, */
  /* update incumbent solution and incumbent value.      */
  if(LB > z) { z := LB; O := {FS}; }
  /* S5(upper bound test): */
  if(UB < z) { goto terminate_problem; }
  /* Compute inconsistent arc pair list IAPL. */
  IAPL := get_iapl(Pi);
  /* If lower bound (score of feasible solution) is less */
  /* than upper bound, execute graph branch operation.   */
  if(LB < UB) { BACL := IAPL; goto branch; }
  /* Lower bound equals to upper bound => optimum solution */
  elsif(LB == UB) {
    O := {FS} ∪ O; /* Add this FS as incumbent solution */
    /* S8(search more optimum solutions) */
    /* (a) existence of an inconsistent arc pair */
    if(IAPL != {}) { BACL := IAPL; goto branch; }
    /* (b) existence of a rival arc */
    BACL := arcs_with_alternatives(FS);
    if(BACL != {}) { goto branch; }
    else { goto terminate_problem; } }

branch: /* S6(branching operation) */
  /* Generate child partial problems based on BACL */
  ChildProblemList := graph_branch(Pi,BACL);
  AP := AP U ChildProblemList - {Pi}; goto search_top;

terminate_problem:  /* S7(termination of Pi) */
  AP := AP - {Pi}; goto search_top;

exit: /* S9(stop) */
  if(z == -1) { Problem P0 has no solution }
  else { O is a set of the optimum solutions }
```

Figure 4: Graph branch algorithm

(1) Partial-problem

Partial-problem P_i in the graph branch algorithm is a problem searching for all the well-formed optimum trees in a DF DF_i consisting of the dependency graph DG_i and constraint matrix CM_i. P_i consists of the following elements.

(a) Dependency graph DG_i
(b) Constraint matrix CM_i
(c) Feasible solution value LB_i
(d) Upper bound value UB_i
(e) Inconsistent arc pair list $IAPL_i$

The constraint matrix is common to all partial-problems, so one CM is shared by all partial-problems. DG_i is represented by "$rem[..]$" which shows a set of arcs to be removed from the whole dependency graph DG. For example, "$rem[b,d]$" represents a partial dependency graph $[a,c,e]$ in the case $DG = [a,b,c,d,e]$. $IAPL_i$ is a list of inconsistent arc pairs. An inconsistent arc pair is an arc pair which does not satisfy some co-occurrence constraint.

(2) Algorithm for Obtaining Feasible Solution and Lower Bound Value

In the graph branch algorithm, a well-formed dependency tree in the dependency graph DG of the partial-problem P is assigned as the feasible solution FS of P[8]. The score of the feasible solution FS is assigned as the lower bound value LB. The function for computing these values get_fs is called a feasible solution/lower bound value function. The details are not shown due to space limitations, but get_fs is realized by the backtrack-based depth-first search algorithm with the optimization based on the arc scores. get_fs assures that the obtained solution satisfies the covering constraint and the arc co-occurrence constraint. The incumbent value z (the best score so far) is replaced by the LB at $S3$ in Figure 4 if needed.

(3) Algorithm for Obtaining Upper Bound

Given a set of arcs A which is a subset of DG, if the set of dependent nodes[9] of arcs in A satisfies the covering constraint, the arc set A is called the well-covered arc set. The maximum well-covered arc set is defined as a well-covered arc set with the highest score. In general, the maximum well-covered arc set does not satisfy the SRC and does not form a tree. In the graph branch algorithm, the score of the maximum well-covered arc set of a dependency graph G is assigned as the upper bound value UB of the partial-problem P. Upper bound function get_ub calculates UB by scanning the arc lists sorted by the surface position of the dependent nodes of the arcs.

(4) Branch Operation

Figure 5 shows a branch operation called a graph branch operation. Child partial-problems of P are constructed as follows:

(a) Search for an inconsistent arc pair (arc_i, arc_j) in the maximum well-covered arc set of the DG of P.

(b) Create child partial-problems P_i, P_j which have new DGs $DG_i = DG - \{arc_j\}$ and $DG_j = DG - \{arc_i\}$ respectively.

Since a solution to P cannot have both arc_i and arc_j simultaneously due to the co-occurrence constraint, the optimum solution of P is obtained from either/both P_i or/and P_j. The child partial-problem is easier than the parent partial-problem because the size of the DG of the child partial-problem is less than that of its parent.

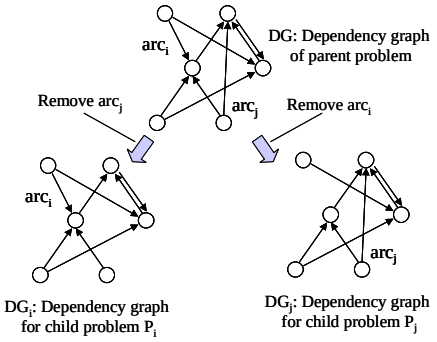


Figure 5: Graph branching

In Figure 4, get_iapl computes the list of inconsistent arc pairs $IAPL$(Inconsistent Arc Pair List) for the maximum well-covered arc set of P_i. Then the graph branch function $graph\_branch$ selects one inconsistent arc pair (arc_i, arc_j) from $IAPL$ for branch operation. The selection criteria for (arc_i, arc_j) affects the efficiency of the algorithm. $graph\_branch$ selects the inconsistent arc pair containing the highest score arc in $BACL$(Branch Arc Candidates List). $graph\_branch$ calculates the upper bound value for a child partial-problem by get_ub and sets it to the child partial-problem.

(5) Selection of Partial-problem

$select\_problem$ employs the best bound search strategy, i.e. it selects the partial-problem which has the maximum bound value among the active partial-problems. It is known that the number of partial-problems decomposed during computation is minimized by this strategy in the case that no dominance tests are applied (Ibaraki, 1978).

(6) Computing All Optimum Solutions

In order to obtain all optimum solutions, partial-problems whose upper bound values are equal to the score of the optimum solution(s) are expanded at $S8$ in Figure 4. In the case that at least one inconsistent arc pair remains in a partial-problem (i.e. $IAPL \neq \{\}$), graph branch is performed based on the inconsistent arc pair. Otherwise, the obtained optimum solution FS is checked if one of the arcs in FS has an equal rival arc by $arcs\_with\_alternatives$ function. The equal rival arc of arc A is an arc whose position and score are equal to those of arc A. If an equal rival arc of an arc in FS exists, a new partial-problem is generated by removing the arc in FS. $S8$ assures that no partial-problem has an upper bound value

[8] A feasible solution may not be optimum but is a possible interpretation of a sentence. Therefore, it can be used as an approximate output when the search process is aborted.

[9] The dependent node of an arc is the node located at the source of the arc.

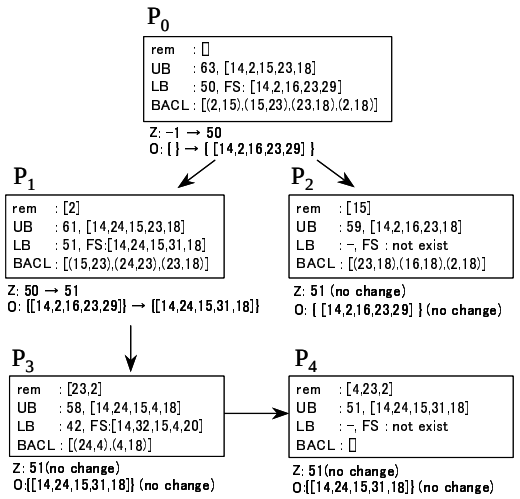


Figure 6: Search diagram

greater than or equal to the score of the optimum solutions when the computation stopped.

4 Example of Optimum Tree Search

This section shows an example of the graph branch algorithm execution using the DF in Fig.3.

4.1 Example of Graph Branch Algorithm

The search process of the B&B method can be shown as a search diagram constructing a partial-problem tree representing the parent-child relation between the partial-problems. Figure 6 is a search diagram for the example DF showing the search process of the graph branch algorithm.

In this figure, box P_i is a partial-problem with its dependency graph rem, upper bound value UB, feasible solution and lower bound value LB and inconsistent arc pair list $IAPL$. Suffix i of P_i indicates the generation order of partial-problems. Updating of global variable z (incumbent value) and O (set of incumbent solutions) is shown under the box. The value of the left-hand side of the arrow is updated to that of right-hand side of the arrow during the partial-problem processing. Details of the behavior of the algorithm in Figure 4 are described below.

In $S1(initialize)$, z, O and AP are set to -1, $\{\}$ and $\{P_0\}$ respectively. The DG of P_0 is that of the example DF. This is represented by $rem = []$. get_ub sets the upper bound value (=63) of P_0 to UB. In practice, this is calculated by obtaining the maximum well-covered arc set of P_0. In $S2(search)$, $select\_problem$ selects P_0 and $get\_fs(P_0)$ is executed. The feasible solution FS and its score LB are calculated to set $FS = [14, 2, 16, 23, 29]$, $LB = 50$ (P_0 in the search diagram). $S3(incumbent\ value\ update)$ updates z and O to new values. Then, $get\_iapl(P_0)$ computes the inconsistent arc pair list $[(2, 15), (15, 23), (23, 18), (2, 18)]$ from the maximum well-covered arc set $[14, 2, 15, 23, 18]$ and set it to $IAPL$. $S5(maximum\ value\ test)$ compares the upper bound value UB and the feasible solution value LB. In this case, $LB < UB$ holds, so $BACL$ is assigned the value of $IAPL$. The next step $S6(branch\ operation)$ executes the $graph\_branch$ function. $graph\_branch$ selects the arc pair with the highest arc score and performs the graph branch operation with the selected arc pair. The following is a $BACL$ shown with the arc names and arc scores.

$$[(nc2[17], pre15[10]), (pre15[10], sub23[10]),$$
$$(sub23[10], vpp18[9]), (nc2[17], vpp18[9])]$$

Scores are shown in [ ]. The arc pair containing the highest arc score is $(2, 15)$ and $(2, 18)$ containing $nc2[17]$. Here, $(2, 15)$ is selected and partial-problems $P_1(rem[2])$ and $P_2(rem[15])$ are generated. P_0 is removed from AP and the new two partial-problems are added to AP resulting in $AP = \{P_1, P_2\}$. Then, based on the best bound search strategy, $S2(search)$ is tried again.

P_1 updates z and O because P_1 obtained a feasible solution better than that in O obtained by P_0. P_2 and P_4 are terminated because they have no feasible solution. P_3 generates a feasible solution but z and O are not updated. This is because the obtained feasible solution is inferior to the incumbent solution in O. The optimum solution(=$\{[14, 24, 15, 31, 18]\}$) is obtained by P_1. The computation from P_2 to P_4 is required to assure that the feasible solution of P_1 is optimum.

5 Experiment

This section describes some experimental results showing the computational amount of the graph branch algorithm.

5.1 Environment and Performance Metric

An existing sentence analysis system[10] (called the oracle system) is used as a generator of the test corpus, the preference knowledge source and the correct answers. Experiment data of 125,320 sentences[11] extracted from English technical docu-

[10] A real-world rule-based machine translation system with a long development history

[11] Sentences ending with a period and parsable by the oracle system.

ments is divided into open data (8605 sentences) and closed data (116,715 sentences). The preference score source, i.e. the WPP frequencies and the dependency relation frequencies are obtained from the closed data. The basic PDG grammar (907 extended CFG rules) is used for generating the DFs for the open data.

The expanded problem number (EPN), a principal computational amount factor of the B&B method, is adopted as the base metric. The following three metrics are used in this experiment.

(a) EPN in total (EPN-T): The number of the expanded problems which are generated in the entire search process.

(b) EPN for the first optimum solution (EPN-F): The number of the expanded problems when the first optimum solution is obtained.

(c) EPN for the last optimum solution (EPN-L): The number of the expanded problems when the last optimum solution is obtained. At this point, all optimum solutions are obtained.

Optimum solution number (OSN) for a problem, i.e. the number of optimum dependency trees in a given DF, gives the lower bound value for all these metrics because one problem generates at most one solution. The minimum value of OSN is 1 because every DF has at least one dependency tree. As the search process proceeds, the algorithm finds the first optimum solution, then the last optimum solution, and finally terminates the process by confirming no better solution is left. Therefore, the three metrics and OSN have the relation OSN $\leq$ EPN-F $\leq$ EPN-L $\leq$ EPN-T. Average values for these are described as Ave:OSN, Ave:EPN-F, Ave:EPN-L and Ave:EPN-T.

5.2 Experimental Results

An evaluation experiment for the open data is performed using a prototype PDG system implemented in Prolog. The sentences containing more than 22 words are neglected due to the limitation of Prolog system resources in the parsing process. 4334 sentences out of the remaining 6882 test sentences are parsable. Unparsable sentences (2584 sentences) are simply neglected in this experiment. The arc precision ratio[12] of the oracle

[12]Correct arc ratio with respect to arcs in the output dependency trees (Hirakawa, 2005).

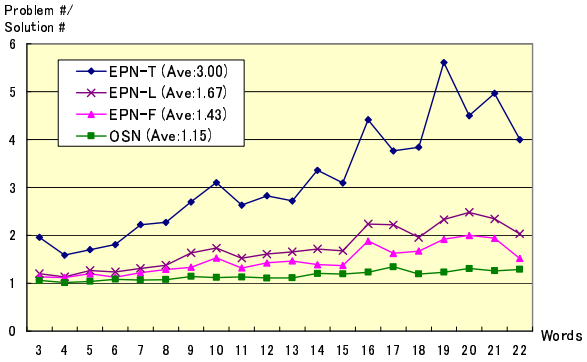


Figure 7: EPN-T, EPN-F EPN-F and OSN

system for 136 sentences in this sentence set is 97.2% with respect to human analysis results.

All optimum trees are computed by the graph branch algorithm described in Section 3.2. Figure 7 shows averages of EPN-T, EPN-L, EPN-F and OSN with respect to sentence length. Overall averages of EPN-T, EPN-L, EPN-F and OSN for the test sentences are 3.0, 1.67, 1.43 and 1.15. The result shows that the average number of problems required is relatively small. The gap between Ave:EPN-T and Ave:EPN-L (3.0-1.67=1.33) is much greater than the gap between Ave:EPN-L and Ave:OSN(1.67-1.15=0.52). This means that the major part of the computation is performed only for checking if the obtained feasible solutions are optimum or not.

According to (Hirakawa, 2001), the experiment on the B&B-based algorithm for the SDG shows the overall averages of AVE:EPN-T, AVE:EPN-F are 2.91, 1.33 and the average CPU time is 305.8ms (on EWS). These values are close to those in the experiment based on the graph branch algorithm. Two experiments show a tendency for the optimum solution to be obtained in the early stage of the search process. The graph branch algorithm is expected to obtain the comparable performance with the SDG search algorithm.

(Hirakawa, 2001) introduced the improved upper bound function g'(P) into the B&B-based algorithm for the SDG and found Ave:EPN-T is reduced from 2.91 to 1.82. The same technique is introduced to the graph branch algorithm and has obtained the reduction of the Ave:EPN-T from 3.00 to 2.68.

The tendency for the optimum solution to be obtained in the early stage of the search process suggests that limiting the number of problems to expand is an effective pruning strategy. Figure 8 shows the ratios of the sentences obtaining the whole problem expansion, the first optimum solu-

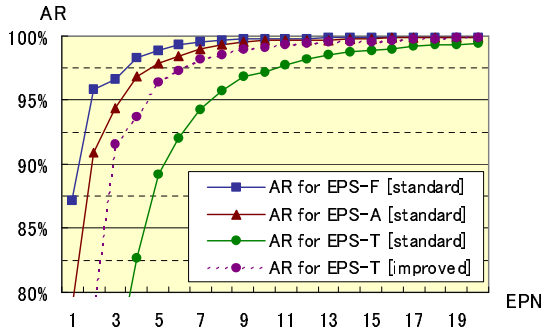


Figure 8: ARs for EPS-F, EPS-A, EPS-T

tion and the last optimum solution to whole sentences with respect to the EPNs. This kind of ratio is called an achievement ratio (AR) in this paper. From Figure 8, the ARs for EPN-T, EPN-L, EPN-F (plotted in solid lines) are 97.1%,99.6%,99.8% respectively at the EPN 10. The dotted line shows the AR for EPN-T of the improved algorithm using g'(P). The use of g'(P) increases the AR for EPN-T from 97.1% to 99.1% at the EPN 10. However, the effect of g'(P) is quite small for EPN-F and EPN-L. This result shows that the pruning strategy based on the EPN is effective and g'(P) works for the reduction of the problems generated in the posterior part of the search processes.

6 Concluding Remarks

This paper has described the graph branch algorithm for obtaining the optimum solution for a DF used in PDG. The well-formedness dependency tree constraints are represented by the constraint matrix of the DF, which has flexible and precise description ability so that controlled non-projectivity is available in PDG framework. The graph branch algorithm assures the search for the optimum trees with arbitrary arc co-occurrence constraints, including the SVOC which has not been treated in DP-based algorithms so far. The experimental result shows the averages of EPN-T, EPN-L and EPN-F for English test sentences are 3.0, 1.67 and 1.43, respectively. The practical code implementation of the graph branch algorithm and its performance evaluation are subjects for future work.

References

Y. J. Chu and T. H. Liu. 1965. On the shortest arborescence of a directed graph. *Science Sinica*, 14.

J. Edmonds. 1967. Optimum branchings. *Journal of Research of the National Bureau of Standards*, 71B:233–240.

J. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of COLING'96*, pages 340–345.

M. Harada and T. Mizuno. 2001. Japanese semantic analysis system sage using edr (in japanese). *Transactions of JSAI*, 16(1):85–93.

H. Hirakawa. 2001. Semantic dependency analysis method for japanese based on optimum tree search algorithm. In *Proceedings of the PACLING2001*.

H. Hirakawa. 2005. Evaluation measures for natural language analyser based on preference dependency grammar. In *Proceedings of the PACLING2005*.

H. Hirakawa. 2006. Preference dependency grammar and its packed shared data structure 'dependency forest' (in japanese). *To appear in Natural Language Processing*, 13(3).

T. Ibaraki. 1978. Branch-and-bounding procedure and state-space representation of combinatorial optimization problems. *Information and Control*, 36,1-27.

S. Kanahe, A. Nasr, and O. Rambow. 1998. Pseudo-projectivity: A polynomially parsable non-projective dependency grammar. In *COLING-ACL'98*, pages 646–52.

N. Katoh and T. Ehara. 1989. A fast algorithm for dependency structure analysis (in japanese). In *Proceedings of 39th Annual Convention of the Information Processing Society*.

S. Lee and K. S. Choi. 1997. Reestimation and best-first parsing algorithm for probablistic dependency grammars. In *Proceedings of the Fifth Workshop on Very Large Corpora*, pages 41–55.

H. Maruyama. 1990. Constraint dependency grammar and its weak generative capacity. *Computer Software*.

R. McDonald, F. Pereira, K. Ribarov, and J. Hajic. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of HLT-EMNLP*, pages 523–530.

J. Nivre and J. Nilsson. 2005. Pseudo-projective dependency parsing. In *ACL-05*, pages 99–106.

K. Ozeki and Y. Zhang. 1999. 最小コスト分割問題としての係り受け解析. In *Proceeding of the Workshop of The Fifth Annual Meeting of The Association for Natural Language Processing*, pages 9–14.

K. Ozeki. 1994. Dependency structure analysis as combinatorial optimization. *Information Sciences*, 78(1-2):77–99.

W. Wang and M. P. Harper. 2004. A statistical constraint dependency grammar (cdg) parser. In *Workshop on Incremental Parsing: Bringing Engineering and Cognition Together (ACL)*, pages 42–49.

Exploring the Potential of Intractable Parsers

Mark Hopkins
Dept. of Computational Linguistics
Saarland University
Saarbrücken, Germany
mhopkins@coli.uni-sb.de

Jonas Kuhn
Dept. of Computational Linguistics
Saarland University
Saarbrücken, Germany
jonask@coli.uni-sb.de

Abstract

We revisit the idea of history-based parsing, and present a history-based parsing framework that strives to be simple, general, and flexible. We also provide a decoder for this probability model that is linear-space, optimal, and anytime. A parser based on this framework, when evaluated on Section 23 of the Penn Treebank, compares favorably with other state-of-the-art approaches, in terms of both accuracy and speed.

1 Introduction

Much of the current research into probabilistic parsing is founded on probabilistic context-free grammars (PCFGs) (Collins, 1996; Charniak, 1997; Collins, 1999; Charniak, 2000; Charniak, 2001; Klein and Manning, 2003). For instance, consider the parse tree in Figure 1. One way to decompose this parse tree is to view it as a sequence of applications of CFG rules. For this particular tree, we could view it as the application of rule "NP → NP PP," followed by rule "NP → DT NN," followed by rule "DT → that," and so forth. Hence instead of analyzing $P(tree)$, we deal with the more modular:

P(NP → NP PP, NP → DT NN, DT → that, NN → money, PP → IN NP, IN → in, NP → DT NN, DT → the, NN → market)

Obviously this joint distribution is just as difficult to assess and compute with as $P(tree)$. However there exist cubic-time dynamic programming algorithms to find the most likely parse if we assume that all CFG rule applications are marginally independent of one another. The problem, of course, with this simplification is that although it is computationally attractive, it is usually too strong of an independence assumption. To mitigate this loss of context, without sacrificing algorithmic tractability, typically researchers annotate the nodes of the parse tree with contextual information. A simple example is the annotation of nodes with their parent labels (Johnson, 1998).

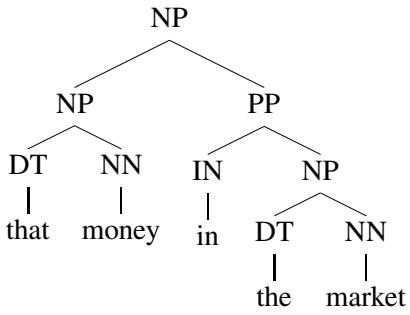


Figure 1: Example parse tree.

The choice of which annotations to use is one of the main features that distinguish parsers based on this approach. Generally, this approach has proven quite effective in producing English phrase-structure grammar parsers that perform well on the Penn Treebank.

One drawback of this approach is its inflexibility. Because we are adding probabilistic context by changing the data itself, we make our data increasingly sparse as we add features. Thus we are constrained from adding too many features, because at some point we will not have enough data to sustain them. We must strike a delicate balance between how much context we want to include versus how much we dare to partition our data set.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 369–376,
Sydney, July 2006.

The major alternative to PCFG-based approaches are so-called *history-based* parsers (Black et al., 1993). These parsers differ from PCFG parsers in that they incorporate context by using a more complex probability model, rather than by modifying the data itself. The tradeoff to using a more powerful probabilistic model is that one can no longer employ dynamic programming to find the most probable parse. Thus one trades assurances of polynomial running time for greater modeling flexibility.

There are two canonical parsers that fall into this category: the decision-tree parser of (Magerman, 1995), and the maximum-entropy parser of (Ratnaparkhi, 1997). Both showed decent results on parsing the Penn Treebank, but in the decade since these papers were published, history-based parsers have been largely ignored by the research community in favor of PCFG-based approaches. There are several reasons why this may be. First is naturally the matter of time efficiency. Magerman reports decent parsing times, but for the purposes of efficiency, must restrict his results to sentences of length 40 or less. Furthermore, his two-phase stack decoder is a bit complicated and is acknowledged to require too much memory to handle certain sentences. Ratnaparkhi is vague about the running time performance of his parser, stating that it is "observed linear-time," but in any event, provides only a heuristic, not a complete algorithm.

Next is the matter of flexibility. The main advantage of abandoning PCFGs is the opportunity to have a more flexible and adaptable probabilistic parsing model. Unfortunately, both Magerman and Ratnaparkhi's models are rather specific and complicated. Ratnaparkhi's, for instance, consists of the interleaved sequence of four different types of tree construction operations. Furthermore, both are inextricably tied to the learning procedure that they employ (decision trees for Magerman, maximum entropy for Ratnaparkhi).

In this work, our goal is to revisit history-based parsers, and provide a general-purpose framework that is (a) simple, (b) fast, (c) space-efficient and (d) easily adaptable to new domains. As a method of evaluation, we use this framework with a very simple set of features to see how well it performs (both in terms of accuracy and running time) on the Penn Treebank. The overarching goal is to develop a history-based hierarchical labeling framework that is viable not only for parsing, but for other application areas that current rely on dynamic programming, like phrase-based machine translation.

2 Preliminaries

For the following discussion, it will be useful to establish some terminology and notational conventions. Typically we will represent variables with capital letters (e.g. X, Y) and sets of variables with bold-faced capital letters (e.g. $\mathbf{X}$, $\mathbf{Y}$). The domain of a variable X will be denoted $dom(X)$, and typically we will use the lower-case correspondent (in this case, x) to denote a value in the domain of X. A *partial assignment* (or simply *assignment*) of a set $\mathbf{X}$ of variables is a function $\mathbf{w}$ that maps a subset $\mathbf{W}$ of the variables of $\mathbf{X}$ to values in their respective domains. We define $dom(\mathbf{w}) = \mathbf{W}$. When $\mathbf{W} = \mathbf{X}$, then we say that $\mathbf{w}$ is a *full assignment* of $\mathbf{X}$. The *trivial assignment* of $\mathbf{X}$ makes no variable assignments.

Let $\mathbf{w}(X)$ denote the value that partial assignment $\mathbf{w}$ assigns to variable X. For value $x \in dom(X)$, let $\mathbf{w}[X = x]$ denote the assignment identical to $\mathbf{w}$ except that $\mathbf{w}[X = x](X) = x$. For a set $\mathbf{Y}$ of variables, let $\mathbf{w}|_{\mathbf{Y}}$ denote the restriction of partial assignment $\mathbf{w}$ to the variables in $dom(\mathbf{w}) \cap \mathbf{Y}$.

3 The Generative Model

The goal of this section is to develop a probabilistic process that generates labeled trees in a manner considerably different from PCFGs. We will use the tree in Figure 2 to motivate our model. In this example, nodes of the tree are labeled with either an A or a B. We can represent this tree using two charts. One chart labels each span with a boolean value, such that a span is labeled $true$ iff it is a constituent in the tree. The other chart labels each span with a label from our labeling scheme (A or B) or with the value $null$ (to represent that the span is unlabeled). We show these charts in Figure 3. Notice that we may want to have more than one labeling scheme. For instance, in the parse tree of Figure 1, there are three different types of labels: word labels, preterminal labels, and nonterminal labels. Thus we would use four 5x5 charts instead of two 3x3 charts to represent that tree.

We will pause here and generalize these concepts. Define a *labeling scheme* as a set of symbols including a special symbol $null$ (this will desig-

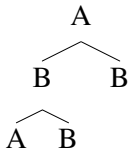


Figure 2: Example labeled tree.

	1	2	3
1	true	true	true
2	-	true	false
3	-	-	true

	1	2	3
1	A	B	A
2	-	B	null
3	-	-	B

Figure 3: Chart representation of the example tree: the left chart tells us which spans are tree constituents, and the right chart tells us the labels of the spans ($null$ means unlabeled).

nate that a given span is unlabeled). For instance, we can define $L^1 = \{null, A, B\}$ to be a labeling scheme for the example tree.

Let $\mathcal{L} = \{L^1, L^2, ...L^m\}$ be a set of labeling schemes. Define a *model variable* of $\mathcal{L}$ as a symbol of the form S_{ij} or L^k_{ij}, for positive integers i, j, k, such that $i \leq j$ and $k \leq m$. Model variables of the form S_{ij} indicate whether span (i, j) is a tree constituent, hence the *domain* of S_{ij} is $\{true, false\}$. Such variables correspond to entries in the left chart of Figure 3. Model variables of the form L^k_{ij} indicate which label from scheme L^k is assigned to span (i, j), hence the *domain* of model variable L^k_{ij} is L^k. Such variables correspond to entries in the right chart of Figure 3. Here we have only one labeling scheme.

Let $\mathbf{V}_\mathcal{L}$ be the (countably infinite) set of model variables of $\mathcal{L}$. Usually we are interested in trees over a given sentence of finite length n. Let $\mathbf{V}^n_\mathcal{L}$ denote the finite subset of $\mathbf{V}_\mathcal{L}$ that includes precisely the model variables of the form S_{ij} or L^k_{ij}, where $j \leq n$.

Basically then, our model consists of two types of decisions: (1) whether a span should be labeled, and (2) if so, what label(s) the span should have. Let us proceed with our example. To generate the tree of Figure 2, the first decision we need to make is how many leaves it will have (or equivalently, how large our tables will be). We assume that we have a probability distribution P_N over the set of positive integers. For our example tree, we draw the value 3, with probability $P_N(3)$.

Now that we know our tree will have three leaves, we can now decide which spans will be constituents and what labels they will have. In other words, we assign values to the variables in $\mathbf{V}^3_\mathcal{L}$. First we need to choose the order in which we will make these assignments. For our example, we will assign model variables in the following order: S_{11}, L^1_{11}, S_{22}, L^1_{22}, S_{33}, L^1_{33}, S_{12}, L^1_{12}, S_{23}, L^1_{23}, S_{13}, L^1_{13}. A detailed look at this assignment process should help clarify the details of the model.

Assigning S_{11}: The first model variable in our order is S_{11}. In other words, we need to decide whether the span $(1, 1)$ should be a constituent. We could let this decision be probabilistically determined, but recall that we are trying to generate a well-formed tree, thus the leaves and the root should always be considered constituents. To handle situations when we would like to make deterministic variable assignments, we supply an auxilliary function $\mathcal{A}$ that tells us (given a model variable X and the history of decisions made so far) whether X should be automatically determined, and if so, what value it should be assigned. In our running example, we ask $\mathcal{A}$ whether S_{11} should be automatically determined, given the previous assignments made (so far only the value chosen for n, which was 3). The so-called *auto-assignment function* $\mathcal{A}$ responds (since S_{11} is a leaf span) that S_{11} should be automatically assigned the value $true$, making span $(1, 1)$ a constituent.

Assigning L^1_{11}: Next we want to assign a label to the first leaf of our tree. There is no compelling reason to deterministically assign this label. Therefore, the auto-assignment function $\mathcal{A}$ declines to assign a value to L^1_{11}, and we proceed to assign its value probabilistically. For this task, we would like a probability distribution over the labels of labeling scheme $L^1 = \{null, A, B\}$, conditioned on the decision history so far. The difficulty is that it is clearly impractical to learn conditional distributions over every conceivable history of variable assignments. So first we distill the important features from an assignment history. For instance, one such feature (though possibly not a good one) could be whether an odd or an even number of nodes have so far been labeled with an A. Our conditional probability distribution is conditioned on the values of these features, instead of the entire assignment history. Consider specifically model variable L^1_{11}. We compute its features (an even number of nodes – zero – have so far been labeled with an A), and then we use these feature values to access the relevant prob-

ability distribution over $\{null, A, B\}$. Drawing from this conditional distribution, we probabilistically assign the value A to variable L^1_{11}.

Assigning S_{22}, L^1_{22}, S_{33}, L^1_{33}: We proceed in this way to assign values to S_{22}, L^1_{22}, S_{33}, L^1_{33} (the S-variables deterministically, and the L^1-variables probabilistically).

Assigning S_{12}: Next comes model variable S_{12}. Here, there is no reason to deterministically dictate whether span $(1, 2)$ is a constituent or not. Both should be considered options. Hence we treat this situation the same as for the L^1 variables. First we extract the relevant features from the assignment history. We then use these features to access the correct probability distribution over the domain of S_{12} (namely $\{true, false\}$). Drawing from this conditional distribution, we probabilistically assign the value $true$ to S_{12}, making span $(1, 2)$ a constituent in our tree.

Assigning L^1_{12}: We proceed to probabilistically assign the value B to L^1_{12}, in the same manner as we did with the other L^1 model variables.

Assigning S_{23}: Now we must determine whether span $(2, 3)$ is a constituent. We could again probabilistically assign a value to S_{23} as we did for S_{12}, but this could result in a hierarchical structure in which both spans $(1, 2)$ and $(2, 3)$ are constituents, which is not a tree. For trees, we cannot allow two model variables S_{ij} and S_{kl} to both be assigned $true$ if they *properly overlap*, i.e. their spans overlap and one is not a subspan of the other. Fortunately we have already established auto-assignment function $\mathcal{A}$, and so we simply need to ensure that it automatically assigns the value $false$ to model variable S_{kl} if a properly overlapping model variable S_{ij} has previously been assigned the value $true$.

Assigning L^1_{23}, S_{13}, L^1_{13}: In this manner, we can complete our variable assignments: L^1_{23} is automatically determined (since span $(2, 3)$ is not a constituent, it should not get a label), as is S_{13} (to ensure a rooted tree), while the label of the root is probabilistically assigned.

We can summarize this generative process as a general modeling tool. Define a *hierarchical labeling process* (HLP) as a 5-tuple $\langle \mathcal{L}, <, \mathcal{A}, \mathcal{F}, \mathcal{P} \rangle$ where:

- $\mathcal{L} = \{L^1, L^2, ..., L^m\}$ is a finite set of *labeling schemes*.
- $<$ is a *model order*, defined as a total ordering of the model variables $\mathbf{V}_\mathcal{L}$ such that for all i, j, k: $S_{ij} < L^k_{ij}$ (i.e. we decide whether a span is a constituent before attempting to label it).
- $\mathcal{A}$ is an *auto-assignment function*. Specifically $\mathcal{A}$ takes three arguments: a model variable Y of $\mathbf{V}_\mathcal{L}$, a partial assignment $\mathbf{x}$ of $\mathbf{V}_\mathcal{L}$, and integer n. The function $\mathcal{A}$ maps this 3-tuple to $false$ if the variable Y should not be automatically assigned a value based on the current history, or the pair $\langle true, y \rangle$, where y is the value in the domain of Y that should be automatically assigned to Y.
- $\mathcal{F} = \{\mathcal{F}^S, \mathcal{F}^1, \mathcal{F}^2, ..., \mathcal{F}^m\}$ is a set of *feature functions*. Specifically, $\mathcal{F}^k$ (resp., $\mathcal{F}^S$) takes four arguments: a partial assignment $\mathbf{x}$ of $\mathbf{V}_\mathcal{L}$, and integers i , j , n such that $1 \leq i \leq j \leq n$. It maps this 4-tuple to a full assignment $\mathbf{f}^k$ (resp., $\mathbf{f}^S$) of some finite set $\mathbf{F}^k$ (resp., $\mathbf{F}^S$) of feature variables.
- $\mathcal{P} = \{P_N, P_S, P_1, P_2, ..., P_m\}$ is a set of probability distributions. P_N is a marginal probability distribution over the set of positive integers, whereas $\{P_S, P_1, P_2, ..., P_m\}$ are conditional probability distributions. Specifically, P_k (respectively, P_S) is a function that takes as its argument a full assignment $\mathbf{f}^k$ (resp., $\mathbf{f}^S$) of feature set $\mathbf{F}^k$ (resp.,

HLPGEN(HLP $\mathcal{H} = \langle \mathcal{L}, <, \mathcal{A}, \mathcal{F}, \mathcal{P} \rangle$):

1. Choose a positive integer n from distribution P_N. Let $\mathbf{x}$ be the trivial assignment of $\mathbf{V}_\mathcal{L}$.
2. In the order defined by $<$, compute step 3 for each model variable Y of $\mathbf{V}^n_\mathcal{L}$.
3. If $\mathcal{A}(Y, \mathbf{x}, n) = \langle true, y \rangle$ for some y in the domain of model variable Y, then let $\mathbf{x} = \mathbf{x}[Y = y]$. Otherwise assign a value to Y from its domain:
   (a) If $Y = S_{ij}$, then let $\mathbf{x} = \mathbf{x}[S_{ij} = s_{ij}]$, where s_{ij} is a value drawn from distribution $P_S(s|\mathcal{F}^S(\mathbf{x}, i, j, n))$.
   (b) If $Y = L^k_{ij}$, then let $\mathbf{x} = \mathbf{x}[L^k_{ij} = l^k_{ij}]$, where l^k_{ij} is a value drawn from distribution $P_k(l^k|\mathcal{F}^k(\mathbf{x}, i, j, n))$.
4. Return $\langle n, \mathbf{x} \rangle$.

Figure 4: Pseudocode for the generative process.

$\mathcal{A}$(variable Y, assignment $\mathbf{x}$, int n):

1. If $Y = S_{ij}$, and there exists a properly overlapping model variable S_{kl} such that $\mathbf{x}(S_{kl}) = true$, then return $\langle true, false \rangle$.
2. If $Y = S_{ii}$ or $Y = S_{1n}$, then return $\langle true, true \rangle$.
3. If $Y = L^k_{ij}$, and $\mathbf{x}(S_{ij}) = false$, then return $\langle true, null \rangle$.
4. Else return $false$.

Figure 5: An example auto-assignment function.

$\mathbf{F}^S$). It maps this to a probability distribution over $dom(L^k)$ (resp., $\{true, false\}$).

An HLP probabilistically generates an assignment of its model variables using the generative process shown in Figure 4. Taking an HLP $\mathcal{H} = \langle \mathcal{L}, <, \mathcal{A}, \mathcal{F}, \mathcal{P} \rangle$ as input, HLPGEN outputs an integer n, and an *$\mathcal{H}$-labeling* $\mathbf{x}$ of length n, defined as a full assignment of $\mathbf{V}^n_{\mathcal{L}}$.

Given the auto-assignment function in Figure 5, every $\mathcal{H}$-labeling generated by HLPGEN can be viewed as a labeled tree using the interpretation: span (i, j) is a constituent iff $S_{ij} = true$; span (i, j) has label $l^k \in dom(L^k)$ iff $L^k_{ij} = l^k$.

4 Learning

The generative story from the previous section allows us to express the probability of a labeled tree as $P(n, \mathbf{x})$, where $\mathbf{x}$ is an $\mathcal{H}$-labeling of length n. For model variable X, define $\mathbf{V}^<_{\mathcal{L}}(X)$ as the subset of $\mathbf{V}_{\mathcal{L}}$ appearing before X in model order $<$. With the help of this terminology, we can decompose $P(n, \mathbf{x})$ into the following product:

$$P_0(n) \cdot \prod_{S_{ij} \in \mathbf{Y}} P_S(\mathbf{x}(S_{ij}) | \mathbf{f}^S_{ij}) \cdot \prod_{L^k_{ij} \in \mathbf{Y}} P_k(\mathbf{x}(L^k_{ij}) | \mathbf{f}^k_{ij})$$

where $\mathbf{f}^S_{ij} = \mathcal{F}^S(\mathbf{x}|_{\mathbf{V}^<_{\mathcal{L}}(S_{ij})}, i, j, n)$ and $\mathbf{f}^k_{ij} = \mathcal{F}^k(\mathbf{x}|_{\mathbf{V}^<_{\mathcal{L}}(L^k_{ij})}, i, j, n)$ and $\mathbf{Y}$ is the subset of $\mathbf{V}^n_{\mathcal{L}}$ that was not automatically assigned by HLPGEN.

Usually in parsing, we are interested in computing the most likely tree given a specific sentence. In our framework, this generalizes to computing: $argmax_{\mathbf{x}} P(\mathbf{x}|n, \mathbf{w})$, where $\mathbf{w}$ is a subassignment of an $\mathcal{H}$-labeling $\mathbf{x}$ of length n. In natural language parsing, $\mathbf{w}$ could specify the constituency and word labels of the leaf-level spans. This would be equivalent to asking: given a sentence, what is its most likely parse?

Let $\mathbf{W} = dom(\mathbf{w})$ and suppose that we choose a model order $<$ such that for every pair of model variables $W \in \mathbf{W}, X \in V_{\mathcal{L}} \backslash \mathbf{W}$, either $W < X$ or W is always auto-assigned. Then $P(\mathbf{x}|n, \mathbf{w})$ can be expressed as:

$$\prod_{S_{ij} \in \mathbf{Y} \backslash \mathbf{W}} P_S(\mathbf{x}(S_{ij}) | \mathbf{f}^S_{ij}) \cdot \prod_{L^k_{ij} \in \mathbf{Y} \backslash \mathbf{W}} P_k(\mathbf{x}(L^k_{ij}) | \mathbf{f}^k_{ij})$$

Hence the distributions we need to learn are probability distributions $P_S(s_{ij}|\mathbf{f}_S)$ and $P_k(l^k_{ij}|\mathbf{f}_k)$. This is fairly straightforward. Given a data bank consisting of labeled trees (such as the Penn Treebank), we simply convert each tree into its $\mathcal{H}$-labeling and use the probabilistically determined variable assignments to compile our training instances. In this way, we compile $k + 1$ sets of training instances that we can use to induce P_S, and the P_k distributions. The choice of which learning technique to use is up to the personal preference of the user. The only requirement is that it must return a conditional probability distribution, and not a hard classification. Techniques that allow this include relative frequency, maximum entropy models, and decision trees. For our experiments, we used maximum entropy learning. Specifics are deferred to Section 6.

5 Decoding

For the PCFG parsing model, we can find $argmax_{tree} P(tree|sentence)$ using a cubic-time dynamic programming-based algorithm. By adopting a more flexible probabilistic model, we sacrifice polynomial-time guarantees. The central question driving this paper is whether we can jettison these guarantees and still obtain good performance in practice. For the decoding of the probabilistic model of the previous section, we choose a depth-first branch-and-bound approach, specifically because of two advantages. First, this approach takes linear space. Second, it is anytime,

HLPDECODE(HLP $\mathcal{H}$, int n, assignment $\mathbf{w}$):

1. Initialize stack S with the pair $\langle \mathbf{x}_\emptyset, 1 \rangle$, where $\mathbf{x}_\emptyset$ is the trivial assignment of $\mathbf{V}_\mathcal{L}$. Let $\mathbf{x}_{best} = \mathbf{x}_\emptyset$; let $p_{best} = 0$. Until stack S is empty, repeat steps 2 to 4.
2. Pop topmost pair $\langle \mathbf{x}, p \rangle$ from stack S.
3. If $p > p_{best}$ and $\mathbf{x}$ is an $\mathcal{H}$-labeling of length n, then: let $\mathbf{x}_{best} = \mathbf{x}$; let $p_{best} = p$.
4. If $p > p_{best}$ and $\mathbf{x}$ is not yet a $\mathcal{H}$-labeling of length n, then:
   (a) Let Y be the earliest variable in $\mathbf{V}^n_\mathcal{L}$ (according to model order $<$) unassigned by $\mathbf{x}$.
   (b) If $Y \in dom(\mathbf{w})$, then push pair $\langle \mathbf{x}[Y = \mathbf{w}(Y)], p \rangle$ onto stack S.
   (c) Else if $\mathcal{A}(Y, \mathbf{x}, n) = \langle true, y \rangle$ for some value $y \in dom(Y)$, then push pair $\langle \mathbf{x}[Y = y], p \rangle$ onto stack S.
   (d) Otherwise for every value $y \in dom(Y)$, push pair $\langle \mathbf{x}[Y = y], p \cdot q(y) \rangle$ onto stack S in ascending order of the value of $q(y)$, where:

$$q(y) = \begin{cases} P_S(y|\mathcal{F}^S(\mathbf{x}, i, j, n)) & \text{if } Y = S_{ij} \\ P_k(y|\mathcal{F}^k(\mathbf{x}, i, j, n)) & \text{if } Y = L^k_{ij} \end{cases}$$

5. Return $\mathbf{x}_{best}$.

Figure 6: Pseudocode for the decoder.

i.e. it finds a (typically good) solution early and improves this solution as the search progresses. Thus if one does not wish the spend the time to run the search to completion (and ensure optimality), one can use this algorithm easily as a heuristic by halting prematurely and taking the best solution found thus far.

The search space is simple to define. Given an HLP $\mathcal{H}$, the search algorithm simply makes assignments to the model variables (depth-first) in the order defined by $<$.

This search space can clearly grow to be quite large, however in practice the search speed is improved drastically by using branch-and-bound backtracking. Namely, at any choice point in the search space, we first choose the least cost child to expand (i.e. we make the most probable assignment). In this way, we quickly obtain a greedy solution (in linear time). After that point, we can continue to keep track of the best solution we have found so far, and if at any point we reach an internal node of our search tree with partial cost greater than the total cost of our best solution, we can discard this node and discontinue exploration of that subtree. This technique can result in a significant aggregrate savings of computation time, depending on the nature of the cost function.

Figure 6 shows the pseudocode for the depth-first branch-and-bound decoder. For an HLP $\mathcal{H} = \langle \mathcal{L}, <, \mathcal{A}, \mathcal{F}, \mathcal{P} \rangle$, a positive integer n, and a partial assignment $\mathbf{w}$ of $\mathbf{V}^n_\mathcal{L}$, the call HLPDECODE($\mathcal{H}$, n, $\mathbf{w}$) returns the $\mathcal{H}$-labeling $\mathbf{x}$ of length n such that $P(\mathbf{x}|n, \mathbf{w})$ is maximized.

6 Experiments

We employed a familiar experimental set-up. For training, we used sections 2–21 of the WSJ section of the Penn treebank. As a development set, we used the first 20 files of section 22, and then saved section 23 for testing the final model. One unconventional preprocessing step was taken. Namely, for the entire treebank, we compressed all unary chains into a single node, labeled with the label of the node furthest from the root. We did so in order to simplify our experiments, since the framework outlined in this paper allows only one label per labeling scheme per span. Thus by avoiding unary chains, we avoid the need for many labeling schemes or more complicated compound labels (labels like "NP-NN"). Since our goal here was not to create a parsing tool but rather to explore the viability of this approach, this seemed a fair concession. It should be noted that it is indeed possible to create a fully general parser using our framework (for instance, by using the above idea of compound labels for unary chains).

The main difficulty with this compromise is that it renders the familiar metrics of labeled precision and labeled recall incomparable with previous work (i.e. the LP of a set of candidate parses with respect to the unmodified test set differs from the LP with respect to the preprocessed test set). This would be a major problem, were it not for the existence of other metrics which measure only the quality of a parser's recursive decomposition of a sentence. Fortunately, such metrics do exist, thus we used *cross-bracketing statistics* as the basic measure of quality for our parser. The cross-bracketing score of a set of candidate parses with

word(i+k) = w	word(j+k) = w
preterminal(i+k) = p	**preterminal(j+k) = p**
label(i+k) = l	**label(j+k) = l**
category(i+k) = c	category(j+k) = c
signature(i,i+k) = s	

Figure 7: Basic feature templates used to determine constituency and labeling of span (i, j). k is an arbitrary integer.

	≤ 40		≤ 100	
	CB	0CB	CB	0CB
Magerman (1995)	1.26	56.6		
Collins (1996)	1.14	59.9		
Klein/Manning (2003)	1.10	60.3	1.31	57.2
this paper	**1.09**	**58.2**	**1.25**	**55.2**
Charniak (1997)	1.00	62.1		
Collins (1999)	0.90	67.1		

Figure 8: Cross-bracketing results for Section 23 of the Penn Treebank.

respect to the unmodified test set is identical to the cross-bracketing score with respect to the preprocessed test set, hence our preprocessing causes no comparability problems as viewed by this metric.

For our parsing model, we used an HLP $\mathcal{H} = \langle \mathcal{L}, <, \mathcal{A}, \mathcal{F}, \mathcal{P} \rangle$ with the following parameters. $\mathcal{L}$ consisted of three labeling schemes: the set L^{wd} of word labels, the set L^{pt} of preterminal labels, and the set L^{nt} of nonterminal labels. The order $<$ of the model variables was the unique order such that for all suitable integers i, j, k, l: (1) $S_{ij} < L^{wd}_{ij} < L^{pt}_{ij} < L^{nt}_{ij}$, (2) $L^{nt}_{ij} < S_{kl}$ iff span (i, j) is strictly shorter than span (k, l) or they have the same length and integer i is less than integer k. For auto-assignment function $\mathcal{A}$, we essentially used the function in Figure 5, modified so that it automatically assigned $null$ to model variables L^{wd}_{ij} and L^{pt}_{ij} for $i \neq j$ (i.e. no preterminal or word tagging of internal nodes), and to model variables L^{nt}_{ii} (i.e. no nonterminal tagging of leaves, rendered unnecessary by our preprocessing step).

Rather than incorporate part-of-speech tagging into the search process, we opted to pretag the sentences of our development and test sets with an off-the-shelf tagger, namely the Brill tagger (Brill, 1994). Thus the object of our computation was HLPDECODE($\mathcal{H}$, n, $\mathbf{w}$), where n was the length of the sentence, and partial assignment $\mathbf{w}$ specified the word and PT labels of the leaves. Given this partial assignment, the job of HLPDECODE was to find the most probable assignment of model variables S_{ij} and L^{nt}_{ij} for $1 \leq i < j \leq n$.

The two probability models, P^S and P^{nt}, were trained in the manner described in Section 4. Two decisions needed to be made: which features to use and which learning technique to employ. As for the learning technique, we used maximum entropy models, specifically the implementation called MegaM provided by Hal Daume (Daumé III, 2004). For P^S, we needed features that would be relevant to deciding whether a given span (i, j) should be considered a constituent. The basic building blocks we used are depicted in Figure 7. A few words of explanation are in order. By $label(k)$, we mean the highest nonterminal label so far assigned that covers word k, or if such a label does not yet exist, then the preterminal label of k (recall that our model order was bottom-up). By $category(k)$, we mean the category of the preterminal label of word k (given a coarser, hand-made categorization of preterminal labels that grouped all noun tags into one category, all verb tags into another, etc.). By $signature(k, m)$, where $k \leq m$, we mean the sequence $\langle label(k), label(k+1), ..., label(m) \rangle$, from which all consecutive sequences of identical labels are compressed into a single label. For instance, $\langle IN, NP, NP, VP, VP \rangle$ would become $\langle IN, NP, VP \rangle$. Ad-hoc conjunctions of these basic binary features were used as features for our probability model P^S. In total, approximately 800,000 such conjunctions were used.

For P^{nt}, we needed features that would be relevant to deciding which nonterminal label to give to a given constituent span. For this somewhat simpler task, we used a subset of the basic features used for P^S, shown in bold in Figure 7. Ad-hoc conjunctions of these boldface binary features were used as features for our probability model P^{nt}. In total, approximately 100,000 such conjunctions were used.

As mentioned earlier, we used cross-bracketing statistics as our basis of comparision. These results as shown in Figure 8. CB denotes the average cross-bracketing, i.e. the overall percentage of candidate constituents that properly overlap with a constituent in the gold parse. 0CB denotes the percentage of sentences in the test set that exhibit no cross-bracketing. With a simple feature set, we manage to obtain performance comparable to the unlexicalized PCFG parser of (Klein and Manning, 2003) on the set of sentences of length

40 or less. On the subset of Section 23 consisting of sentences of length 100 or less, our parser slightly outperforms their results in terms of average cross-bracketing. Interestingly, our parser has a lower percentage of sentences exhibiting no cross bracketing. To reconcile this result with the superior overall cross-bracketing score, it would appear that when our parser does make bracketing errors, the errors tend to be less severe.

The surprise was how quickly the parser performed. Despite its exponential worst-case time bounds, the search space turned out to be quite conducive to depth-first branch-and-bound pruning. Using an unoptimized Java implementation on a 4x Opteron 848 with 16GB of RAM, the parser required (on average) less than 0.26 seconds per sentence to optimally parse the subset of Section 23 comprised of sentences of 40 words or less. It required an average of 0.48 seconds per sentence to optimally parse the sentences of 100 words or less (an average of less than 3.5 seconds per sentence for those sentences of length 41-100). As noted earlier, the parser requires space linear in the size of the sentence.

7 Discussion

This project began with a question: can we develop a history-based parsing framework that is simple, general, and effective? We sought to provide a versatile probabilistic framework that would be free from the constraints that dynamic programming places on PCFG-based approaches. The work presented in this paper gives favorable evidence that more flexible (and worst-case intractable) probabilistic approaches can indeed perform well in practice, both in terms of running time and parsing quality.

We can extend this research in multiple directions. First, the set of features we selected were chosen with simplicity in mind, to see how well a simple and unadorned set of features would work, given our probabilistic model. A next step would be a more carefully considered feature set. For instance, although lexical information was used, it was employed in only a most basic sense. There was no attempt to use head information, which has been so successful in PCFG parsing methods.

Another parameter to experiment with is the model order, i.e. the order in which the model variables are assigned. In this work, we explored only one specific order (the left-to-right, leaves-to-head assignment) but in principle there are many other feasible orders. For instance, one could try a top-down approach, or a bottom-up approach in which internal nodes are assigned immediately after all of their descendants' values have been determined.

Throughout this paper, we strove to present the model in a very general manner. There is no reason why this framework cannot be tried in other application areas that rely on dynamic programming techniques to perform hierarchical labeling, such as phrase-based machine translation. Applying this framework to such application areas, as well as developing a general-purpose parser based on HLPs, are the subject of our continuing work.

References

Ezra Black, Fred Jelinek, John Lafferty, David M. Magerman, Robert Mercer, and Salim Roukos. 1993. Towards history-based grammars: using richer models for probabilistic parsing. In *Proc. ACL*.

Eric Brill. 1994. Some advances in rule-based part of speech tagging. In *Proc. AAAI*.

Eugene Charniak. 1997. Statistical parsing with a context-free grammar and word statistics. In *Proc. AAAI*.

Eugene Charniak. 2000. A maximum entropy-inspired parser. In *Proc. NAACL*.

Eugene Charniak. 2001. Immediate-head parsing for language models. In *Proc. ACL*.

Michael Collins. 1996. A new statistical parser based on bigram lexical dependencies. In *Proc. ACL*.

Michael Collins. 1999. *Head-driven statistical models for natural language parsing*. Ph.D. thesis, University of Pennsylvania.

Hal Daumé III. 2004. Notes on CG and LM-BFGS optimization of logistic regression. Paper available at http://www.isi.edu/ hdaume/docs/daume04cg-bfgs.ps, implementation available at http://www.isi.edu/ hdaume/megam/, August.

Mark Johnson. 1998. Pcfg models of linguistic tree representations. *Computational Linguistics*, 24:613–632.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proc. ACL*.

David M. Magerman. 1995. Statistical decision-tree models for parsing. In *Proc. ACL*.

Adwait Ratnaparkhi. 1997. A linear observed time statistical parser based on maximum entropy models. In *Proc. EMNLP*.

Transformation-based Interpretation of Implicit Parallel Structures: Reconstructing the meaning of *vice versa* and similar linguistic operators

Helmut Horacek
Fachrichtung Informatik
Universität des Saarlandes
66041 Saarbrücken, Germany
horacek@ags.uni-sb.de

Magdalena Wolska
Fachrichtung Allgemeine Linguistik
Universität des Saarlandes
66041 Saarbrücken, Germany
magda@coli.uni-sb.de

Abstract

Successful participation in dialogue as well as understanding written text requires, among others, interpretation of specifications implicitly conveyed through parallel structures. While those whose reconstruction requires insertion of a missing element, such as gapping and ellipsis, have been addressed to a certain extent by computational approaches, there is virtually no work addressing parallel structures headed by *vice versa*-like operators, whose reconstruction requires transformation. In this paper, we address the meaning reconstruction of such constructs by an informed reasoning process. The applied techniques include building deep semantic representations, application of categories of patterns underlying a formal reconstruction, and using pragmatically-motivated and empirically justified preferences. We present an evaluation of our algorithm conducted on a uniform collection of texts containing the phrases in question.

1 Introduction

Specifications implicitly conveyed through parallel structures are an effective means of human communication. Handling these utterances adequately is, however, problematic for a machine since a formal reconstruction of the representation may be associated with ambiguities, typically requiring some degree of context understanding and domain knowledge in their interpretation. While parallel structures whose reconstruction mainly requires insertion, such as gapping and ellipsis, have been addressed to a certain extent by computational approaches, there is virtually no work addressing parallel structures whose reconstruction requires transformation. Several linguistic operators create specifications of this kind, including: *the other way (a)round*, *vice-versa*, and *analogously*. Consider, for example, the following statement made by a student in an experiment with a simulated tutoring system for proving theorems in elementary set theory (Benzmüller et al., 2003): "If all A are contained in $K(B)$ and this also holds *the other way round*, these must be identical sets" (K stands for set complement). The interpretation of the *the other way round* operator is ambiguous here in that it may operate on immediate dependents ("all $K(B)$ are contained in A") or on the embedded dependents ("all B are contained in $K(A)$") of the verb "contain". The fact that the *Containment* relation is asymmetric and the context of the task – proving that "If $A \subseteq K(B)$, then $B \subseteq K(A)$" holds – suggest that the second interpretation is meant. Assuming this more plausible reading enables a more goal-oriented dialog: the tutorial system can focus on a response to the false conclusion made by the student about the identity of the sets in question, rather than starting a boring clarification subdialog.

The above example and several similar others motivated us to look more systematically at lexical devices that create specifications of this kind. We address the interpretation of such structures by a well-informed reasoning process. Applied techniques include building deep semantic representations, application of patterns underlying formal reconstruction, and using pragmatically-motivated and empirically justified preferences.

The outline of the paper is as follows: We describe phenomena in question. Then we illustrate our natural language analysis techniques. We cate-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 377–384,
Sydney, July 2006.

gorize underlying interpretation patterns, describe the reconstruction algorithm, and evaluate it.

2 Data Collected From Corpora

In order to learn about cross-linguistic regularities in reconstructing the underlying form of propositions specified by *vice versa* or similar operators, we first looked at several English and German corpora. These included, among others, the Negra, the Frankfurter Rundschau, the Europarl corpora and a corpus of tutorial dialogs on mathematics (Wolska et al., 2004). We also performed several internet searches. We looked at the German phrases *andersrum* and *umgekehrt*, and their English equivalents *vice versa* and *the other way (a)round*. We only considered instances where the parallel structure with a pair of items swapped is not stated explicitly. We excluded cases of the use of *umgekehrt* as a discourse marker, cases in which the transformation needed is of purely lexical nature, such as turning "augment" into "reduce", and instances of *andersrum* as expressing a purely physical change, such as altering the orientation of an object (cf. the Bielefeld corpus[1]).

The classification of *vice versa* utterances presented in Figure 1, reflects the role of the items that must be swapped to build the parallel proposition conveyed implicitly. The examples demonstrate that the task of reconstructing the proposition left implicit in the text may be tricky.

The first category concerns swapping two case role fillers or *Arguments* of a predicate head. This may be applied to Agent and Patient dependents, as in (1), or to two directional roles as in (2). In the last example in this category, complications arise due to the fact that one of the arguments is missing on the surface and needs to be contextually inserted prior to building the assertions with exchanged directional arguments. Moreover, the swap can also work across clauses as in (3). Complex interrelations may occur when the fillers themselves are composed structures, is in (4), which also makes swapping other pairs of items structurally possible. In this example, the need for exchanging the persons including their mentioned body parts rather than the mere body parts or just the persons requires world knowledge.

The second category comprises swapping applied to *modifiers* of two arguments rather than the arguments themselves. An example is (5); the utterance is ambiguous since, from a purely structural point of view, it could also be categorized as an *Argument* swap, however, given world knowledge, this interpretation is rather infelicitous. Similarly to (3), a contextually-motivated enhancement prior to applying a swapping operation is required in (6); here: a metonymic extension, i.e. expanding the "strings" to "the strings' tones".

The third category comprises occurrences of a "mixed" form of the first two with a modifier substituted for an argument which, in turn, takes the role of the modifier in the reconstructed form. The first example, (7), has already been discussed in the Introduction. The next one, (8), illustrates repeated occurrences of the items to be swapped. Moreover, swapping the items A and B must be propagated to the included formula. The next example, (9), is handled by applying the exchange on the basis of the surface structure: swapping the properties of a triangle for the reconstructed assertion. If a deeper structure of the sentence's meaning is built, this would amount to an implication expressing the fact that a triangle with two sides of equal length is a triangle that has two equal angles. For such a structure, the reconstruction would fall into the next category, exchange of the order of two propositions: here, reversing the implication. In (10), the lexeme "Saxophonist" needs to be expanded into "Saxophone" and "Spieler" ("player"), prior to performing the exchange.

The fourth category involves a swap of entire *Propositions*; in the domain of mathematics, this may pertain to formulas. In (11), swapping applies to the sides of the equation descriptively referred to by the distributivity law. In (12), this applies to the arguments of the set inclusion relation, when the arguments are interpreted as propositions. The last example, (13), requires a structural recasting in order to apply the appropriate swapping operation. When the utterance is rebuilt around the $RESULT$ relation, expressed as an optional case role on the surface, swapping the two propositions – "branching out of languages" and "geographical separation" – yields the desired result.

3 The Interpretation Procedure

In this section, we illustrate our technical contribution. It consists of three parts, each dealt with in a separate subsection: (1) the linguistic/semantic analysis, (2) definitions of rules that support building parallel structures, and (3) the algorithm.

[1] http://www.sfb360.uni-bielefeld.de/

Argument swap	(1)	Technological developments influence the regulatory framework and vice versa.
	(2)	It discusses all modes of transport from the European Union to these third countries and vice versa.
	(3)	Ok – so the affix on the verb is the trigger and the NP is the target. . . . No; the other way round
	(4)	Da traf Völler mit seinem Unterarm auf die Hüfte des für Glasgow Rangers spielenden Ukrainers, oder umgekehrt *Then Völler with his lower arm hit the hip of the Ukrainian playing for Glasgow Rangers, or the other way round*
Modifier swap	(5)	Nowadays, a surgeon in Rome can operate on an ill patient – usually an elderly patient – in Finland or Belgium and vice versa.
	(6)	Der Ton der Klarinette ist wirklich ganz komplementär zu den Seiteninstrumenten und umgekehrt *The clarinet's tone is really very complimentary to strings and vice-versa*
Mixed swap	(7)	Wenn alle A in $K(B)$ enthalten sind und dies auch umgekehrt gilt, muß es sich um zwei identische Mengen handeln *If all A are contained in $K(B)$ and this also holds* vice-versa*, these must be identical sets*
	(8)	Dann ist das Komplement von Menge A in Bezug auf B die Differenz $A/B = K(A)$ und umgekehrt *Then the complement of set A in relation to B is the difference $A/B = K(A)$ and vice-versa*
	(9)	Ein Dreieck mit zwei gleichlangen Seiten hat zwei gleichgroße Winkel und umgekehrt *A triangle with two sites of equal length has two angles of equal size, and vice-versa*
	(10)	. . . Klarinette für Saxophonist und umgekehrt . . . *. . . a clarinet for a saxophonist and vice-versa . . .*
Proposition swap	(11)	Man muß hier das Gesetz der Distributivität von Durchschnitt über Vereinigung umgekehrt anwenden *It is necessary here to apply the law of distributivity of intersection over union in reverse direction*
	(12)	Es gilt: $P(C \cup (A \cap B)) \subseteq P(C) \cup P(A \cap B)$. . . . . Nein, andersrum. *It holds: $P(C \cup (A \cap B)) \subseteq P(C) \cup P(A \cap B)$. . . . . No, the other way round.*
	(13)	Wir wissen, daß sich Sprachen in Folge von geographischer Separierung auseinanderentwickeln, und nicht umgekehrt *We know that languages branch out as a result of geographical separation, not the other way round*

Figure 1: Examples of utterances with *vice versa* or similar operators

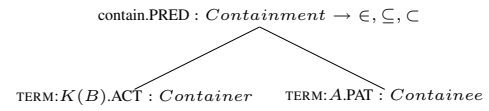


Figure 2: Interpreted representation of the utterance "all A are contained in $K(B)$"

3.1 Linguistic Analysis

The linguistic analysis consists of semantic parsing followed by contextually motivated embedding and enhancements. We assume a deep semantic dependency-based analysis of the source text. The input to our reconstruction algorithm is a relational structure representing a dependency-based deep semantics of the utterance, e.g. in the sense of Prague School sentence meaning, as employed in the Functional Generative Description (FGD) at the tectogrammatical level (Sgall et al., 1986). In FGD, the central frame unit of a clause is the head verb which specifies the *tectogrammatical relations* (TRs) of its dependents (*participants/modifications*). Every valency frame specifies, moreover, which modifications are *obligatory* and which *optional*. For example, the utterance (7) (see Figure 1.) obtains the interpretation presented in Figure 2.[2] which, in the context of an informal verbalization of a step in a naive set theory proof, translates into the following formal statement: "$\forall x.x \in A \Rightarrow x \in K(B)$".

The meaning representations are embedded within discourse context and discourse relations between adjacent utterances are inferred where possible, based on the linguistic indicators (discourse markers). The nodes (heads) and dependency relations of the interpreted dependency structures as well as discourse-level relations serve as input to instantiate the reconstruction patterns. Contextual enhancements (e.g. lexical or metonymic extensions) driven by the reconstruction requirements may be carried out.

Based on analysis of corpora, we have identified combinations of dependency relations that commonly participate in the swapping operation called for by the *vice versa* phrases. Examples of pairs of such relations at sentence level are shown in Figure 3.[3] Similarly, in the discourse context, arguments in, for example, $CAUSE$, $RESULT$, $CONDITION$, $SEQUENCE$ or $LIST$ relations are likely candidates for a swapping operation. During processing, we use the association table as a preference criterion for selecting candidate relations to instantiate patterns. If one of the elements of a candidate pair is an *optional argument* that is not realized in the given sentence, we look at the preceding context to find the first instance of the missing element. Additionally, utterance (10) would call for more complex procedures to identify the required metonymic expansion.

Exchangeable(ACTOR, PATIENT)
Exchangeable(DIRECTION-WHERE-FROM, DIRECTION-WHERE-TO)
Exchangeable(TIME-TILL-WHEN, TIME-FROM-WHEN)
Exchangeable(CAUSE, PRED)
Exchangeable(CONDITION, PRED)

Figure 3: Examples of exchangeable relations

3.2 Interpretation Patterns

In order to accomplish the formal reconstruction task, we define rules that encapsulate specifications for building the implicit parallel text on the basis of the corresponding co-text. The rules consist of a pattern and an action part. Patterns are matched against the output of a parser on a text portion in question, by identifying relevant case roles, and giving access to their fillers. Moreover, the patterns test constraints on compatibility of candidates for swapping operations. The actions apply recasting operations on the items identified by the patterns to build the implicit parallel text.

Within patterns, we perform category membership tests on the representation. Assuming x referring to a semantic representation, $Pred(x)$ is a logical function that checks if x has a $Pred$-feature, i.e., it is an atomic proposition. Similarly, $Conj(x)$ and $Subord(x)$ perform more specific tests for complex propositions: coordination or subordination, respectively. Moreover, $Pred_1(x, x_1)$ accesses the first proposition and binds it to x_1, while $Pred_2(x, x_2)$ does the same for the second one. Within a proposition, arguments and modifiers are accessed by $Case(x, y)$, where y specifies the filler of $Case$ in x, and indices express constraints on identity or distinctiveness of the relations. $Case^+$ is a generalization of $Case$ for iterative embeddings, where individual cases in the chain are not required to be

[2] We present a simplified schematic representation of the tectogrammatical representations. Where necessary, for space reasons, irrelevant parts are omitted.

[3] $PRED$ is the immediate predicate head of the corresponding relation.

1a. *Argument* swap within the same clause

$$Pred(x) \wedge Case_1(x, y) \wedge Case_2(x, z) \wedge Type - compatible(y, z) \wedge Exchangeable(Case_1, Case_2) \rightarrow Swap(x, y, z, x_p)$$

1b. *Argument* swap across two clauses

$$Conj(x) \wedge Case_1(x, y) \wedge Case(y, u) \wedge Case_2(x, z) \wedge Case(z, v) \rightarrow Swap(x, u, v, x_p)$$

2. *Modifier* swap

$$Pred(x) \wedge Case_1(x, y) \wedge Case^{+}_{11}(y, u) \wedge Case_2(x, z) \wedge Case^{+}_{21}(z, v) \wedge \neg(Case_1 = Case_2) \wedge Type - compatible(u, v) \rightarrow Swap(x, u, v, x_p)$$

3. *Mixed* swap

$$Pred(x) \wedge Case_1(x, y) \wedge Case_{11}(y, u) \wedge Case_2(x, z) \wedge \neg(Case_1 = Case_2) \wedge Type - compatible(u, z) \rightarrow Swap(x, u, z, x_p)$$

4. *Proposition* swap

$$Subord(x) \wedge Case_1(x, y) \wedge Case_2(x, z) \wedge \neg(Case_1 = Case_2) \rightarrow Swap(x, y, z, x_p)$$

Figure 4: Reconstruction patterns

1. *Lexical expansion*

$$Pred(x) \wedge Case_1(x, y) \wedge Lex - Expand(y, u, Case, v) \wedge Case_2(x, z) \wedge \neg(Case_1 = Case_2) \wedge Type - compatible(v, z) \rightarrow Swap(x, y, Case(u, v), x_p) \wedge Swap(x_p, z, v, x_p)$$

2. *Recast optional case as head of an obligatory*

$$Pred(x) \wedge Case_1(x, u) \wedge Case_2(x, v) \wedge Type(u, tu) \wedge Type(v, tv) \wedge Recastable(tv, Case_2, tu, Case_3) \wedge Case_3(x, w) \wedge Type - compatible(v, w) \wedge \neg(Case_1 = Case_2) \wedge \neg(Case_1 = Case_3) \wedge \neg(Case_2 = Case_3) \rightarrow Swap(x, u, v, x_p) \wedge Add(x_p, Case_3(v, u)) \wedge Remove(x_p, Case_2)$$

3. *Recast an optional case as a discourse relation*

$$Pred(x) \wedge Case(x, y) \wedge Member(Case, Subords) \rightarrow Build(Case(x_p, Case_2(x_p, y) \wedge Case_1(x_p, Remove(x, y))$$

Figure 5: Recasting rules

identical. In addition to access predicates, there are test predicates that express constraints on the identified items. The most basic one is $Type\text{-}compatible(x, y)$, which tests whether the types of x and y are compatible according to an underlying domain ontology. A more specific test is performed by $Exchangeable(Case_1, Case_2)$ to access the associations specified in the previous section. The action part of the patterns is realized by $Swap(x, y, z, x_p)$ which replaces all occurrences of x in z by y and vice-versa, binding the result to x_p. Different uses of this operation result in different instantiations of y and z with respect to the overarching structure x.

There are patterns for each category introduced in Section 2 (see Figure 4). All patterns are tested on a structure x and, if successful, the result is bound to x_p. For *Argument* swap there are two patterns. If the scope of the swap is a single clause (1a), two arguments (case roles) identified as exchangeable are picked. Their fillers must be compatible in types. If the swapping overarches two clauses (1b), the connecting relation must be a conjunction and subject to swapping are arguments in the same relations. For *Modifier* swap (2), type compatible modifiers of distinct arguments are picked. For *Mixed* swap (3), a dependent is picked, as in (1a) and a type-compatible modifier of another argument, as in (2). *Proposition* swap (4) inverts the order of the two clauses.

In addition to the the pattern matching tests, the *Argument* and the *Proposition* swap operations undergo a feasibility test if knowledge is available about symmetry or asymmetry of the relation (the $Pred$ feature) whose cases are subject to the swapping operation: if such a relation is known as asymmetric, the result is considered implausible due to semantic reasons, if it is symmetric, due to pragmatic reasons since the converse proposition conveys no new information; in both cases such a swapping operation is not carried out.

To extend the functionality of the patterns, we defined a set of recasting rules (Figure 5) invoked to reorganize the semantic representation prior to testing applicability of a suitable reconstruction rule. In contrast to inserting incomplete information contextually and expanding metonymic relations the recasting operations are intended purely to accommodate semantic representations for this purpose. We have defined three recasting rules (numbered accordingly in Figure 5):

1. *Lexical recasting*

   The semantics of some lexemes conflates the meaning of two related items. If one of them is potentially subject to swapping, it is not accessible for the operation without possibly af-

Build-Parallel-Structure (x)

1. Determine scopes for applying swap operations
$Structures \leftarrow \epsilon$
if $Pred(x)$ **then** $Scopes \leftarrow \{x\}$ **else**
  if $Subord(x) \vee Conj(x) \wedge Case_2(x, z)$
    then $Scopes \leftarrow \{z, x\}$
endif endif

2. Match patterns and build swapped structures

forall $Scope_1$ in $Scopes$ **do**
  $Structures \leftarrow Structures \cup$
  $< X - swap(Scope_1) >$
  $< X - swap(Y - recast(Scope_1)) >$
end forall
return $Sort(Apply - priorities(Structures))$

Figure 6: Reconstruction algorithm

fecting the other so closely related to it. The representation of such lexemes is expanded, provided there is a sister case with a filler that is type compatible.

2. *Case recasting*

   The dependency among items may not be reflected by the dependencies in the linguistic structure. Specifically, a dependent item may appear as a sister case in overarching case frame. The purpose of this operation is to build a uniform representation, by removing the dependent case role filler and inserting it as a modifier of the item it is dependent on.

3. *Proposition recasting*

   Apart from expressing a discourse relation by a connective, a proposition filling a subordinate relation may also be expressed as a case role (argument). Again, uniformity is obtained through lifting the argument (case filler) and expressing the discourse relation as a multiple clause construct.

Additional predicates are used to implement recasting operations. For example, the predicate $Lex{-}Expand(y, u, Case, v)$ re-expresses the semantics of y by u, accompanied by a $Case$ role filled by v. $Type(x, y)$ associates the type y with x. The type information is used to access $Recastable(t_1, C_1, t_2, C_2)$ table to verify whether case C_1 with a t_1-type filler can also be expressed as case C_2 with type t_2. $Build(x)$ creates a new structure x. $Remove(x, y)$ is realized as a function, deleting occurrences of y in x, and $Add(x, y)$ expands x by an argument y.

3.3 The Structure Building Algorithm

In this section, we describe how we build implicitly conveyed parallel structures based on the definitions of swapping operations with optional incorporation of recasting operations if needed. The procedure consists of two main parts (see Figure 6). In the first part, the scope for applying the swapping rules defined in Figure 4 is determined, and in the second part, the results obtained by executing the rules are collected. Due to practical reasons, we introduce simplifications concerning the scope of *vice-versa* in the current formulation of the procedure. While the effect of this operator may range over entire paragraphs in some involved texts, we only consider single sentences with at most two coordinated clauses or one subordinated clause. We feel that this restriction is not severe for uses in application-oriented systems.

The procedure *Build-Parallel-Structure* takes the last input sentence x, examines its clause structure, and binds potential scopes to variable $Scopes$. For composed sentences, the entire sentence (x) as well as the second clause ($Case_2(x, z)$) is a potential scope for building parallel structures.

In the second part of the procedure, each swapping pattern is tested for the two potential scopes, and results are accumulated in $Structures$. The call $< X - swap(Scope_1) >$, with X being either $Case$, $Argument$, $Mixed$, or $Prop$ expresses building a set of all possible instantiations of the pattern specified when applied to $Scope_1$. Some of these operations are additionally invoked with alternative parameters which are accommodated by a recasting operation fitting to the pattern used, that call being $< X - swap(Y - recast(Scope_1)) >$, where Y is $Case$, Lex, or $Prop$. Finally, if multiple readings are generated, they are ranked according to the following prioritized criteria:

1. The nearest scope is preferred;
2. Operations swapping "duals", such as left-right, are given priority;
3. Candidate phrases are matched against the corpus; items with higher bigram frequencies are preferred.

Linguistic analysis, structure reconstruction patterns, recasting rules, and the algorithms operating on top of these structures are formulated in a domain-independent way, also taking care that the tasks involved are clearly separated. Hence, it is up to a concrete application to elaborate lexical

semantic definitions required (e.g. for a saxophonist to capture example (10) in Figure 1) to define the tables $Exchangeable$ and $Recastable$, and to enhance preference criteria.

4 Evaluation

We conducted an evaluation of the parallel structure building algorithm on a sample of sentences from Europarl (Koehn, 2002), a parallel corpus of professionally translated proceedings of the European Parliament aligned at the document and sentence level. At this point, we were able to conduct only manual evaluation. This is mainly due to the fact that we did not have access to a wide-coverage semantic dependency parser for English and German.[4] In this section, we present our corpus sample and the evaluation results.

Evaluation sample To build the evaluation sample, we used sentence- and word-tokenized English German part of Europarl. Using regular expressions, we extracted sentences with the following patterns: (i) for English, phrases *the other way a*round* or *vice versa* (ii) for German: (ii-1) the word *umgekehrt* preceded by a sequence of *und* ("and"), *oder* ("or"), *sondern* ("but"), *aber* ("but") or comma, optional one or two tokens and optional *nicht* ("not"), (ii-2) the word *umgekehrt* preceded by a sequence *gilt* ("holds") and one or two optional tokens, (ii-3): the word *anders(he)*rum*. We obtained 137 sentences.

Next, given the present limitation of our algorithm (see Section 3.3), we manually excluded those whose interpretation involved the preceding sentence or paragraph,[5] as well as those in which the interpretation was explicitly spelled out. There were 27 such instances. Our final evaluation sample consisted of 110 sentences: 82 sentences in English–German pairs and 28 German-only.[6]

[4] In the future, we are planning an automated evaluation in which as input to the implemented algorithm we would pass manually built dependency structures.

[5] For example, sentences such as: "Mr President , concerning Amendment No 25 , I think the text needs to be looked at because in the original it is the other way round to how it appears in the English text ."

[6] The reason for this split is that the English equivalents of the German sentences containing the word *umgekehrt* may contain phrases other than *the other way round* or *vice versa*. Depending on context, phrases such as *conversely*, *in* or *the reverse*, *the opposite*, *on the contrary* may be used. Here, we targeted only *the other way round* and *vice versa* phrases. If the German translation contained the word *umgekehrt*, and the English source one of the alternatives to our target, in the evaluation we included only the German sentence.

Category	No. of instances
Arg	64
Modifier	5
Arg/Mod	3
Mixed	6
Arg/Mixed	2
Prop	1
Arg/Prop	1
Lex	18
Other	10
Total	110

Table 1: Distribution of patterns

Distribution of categories We manually categorized the structures in our sample and marked the elements of the dependency structures that participate in the transformation. Table 1. presents the distribution of structure categories. We explicitly included counts for alternative interpretations. For example *Arg/Mod* means that either the *Argument* or *Modifier* transformation can be applied with the same effect, as in the sentence "External policy has become internal policy, and vice versa": either the words "external" and "internal" may be swapped (*Modifier*), or the whole NPs "external policy" and "internal policy" (*Argument*). *Lex* means that none of the patterns was applicable and a lexical paraphrase (such as use of an antonym) needed to be performed in order to reconstruct the underlying semantics (i.e. no parallel structure was involved). *Other* means that there was a parallel structure involved, however, none of our patterns covered the intended transformation.

Evaluation results The evaluation results are presented in Tables 2. and 3. Table 2. shows an overview of the results. The interpretation of the result categories is as follows:

Correct: the algorithm returned the intended reading as a unique interpretation (this includes correct identification of "lexical paraphrases" (the *Lex* category in Table 1.);

Ambig.: multiple results were returned with the intended reading among them;

Wrong: the algorithm returned a wrong result (if multiple results, then the intended one was not included);

Failed: the algorithm failed to recognize a parallel structure where one existed because no known pattern matched.

Table 3. shows within-category results. Here, Correct result for *Other* means that the algorithm correctly identified 8 cases to which no current pattern applied. The two Wrong results for *Other*

Result	No. of instances
Correct	75
Ambig.	21
Wrong	4
Failed	10
Total	110

Table 2: Evaluation results

Category	Correct	Ambig.	Wrong	Failed	Total
Arg	46	17	0	1	64
Mod	3	2	0	0	5
Arg/Mod	3	–	0	0	3
Mixed	4	2	0	0	6
Arg/Mixed	2	–	0	0	2
Prop	1	0	0	0	1
Arg/Prop	0	–	0	1	1
Lex	16	0	2	0	18
Other	8	0	2	0	10

Table 3: Within-category results

mean that a pattern was identified, however, this pattern was not the intended one. In two cases (false-negatives), the algorithm failed to identify a pattern even though it fell into one of the known categories (*Argument* and *Prop*).

Discussion The most frequently occurring pattern in our sample is *Argument*. This is often a plausible reading. However, in 3 of the 4 false-positives (Wrong results), the resolved incorrect structure was *Arg*. If we were to take *Arg* as baseline, aside from missing the other categories (altogether 12 instances), we would obtain the final result of 63 Correct (as opposed to 96; after collapsing the Correct and Ambig. categories) and 15 (as opposed to 4) Wrong results.

Let us take a closer look at the false-negative cases and the missed patterns. Two missed known categories involved multiple arguments of the main head: a modal modifier (modal verb) and an additive particles ("also") in one case, and in the other, rephrasing after transformation. To improve performance on cases such as the former, we could incorporate an exclusion list of dependents that the transformation should disregard.

Among the patterns currently unknown to the algorithm, we found four types (one instance of each in the sample) that we can anticipate as frequently recurring: aim and recipient constructs involving a head and its Aim- and Beneficiary-dependent respectively, a temporal-sequence in which the order of the sequence elements is reversed, and a comparative structure with swapped relata. The remaining 6 structures require a more involved procedure: either the target dependent is deeply embedded or paraphrasing and/or morphological transformation of the lexemes is required.

5 Conclusions and Future Research

In this paper, we presented techniques of formal reconstruction of parallel structures implicitly specified by *vice versa* or similar operators. We addressed the problem by a domain-independent analysis method that uses deep semantics and contextually enhanced representations, exploits recasting rules to accommodate linguistic variations into uniform expressions, and makes use of patterns to match parallel structure categories.

Although we dedicated a lot of effort to building a principled method, the success is limited with respect to the generality of the problem: in some cases, the scope of reconstruction overarches entire paragraphs and deciding about the form requires considerable inferencing (cf. collection at http://www.chiasmus.com/). For our purposes, we are interested in expanding our method to other kinds of implicit structures in the tutorial context, for example, interpretations of references to analogies, in the case of which structure accommodation and swapping related items should also be prominent parts.

References

C. Benzmüller, A. Fiedler, M. Gabsdil, H. Horacek, I. Kruijff-Korbayová, M. Pinkal, J. Siekmann, D. Tsovaltzi, B.Q. Vo, and M. Wolska. 2003. A Wizard-of-Oz experiment for tutorial dialogues in mathematics. In *Supplementary Proceedings of the 11th Conference on Artifi cial Intelligence in Education (AIED-03); Vol. VIII. Workshop on Advanced Technologies for Mathematics Education*, pages 471–481, Sydney, Australia.

P. Koehn. 2002. Europarl: A multilingual corpus for evaluation of machine translation, Draft, Unpublished.

P. Sgall, E. Hajičová, and J. Panevová. 1986. *The meaning of the sentence in its semantic and pragmatic aspects*. Reidel Publishing Company, Dordrecht, The Netherlands.

M. Wolska, B.Q. Vo, D. Tsovaltzi, I. Kruijff-Korbayová, E. Karagjosova, H. Horacek, M. Gabsdil, A. Fiedler, and C. Benzmüller. 2004. An annotated corpus of tutorial dialogs on mathematical theorem proving. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC-04)*, pages 1007–1010, Lisbon, Potugal.

When Conset meets Synset: A Preliminary Survey of an Ontological Lexical Resource based on Chinese Characters

Shu-Kai Hsieh
Institute of Linguistics
Academia Sinica
Taipei, Taiwan
shukai@gate.sinica.edu.tw

Chu-Ren Huang
Institute of Linguistics
Academia Sinica
Taipei, Taiwan
churen@gate.sinica.edu.tw

Abstract

This paper describes an on-going project concerning with an ontological lexical resource based on the abundant conceptual information grounded on Chinese characters. The ultimate goal of this project is set to construct a cognitively sound and computationally effective character-grounded machine-understandable resource.

Philosophically, Chinese ideogram has its ontological status, but its applicability to the NLP task has not been expressed explicitly in terms of language resource. We thus propose the first attempt to locate Chinese characters within the context of ontology. Having the primary success in applying it to some NLP tasks, we believe that the construction of this knowledge resource will shed new light on theoretical setting as well as the construction of Chinese lexical semantic resources.

1 Introduction

In the history of western linguistics, writing has long been viewed as a surrogate or substitute for speech, the latter being the primary vehicle for human communication. Such "surrogational model" which neglects the systematicity of writing in its own right has also occupied the predominant views in current computational linguistic studies. This paper is set to provide a quite different perspective along with the Eastern philological tradition of the study of scripts, especially the ideographic one i.e., Chinese characters (Hanzi). We believe that the conceptual knowledge information which has been *grounded* on Chinese characters can be used as a cognitively sound and computationally effective ontological lexical resource in performing some NLP tasks, and it will have contribution to the development of *Semantic Web* as well.

2 Background Issues of Chinese Ideographic Writing

2.1 Ideographic Script and Conceptual Knowledge

From the view of writing system and cognition, human conceptual information has been regarded as being *wired* in ideographic scripts. However, in reviewing the contemporary linguistic literatures concerning with the discussions of the essence of Chinese writing system, we found that the main theoretical dispute lies in the fact that, both structural descriptions and psycholinguistic modeling seem to presume that the notions of *ideography* and *phonography* are mutually exclusive.

To break the theoretical impassé, we take a *pragmatic* position in claiming the tripartite properties of Chinese characters: They are *logographic* (morpho-syllabic) in essence, function *phonologically* at the same time, and can be interpreted *ideographically* and implemented as concept instances by computers.

2.2 Chinese Wordhood

Roughly put, a Chinese character is regarded as an ideographic symbol representing *syllable* and *meaning* of a "morpheme" in spoken Chinese.

But unlike most affixing languages, Chinese has a large class of *morphemes* - which Packard (2000) calls "bound roots" - that possess certain *affixal* properties (namely, they are bound and productive in forming words), but encode **lexical** rather than

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 385–390,
Sydney, July 2006.

grammatical information. These may occur as either the left- or right-hand component of a word. For example, the *morpheme* 輸 (/shu/; "transport") can be used as either the first *morpheme* (e.g., 輸入 (/yùn-rù/; transport-into "import"), or the second *morpheme* (e.g., 運輸 /yùn-shu/; transit-transport "conveyance") of a dissyllabic word, but cannot occur in isolation.

The fuzzy boundary between free and bound morphemes is directly related to the notorious controversial notion of Chinese Wordhood. There are multiple studies showing that to a large extent, (trained or untrained) native speakers of Chinese disagree on what a (free) morpheme/word/compound is.

Such difficulty could be traced back to its historical facts. In modern Mandarin Chinese, there is a strong tendency toward dissyllabic words, while the predominant monosyllabic words in ancient Chinese remain more or less a closed set. But the conceptual knowledge encoded in monosyllabic morphemes still have their influence even on contemporary texts, and thus resulting the difficulties of word-marking decision.

3 Theoretical Setting

Yu et al (1999) reported that a *Morpheme Knowledge Base of Modern Chinese* according to all Chinese characters in GB2312-80 code has been constructed by the institute of Computational Linguistics of Peking University. This Morpheme Knowledge Base has been later integrated into the project called "Grammatical Knowledge Base of Contemporary Chinese".

It is noted that the "morphemes" adopted in this database are monosyllabic "bound morphemes". As for "free morphemes", that is, characters which can be independently used as words, are not included in the Knowledge Base. For example, the *monosyllabic character* 梳 (/shu/,"comb") has (at least) two senses. For the verbal sense ("to comb"), it can be used as a *word*; for the nominal sense ("a comb"), it can only be used in combining with other morphemes. Therefore, only the nominal sense of 梳 is included in the Knowledge Base. However, such *morpheme-based* approach can hardly escape from facing with the difficult decision of free/bound distinction in contemporary Chinese.

3.1 Hanzi/Word Space Model

Based on the consideration mentioned above, in this paper, we will propose a *historical, conventionalized, pre-theoretical* perspective in viewing the lexical and knowledge information within Chinese characters. In Figure 1, (a) illustrates a naive Hanzi space, while (d) shows a linguistic theory-laden result of Hanzi/Word space, where green areas denote to *words*, consisting of 1 to 4 characters. The decision of words (green) and non-words (white) in the space is based on certain perspectives (be it psycholinguistic or computational linguistic). Instead, we take the traditional philological construct of Hanzi into consideration. By analyzing the *conceptual* relations between characters (b) which scatter among diverse lexical resources, we construct an top-level ontology with Hanzi as its instances (c). Rather than $(a) \rightarrow (d)$, which is a predominant approach in contemporary linguistic theoretical construction of Chinese Wordhood, we believe that the proposed approach $(a) \rightarrow (b) \rightarrow (c) \rightarrow (d)$ could not only enclose the implicit conceptual information evolutionarily *encoded* in Chinese characters, but also provide a more clear knowledge scenario for the interaction of characters/words in modern linguistic theoretical setting.

3.2 Conset and Character Ontology

The new model that we propose here is called **HanziNet**. It relies on a novel notion called **conset** and a coarsely grained **upper-level ontology** of characters.

In comparison with synset, which has become a core notion in the construction of Wordnet-like lexical semantic resources, we will argue that there is a crucial difference between Word-based lexical resource and character-based lexical resource, in that they rest with finely-differentiated information contents represented by the nodes of network. A **synset**, or synonym set in WordNet contains a group of words,[1] and each of which is synonymous with the other words in the same synset. In WordNet's design, each synset can be viewed as a *concept* in a taxonomy, While in HanziNet, we are seeking to align Hanzi which share a given putatively primitive meaning extracted from traditional philological resources, so a new term **conset** (concept set) is proposed. A *conset* contains

[1] To put it exactly, it contains a group of lexical units, which can be words or collocations.

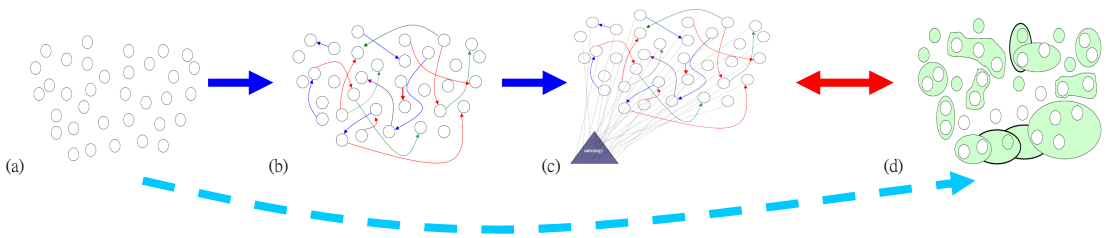


Figure 1: Illustrations of Hanzi/Word Spaces

a group of *Chinese characters similar in concept*, and each of which shares with similar conceptual information with the other characters in the same conset.[2]

The relations between consets constitute a character ontology. Formally, it is a tree-structured conceptual taxonomy in terms of which only two kinds of relations are allowed: the *INSTANCE-OF* (i.e., characters are instances of consets) and *IS-A* relations (i.e., consets are hypernyms/hyponyms to other consets).

Currently, frequently used monosyllabic characters are assigned to *at least* one of 309 consets. Following are some examples:

conset 126 (SUBJECTIVE → EXCITABILITY → ABILITY → ORGANIC FUNCTION)
吸、品、嚐、嚼、嚥、吞、饌、茹、飲,

conset 130 (SUBJECTIVE → EXCITABILITY → ABILITY → SKILLS)
摘、榨、拾、拔、提、攝、選,

conset 133 (SUBJECTIVE → EXCITABILITY → ABILITY → INTELLECT)
牟、謀、考、選、錄、記、聽,

In fact, the core assumption behind the *synset/conset* distinction is non-trivial. In this project, we assume a hypothesis of the *locality* of **Concept Gestalt** and the *context-sensibility* of **Word Sense** concerning with Chinese characters. That is, characters carry two meaning dimensions: on the one hand, they are *lexicalized* concepts; on the other hands, they can be observed linguistically as bound root morphemes and monomorphemic words according to their independent usage in modern Chinese texts.

Figure 2 shows a schematic diagram of our proposed model. In Aitchison's (2003) terms, for the character level, we take an "atomic globule" network viewpoint, where the characters - realized as instances of *core concept Gestalt* - which share similar conceptual information, cluster together. The relationships between these concept Gestalt form a *rooted tree structure*. Characters are thus assigned to the leaves of the tree in terms of an assemblage of bits. For the word level, we take the "cobweb" viewpoint, as *words* -built up from a pool of characters- are connected to each other through lexical semantic relations. In such case, the network does not form a tree structure but a more complex, long-range highly-correlated *random acyclic graphic structure*.

4 Hanzi-grounded Ontological CharacterNet

In light of the previous consideration, this section attempts to further clarify the building blocks of the **HanziNet** system, – a Hanzi-grounded ontological Character Net – with the goal to arrive at a working model which will serve as a framework for ontological knowledge processing. Briefly, HanziNet is consisted of two main parts:

[2] At the time of writing, about 3,600 characters have been finished in their information construction.

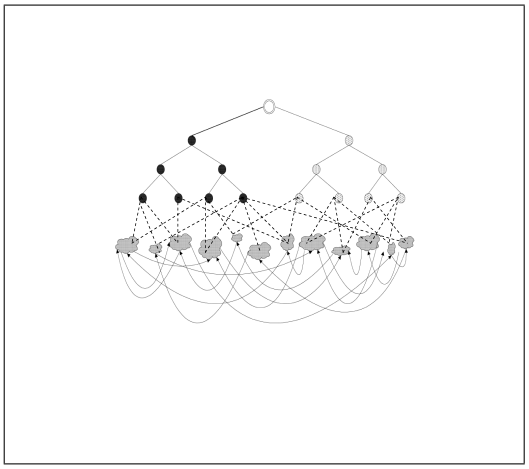


Figure 2: The Schematic Representation of character-triggered tree-like conceptual hierarchy and word-based semantic network

a character-stored machine-readable lexicon and a top-level character ontology.

4.1 Hanzi-grounded Lexicon and Ontology

The current lexicon contains over 5000 characters, and 30,000 derived words in total.[3]

The building of the lexical specification of the entries in HanziNet includes various aspects of Hanzi:

1. *Conset(s)*: The conceptual code is the core part of the MRD lexicon in HanziNet. Concepts in HanziNet are indicated by means of a label (conset name) with a code form. In order to increase the efficiency, an ideal strategy is to adopt the Huffmann-coding-like method, by encoding the conceptual structure of Hanzi as a pattern of bits set within a bit string.[4] The *coding* thus refers to the assignment of code sequences to an character. The sequence of edges from the root to any character yields the code for that character, and the number of bits varies from one character to another. Currently, for each conset (309 in total) there are 12 characters assigned on the average; for each character, it is assigned to 2-3 consets on the average.[5]
2. *Character Semantic Head (CSH) and Character Semantic Modifier (CSM) division.*[6]
3. Shallow parts of speech (mainly Nominal(N) and Verbal(V) tags)
4. Gloss of *prototypical meaning*
5. List of *combined words with statistics calculated from corpus*, and
6. Further aspects such as *character types and cognates*: According to ancient study, characters can be compartmentalized into six groups based on the six classical principles of character construction. Character type here means which group the character belongs to. And the term *cognate* here is defined as characters that share the same *CSH* or *CSM*. Figure 3 shows a snapshot of this lexicon.

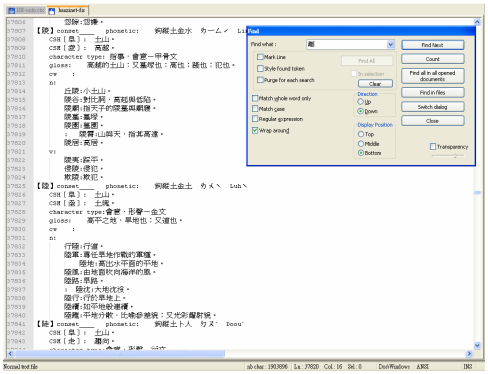


Figure 3: The character-stored lexicon: a snapshot

The second core component of the proposed resource is a set of hierarchically related *Top Concepts* called *Top-level Ontology* (or Upper ontology). This is similar to EuroWordnet 1.2, which is

[3]Since this lexicon aims at establishing an knowledge resource for modern Chinese NLP, characters and words are mostly extracted from the Academia Sinica Balanced Corpus of Modern Chinese (http://www.sinica.edu.tw/SinicaCorpus/), those characters and words which have probably only appeared in classical literary works, (considered *ghost words* in the lexicography), will be discarded.

[4]This is inspired by Chu (1999)'s works.

[5]The disputing point here is that, if some of the monosyllabic morphemes are taken as *words*, they should be very ambiguous in the daily linguistic context, at least more ambiguous than the dissyllabic words. However, as we argued previously, HanziNet takes a different perspective in locating theoretical roles of Hanzi.

[6]This distinction is made based on the glyphographical consideration, which has been a crucial topic in the studies of traditional Chinese scriptology. Due to the limited space, this will not be discussed here.

also enriched with the *Top Ontology* and the set of *Base Concepts* (Vossen 1998).

As mentioned, a tentative set of 309 *conset*, a kind of ontological categories in contrast with synset has been proposed [7], and over 5000 characters have been used as instances in populating the character ontology.

Methodologically, following the basic line of *OntoClear* approach (Guarino and Welty (2002)), we use *simple monotonic inheritance* in our ontology design, which means that each node inherits properties only from a single ancestor, and the inherited value cannot be overwritten at any point of the ontology. The decision to keep the relations to one single parent was made in order to guarantee that the structure would be able to grow indefinitely and still be manageable, i.e. that the transitive quality of the relations between the nodes would not degenerate with size. Figure 4 shows a snapshot of the character ontology.

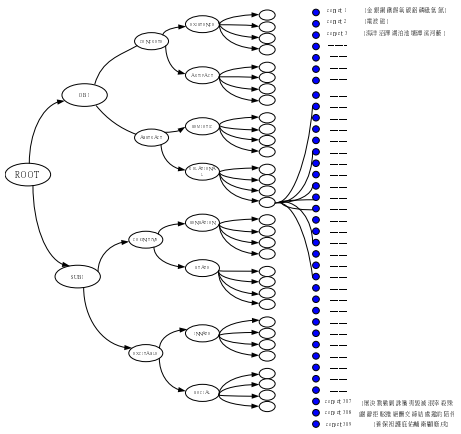


Figure 4: The character ontology: a snapshot

4.2 Characters in a Small World

In addition, an experiment concerning the *character network* that was based on the meaning aspects of characters, was performed from a statistical point of view. It was found that this character network, like many other linguistic semantic networks (such as WordNet), exhibits a *small-world* property (Watt 1998), characterized by sparse connectivity, small average shortest paths between characters, and strong local clustering. Moreover, due to its dynamic property, it appears to exhibit an asymptotic *scale-free* (Barabasi 1999) feature with the connectivity of power laws distribution, which is found in many other network systems as well.

[7] It would be interesting to compare consets with the basic 400 nodes in the upper region proposed by Hovy(2005).

Table 1: Statistical characteristics of the character network: $\mathcal{N}$ is the total number of nodes(characters), $\overline{k}$ is the average number of links per node, $\mathcal{C}$ is the clustering coefficient, and $\overline{L}$ is the average shortest-path length, and L_{max} is the maximum length of the shortest path between a pair of characters in the network.

	N	$\overline{k}$	$\mathcal{C}$	$\overline{\mathcal{L}}$
Actual configuration	6493	350	0.64	2.0
Random configuration	6493	350	0.06	1.5

Our first result is that our proposed conceptual network is highly clustered and at the same time and has a very small length, i.e., it is a small world model in the *static* aspect. Specifically, $\mathcal{L} \gtrsim \mathcal{L}_{random}$ but $\mathcal{C} \gg \mathcal{C}_{random}$. Results for the network of characters, and a comparison with a corresponding random network with the same parameters are shown in Table 1. $\mathcal{N}$ is the total number of nodes (characters), $\overline{k}$ is the average number of links per node, $\mathcal{C}$ is the clustering coefficient, and $\mathcal{L}$ is the average shortest path.

4.3 HanziNet in the Global Wordnet Grid

In order to promote a semantic and ontological interoperability, we have aligned conset with the 164 Base Concepts, a shared set of concepts from EWN in terms of Wordnet synsets and SUMO definitions, which has been currently proposed in the international collaborative platform of *Global Wordnet Grid*.

5 Applications and Future Development

5.1 Sense Prediction and Disambiguation

Based on the initial version of the proposed resources, Hsieh (2005b) has proposed a semantic class prediction model which aims to gain the possible semantic classes of unknown two-characters words. The results obtained shows that, with this knowledge resource, the system can achieve fairly high level of performance. Meaning relevant NLP Tasks such as Word Sense Disambiguation are also in preparation.

5.2 Interfacing Hantology, HanziNet and Chinese Wordnet

Interfacing ontologies and lexical resources has been a research topic in the coming age of semantic web. In the case of Chinese, three existing lexical resources (意符 Radicals::Hantology (Chou and Huang (2005))- 字 Characters::HanziNet - 詞 Words::Chinese Wordnet) constitutes an integrated 3-level knowledge scenario which would provide important insights into the problems of understanding the complexities and its interaction with Chinese natural language.

6 Conclusion

In conclusion, the goal of this research is set to survey the unique characteristics of Chinese Ideographs.

Though it has been well understood and agreed upon in cognitive linguistics that concepts can be represented in many ways, using various constructions at different syntactical levels, conceptual representation at the script level has been unfortunately both undervalued and under-represented in computational linguistics. Therefore, the Hanzi-driven conceptual approach in this thesis might require that we consider the Chinese writing system from a perspective that is not normally found in canonical treatments of writing systems in contemporary linguistics.

Against the deep-seated tradition in contemporary Chinese linguistics, which views the use of Chinese characters in scientific theories as a manifestation of mathematical immaturity and interpretational subjectivity, we propose the first lexical knowledge resource based on Chinese characters in the field of linguistic as well as in the NLP.

It is noted that HanziNet, as a general knowledge resource, should not claim to be a sufficient knowledge resource in and of itself, but instead seek to provide a groundwork for the incremental integration of other knowledge resources for language processing tasks. In order to augment HanziNet, additional information will needed to be incorporated and mapped into HanziNet. This leads us to several avenues of future research.

Acknowledgements

The authors would like to thank the anonymous referees for constructive comments. Thanks also go to the institute of linguistics of Academia Sinica for their kindly data support.

References

Aitchison, Jean. 2003. Words in the mind: an introduction to the mental lexicon. Blackwell publishing.

Barabasi, Albert-Laszlo and Reka Albert. 1999. Emergence of scaling in random networks. *Science*, 286:509-512.

Chou, Ya-Min and Chu-Ren Huang. 2005. Hantology: An ontology based on conventionalized conceptualization. *OntoLex Workshop*, Korea.

Chu, Bong-Foo. 1999-. http://www.cbflabs.com

Guarino, Nicola and Chris Welty. 2002. Evaluating ontological decisions with OntoClean. In: *Communications of the ACM*. 45(2):61-65

Hovy, E.H. 2005. Methodologies for the Reliable Construction of Ontological Knowledge. In : F. Dau, M.-L. Mugnier, and G. Stumme (eds), *Conceptual Structures: Common Semantics for Sharing Knowledge.* Proceedings of the 13th Annual International Conference on Conceptual Structures (ICCS 2005). Kassel, Germany.

Hsieh, Shu-Kai. 2005(a). HanziNet: An enriched conceptual network of Chinese characters. *The 5rd workshop on Chinese lexical semantics*, China: Xiamen.

Hsieh, Shu-Kai. 2005(b). Word Meaning Inducing via Character Ontology. *IJINLP, SIGHAN Workshop*, Jijeu Island, South Korea.

Packard, J. L. 2000. *The morphology of Chinese*. Cambridge, UK: Cambridge University Press.

Steyvers, M. and Tenenbaum, J.B. 2002 The Large-Scale Structure of Semantic Networks: Statistical Analyses and a Model of Semantic Growth. *Cognitive Science*.

Watts, D. J. and Strogatz, S. H. 1998. Collective dynamics of 'small-world' networks. *Nature* 393:440-42.

Yu, Shiwen, Zhu Xuefeng and Li Feng. 1999. The development and application of modern Chinese morpheme knowledge base.[in Chinese]. In: 世界漢語教學, No.2. pp38-45.

Spontaneous Speech Understanding for Robust Multi-Modal Human-Robot Communication

Sonja Hüwel, Britta Wrede
Faculty of Technology, Applied Computer Science
Bielefeld University, 33594 Bielefeld, Germany
shuewel,bwrede@techfak.uni-bielefeld.de

Abstract

This paper presents a speech understanding component for enabling robust situated human-robot communication. The aim is to gain semantic interpretations of utterances that serve as a basis for multi-modal dialog management also in cases where the recognized word-stream is not grammatically correct. For the understanding process, we designed semantic processable units, which are adapted to the domain of situated communication. Our framework supports the specific characteristics of spontaneous speech used in combination with gestures in a real world scenario. It also provides information about the dialog acts. Finally, we present a processing mechanism using these concept structures to generate the most likely semantic interpretation of the utterances and to evaluate the interpretation with respect to semantic coherence.

1 Introduction

Over the past years interest in mobile robot applications has increased. One aim is to allow for intuitive interaction with a personal robot which is based on the idea that people want to communicate in a natural way (Breazeal et al., 2004)(Dautenhahn, 2004). Although often people use speech as the main modality, they tend to revert to additional modalities such as gestures and mimics in face-to-face situations. Also, they refer to objects in the physical environment. Furthermore, speech, gestures and information of the environment are used in combination in instructions for the robot. When participants perceive a shared environment and act in it we call this communication "situated" (Milde et al., 1997). In addition to these features that are characteristic for situated communication, situated dialog systems have to deal with several problems caused by spontaneous speech phenomena like ellipses, indirect speech acts or incomplete sentences. Large pauses or breaks occur inside an utterance and people tend to correct themselves. Utterances often do not follow a standard grammar as written text.

Service robots have not only to be able to cope with this special kind of communication but they also have to cope with noise that is produced by their own actuators or the environment. Speech recognition in such scenarios is a complex and difficult task, leading to severe degradations of the recognition performance. The goal of this paper is to present a framework for human-robot interaction (HRI) that enables robust interpretation of utterances under the specific conditions in HRI.

2 Related Work

Some of the most explored speech processing systems are telephone-based information systems. Their design rather differs from that of situated HRI. They are uni-modal so that every information has to be gathered from speech. However, speech input is different as users utter longer phrases which are generally grammatically correct. These systems are often based on a large corpus and can therefore be well trained to perform satisfactory speech recognition results. A prominent example for this is the telephone based weather forecast information service JUPITER (Zue et al., 2000).

[1]This work has been supported by the European Union within the 'Cognitive Robot Companion' (COGNIRON) project (FP6-IST-002020) and by the German Research Foundation within the Graduate Program 'Task Oriented Communication'.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 391–398,
Sydney, July 2006.

Over the past years interest increased in mobile robot applications where the challenges are even more complex. While many of these problems (person tracking, attention, path finding) are already in the focus of research, robust speech understanding has not yet been extensively explored in the context of HRI. Moreover, interpretation of situated dialogs in combination with additional knowledge sources is rarely considered. Recent projects with related scope are the mobile robots CARL (Lopes et al., 2005) and ALBERT (Rogalla et al., 2002), and the robotic chandelier Elvis (Juster and Roy, 2004). The main task of the robot CARL is robust language understanding in context of knowledge acquisition and management. It combines deep and shallow parsing to achieve robustness. ALBERT is designed to understand speech commands in combination with gestures and object detection with the task to handle dishes. The home lighting robot Elvis gets instructions about lighting preferences of a user via speech and gestural input. The robot itself has a fixed position but the user may walk around in the entire room. It uses keyword spotting to analyze the semantic content of speech. As speech recognition in such robot scenarios is a complex and difficult task, in these systems the speech understanding analysis is constrained to a small set of commands and not oriented towards spontaneous speech. However, deep speech understanding is necessary for more complex human robot interaction.

There is only little research in semantic speech analysis of spontaneous speech. A widely used approach of interpreting sentences is the idea of case grammar (Bruce, 1975). Each verb has a set of named slots, that can be filled by other slots, typically nouns. Syntactic case information of words inside a sentence marks the semantic roles and thus, the corresponding slots can be filled. Another approach of processing spontaneous speech by using semantic information for the Air Travel Information Service (ATIS) task is implemented in the Phoenix system (Ward, 1994). Slots in frames represent the basic semantic entities known to the system. A parser using semantic grammars maps input onto these frame representations. The idea of our approach is similar to that of the Phoenix system, in that we also use semantic entities for extracting information. Much effort has been made in the field of parsing strategies combined with semantic information. These systems support preferably task oriented dialog systems, e.g., the ATIS task as in (Popescu et al., 2004) and (Milward, 2000), or virtual world scenarios (Gorniak and Roy, 2005), which do not have to deal with uncertain visual input. The aim of the FrameNet project (Fillmore and Baker, 2001) is to create a lexicon resource for English, where every entry receives a semantic frame description.

In contrast to other presented approaches we focus on deep semantic analysis of situated spontaneous speech.Written language applications have the advantage to be trainable on large corpora, which is not the case for situated speech based applications. And furthermore, interpretation of situated speech depends on environmental information. Utterances in this context are normally less complex, still our approach is based on a lexicon that allows a broad variety of utterances. It also takes speech recognition problems into account by ignoring non-consistent word hypotheses and scoring interpretations according to their semantic completeness. By adding pragmatic information, natural dialog processing is facilitated.

3 Situated Dialog Corpus

With our robot BIRON we want to improve social and functional behavior by enabling the system to carry out a more sophisticated dialog for handling instructions. One scenario is a home-tour where a user is supposed to show the robot around the home. Another scenario is a plant-watering task, where the robot is instructed to water different plants. There is only little research on multimodal HRI with speech-based robots. A study how users interact with mobile office robots is reported in (Hüttenrauch et al., 2003). However, in this evaluation, the integration of different modalities was not analyzed explicitly. But even though the subjects were not allowed to use speech and gestures in combination, the results support that people tended to communicate in a multi-modal way, nevertheless.

To receive more detailed information about the instructions that users are likely to give to an assistant in home or office we simulated this scenario and recorded 14 dialogs from German native speakers. Their task was to instruct the robot to water plants. Since our focus in this stage of the development of our system lies on the situatedness of the conversation, the robot was simply replaced by a human pretending to be a robot. The subjects

were asked to act as if it would be a robot. As proposed in (Lauriar et al., 2001), a preliminary user study is necessary to reduce the number of repair dialogs between user and system, such as queries. The corpus provides data necessary for the design of the dialog components for multi-modal interaction. We also determined the lexicon and obtained the SSUs that describe the scene and tasks for the robot.

The recorded dialogs feature the specific nature of dialog situations in multi-modal communication situations. The analysis of the corpus is presented in more detail in (Hüwel and Kummert, 2004). It confirms that spontaneously spoken utterances seldom respect the standard grammar and structure of written sentences. People tend to use short phrases or single words. Large pauses often occur during an utterance or the utterance is incomplete. More interestingly, the multi-modal data shows that 13 out of 14 persons used pointing gestures in the dialogs to refer to objects. Such utterances cannot be interpreted without additional information of the scene. For example, an utterance such as "this one" is used with a pointing gesture to an object in the environment. We realize, of course, that for more realistic behavior towards a robot a real experiment has to be performed. However this time- and resource-efficient procedure allowed us to build a system capable of facilitating situated communication with a robot. The implemented system has been evaluated with a real robot (see section 7). In the prior version we used German as language, now the dialog system has adapted to English.

4 The Robot Assistant BIRON

The aim of our project is to enable intuitive interaction between a human and a mobile robot. The basis for this project is the robot system BIRON (et. al, 2004). The robot is able to visually track persons and to detect and localize sound sources.

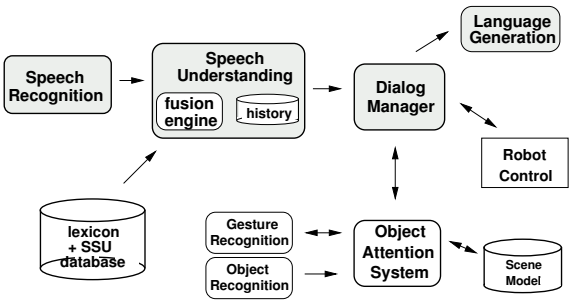


Figure 1: Overview of the BIRON dialog system architecture

The robot expresses its focus of attention by turning the camera into the direction of the person currently speaking. From the orientation of the person's head it is deduced whether the speaker addresses the robot or not. The main modality of the robot system is speech but the system can also detect gestures and objects. Figure 1 gives an overview of the architecture of BIRON's multi-modal interaction system. For the communication between these modules we use an XML based communication framework (Fritsch et al., 2005). In the following we will briefly outline the interacting modules of the entire dialog system with the speech understanding component.

Speech recognition: If the user addresses BIRON by looking in its direction and starting to speak, the speech recognition system starts to analyze the speech data. This means that once the attention system has detected that the user is probably addressing the robot it will route the speech signal to the speech recognizer. The end of the utterance is detected by a voice activation detector. Since both components can produce errors the speech signal sent to the recognizer may contain wrong or truncated parts of speech. The speech recognition itself is performed with an incremental speaker-independent system (Wachsmuth et al., 1998), based on Hidden Markov Models. It combines statistical and declarative language models to compute the most likely word chain.

Dialog manager: The dialog management serves as the interface between speech analysis and the robot control system. It also generates answers for the user. Thus, the speech analysis system transforms utterances with respect to gestural and scene information, such as pointing gestures or objects in the environment, into instructions for the robot. The dialog manager in our application is agent-based and enables a multi-modal, mixed ini-

tiative interaction style (Li et al., 2005). It is based on semantic entities which reflect the information the user uttered as well as discourse information based on speech-acts. The dialog system classifies this input into different categories as e.g., instruction, query or social interaction. For this purpose we use discourse segments proposed by Grosz and Sidner (Grosz and Sidner, 1986) to describe the kind of utterances during the interaction. Then the dialog manager can react appropriately if it knows whether the user asked a question or instructed the robot. As gesture and object detection in our scenario is not very reliable and time-consuming, the system needs verbal hints of scene information such as pointing gestures or object descriptions to gather information of the gesture detection and object attention system.

5 Situated Concept Representations

Based on the situated conversational data, we designed "situated semantic units" (SSUs) which are suitable for fast and automatic speech understanding. These SSUs basically establish a network of strong (mandatory) and weak (optional) relations of sematic concepts which represent world and discourse knowledge. They also provide ontological information and additional structures for the integration of other modalities. Our structures are inspired by the idea of frames which provide semantic relations between parts of sentences (Fillmore, 1976).

Till now, about 1300 lexical entries are stored in our database that are related to 150 SSUs. Both types are represented in form of XML structures. The lexicon and the concept database are based on our experimental data of situated communication (see section 3) and also on data of a home-tour scenario with a real robot. This data has been annotated by hand with the aim to provide an appropriate foundation for human-robot interaction. It is also planned to integrate more tasks for the robot as, e.g., courier service. This can be done by only adding new lexical entries and corresponding SSUs without spending much time in reorganization. Each lexical entry in our database contains a semantic association to the related SSUs. Therefore, equivalent lexical entries are provided for homonyms as they are associated to different concepts.

In figure 2 the SSU *Showing* has an open link to the SSUs *Actor* and *Object*. Missing links to

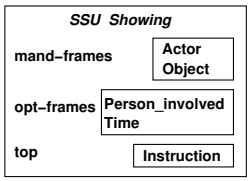


Figure 2: *Schematic SSU "Showing" for utterances like "I show you my poster tomorrow".*

strongly connected SSUs are interpreted as missing information and are thus indicators for the dialog management system to initiate a clarification question or to look for information already stored in the scene model (see fig. 1). The SSUs also have connections to optional arguments, but they are less important for the entire understanding process.

The SSUs also include ontological information, so that the relations between SSUs can be described as general as possible. For example, the SSU *Building_subpart* is a sub-category of *Object*. In our scenario this is important as for example the unit *Building_subpart* related to the concept"wall" has a fixed position and can be used as navigation-support in contrast to other objects. The top-category is stored in the entry *top*, a special item of the SSU. By the use of ontological information, SSUs also differentiate between task and communication related information and thereby support the strategy of the dialog manager to decouple task from communication structure. This is important in order to make the dialog system independent of the task and enable scalable interaction capabilities. For example the SSU *Showing* belongs to the discourse type *Instruction*. Other types important for our domain are *Socialization*, *Description*, *Confirmation*, *Negation*, *Correction*, and *Query*. Further types may be included, if necessary.

In our domain, missing information in an utterance can often be acquired from the scene. For example the utterance "look at this" and a pointing gesture to a table will be merged to the meaning "look at the table". To resolve this meaning, we use hints of co-verbal gestures in the utterance. Words as "this one" or "here" are linked to the SSU *Potential_gesture*, indicating a relation between speech and gesture. The timestamp of the utterance enables temporal alignment of speech and gesture. Since gesture recognition is expensive in computing time and often not well-defined, such linguistic hints can reduce these costs dra-

matically.

The utterance “that” can also represent an anaphora, and is analyzed in both ways, as anaphora and as gesture hint. Only if there is no gesture, the dialog manager will decide that the word probably was used in an anaphoric manner.

Since we focus on spontaneous speech, we cannot rely on the grammar, and therefore the semantic units serve as the connections between the words in an utterance. If there are open connections interpretable as missing information, it can be inferred what is missing and be integrated by the contextual knowledge. This structure makes it easy to merge the constituents of an utterance solely by semantic relations without additional knowledge of the syntactic properties. By this, we lose information that might be necessary in several cases for disambiguation of complex utterances. However, spontaneous speech is hard to parse especially since speech recognition errors often occur on syntactically relevant morphemes. We therefore neglect the cases which tend to occur very rarely in HRI scenarios.

6 Semantic Processing

In order to generate a semantic interpretation of an utterance, we use a special mechanism, which unifies words of an utterance into a single structure. The system also considers the ontological information of the SSUs to generate the most likely interpretation of the utterance. For this purpose, the mechanism first associates lexical entries of all words in the utterance with the corresponding SSUs. Then the system tries to link all SSUs together into one connected uniform. Some SSUs provide open links to other SSUs, which can be filled by semantic related SSUs. The SSU *Beside* for example provides an open link to *Object*. This SSU can be linked to all *Object* entities and to all subtypes of *Object*. Thus, an utterance as ”next to the door” can be linked together to form a single structure (see fig. 3). The SSUs which possess open links are central for this mechanism, they represent roots for parts of utterances. However, these units can be connected by other roots, likewise to generate a tree representing semantic relations inside an utterance.

The fusion mechanism computes in its best case in linear time and in worst case in square time. A scoring function underlies the mechanism: the more words can be combined, the better is the rating. The system finally chooses the structure with the highest score. Thus, it is possible to handle semantic variations of an utterance in parallel, such as homonyms. Additionally, the rating is helpful to decide whether the speech recognition result is reliable or not. In this case, the dialog manager can ask the user for clarification. In the next version we will use a more elaborate evaluation technique to yield better results such as rating the amount of concept-relations and missing relations, distinguish between important and optional relations, and prefer relations to words nearby.

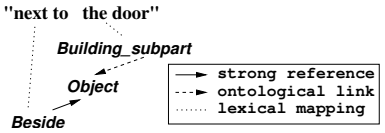


Figure 3: *Simplified parse tree example* .

A converter forwards the result of the mechanism as an XML-structure to the dialog manager. A segment of the result for the dialog manager is presented in Figure 4. With the category-descriptions the dialog-module can react fast on the user’s utterance without any further calculation. It uses them to create inquiries to the user or to send a command to the robot control system, such as “look for a gesture”, “look for a blue object”, or “follow person”. If the interpreted utterance does not fit to any category it gets the value *fragment*. These utterances are currently interpreted in the same way as partial understandings and the dialog manager asks the user to provide more meaningful information.

Figure 1 illustrates the entire architecture of the speech understanding system and its interfaces to other modules. The SSUs and the lexicon are stored in an external XML-databases. As the speech understanding module starts, it first reads these databases and converts them into internal data-structures stored in a fast accessible hash table. As soon as the module receives results from speech recognition, it starts to merge. The mechanism also uses a *history*, where former parts of utterances are stored and which are also integrated in the fusing mechanism. The speech understanding system then converts the best scored result into a semantic XML-structure (see Figure 4) for the dialog manager.

```
<metaInfo>
<time>1125573609635</time>
<status>full</status>
</metaInfo>
<semanticInfo>
<u>what can you do</u>
<category>query</category>
<content>
  <unit = Question_action>
     <name>what</name>
     <unit = Action>
        <name>do</name>
        <unit = Ability>
           <name>can</name>
           <unit = Proxy>
              <name>you</name>
        ...

<u>this is a green cup</u>
<category>description</category>
<content>
  <unit = Existence>
     <name>is</name>
     <unit = Object_kitchen>
        <name>cup</name>
        <unit = Potential_gesture>
           <name>this</name>
        </unit>
        <unit = Color>
           <name>green</name>
        </unit>
      ...
```

Figure 4: Two segments of the speech understanding results for the utterances *"what can you do" and "this is a green cup"*.

6.1 Situated Speech Processing

Our approach has various advantages dealing with spontaneous speech. Double uttered words as in the utterance "look - look here" are ignored in our approach. The system still can interprete the utterance, then only one word is linked to the other words. Corrections inside an utterance as "the left em right cube" are handled similar. The system generates two interpretations of the utterance, the one containing left the other right. The system chooses the last one, since we assume that corrections occur later in time and therefore more to the right. The system deals with pauses inside utterances by integrating former parts of utterances stored in the *history*. The mechanism also processes incomplete or syntactic incorrect utterances. To prevent sending wrong interpretations to the dialog-manager the scoring function rates the quality of the interpretation as described above. In our system we also use scene information to evaluate the entire correctness so that we do not only have to rely on the speech input. In case of doubt the dialog-manager requests to the user.

For future work it is planned to integrate additional information sources, e.g., inquiries of the dialog manager to the user. The module will also store these information in the *history* which will be used for anaphora resolution and can also be used to verify the output of the speech recognition.

```
User1: Robot look - do you see?
       This - is a cow. Funny.
       Do you like it? ...
User2: Look here robot - a cup.
       Look here a - a keyboard.
       Let's try that one. ...
User3: Can you walk in this room?
       Sorry, can you repeat your answer?
       How fast can you move? ...
```

Figure 5: Excerptions of the utterances during the experiment setting.

7 Evaluation

For the evaluation of the entire robot system BIRON we recruited 14 naive user between 12 and 37 years with the goal to test the intuitiveness and the robustness of all system modules as well as its performance. Therefore, in the first of two runs the users were asked to familiarize themselves with the robot without any further information of the system. In the second run the users were given more information about technical details of BIRON (such as its limited vocabulary). We observed similar effects as described in section 2. In average, one utterance contained 3.23 words indicating that the users are more likely to utter short phrases. They also tend to pause in the middle of an utterance and they often uttered so called meta-comments such as "that"s fine". In figure 5 some excerptions of the dialogs during the experiment settings are presented.

Thus, not surprisingly the speech recognition error rate in the first run was 60% which decreased in the second run to 42%, with an average of 52%. High error rate seems to be a general problem in settings with spontaneous speech as other systems also observed this problem (see also (Gorniak and Roy, 2005)). But even in such a restricted experiment setting, speech understanding will have to deal with speech recognition error which can never be avoided.

In order to address the two questions of (1) how well our approach of automatic speech understanding (ASU) can deal with automatic speech recognition (ASR) errors and (2) how its performance compares to syntactic analysis, we performed two analyses. In order to answer question (1) we compared the results from the semantic analysis based on the real speech recognition re-

sults with an accuracy of 52% with those based on the really uttered words as transcribed manually, thus simulating a recognition rate of 100%. In total, the semantic speech processing received 1642 utterances from the speech recognition system. From these utterances 418 utterances were randomly chosen for manual transcription and syntactic analysis. All 1642 utterances were processed and performed on a standard PC with an average processing time of 20ms, which fully fulfills the requirements of real-time applications. As shown in Table 1 39% of the results were rated as complete or partial misunderstandings and 61% as correct utterances with full semantic meaning. Only 4% of the utterances which were correctly recognized were misinterpreted or refused by the speech understanding system. Most errors occurred due to missing words in the lexicon.

Thus, the performance of the speech understanding system (ASU) decreases to the same degree as that of the speech recognition system (ASR): with a 50% ASR recognition rate the number of non-interpretable utterances is doubled indicating a linear relationship between ASR and ASU.

For the second question we performed a manual classification of the utterances into syntactically *correct* (and thus parseable by a standard parsing algorithm) and *not-correct*. Utterances following the English standard grammar (e.g. imperative, descriptive, interrogative) or containing a single word or an NP, as to be expected in answers, were classified as correct. Incomplete utterances or utterances with a non-standard structure (as occurred often in the baby-talk style utterances) were rated as not-correct. In detail, 58 utterances were either truncated at the end or beginning due to errors of the attention system, resulting in utterances such as "where is", "can you find", or "is a cube". These utterances also include instances where users interrupted themselves. In 51 utterances we found words missing in our lexicon database. 314 utterances where syntactically correct, whereas in 28 of these utterances a lexicon entry is missing in the system and therefore would lead to a failure of the parsing mechanism. 104 utterances have been classified as syntactically not-correct.

	ASR=100%	ASR=52%
ASU not or part. interpret.	15%	39%
ASU fully interpretable	84%	61%

Table 1: Semantic Processing results based on different word recognition accuracies.

In contrast, the result from our mechanism performed significantly better. Our system was able to interprete 352 utterances and generate a full semantic interpretation, whereas 66 utterances could only be partially interpreted or were marked as not interpretable. 21 interpretations of the utterances were semantically incorrect (labeled from the system wrongly as correct) or were not assigned to the correct speech act, e.g., "okay" was assigned to no speech act (*fragment*) instead to *confirmation*. Missing lexicon entries often lead to partial interpretations (20 times) or sometimes to complete misinterpretations (8 times). But still in many cases the system was able to interprete the utterance correctly (23 times). For example "can you go for a walk with me" was interpreted as "can you go with me" only ignoring the unknown "for a walk".The utterance "can you come closer" was interpreted as a partial understanding "can you come" (ignoring the unknown word "closer"). The results are summarized in Table 2.

As can be seen the semantic error rate with 15% non-interpretable utterances is just half of the syntactic correctness with 31%. This indicates that the semantic analysis can recover about half of the information that would not be recoverable from syntactic analysis.

	ASU		Synt. cor.
not or part. interpret.	15%	not-correct	31%
fully interpret.	84%	correct	68%

Table 2: Comparison of semantic processing result with syntactic correctness based on a 100% word recognition rate.

8 Conclusion and Outlook

In this paper we have presented a new approach of robust speech understanding for mobile robot assistants. It takes into account the special characteristics of situated communication and also the difficulty for the speech recognition to process utterances correctly. We use special concept structures for situated communication combined with an automatic fusion mechanism to generate semantic structures which are necessary for the dialog manager of the robot system in order to respond adequately.

This mechanism combined with the use of our

SSUs has several benefits. First, speech is interpreted even if speech recognition does not always guarantee correct results and speech input is not always grammatically correct. Secondly, the speech understanding component incorporates information about gestures and references to the environment. Furthermore, the mechanism itself is domain-independent. Both, concepts and lexicon can be exchanged in context of a different domain.

This semantic analysis already produces elaborated interpretations of utterances in a fast way and furthermore, helps to improve robustness of the entire speech processing system. Nevertheless, we can improve the system. In our next phase we will use a more elaborate scoring function technique and use the correlations of mandatory and optional links to other concepts to perform a better evaluation and also to help the dialog manager to find clues for missing information both in speech and scene. We will also use the evaluation results to improve the SSUs to get better results for the semantic interpretation.

References

C. Breazeal, A. Brooks, J. Gray, G. Hoffman, C. Kidd, H. Lee, J. Lieberman, A. Lockerd, and D. Mulanda. 2004. Humanoid robots as cooperative partners for people. *Int. Journal of Humanoid Robots*.

B. Bruce. 1975. Case systems for natural language. *Artificial Intelligence*, 6:327–360.

K. Dautenhahn. 2004. Robots we like to live with?! - a developmental perspective on a personalized, lifelong robot companion. In *Proc. Int. Workshop on Robot and Human Interactive Communication (RO-MAN)*.

A. Haasch et. al. 2004. BIRON – The Bielefeld Robot Companion. In E. Prassler, G. Lawitzky, P. Fiorini, and M. Hägele, editors, *Proc. Int. Workshop on Advances in Service Robotics*, pages 27–32. Fraunhofer IRB Verlag.

C. J. Fillmore and C. F. Baker. 2001. Frame semantics for text understanding. In *Proc. of WordNet and Other Lexical Recources Workshop*. NACCL.

C. J. Fillmore. 1976. Frame semantics and the nature of language. In *Annals of the New York Academy of Sciences: Conf. on the Origin and Development of Language and Speech*, volume 280, pages 20–32.

J. Fritsch, M. Kleinehagenbrock, A. Haasch, S. Wrede, and G. Sagerer. 2005. A flexible infrastructure for the development of a robot companion with extensible HRI-capabilities. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 3419–3425.

P. Gorniak and D. Roy. 2005. Probabilistic Grounding of Situated Speech using Plan Recognition and Reference Resolution. In *ICMI*. ACM Press.

B. J. Grosz and C. L. Sidner. 1986. Attention, intention, and the structure of discourse. *Computational Linguistics*, 12(3):175–204.

H. Hüttenrauch, A. Green, K. Severinson-Eklundh, L. Oestreicher, and M. Norman. 2003. Involving users in the design of a mobile office robot. *IEEE Transactions on Systems, Man and Cybernetics, Part C*.

S. Hüwel and F. Kummert. 2004. Interpretation of situated human-robot dialogues. In *Proc. of the 7th Annual CLUK*, pages 120–125.

J. Juster and D. Roy. 2004. Elvis: Situated Speech and Gesture Understanding for a Robotic Chandelier. In *Proc. Int. Conf. Multimodal Interfaces*.

S. Lauriar, G. Bugmann, T. Kyriacou, J. Bos, and E. Klein. 2001. Personal robot training via natural language instructions. *IEEE Intelligent Systems, 16:3*, pages 38–45.

S. Li, A. Haasch, B. Wrede, J. Fritsch, and G. Sagerer. 2005. Human-style interaction with a robot for cooperative learning of scene objects. In *Proc. Int. Conf. on Multimodal Interfaces*.

L. Seabra Lopes, A. Teixeira, M. Quindere, and M. Rodrigues. 2005. From robust spoken language understanding to knowledge aquisition and management. In *EUROSPEECH 2005*.

J. T. Milde, K. Peters, and S. Strippgen. 1997. Situated communication with robots. In *First Int. Workshop on Human-Computer-Conversation*.

D. Milward. 2000. Distributing representation for robust interpretation of dialogue utterances. In *ACL*.

A.-M. Popescu, A. Armanasu, O. Etzioni, D. Ko, and A. Yates. 2004. Modern natural language interfaces to databases: Composing statistical parsing with semantic tractability. In *Proc. of COLING*.

O. Rogalla, M. Ehrenmann, R. Zöllner, R. Becher, and R. Dillmann. 2002. Using gesture and speech control for commanding a robot assistant. In *Proc. of the 11th IEEE Int. Workshop on Robot and Human interactive Communication*, pages 454–459. ROMAN.

S. Wachsmuth, G. A. Fink, and G. Sagerer. 1998. Integration of parsing and incremental speech recognition. In *EUSIPCO*, volume 1, pages 371–375.

W. Ward. 1994. Extracting Information From Spontaneous Speech. In *ICRA*, pages 83–86. IEEE Press.

V. Zue, S. Seneff, J. Glass, J. Polifronti, C. Pao, T. J. Hazen, and L. Hetherington. 2000. JUPITER: A telephone-based conversational interface for weather information. *IEEE Transactions on Speech and Audio Processing*, pages 100–112, January.

Efficient sentence retrieval based on syntactic structure

Ichikawa Hiroshi, Hakoda Keita, Hashimoto Taiichi and **Tokunaga Takenobu**
Department of Computer Science, Tokyo Institute of Technology
{ichikawa,hokoda,taiichi,take}@cl.cs.titech.ac.jp

Abstract

This paper proposes an efficient method of sentence retrieval based on syntactic structure. Collins proposed Tree Kernel to calculate structural similarity. However, structual retrieval based on Tree Kernel is not practicable because the size of the index table by Tree Kernel becomes impractical. We propose more efficient algorithms approximating Tree Kernel: Tree Overlapping and Subpath Set. These algorithms are more efficient than Tree Kernel because indexing is possible with practical computation resources. The results of the experiments comparing these three algorithms showed that structural retrieval with Tree Overlapping and Subpath Set were faster than that with Tree Kernel by 100 times and 1,000 times respectively.

1 Introduction

Retrieving similar sentences has attracted much attention in recent years, and several methods have been already proposed. They are useful for many applications such as information retrieval and machine translation. Most of the methods are based on frequencies of surface information such as words and parts of speech. These methods might work well concerning similarity of topics or contents of sentences. Although the surface information of two sentences is similar, their syntactic structures can be completely different (Figure 1). If a translation system regards these sentences as similar, the translation would fail. This is because conventional retrieval techniques exploit only similarity of surface information such as words and parts-of-speech, but not more abstract information such as syntactic structures.

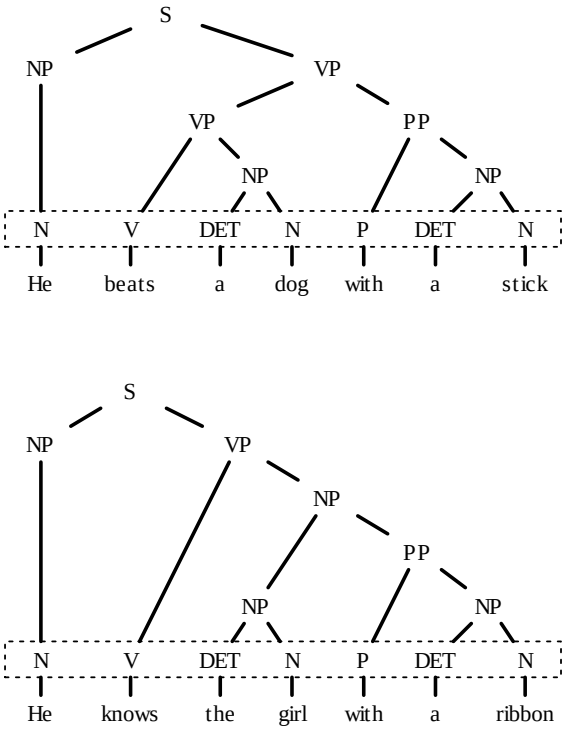


Figure 1: Sentences similar in appearance but differ in syntactic structure

Collins *et al.* (Collins, 2001a; Collins, 2001b) proposed Tree Kernel, a method to calculate a similarity between syntactic structures. Tree Kernel defines the similarity between two syntactic structures as the number of shared subtrees. Retrieving similar sentences in a huge corpus requires calculating the similarity between a given query and each of sentences in the corpus. Building an index table in advance could improve retrieval efficiency, but indexing with Tree Kernel is impractical due to the size of its index table.

In this paper, we propose two efficient algo-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 399–406,
Sydney, July 2006.

rithms to calculate similarity of syntactic structures: Tree Overlapping and Subpath Set. These algorithms are more efficient than Tree Kernel because it is possible to make an index table in reasonable size. The experiments comparing these three algorithms showed that Tree Overlapping is 100 times faster and Subpath Set is 1,000 times faster than Tree Kernel when being used for structural retrieval.

After briefly reviewing Tree Kernel in section 2, in what follows, we describe two algorithms in section 3 and 4. Section 5 describes experiments to compare these three algorithms and discussion on the results. Finally, we conclude the paper and look at the future direction of our research in section 6.

2 Tree Kernel

2.1 Definition of similarity

Tree Kernel is proposed by Collins *et al.* (Collins, 2001a; Collins, 2001b) as a method to calculate similarity between tree structures. Tree Kernel defines similarity between two trees as the number of shared subtrees. Subtree S of tree T is defined as any tree subsumed by T, and consisting of more than one node, and all child nodes are included if any.

Tree Kernel is not always suitable because the desired properties of similarity are different depending on applications. Takahashi *et al.* proposed three types of similarity based on Tree Kernel (Takahashi, 2002). We use one of the similarity measures (equation (1)) proposed by Takahashi *et al.*

$$K_C(T_1, T_2) = \max_{n_1 \in N_1,\ n_2 \in N_2} C(n_1, n_2) \qquad (1)$$

where $C(n_1, n_2)$ is the number of shared subtrees by two trees rooted at nodes n_1 and n_2.

2.2 Algorithm to calculate similarity

Collins *et al.* (Collins, 2001a; Collins, 2001b) proposed an efficient method to calculate Tree Kernel by using $C(n_1, n_2)$ as follows.

- If the productions at n_1 and n_2 are different $C(n_1, n_2) = 0$
- If the productions at n_1 and n_2 are the same, and n_1 and n_2 are pre-terminals, then $C(n_1, n_2) = 1$
- Else if the productions at n_1 and n_2 are the same and n_1 and n_2 are not pre-terminals,

$$C(n_1, n_2) = \prod_{i=1}^{nc(n_1)} (1 + C(ch(n_1, i), ch(n_2, i))) \qquad (2)$$

where $nc(n)$ is the number of children of node n and $ch(n, i)$ is the i'th child node of n. Equation (2) recursively calculates C on its child node, and calculating Cs in postorder avoids recalculation. Thus, the time complexity of $K_C(T_1, T_2)$ is $O(mn)$, where m and n are the numbers of nodes in T_1 and T_2 respectively.

2.3 Algorithm to retrieve sentences

Neither Collins nor Takahashi discussed retrieval algorithms using Tree Kernel. We use the following simple algorithm. First we calculate the similarity $K_C(T_1, T_2)$ between a query tree and every tree in the corpus and rank them in descending order of K_C.

Tree Kernel exploits all subtrees shared by trees. Therefore, it requires considerable amount of time in retrieval because similarity calculation must be performed for every pair of trees. To improve retrieval time, an index table can be used in general. However, indexing by all subtrees is difficult because a tree often includes millions of subtrees. For example, one sentence in Titech Corpus (Noro et al., 2005) with 22 words and 87 nodes includes 8,213,574,246 subtrees. The number of subtrees in a tree with N nodes is bounded above by 2^N.

3 Tree Overlapping

3.1 Definition of similarity

When putting an arbitrary node n_1 of tree T_1 on node n_2 of tree T_2, there might be the same production rule overlapping in T_1 and T_2. We define $C_{TO}(n_1, n_2)$ as the number of such overlapping production rules when n_1 overlaps n_2 (Figure 2).

We will define $C_{TO}(n_1, n_2)$ more precisely. First we define $L(n_1, n_2)$ of node n_1 of T_1 and node n_2 of T_2. $L(n_1, n_2)$ represents a set of pairs of nodes which overlap each other when putting n_1 on n_2. For example in Figure 2, $L(b_1^1, b_1^2) = \{(b_1^1, b_1^2), (d_1^1, d_1^2), (e_1^1, e_1^2), (g_1^1, g_1^2), (i_1^1, j_1^2)\}$.
$L(n_1, n_2)$ is defined as follows. Here n_i and m_i are nodes of tree T_i, $ch(n, i)$ is the i'th child of node n.

1. $(n_1, n_2) \in L(n_1, n_2)$

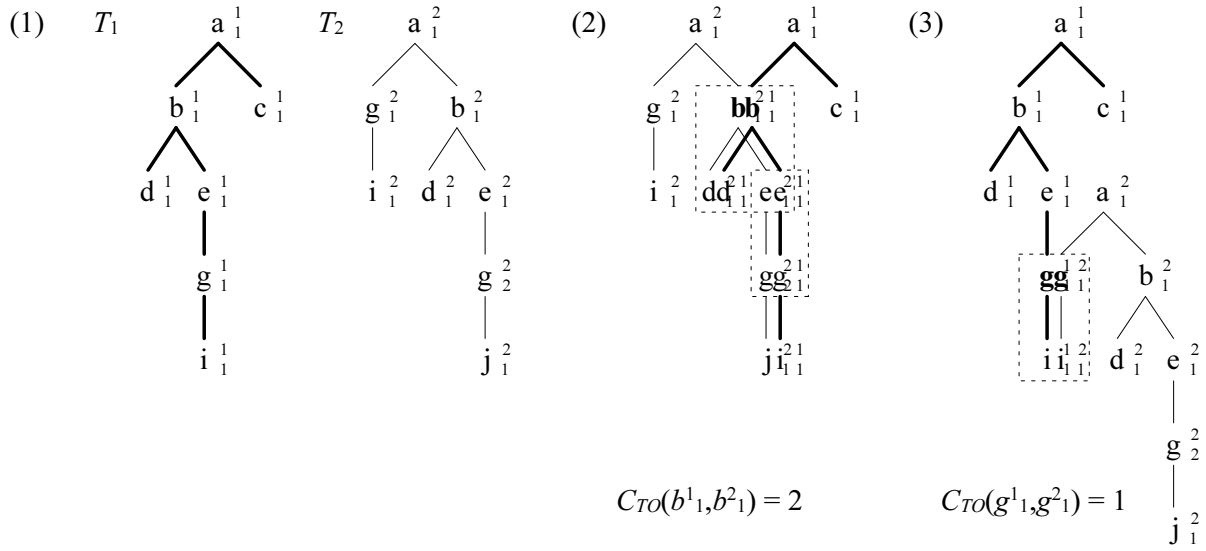


Figure 2: Example of similarity calculation

2. If $(m_1, m_2) \in L(n_1, n_2)$, $(ch(m_1, i), ch(m_2, i)) \in L(n_1, n_2)$

3. If $(ch(m_1, i), ch(m_2, i)) \in L(n_1, n_2)$, $(m_1, m_2) \in L(n_1, n_2)$

4. $L(n_1, n_2)$ includes only pairs generated by applying 2. and 3. recursively.

$C_{TO}(n_1, n_2)$ is defined by using $L(n_1, n_2)$ as follows.

$$C_{TO}(n_1, n_2) = \left| \left\{ (m_1, m_2) \;\middle|\; \begin{array}{l} m_1 \in NT(T_1) \\ \wedge\, m_2 \in NT(T_2) \\ \wedge\, (m_1, m_2) \in L(n_1, n_2) \\ \wedge\, PR(m_1) = PR(m_2) \end{array} \right\} \right|, \quad (3)$$

where $NT(T)$ is a set of nonterminal nodes in tree T, $PR(n)$ is a production rule rooted at node n.

Tree Overlapping similarity $S_{TO}(T_1, T_2)$ is defined as follows by using $C_{TO}(n_1, n_2)$.

$$S_{TO}(T_1, T_2) = \max_{n_1 \in NT(T_1)\ n_2 \in NT(T_2)} C_{TO}(n_1, n_2) \quad (4)$$

This formula corresponds to equation (1) of Tree Kernel.

As an example, we calculate $S_{TO}(T_1, T_2)$ in Figure 2 (1). Putting b_1^1 on b_1^2 gives Figure 2 (2) in which two production rules $b \rightarrow d\ e$ and $e \rightarrow g$ overlap respectively. Thus, $C_{TO}(b_1^1, b_1^2)$ becomes 2. While overlapping g_1^1 and g_1^2 gives Figure 2 (3) in which only one production rule $g \rightarrow i$ overlaps. Thus, $C_{TO}(g_1^1, g_1^2)$ becomes 1. Since there are no other node pairs which gives larger C_{TO} than 2, $S_{TO}(T_1, T_2)$ becomes 2.

Table 1: Example of the index table

p	$I[p]$
$a \rightarrow b\ c$	$\{a_1^1\}$
$b \rightarrow d\ e$	$\{b_1^1, b_1^2\}$
$e \rightarrow g$	$\{e_1^1, e_1^2\}$
$g \rightarrow i$	$\{g_1^1, g_1^2\}$
$a \rightarrow g\ b$	$\{a_1^2\}$
$g \rightarrow j$	$\{g_1^2\}$

3.2 Algorithm

Let us take an example in Figure 3 to explain the algorithm. Suppose that T_0 is a query tree and the corpus has only two trees, T_1 and T_2.

The method to find the most similar tree to a given query tree is basically the same as Tree Kernel's (section 2.2). However, unlike Tree Kernel, Tree Overlapping-based retrieval can be accelerated by indexing the corpus in advance. Thus, given a tree corpus, we build an index table $I[p]$ which maps a production rule p to its occurrences. Occurrences of production rules are represented by their left-hand side symbols, and are distinguished with respect to trees including the rule and

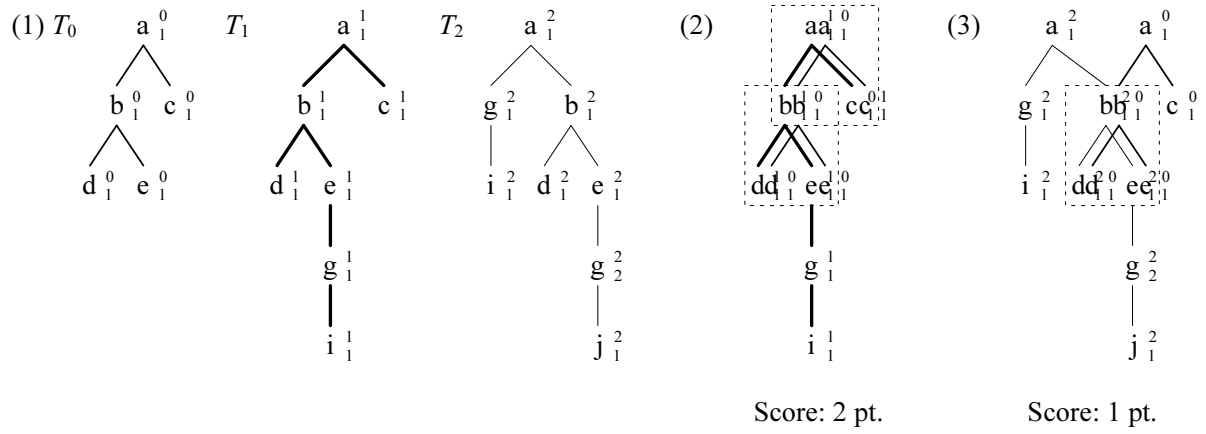


Figure 3: Example of Tree Overlapping-based retrieval

the position in the tree. $I[p]$ is defined as follows.

$$I[p] = \left\{ m \;\middle|\; \begin{array}{l} T \in F \\ \wedge\, m \in NT(T) \\ \wedge\, p = PR(m) \end{array} \right\} \qquad (5)$$

where F is the corpus (here $\{T_1, T_2\}$) and the meaning of other symbols is the same as the definition of C_{TO} (equation (3)).

Table 1 shows an example of the index table generated from T_1 and T_2 in Figure 3 (1). In Table 1, a superscript of a nonterminal symbol identifies a tree, and a subscript identifies a position in the tree.

By using the index table, we calculate $C[n, m]$ with the following algorithm.

for all (n, m) **do** $C[n, m] := 0$ **end**
foreach n **in** $NT(T_0)$ **do**
  foreach m **in** $I[PR(n)]$ **do**
    $(n', m') := top(n, m)$
    $C[n', m'] := C[n', m'] + 1$
  end
end

where $top(n, m)$ returns the upper-most pair of overlapped nodes when node n and m overlap. The value of top uniquely identifies a situation of overlapping two trees. Function $top(n, m)$ is calculated by the following algorithm.

function $top(n, m)$;
begin
  $(n', m') := (n, m)$
  while $order(n') = order(m')$ **do**
    $n' := parent(n')$
    $m' := parent(m')$
  end
  return (n', m')
end

where $parent(n)$ is the parent node of n, and $order(n)$ is the order of node n among its siblings. Table 2 shows example values of $top(n, m)$ generated by overlapping T_0 and T_1 in Figure 3. Note that top maps every pair of corresponding nodes in a certain overlapping situation to a pair of the upper-most nodes of that situation. This enables us to use the value of top as an identifier of a situation of overlap.

Table 2: Examples of $top(n, m)$

(n, m)	$top(n, m)$
(a_1^0, a_1^1)	(a_1^0, a_1^1)
(b_1^0, b_1^1)	(a_1^0, a_1^1)
(c_1^0, c_1^1)	(a_1^0, a_1^1)

Now $C[top(n, m)] = C_{TO}(n, m)$, therefore the tree similarity between a query tree T_0 and each tree T in the corpus $S_{TO}(T_0, T)$can be calculated by:

$$S_{TO}(T_0, T) = \max_{n \in NT(T_0),\, m \in NT(T)} C[top(n, m)] \qquad (6)$$

3.3 Comparison with Tree Kernel

The value of $S_{TO}(T_1, T_2)$ roughly corresponds to the number of production rules included in the largest sub-tree shared by T_1 and T_2. Therefore, this value represents the size of the subtree shared

by both trees, like Tree Kernel's K_C, though the definition of the subtree size is different.

One difference is that Tree Overlapping considers shared subtrees even though they are split by a nonshared node as shown in Figure 4. In Figure 4, T_1 and T_2 share two subtrees rooted at b and c, but their parent nodes are not identical. While Tree Kernel does not consider the superposition putting node a on h, Tree Overlapping considers putting a on h and assigns count 2 to this superposition.

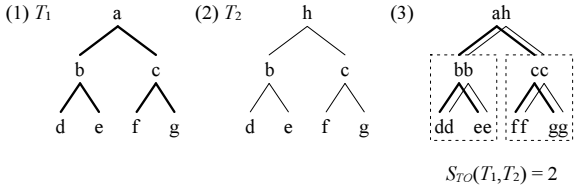


Figure 4: Example of counting two separated shared subtrees as one

Another, more important, difference is that Tree Overlapping retrieval can be accelerated by indexing the corpus in advance. The number of indexes is bounded above by the number of production rules, which is within a practical index size.

4 Subpath Set

4.1 Definition of similarity

Subpath Set similarity between two trees is defined as the number of subpaths shared by the trees. Given a tree, its subpaths is defined as a set of every path from the root node to leaves and their partial paths.

Figure 5 (2) shows all subpaths in T_1 and T_2 in Figure 5(1). Here we denotes a path as a sequence of node names such as (a, b, d). Therefore, Subpath Set similarity of T_1 and T_2 becomes 15.

4.2 Algorithm

Suppose T_0 is a query tree, TS is a set of trees in the corpus and $P(T)$ is a set of subpaths of T. We can build an index table $I[p]$ for each production rule p as follows.

$$I[p] = \{T | T \in TS \wedge p \in P(T)\} \qquad (7)$$

Using the index table, we can calculate the number of shared subpaths by T_0 and T, $S[T]$, by the following algorithm:

for all T $S[T] := 0$;
foreach p **in** $P(T_0)$ **do**
  foreach T **in** $I[p]$ **do**
    $S[T] := S[T] + 1$
  end
end

4.3 Comparison with Tree Kernel

As well as Tree Overlapping, Subpath Set retrieval can be accelerated by indexing the corpus. The number of indexes is bounded above by $L \times D^2$ where L is the maximum number of leaves of trees (the number of words in a sentence) and D is the maximum depth of syntactic trees. Moreover, considering a subpath as an index term, we can use existing retrieval tools.

Subpath Set uses less structural information than Tree Kernel and Tree Overlapping. It does not distinguish the order and number of child nodes. Therefore, the retrieval result tends to be noisy. However, Subpath Set is faster than Tree Overlapping, because the algorithm is simpler.

5 Experiments

This section describes the experiments which were conducted to compare the performance of structure retrieval based on Tree Kernel, Tree Overlapping and Subpath Set.

5.1 Data

We conducted two experiments using different annotated corpora. Titech corpus (Noro et al., 2005) consists of about 20,000 sentences of Japanese newspaper articles (Mainiti Shimbun). Each sentence has been syntactically annotated by hand. Due to the limitation of computational resources, we used randomly selected 2,483 sentences as a data collection.

Iwanami dictionary (Nishio et al., 1994) is a Japanese dictionary. We extracted 57,982 sentences from glosses in the dictionary. Each sentences was analyzed with a morphological analyzer, ChaSen (Asahara et al., 1996) and the MSLR parser (Shirai et al., 2000) to obtain syntactic structure candidates. The most probable structure with respect to PGLR model (Inui et al., 1996) was selected from the output of the parser. Since they were not investigated manually, some sentences might have been assigned incorrect structures.

5.2 Method

We conducted two experiments Experiment I and Experiment II with different corpora. The queries

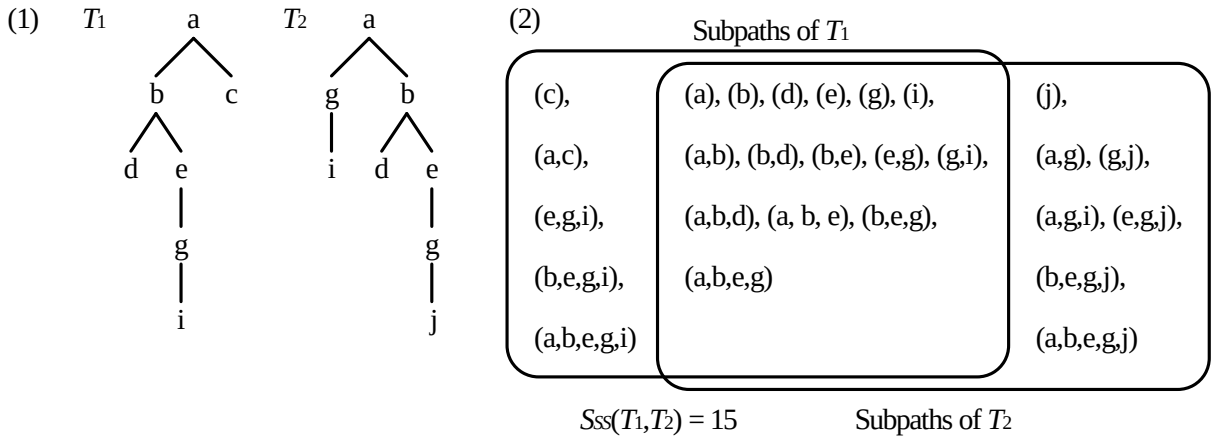


Figure 5: Example of subpaths

were extracted from these corpora. The algorithms described in the preceding sections were implemented with Ruby 1.8.2. Table 3 outlines the experiments.

Table 3: Summary of experiments

Experiment	I	II
Target corpus	Titech Corpus	Iwanami dict.
Corpus size	2,483 sent.	57,982 sent.
No. of queries	100	1,000
CPU	Intel Xeon (2.4GHz)	PowerPC G5 (2.3GHz)
Memory	2GB	2GB

5.3 Results and discussion

Since we select a query from the target corpus, the query is always ranked in the first place in the retrieval result. In what follows, we exclude the query tree as an answer from the result.

We evaluated the algorithms based on the following two factors: average retrieval time (CPU time) (Table 4) and the rank of the tree which was top-ranked in other algorithm (Table 5). For example, in Experiment I of Table 5, the column "$\geq$5th" of the row "TO/TK" means that there were 73 % of the cases in which the top-ranked tree by Tree Kernel (TK) was ranked 5th or above by Tree Overlapping (TO).

We consider Tree Kernel (TK) as the baseline method because it is a well-known existing similarity measure and exploits more information than others. Table 4 shows that in both corpora, the retrieval speed of Tree Overlapping (TO) is about 100 times faster than that of Tree Kernel, and the retrieval speed of Subpath Set (SS) is about 1,000 times faster than that of Tree Kernel. This results show we have successfully accelerated the retrieval speed.

Table 4: Average retrieval time per query [sec]

Algorithm	Experiment I	Experiment II
TK	529.42	3796.1
TO	6.29	38.3
SS	0.47	5.1

The retrieval time of Tree Overlapping, 6.29 and 38.3 sec./per query, seems be a bit long. However, we can shorten this time if we tune the implementation by using a compiler-type language. Note that the current implementation uses Ruby, an interpreter-type language.

Comparing Tree Overlapping and Subpath Set with respect to Tree Kernel (see rows "TK/TO" and "TK/SS"), the top-ranked trees by Tree Kernel are ranked in higher places by Tree Overlapping than by Subpath Set. This means Tree Overlapping is better than Subpath Set in approximating Tree Kernel.

Although the corpus of Experiment II is 20 times larger than that of Experiment I, the figures of Experiment II is better than that of Experiment I in Table 5. This could be explained as follows. In Experiment II, we used sentences from glosses in the dictionary, which tend to be formulaic and short. Therefore we could find similar sentences easier than in Experiment I.

To summarize the results, when being used in

Table 5: The rank of the top-ranked tree by other algorithm [%]

Experiment I

A/B	1st	$\geq$ 5th	$\geq$ 10th
TO/TK	34.0	73.0	82.0
SS/TK	16.0	35.0	45.0
TK/TO	29.0	41.0	51.0
SS/TO	27.0	49.0	58.0
TK/SS	17.0	29.0	37.0
TO/SS	29.0	58.0	69.0

Experiment II

A/B	1st	$\geq$ 5th	$\geq$ 10th
TO/TK	74.6	88.0	92.0
SS/TK	65.3	78.8	84.1
TK/TO	71.1	81.0	84.6
SS/TO	73.4	86.0	89.8
TK/SS	65.5	75.9	79.7
TO/SS	76.1	87.7	92.0

similarity calculation of tree structure retrieval, Tree Overlapping approximates Tree Kernel better than Subpath Set, while Subpath Set is faster than Tree Overlapping.

6 Conclusion

We proposed two fast algorithms to retrieve sentences which have a similar syntactic structure: Tree Overlapping (TO) and Subpath Set (SS). And we compared them with Tree Kernel (TK) to obtain the following results.

- Tree Overlapping-based retrieval outputs similar results to Tree Kernel-based retrieval and is 100 times faster than Tree Kernel-based retrieval.
- Subpath Set-based retrieval is not so good at approximating Tree Kernel-based retrieval, but is 1,000 times faster than Tree Kernel-based retrieval.

Structural retrieval is useful for annotationg corpora with syntactic information (Yoshida et al., 2004). We are developing a corpus annotation tool named "eBonsai" which supports human to annotate corpora with syntactic information and to retrieve syntactic structures. Integrating annotation and retrieval enables annotators to annotate a new instance with looking back at the already annotated instances which share the similar syntactic structure with the current one. For such purpose, Tree Overlapping and Subpath Set algorithms contribute to speed up the retrieval process, thus make the annotation process more efficient.

However, "similarity" of sentences is affected by semantic aspects as well as structural aspects. The output of the algorithms do not always conform with human's intuition. For example, the two sentences in Figure 6 have very similar structures including particles, but they are hardly considered similar from human's viewpoint. With this respect, it is hardly to say which algorithm is superior to others.

As a future work, we need to develop a method to integrate both content-based and structure-based similarity measures. To this end, we have to evaluate the algorithms in real application environments (e.g. information retrieval and machine translation) because desired properties of similarity are different depending on applications.

References

Asahara, M. and Matsumoto, Y., Extended Models and Tools for High-performance Part-of-Speech Tagger. Proceedings of COLING 2000, 2000.

Collins, M. and Duffy, N. Parsing with a Single Neuron: Convolution Kernels for Natural Language Problems. Technical report UCSC-CRL-01-01, University of California at Santa Cruz, 2001.

Collins, M. and Duffy, N. Convolution Kernels for Natural Language. In Proceedings of NIPS 2001, 2001.

Inui, K., Shirai, K., Tokunaga T. and Tanaka H., The Integration of Statistics-based Techniques in the Analysis of Japanese Sentences. Special Interest Group of Natural Language Processing, Information Processing Society of Japan, Vol. 96, No. 114, 1996.

Nagao, M. A framework of a mechanical translation between Japanese and English by analogy principle. In Alick Elithorn and Ranan Banerji, editors, Artifical and Human Intelligence, pages 173-180. Amsterdam, 1984.

Noro, T., Koike, C., Hashimoto, T., Tokunaga, T. and Tanaka, H. Evaluation of a Japanese CFG Derived from a Syntactically Annotated Corpus with respect to Dependency Measures, The 5th Workshop on Asian Language Resources, pp.9-16, 2005.

Nishio, M., Iwabuchi, E. and Mizutani, S. (ed.) Iwanami Kokugo Jiten, Iwanamishoten, 5th Edition, 1994.

Shirai, K., Ueki, M. Hashimoto, T., Tokunaga, T. and Tanaka, H., MSLR Parser Tool Kit - Tools for Natural Language Analysis. Journal of Natural Language

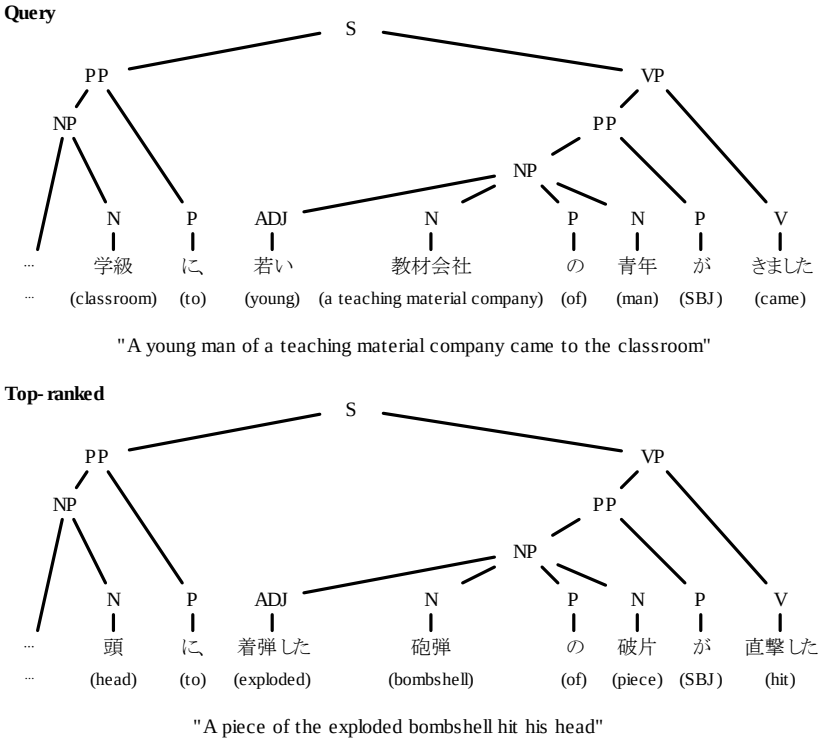


Figure 6: Example of a retrieved similar sentence

Processing, Vol. 7, No. 5, pp. 93-112, 2000. (in Japanese)

Somers, H., McLean, I., Jones, D. Experiments in multilingual example-based generation. CSNLP 1994: 3rd conference on the Cognitive Science of Natural Language Processing, Dublin, 1994.

Takahashi, T., Inui K., and Matsumoto, Y.. Methods of Estimating Syntactic Similarity. Special Interest Group of Natural Language Processing, Information Processing Society of Japan, NL-150-7, 2002. (in Japanese)

Yoshida, K., Hashimoto, T., Tokunaga, T. and Tanaka, H.. Retrieving annotated corpora for corpus annotation. Proceedings of 4th International Conference on Language Resources and Evaluation: LREC 2004. pp.1775 – 1778. 2004.

Towards the Orwellian Nightmare

Separation of Business and Personal Emails

Sanaz Jabbari, Ben Allison, David Guthrie, Louise Guthrie
Department of Computer Science
University of Sheffield
211 Portobello St.
Sheffield
S1 4DP
{s.jabbari, b.allison, d.guthrie, l.guthrie}@dcs.shef.ac.uk

Abstract

This paper describes the largest scale annotation project involving the Enron email corpus to date. Over 12,500 emails were classified, by humans, into the categories "Business" and "Personal", and then subcategorised by type within these categories. The paper quantifies how well humans perform on this task (evaluated by inter-annotator agreement). It presents the problems experienced with the separation of these language types. As a final section, the paper presents preliminary results using a machine to perform this classification task.

1 Introduction

Almost since it became a global phenomenon, computers have been examining and reasoning about our email. For the most part, this intervention has been well natured and helpful – computers have been trying to protect us from attacks of unscrupulous blanket advertising mail shots. However, the use of computers for more nefarious surveillance of email has so far been limited. The sheer volume of email sent means even government agencies (who can legally intercept all mail) must either filter email by some preconceived notion of what is interesting, or they must employ teams of people to manually sift through the volumes of data. For example, the NSA has had massive parallel machines filtering e-mail traffic for at least ten years.

The task of developing such automatic filters at research institutions has been almost impossible, but for the opposite reason. There is no shortage of willing researchers, but progress has been hampered by the lack of any data – one's email is often hugely private, and the prospect of surrendering it, in its entirety, for research purposes is somewhat unsavoury.

Recently, a data resource has become available where exactly this condition (several hundred people's entire email archive) has been satisfied – the Enron dataset. During the legal investigation of the collapse of Enron, the FERC (Federal Energy Regulatory Commission) seized the emails of every employee in that company. As part of the process, the collection of emails was made public and subsequently prepared for research use by researchers at Carnegie Melon University (Klimt and Yang, 2004).Such a corpus of authentic data, on such a large scale, is unique, and an invaluable research tool. It then falls to the prospective researcher to decide which divisions in the language of email are interesting, which are possible, and how the new resource might best be used.

Businesses which offer employees an email system at work (and there are few who do not) have always known that they possess an invaluable resource for monitoring their employees' work habits. During the 1990s, UK courts decided that that an employee's email is not private – in fact, companies can read them at will. However, for exactly the reasons described above, automatic monitoring has been impossible, and few businesses have ever considered it sufficiently important to employ staff to monitor the email use of other staff. However, in monitoring staff productivity, few companies would decline the use of a system which could analyse the email habits of its employees, and report the percentage of time which each employee was spending engaged in non-work related email activities.

The first step in understanding how this problem might be tackled by a computer, and if it is even feasible for this to happen, is to have humans perform the task. This paper describes the process of having humans annotate a corpus of emails, classifying each as to whether they are business or personal, and then attempting to classify the type of business or personal mail being considered.

A resource has been created to develop a system able to make these distinctions automatically. Furthermore, the process of subcategorising types of business and personal has allowed invaluable insights into the areas

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 407–411,
Sydney, July 2006.

where confusion can occur, and how these confusions might be overcome.

The paper presents an evolution of appropriate sub-categories, combined with analysis of performance (measured by inter-annotator agreement) and reasons for any alterations. It addresses previous work done with the Enron dataset, focusing particularly on the work of Marti Hearst at Berkeley who attempted a smaller-scale annotation project of the Enron corpus, albeit with a different focus. It concludes by suggesting that in the main part (with a few exceptions) the task is possible for human annotators. The project has produced a set of labeled messages (around 14,000, plus double annotations for approximately 2,500) with arguably sufficiently high business-personal agreement that machine learning algorithms will have sufficient material to attempt the task automatically.

2 Introduction to the Corpus

Enron's email was made public on the Web by FERC (Federal Energy Regulatory Commission), during a legal investigation on Enron Corporation. The emails cover 92 percent of the staff's emails, because some messages have been deleted "as part of a redaction effort due to requests from affected employees". The dataset was comprised of 619,446 messages from 158 users in 3,500 folders. However, it turned out that the raw data set was suffering from various data integrity problems. Various attempts were made to clean and prepare the dataset for research purposes. The dataset used in this project was the March 2, 2004 version prepared at Carnegie Mellon University, acquired from http://www.cs.cmu.edu/~enron/. This version of the dataset was reduced to 200,399 emails by removing some folders from each user. Folders like "discussion threads" and "all documents", which were machine generated and contained duplicate emails, were removed in this version.

There were on average 757 emails per each of the 158 users. However, there are between one and 100,000 emails per user. There are 30,091 threads present in 123,091 emails. The dataset does not include attachments. Invalid email addresses were replaced with "user@enron.com". When no recipient was specified the address was replaced with "no_address@enron.com" (Klimt and Yang, 2005).

3 Previous Work with the Dataset

The most relevant piece of work to this paper was performed at Berkeley. Marti Hearst ran a small-scale annotation project to classify emails in the corpus by their type and purpose (Email annotation at Berkely). In total, approximately 1,700 messages were annotated by two distinct annotators. Annotation categories captured four dimensions, but broadly speaking they reflected the following qualities of the email: coarse genre, the topic of the email if business was selected, information about any forwarded or included text and the emotional tone of the email. However, the categories used at the Berkeley project were incompatible with our requirements for several reasons: that project allowed multiple labels to be assigned to each email; the categories were not designed to facilitate discrimination between business and personal emails; distinctions between topic, genre, source and purpose were present in each of the dimensions; and no effort was made to analyse the inter-annotator agreement (Email annotation at Berkely).

User-defined folders are preserved in the Enron data, and some research efforts have used these folders to develop and evaluate machine-learning algorithms for automatically sorting emails (Klimt and Yang, 2004). However, as users are often inconsistent in organising their emails, so the training and testing data in these cases are questionable. For example, many users have folders marked "Personal", and one might think these could be used as a basis for the characterisation of personal emails. However, upon closer inspection it becomes clear that only a tiny percentage of an individual's personal emails are in these folders. Similarly, many users have folders containing exclusively personal content, but without any obvious folder name to reveal this. All of these problems dictate that for an effective system to be produced, large-scale manual annotation will be necessary.

Researchers at Queen's University, Canada (Keila, 2005) recently attempted to categorise and identify deceptive messages in the Enron corpus. Their method used a hypothesis from deception theory (e.g., deceptive writing contains cues such as reduced frequency of first-person pronouns and increased frequency of "negative emotion" words) and as to what constitutes deceptive language. Single value decomposition (SVD) was applied to separate the emails, and a manual survey of the results allowed them to conclude that this classification method for detecting deception in email was promising.

Other researchers have attempted to analyse the Enron emails from a network analytic perspective (Deisner, 2005). Their goal was to analyse the flow of communication between employees at times of crisis, and develop a characterisation for the state of a communication network in such difficult times, in order to identify looming crises in other companies from the state of their communication networks. They compared the network flow of email in October 2000 and October 2001.

4 Annotation Categories for this Project

Because in many cases there is no definite line between business emails and personal emails, it was decided to mark emails with finer categories than

Business and Personal. This subcategorising not only helped us to analyse the different types of email within business and personal emails, but it helped us to find the nature of the disagreements that occurred later on, in inter-annotation. In other words, this process allowed us to observe patterns in disagreement.

Obviously, the process of deciding categories in any annotation project is a fraught and contentious one. The process necessarily involves repeated cycles of category design, annotation, inter-annotation, analysis of disagreement, category refinement. While the process described above could continue ad infinitum, the sensible project manager must identify were this process is beginning to converge on a set of well-defined but nonetheless intuitive categories, and finalise them.

Likewise, the annotation project described here went through several evolutions of categories, mediated by input from annotators and other researchers. The final categories chosen were:

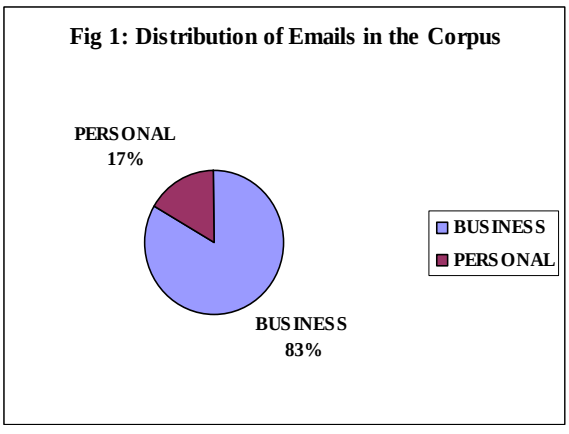
Business: Core Business, Routine Admin, Inter-Employee Relations, Solicited/soliciting mailing, Image.

Personal: Close Personal, Forwarded, Auto generated emails.

5 Annotation and Inter-Annotation

Based on the categories above, approximately 12,500 emails were single-annotated by a total of four annotators.

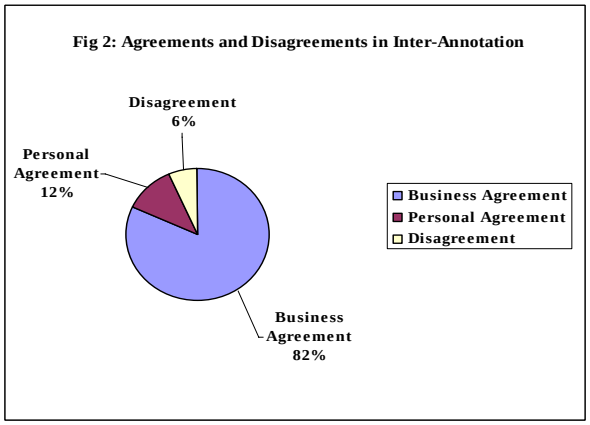
The results showed that around 83% of the emails were business related, while 17% were personal. The company received one personal email for every five business emails.



A third of the received emails were “Core Business” and a third were “Routine Admin”. All other categories comprised the remaining third of the emails. One could conclude that approximately one third of emails received at Enron were discussions of policy, strategy, legislation, regulations, trading, and other high-level business matters. The next third of received emails were about the peripheral, routine matters of the company. These are emails related to HR, IT administration, meeting scheduling, etc. which can be regarded as part of the common infrastructure of any large scale corporation.

The rest of the emails were distributed among personal emails, emails to colleagues, company news letters, and emails received due to subscription. The biggest portion of the last third, are emails received due to subscription, whether the subscription be business or personal in nature.

In any annotation project consistency should be measured. To this end 2,200 emails were double annotated between four annotators. As Figure 2 below shows, for 82% of the emails both annotators agreed that the email was business email and in 12% of the emails, both agreed on them being personal. Six percent of the emails were disagreed upon.



By analysing the disagreed categories, some patterns of confusion were found.

Around one fourth of the confusions were solicited emails where it was not clear whether the employee was subscribed to a particular newsletter group for his personal interest, private business, or Enron’s business. While some subscriptions were clearly personal (e.g. subscription to latest celebrity news) and some were clearly business related (e.g. Daily Energy reports), for some it was hard to identify the intention of the subscription (e.g. New York Times).

Eighteen percent of the confusions were due to emails about travel arrangements, flight and hotel booking confirmations, where it was not clear whether the personal was acting in a business or personal role.

Thirteen percent of the disagreements were upon whether an email is written between two Enron employees as business colleagues or friends. The emails such as "shall we meet for a coffee at 2:00?" If insufficient information exists in the email, it can be hard to draw the line between a personal relationship and a relationship between colleagues. The annotators were advised to pick the category based on the formality of the language used in such emails, and reading between the lines wherever possible.

About eight percent of the disagreements were on emails which were about services that Enron provides for its employees. For example, the Enron's running club is seeking for runners, and sending an ad to Enron's employers. Or Enron's employee's assistance Program (EAP), sending an email to all employees, letting them know that in case of finding themselves in stressful situations they can use some of the services that Enron provides for them or their families.

One theme was encountered in many types of confusions: namely, whether to decide an e-mail's category based upon its topic or its form. For example, should an email be categorised because it is scheduling a meeting or because of the subject of the meeting being scheduled? One might consider this a distinction by topic or by genre.

As the result, final categories were created to reflect topic as the only dimension to be considered in the annotation. "Solicited/Soliciting mailing", "Solicited/Auto generated mailing" and "Forwarded" were removed and "Keeping Current", "Soliciting" were added as business categories and "Personal Maintenance" and "Personal Circulation" were added as personal categories. The inter-annotation agreement was measured for one hundred and fifty emails, annotated by five annotators. The results confirmed that these changes had a positive effect on the accuracy of annotation.

6 Preliminary Results of Automatic Classification

Some preliminary experiments were performed with an automatic classifier to determine the feasibility of separating business and personal emails by machine. The classifier used was a probabilistic classifier based upon the distribution of distinguishing words. More information can be found in (Guthrie and Walker, 1994).

Two categories from the annotation were chosen which were considered to typify the broad categories – these were Core Business (representing business) and Close Personal (representing personal). The Core Business class contains 4,000 messages (approx 900,000 words), while Close Personal contains approximately 1,000 messages (220,000 words).

The following table summarises the performance of this classifier in terms of Recall, Precision and F-Measure and accuracy:

Class	Recall	Precision	F-Measure	Accuracy
Business	0.99	0.92	0.95	0.99
Personal	0.69	0.95	0.80	0.69
AVERAGE	**0.84**	**0.94**	**0.88**	**0.93**

Based upon the results of this experiment, one can conclude that automatic methods are also suitable for classifying emails as to whether they are business or personal. The results indicate that the business category is well represented by the classifier, and given the disproportionate distribution of emails, the classifier's tendency towards the business category is understandable.

Given that our inter-annotator agreement statistic tells us that humans only agree on this task 94% of the time, preliminary results with 93% accuracy (the statistic which correlates exactly to agreement) of the automatic method are encouraging. While more work is necessary to fully evaluate the suitability of this task for application to a machine, the seeds of a fully automated system are sown.

7 Conclusion

This paper describes the process of creating an email corpus annotated with business or personal labels. By measuring inter-annotator agreement it shows that this process was successful. Furthermore, by analysing the disagreements in the fine categories, it has allowed us to characterise the areas where the business/personal decisions are difficult.

In general, the separation of business and personal mails is a task that humans can perform. Part of the project has allowed the identification of the areas where humans cannot make this distinction (as demonstrated by inter-annotator agreement scores) and one would not expect machines to perform the task under these conditions either. In all other cases, where the language is not ambiguous as judged by human annotators, the challenge has been made to automatic classifiers to match this performance.

Some initial results were reported where machines attempted exactly this task. They showed that accuracy almost as high as human agreement was achieved by the system. Further work, using much larger sets and incorporating all types of business and personal emails, is the next logical step.

Any annotation project will encounter its problems in deciding appropriate categories. This paper described the various stages of evolving these categories to a stage where they are both intuitive and logical and also, produce respectable inter-annotator agreement scores. The work is still in progress in ensuring maximal consistency within the data set and refining the precise definitions of the categories to avoid possible overlaps.

References

Brian Klimt and Yiming Yang. 2004. *Introducing the Enron Email Corpus*, Carnegie Mellon University.

Brian Klimt and Yiming Yang. 2004. *The Enron Corpus: A New Data Set for Email Classification Research*. Carnegie Mellon University.

Email Annotation at Berkely
http://bailando.sims.berkeley.edu/enron_email.html

Jana Diesner and Kathleen M. Karley. 2005. *Exploration of Communication Networks from the Enron Email Corpus*, Carnegie Mellon University

Louise Guthrie, Elbert Walker and Joe Guthrie. 1994 *Document classification by machine: Theory and practice*. Proc. of COLING'94

Parambir S. Keila and David B. Skillcorn. 2005. *Detecting Unusual and Deceptive Communication in Email*. Queen's University, CA

Exploiting Non-local Features for Spoken Language Understanding

Minwoo Jeong and **Gary Geunbae Lee**
Department of Computer Science & Engineering
Pohang University of Science and Technology,
San 31 Hyoja-dong, Nam-gu
Pohang 790-784, Korea
{stardust,gblee}@postech.ac.kr

Abstract

In this paper, we exploit non-local features as an estimate of long-distance dependencies to improve performance on the statistical spoken language understanding (SLU) problem. The statistical natural language parsers trained on text perform unreliably to encode non-local information on spoken language. An alternative method we propose is to use trigger pairs that are automatically extracted by a feature induction algorithm. We describe a light version of the inducer in which a simple modification is efficient and successful. We evaluate our method on an SLU task and show an error reduction of up to 27% over the base local model.

1 Introduction

For most sequential labeling problems in natural language processing (NLP), a decision is made based on local information. However, processing that relies on the Markovian assumption cannot represent higher-order dependencies. This long-distance dependency problem has been considered at length in computational linguistics. It is the key limitation in bettering sequential models in various natural language tasks. Thus, we need new methods to import *non-local* information into sequential models.

There are two types of method for using non-local information. One is to add edges to structure to allow higher-order dependencies and another is to add features (or observable variables) to encode the non-locality. An additional consistent edge of a linear-chain conditional random field (CRF) explicitly models the dependencies between distant occurrences of similar words (Sutton and McCallum, 2004; Finkel et al., 2005). However, this approach requires additional time complexity in inference/learning time and it is only suitable for representing constraints by enforcing label consistency. We wish to identify ambiguous labels with more general dependency without additional time cost in inference/learning time.

Another approach to modeling non-locality is to use observational features which can capture non-local information. Traditionally, many systems prefer to use a syntactic parser. In a language understanding task, the head word dependencies or parse tree path are successfully applied to learn and predict semantic roles, especially those with ambiguous labels (Gildea and Jurafsky, 2002). Although the power of syntactic structure is impressive, using the parser-based feature fails to encode correct global information because of the low accuracy of a modern parser. Furthermore the inaccurate result of parsing is more serious in a spoken language understanding (SLU) task. In contrast to written language, spoken language loses much information including grammar, structure or morphology and contains some errors in automatically recognized speech.

To solve the above problems, we present one method to exploit non-local information – the trigger feature. In this paper, we incorporate trigger pairs into a sequential model, a linear-chain CRF. Then we describe an efficient algorithm to extract the trigger feature from the training data itself. The framework for inducing trigger features is based on the Kullback-Leibler divergence criterion which measures the improvement of log-likelihood on the current parameters by adding a new feature (Pietra et al., 1997). To reduce the cost of feature selection, we suggest a modified

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 412–419,
Sydney, July 2006.

version of an inducing algorithm which is quite efficient. We evaluate our method on an SLU task, and demonstrate the improvements on both transcripts and recognition outputs. On a real-world problem, our modified version of a feature selection algorithm is very efficient for both performance and time complexity.

2 Spoken Language Understanding as a Sequential Labeling Problem

2.1 Spoken Language Understanding

The goal of SLU is to extract semantic meanings from recognized utterances and to fill the correct values into a semantic frame structure. A semantic frame (or template) is a well-formed and machine readable structure of extracted information consisting of slot/value pairs. An example of such a reference frame is as follows.

<s> i wanna go from denver to new york on november eighteenth </s>
FROMLOC.CITY_NAME = denver
TOLOC.CITY_NAME = new york
MONTH_NAME = november
DAY_NUMBER = eighteenth

This example from air travel data (CU-Communicator corpus) was automatically generated by a Phoenix parser and manually corrected (Pellom et al., 2000; He and Young, 2005). In this example, the slot labels are two-level hierarchical; such as FROMLOC.CITY_NAME. This hierarchy differentiates the semantic frame extraction problem from the named entity recognition (NER) problem.

Regardless of the fact that there are some differences between SLU and NER, we can still apply well-known techniques used in NER to an SLU problem. Following (Ramshaw and Marcus, 1995), the slot labels are drawn from a set of classes constructed by extending each label by three additional symbols, Beginning/Inside/Outside (B/I/O). A two-level hierarchical slot can be considered as an integrated flattened slot. For example, FROMLOC.CITY_NAME and TOLOC.CITY_NAME are different on this slot definition scheme.

Now, we can formalize the SLU problem as a sequential labeling problem, $\mathbf{y}^* = \arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x})$. In this case, input word sequences $\mathbf{x}$ are not only lexical strings, but also multiple linguistic features. To extract semantic frames from utterance inputs, we use a linear-chain CRF model; a model that assigns a joint probability distribution over labels which is conditional on the input sequences, where the distribution respects the independent relations encoded in a graph (Lafferty et al., 2001).

A linear-chain CRF is defined as follows. Let $\mathcal{G}$ be an undirected model over sets of random variables $\mathbf{x}$ and $\mathbf{y}$. The graph $\mathcal{G}$ with parameters $\Lambda = \{\lambda, \ldots\}$ defines a conditional probability for a state (or label) sequence $\mathbf{y} = y_1, \ldots, y_T$, given an input $\mathbf{x} = x_1, \ldots, x_T$, to be

$$P_\Lambda(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_\mathbf{x}} \exp\left(\sum_{t=1}^{T}\sum_{k} \lambda_k f_k(y_{t-1}, y_t, \mathbf{x}, t)\right)$$

where $Z_\mathbf{x}$ is the normalization factor that makes the probability of all state sequences sum to one. $f_k(y_{t-1}, y_t, \mathbf{x}, t)$ is an arbitrary linguistic feature function which is often binary-valued in NLP tasks. λ_k is a trained parameter associated with feature f_k. The feature functions can encode any aspect of a state transition, $y_{t-1} \rightarrow y_t$, and the observation (a set of observable features), $\mathbf{x}$, centered at the current time, t. Large positive values for λ_k indicate a preference for such an event, while large negative values make the event unlikely.

Parameter estimation of a linear-chain CRF is typically performed by conditional maximum log-likelihood. To avoid overfitting, the 2-norm regularization is applied to penalize on weight vector whose norm is too large. We used a limited memory version of the quasi-Newton method (L-BFGS) to optimize this objective function. The L-BFGS method converges super-linearly to the solution, so it can be an efficient optimization technique on large-scale NLP problems (Sha and Pereira, 2003).

A linear-chain CRF has been previously applied to obtain promising results in various natural language tasks, but the linear-chain structure is deficient in modeling long-distance dependencies because of its limited structure (n-th order Markov chains).

2.2 Long-distance Dependency in Spoken Language Understanding

In most sequential supervised learning problems including SLU, the feature function $f_k(y_{t-1}, y_t, \mathbf{x}_t, t)$ indicates only local information

for practical reasons. With sufficient local context (e.g. a sliding window of width 5), inference and learning are both efficient.

However, if we only use local features, then we cannot model long-distance dependencies. Thus, we should incorporate non-local information into the model. For example, figure 1 shows the long-distance dependency problem in an SLU task. The same two word tokens "dec." should be classified differently, `DEPART.MONTH` and `RETURN.MONTH`. The dotted line boxes represent local information at the current decision point ("dec."), but they are exactly the same in two distinct examples. Moreover, the two states share the same previous sequence `(O, O, FROMLOC.CITY_NAME-B, O, TOLOC.CITY_NAME-B, O)`. If we cannot obtain higher-order dependencies such as "fly" and "return," then the linear-chain CRF cannot classify the correct labels between the two same tokens. To solve this problem, we propose an approach to exploit non-local information in the next section.

3 Incorporating Non-local Information

3.1 Using Trigger Features

To exploit non-local information to sequential labeling for a statistical SLU, we can use two approaches; a syntactic parser-based and a data-driven approach. Traditionally, information extraction and language understanding fields have usually used a syntactic parser to encode global information (e.g. parse tree path, governing category, or head word) over a local model. In a semantic role labeling task, the syntax and semantics are correlated with each other (Gildea and Jurafsky, 2002), that is, the global structure of the sentence is useful for identifying ambiguous semantic roles. However the problem is the poor accuracy of the syntactic parser with this type of feature. In addition, recognized utterances are erroneous and the spoken language has no capital letters, no additional symbols, and sometimes no grammar, so it is difficult to use a parser in an SLU problem.

Another solution is a data-driven method, which uses statistics to find features that are approximately modeling long-distance dependencies. The simplest way is to use identical words in history or lexical co-occurrence, but we wish to use a more general tool; triggering. The trigger word pairs are introduced by (Rosenfeld, 1994). A trigger pair is the basic element for extracting information from the long-distance document history. In language modeling, n-gram based on the Markovian assumption cannot represent higher-order dependencies, but it can automatically extract trigger word pairs from data. The pair $(A \rightarrow B)$ means that word A and B are significantly correlated, that is, when A occurs in the document, it triggers B, causing its probability estimate to change.

To select reasonable pairs from arbitrary word pairs, (Rosenfeld, 1994) used averaged mutual information (MI). In this scheme, the MI score of one pair is $MI(A;B) =$

$$P(A,B)\log\frac{P(B|A)}{P(B)} + P(A,\bar{B})\log\frac{P(\bar{B}|A)}{P(\bar{B})} + P(\bar{A},B)\log\frac{P(B|\bar{A})}{P(\bar{B})} + P(\bar{A},\bar{B})\log\frac{P(\bar{B}|\bar{A})}{P(\bar{B})}.$$

Using the MI criterion, we can select correlated word pairs. For example, the trigger pair (dec.→return) was extracted with score 0.001179 in the training data[1]. This trigger word pair can represent long-distance dependency and provide a cue to identify ambiguous classes. The MI approach, however, considers only lexical collocation without reference labels $\mathbf{y}$, and MI based selection tends to excessively select the irrelevant triggers. Recall that our goal is to find the significantly correlated trigger pairs which improve the model. Therefore, we use a more appropriate selection method for sequential supervised learning.

3.2 Selecting Trigger Feature

We present another approach to extract relevant triggers and exploit them in a linear-chain CRF. Our approach is based on an automatic feature induction algorithm, which is a novel method to select a feature in an exponential model (Pietra et al., 1997; McCallum, 2003). We follow McCallum's work which is an efficient method to induce features in a linear-chain CRF model. Following the framework of feature inducing, we start the algorithm with an empty set, and iteratively increase the bundle of features including local features and trigger features. Our basic assumption, however, is that the local information should be included because the local features are the basis of the decision to identify the classes, and they reduce the

[1] In our experiment, the pair (dec.→fly) cannot be selected because this MI score is too low. However, the trigger pair is a binary type feature, so the pair (dec.→return) is enough to classify the two cases in the previous example.

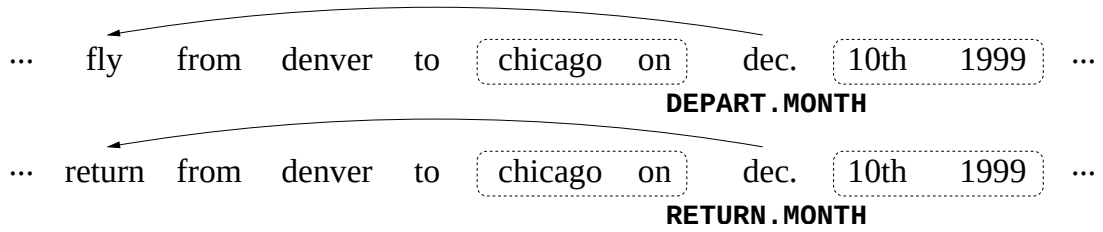


Figure 1: An example of a long-distance dependency problem in spoken language understanding. In this case, a word token "dec." with local feature set (dotted line box) is ambiguous for determining the correct label (DEPART.MONTH or RETURN.MONTH).

mismatch between training and testing tasks. Furthermore, this assumption leads us to faster training in the inducing procedure because we can only consider additional trigger features.

Now, we start the inducing process with local features rather than an empty set. After training the base model $\Lambda^{(0)}$, we should calculate the gains, which measure the effect of adding a trigger feature, based on the local model parameter $\Lambda^{(0)}$. The gain of the trigger feature is defined as the improvement in log-likelihood of the current model $\Lambda^{(i)}$ at the i-th iteration according to the following formula:

$$\begin{aligned}\hat{G}_{\Lambda^{(i)}}(g) &= \max_{\mu} G_{\Lambda^{(i)}}(g,\mu) \\ &= \max_{\mu}\left\{L_{\Lambda^{(i)}+g,\mu} - L_{\Lambda^{(i)}}\right\}\end{aligned}$$

where μ is a parameter of a trigger feature to be found and g is a corresponding trigger feature function. The optimal value of μ can be calculated by Newton's method.

By adding a new candidate trigger, the equation of the linear-chain CRF model is changed to an additional feature model as $P_{\Lambda^{(i)}+g,\mu}(\mathbf{y}|\mathbf{x}) =$

$$\frac{P_{\Lambda^{(i)}}(\mathbf{y}|\mathbf{x})\exp\left(\sum_{t=1}^{T}\mu g(y_{t-1},y_t,\mathbf{x},t)\right)}{Z_{\mathbf{x}}(\Lambda^{(i)},g,\mu)}.$$

Note that $Z_{\mathbf{x}}(\Lambda^{(i)},g,\mu)$ is the marginal sum over all states of $\mathbf{y}'$. Following (Pietra et al., 1997; McCallum, 2003), the mean field approximation and agglomerated features allows us to treat the above calculation as the independent inference problem rather than sequential inference. We can evaluate the probability of state $\mathbf{y}$ with an adding trigger pair given observation $\mathbf{x}$ separately as follows.

$$P_{\Lambda^{(i)}+g,\mu}(y|\mathbf{x},t) = \frac{P_{\Lambda^{(i)}}(y|\mathbf{x},t)\exp\left(\mu g(y_t,\mathbf{x},t)\right)}{Z_{\mathbf{x}}(\Lambda^{(i)},g,\mu)}$$

Here, we introduce a second approximation. We use the individual inference problem over the unstructured maximum entropy (ME) model whose state variable is independent from other states in history. The background of our approximation is that the state independent problem of CRF can be relaxed to ME inference problem without the state-structured model. In the result, we calculate the gain of candidate triggers, and select trigger features over a light ME model instead of a huge computational CRF model[2].

We can efficiently assess many candidate trigger features in parallel by assuming that the old features remain fixed while estimating the gain. The gain of trigger features can be calculated on the old model that is trained with the local and added trigger pairs in previous iterations. Rather than summing over all training instances, we only need to use the mislabeled N tokens by the current parameter $\Lambda^{(i)}$ (McCallum, 2003). From misclassified instances, we generate the candidates of trigger pairs, that is, all pairs of current words and others within the sentence. With the candidate feature set, the gain is

$$\begin{aligned}\hat{G}_{\Lambda^{(i)}}(g) &= N\hat{\mu}\tilde{E}[g] \\ &- \sum_{j=1}^{N}\log\left(E_{\Lambda^{(i)}}[\exp(\hat{\mu}g)|\mathbf{x}_j]\right) - \frac{\hat{\mu}^2}{2\sigma^2}.\end{aligned}$$

Using the estimated gains, we can select a small portion of all candidates, and retrain the model with selected features. We iteratively perform the selection algorithm with some stop conditions (excess of maximum iteration or no added feature up to the gain threshold). The outline of the induction

[2]The ME model cannot represent the sequential structure and the resulting model is different from CRF. Nevertheless, we empirically prove that the effect of additional trigger features on both ME and approximated CRF (without regarding edge-state) are similar (see the experiment section).

Algorithm InduceLearn($\mathbf{x}$,$\mathbf{y}$)
$triggers \leftarrow \{\varepsilon\}$ and $i \leftarrow 0$
while $|pairs| > 0$ and $i < maxiter$ **do**
  $\Lambda^{(i)} \leftarrow \mathrm{TrainME}(\mathbf{x}, \mathbf{y})$
  $P(\mathbf{y}_e|\mathbf{x}_e) \leftarrow \mathrm{Evaluate}(\mathbf{x}, \mathbf{y}, \Lambda^{(i)})$
  $\mathbf{c} \leftarrow \mathrm{MakeCandidate}(\mathbf{x}_e)$
  $G_{\Lambda^{(i)}} \leftarrow \mathrm{EstimateGain}(\mathbf{c}, P(\mathbf{y}_e|\mathbf{x}_e))$
  $pairs \leftarrow \mathrm{SelectTrigger}(\mathbf{c}, G_{\Lambda^{(i)}})$
  $\mathbf{x} \leftarrow \mathrm{UpdateObs}(\mathbf{x}, pairs)$
  $triggers \leftarrow triggers \cup pairs$ and $i \leftarrow i + 1$
end while
$\Lambda^{(i+1)} \leftarrow \mathrm{TrainCRF}(\mathbf{x}, \mathbf{y})$
return $\Lambda^{(i+1)}$

Figure 2: Outline of trigger feature induction algorithm

algorithms is described in figure 2. In the next section, we empirically prove the effectiveness of our algorithm.

The trigger pairs introduced by (Rosenfeld, 1994) are just word pairs. Here, we can generalize the trigger pairs to any arbitrary pairs of features. For example, the feature pair (of→B-PP) is useful in deciding the correct answer `PERIOD_OF_DAY-I` in "in the middle of the day." Without constraints on generating the pairs (e.g. at most 3 distant tokens), the candidates can be arbitrary conjunctions of features[3]. Therefore we can explore any features including local conjunction or non-local singleton features in a uniform framework.

4 Experiments

4.1 Experimental Setup

We evaluate our method on the CU-Communicator corpus. It consists of 13,983 utterances. The semantic categories correspond to city names, time-related information, airlines and other miscellaneous entities. The semantic labels are automatically generated by a Phoenix parser and manually corrected. In the data set, the semantic category has a two-level hierarchy: 31 first level classes and 7 second level classes, for a total of 62 class combinations. The data set is 630k words with 29k entities. Roughly half of the entities are time-related information, a quarter of the entities are city names, a tenth are state and country names, and a fifth are airline and airport names. For the second level hierarchy, approximately three quarters of the entities are "`NONE`", a tenth are "`TOLOC`", a tenth are "`FROMLOC`", and the remaining are "`RETURN`", "`DEPART`", "`ARRIVE`", and "`STOPLOC`."

For spoken inputs, we used the open source speech recognizer Sphinx2. We trained the recognizer with only the domain-specific speech corpus. The reported accuracy for Sphinx2 speech recognition is about 85%, but the accuracy of our speech recognizer is 76.27%; we used only a subset of the data without tuning and the sentences of this subset are longer and more complex than those of the removed ones, most of which are single-word responses.

All of our results have averaged over 5-fold cross validation with an 80/20 split of the data. As it is standard, we compute precision and recall, which are evaluated on a per-entity basis and combined into a micro-averaged F1 score (F1 = 2PR/(P+R)).

A final model (a first-order linear chain CRF) is trained for 100 iterations with a Gaussian prior variance of 20, and 200 or fewer trigger features (down to a gain threshold of 1.0) for each round of inducing iteration (100 iterations of L-BFGS for the ME inducer and 10∼20 iterations of L-BFGS for the CRF inducer). All experiments are implemented in C++ and executed on Linux with XEON 2.8 GHz dual processors and 2.0 Gbyte of main memory.

4.2 Empirical Results

We list the feature templates used by our experiment in figure 3. For local features, we use the indicators for specific words at location i, or locations within five words of i ($-2, -1, 0, +1, +2$ words on current position i). We also use the part-of-speech (POS) tags and phrase labels with partial parsing. Like words, the two basic linguistic features are located within five tokens. For comparison, we exploit the two groups of non-local syntax parser-based features; we use Collins parser and extract this type of features from the parse trees. The first consists of the head word and POS-tag of the head word. The second group includes governing category and parse tree paths introduced by semantic role labeling (Gildea and Jurafsky, 2002). Following the previous studies

[3] In our experiment, we do not consider the local conjunctions because we wish to capture the effect of long-distance entities.

Local feature templates
-lexical words -part-of-speech (POS) tags -phrase chunk labels
Grammar-based feature templates
-head word / POS-tag -parse tree path and governing category
Trigger feature templates
-word pairs ($w_i \rightarrow w_j$), $\lvert i-j \rvert > 2$ -feature pairs between words, POS-tags, and chunk labels ($f_i \rightarrow f_j$), $\lvert i-j \rvert > 2$ -null pairs ($\varepsilon \rightarrow w_j$)

Figure 3: Feature templates

of semantic role labeling, the parse tree path improves the classification performance of semantic role labeling. Finally, we use the trigger pairs that are automatically extracted from the training data. Avoiding the overlap of local features, we add the constraint $|i-j| > 2$ for the target word w_j. Note that null pairs are equivalent to long-distance singleton word features w_j.

To compute feature performance, we begin with word features and iteratively add them one-by-one so that we achieve the best performance. Table 1 shows the empirical results of local features, syntactic parser-based features, and trigger features respectively. The two F1 scores for text transcripts (Text) and outputs recognized by an automatic speech recognizer (ASR) are listed. We achieved F1 scores of 94.79 and 71.79 for Text and ASR inputs using only word features. The performance is decreased by adding the additional local features (POS-tags and chunk labels) because the pre-processor brings more errors to the system for spoken dialog.

The parser-based and trigger features are added to two baselines: word only and all local features. The result shows that the trigger feature is more robust to an SLU task than the features generated from the syntactic parser. The parse tree path and governing category show a small improvement of performance over local features, but it is rather insignificant (word vs. word+path, McNemar's test (Gillick and Cox, 1989); $p = 0.022$). In contrast, the trigger features significantly improve the performance of the system for both Text and ASR inputs. The differences between the trigger and the others are statistically significant (McNemar's test; $p < 0.001$ for both Text and ASR).

Table 1: The result of local features, parser-based features and trigger features

Feature set	F1 (Text)	F1 (ASR)
word (w)	94.79	71.79
w + POStag (p)	94.57	71.61
w + chunk (c)	94.70	71.64
local (w+p+c)	94.41	71.60
w + head (h)	94.55	71.76
w + path (t)	95.07	72.17
w + h + t	94.84	72.09
local + head (h)	94.17	71.39
local + path (t)	94.80	71.89
local + h + t	94.51	71.67
w + trigger	**96.18**	**72.95**
local + trigger	96.04	72.72

Next, we compared the two trigger selection methods; mutual information (MI) and feature induction (FI). Table 2 shows the experimental results of the comparison between MI and FI approaches (with the local feature set; w+p+c). For the MI-based approach, we should calculate an averaged MI for each word pair appearing in a sentence and cut the unreliable pairs (down to threshold of 0.0001) before training the model. In contrast, the FI-based approach selects reliable triggers which should improve the model in training time. Our method based on the feature induction algorithm outperforms simple MI-based methods. Fewer features are selected by FI, that is, our method prunes the event pairs which are highly correlated, but not relevant to models. The extended feature trigger ($f_i \rightarrow f_j$) and null triggers ($\varepsilon \rightarrow w_j$) improve the performance over word trigger pairs ($w_i \rightarrow w_j$), but they are not statistically significant (vs. ($f_i \rightarrow f_j$); $p = 0.749$, vs. ($\{\varepsilon, w_i\} \rightarrow w_j$); $p = 0.294$). Nevertheless, the null pairs are effective in reducing the size of trigger features.

Figure 4 shows a sample of triggers selected by MI and FI approaches. For example, the trigger "morning $\rightarrow$ return" is ranked in first of FI but 66th of MI. Moreover, the top 5 pairs of MI are not meaningful, that is, MI selects many functional word pairs. The MI approach considers only lexical collocation without reference labels, so the FI method is more appropriate to sequential supervised learning.

Finally, we wish to justify that our modified

Table 2: Result of the trigger selection methods

Method	Avg. # triggers	F1 (Text)	F1 (ASR)	McNemar's test (vs. MI)
MI ($w_i \rightarrow w_j$)	1,713	95.20	72.12	-
FI ($w_i \rightarrow w_j$)	702	96.04	72.72	$p < 0.001$
FI ($f_i \rightarrow f_j$)	805	96.04	72.76	$p < 0.001$
FI ($\{\varepsilon, w_i\} \rightarrow w_j$)	**545**	**96.14**	**72.80**	$p < 0.001$

Mutual Information	Feature Induction
[1] from→like	[1] morning→return
[2] on→to	[2] morning→on
[3] to→i	[3] morning→to
[4] on→from	[4] afternoon→on
[5] from→i	[5] afternoon→return
[41] afternoon→return	[6] afternoon→to
[66] morning→return	[15] morning→leaving
[89] morning→leaving	[349] december→return
[1738] london→fly	[608] illinois→airport

Figure 4: A sample of triggers extracted by two methods

version of an inducing algorithm is efficient and maintains performance without any drawbacks. We proposed two approximations: starting with local features (Approx. 1) and using an unstructured model on the selection stage (Approx. 2), Table 3 shows the results of variant versions of the algorithm. Surprisingly, the selection criterion based on ME (the unstructured model) is better than CRF (the structured model) not only for time cost but also for the performance on our experiment[4]. This result shows that local information provides the fundamental decision clues. Our modification of the algorithm to induce features for CRF is sufficiently fast for practical usage.

5 Related Work and Discussion

The most relevant previous work is (He and Young, 2005) who describes an generative approach – hidden vector state (HVS) model. They used 1,178 test utterances with 18 classes for 1st level label, and published the resulting F1 score of 88.07. Using the same test data and classes, we achieved the 92.77 F1-performance, as well as 39% of error reduction compared to the previous result. Our system uses a discriminative approach, which directly models the conditional distribution, and it is sufficient for classification task. To capture long-distance dependency, HVS uses a context-free model, which increases the complexity of models. In contrast, we use non-local trigger features, which are relatively easy to use without having additional complexity of models.

Trigger word pairs are introduced and successfully applied in a language modeling task. (Rosenfeld, 1994) demonstrated that the trigger word pairs improve the perplexity in ME-based language models. Our method extends this idea to sequential supervised learning problems. Our trigger selection criterion is based on the automatic feature inducing algorithm, and it allows us to generalize the arbitrary pairs of features.

Our method is based on two works of feature induction on an exponential model, (Pietra et al., 1997) and (McCallum, 2003). Our induction algorithm builds on McCallum's method which presents an efficient procedure to induce features on CRF. (McCallum, 2003) suggested using only the mislabeled events rather than the whole training events. This intuitional suggestion has offered us fast training. We added two additional approximations to reduce the time cost; 1) an inducing procedure over a conditional non-structured inference problem rather than an approximated sequential inference problem, and 2) training with a local feature set, which is the basic information to identify the labels.

In this paper, our approach describes how to exploit non-local information to a SLU problem. The trigger features are more robust than grammar-based features, and are easily extracted from the data itself by using an efficient selection algorithm.

[4]In our analysis, 10∼20 iterations for each round of inducing procedure are insufficient in optimizing the model in CRF (empty) inducer. Thus, the resulting parameters are under-fitted and selected features are infeasible. We need more iteration to fit the parameters, but they require too much learning time ($>$ 1 day).

Table 3: Comparison of variations in the induction algorithm (performed on one of the 5-fold validation sets); columns are induction and total training time (h:m:s), number of trigger and total features, and f-score on test data.

Inducer type	Approx.	Induction/total time	# triggers/features	F1 (Text)	F1 (ASR)
CRF (empty)	No approx.	3:55:01 / 5:27:13	682 / 2,693	90.23	67.60
CRF (local)	Approx. 1	1:25:28 / 2:56:49	750 / 5,241	94.87	71.65
ME (empty)	Approx. 2	20:57 / 1:54:22	618 / 2,080	94.85	71.46
ME (local)	Approx. 1+2	**6:30 / 1:36:14**	**608 / 5,099**	**95.17**	**71.81**

6 Conclusion

We have presented a method to exploit non-local information into a sequential supervised learning task. In a real-world problem such as statistical SLU, our model performs significantly better than the traditional models which are based on syntactic parser-based features. In comparing our selection criterion, we find that the mutual information tends to excessively select the triggers while our feature induction algorithm alleviates this issue. Furthermore, the modified version of the algorithm is practically fast enough to maintain its performance particularly when the local features are offered by the starting position of the algorithm.

In this paper, we have focused on a sequential model such as a linear-chain CRF. However, our method can also be naturally applied to arbitrary structured models, thus the first alternative is to combine our methods with a skip-chain CRF (Sutton and McCallum, 2004). Applying and extending our approach to other natural language tasks (which are difficult to apply a parser to) such as information extraction from e-mail data or biomedical named entity recognition is a topic of future work.

Acknowledgements

We thank three anonymous reviewers for helpful comments. This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment). (IITA-2005-C1090-0501-0018)

References

J. R. Finkel, T. Grenager, and C. Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of ACL'05*, pages 363–370.

D. Gildea and D. Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

L. Gillick and S. Cox. 1989. Some statistical issues in the comparison of speech recognition algorithms. In *Proceedings of ICASSP*, pages 532–535.

Y. He and S. Young. 2005. Semantic processing using the hidden vector state model. *Computer Speech & Language*, 19(1):85–106.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, pages 282–289.

A. McCallum. 2003. Efficiently inducing features of conditional random fields. In *Proceedings of UAI*, page 403.

B. L. Pellom, W. Ward, and S. S. Pradhan. 2000. The cu communicator: An architecture for dialogue systems. In *Proceedings of ICSLP*.

S. Della Pietra, V. J. Della Pietra, and J. Lafferty. 1997. Inducing features of random fields. *IEEE Trans. Pattern Anal. Mach. Intell*, 19(4):380–393.

L. A. Ramshaw and M. P. Marcus. 1995. Text chunking using transformation-based learning. In *3rd Workshop on Very Large Corpora*, pages 82–94.

R. Rosenfeld. 1994. Adaptive statistical language modeling: A maximum entropy approach. Technical report, School of Computer Science Carnegie Mellon University.

F. Sha and F. Pereira. 2003. Shallow parsing with conditional random fields. In *Proceedings of HLT/NAACL'03*.

C. Sutton and A. McCallum. 2004. Collective segmentation and labeling of distant entities in information extraction. In *ICML Workshop on Statistical Relational Learning*.

Analysis and Repair of Name Tagger Errors

Heng Ji **Ralph Grishman**
Department of Computer Science
New York University
New York, NY, 10003, USA
hengji@cs.nyu.edu grishman@cs.nyu.edu

Abstract

Name tagging is a critical early stage in many natural language processing pipelines. In this paper we analyze the types of errors produced by a tagger, distinguishing name classification and various types of name identification errors. We present a joint inference model to improve Chinese name tagging by incorporating feedback from subsequent stages in an information extraction pipeline: name structure parsing, cross-document coreference, semantic relation extraction and event extraction. We show through examples and performance measurement how different stages can correct different types of errors. The resulting accuracy approaches that of individual human annotators.

1 Introduction

High-performance named entity (NE) tagging is crucial in many natural language processing tasks, such as information extraction and machine translation. In 'traditional' pipelined system architectures, NE tagging is one of the first steps in the pipeline. NE errors adversely affect subsequent stages, and error rates are often *compounded* by later stages.

However, (Roth and Yi 2002, 2004) and our recent work have focused on incorporating richer linguistic analysis, using the "feedback" from later stages to improve name taggers. We expanded our last year's model (Ji and Grishman, 2005) that used the results of coreference analysis and relation extraction, by adding 'feedback' from more information extraction components – name structure parsing, cross-document coreference, and event extraction – to incrementally rerank the multiple hypotheses from a baseline name tagger.

While together these components produced a further improvement on last year's model, our goal in this paper is to look behind the overall performance figures in order to understand how these varied components contribute to the improvement, and compare the remaining system errors with the human annotator's performance. To this end, we shall decompose the task of name tagging into two subtasks

- Name Identification – The process of identifying name boundaries in the sentence.
- Name Classification – Given the correct name boundaries, assigning the appropriate name types to them.

and observe the effects that different components have on errors of each type. Errors of identification will be further subdivided by type (missing names, spurious names, and boundary errors). We believe such detailed understanding of the benefits of joint inference is a prerequisite for further improvements in name tagging performance.

After summarizing some prior work in this area, describing our baseline NE tagger, and analyzing its errors, we shall illustrate, through a series of examples, the potential for feedback to improve NE performance. We then present some details on how this improvement can be achieved through hypothesis reranking in the extraction pipeline, and analyze the results in terms of different types of identification and classification errors.

2 Prior Work

Some recent work has incorporated global information to improve the performance of name taggers.

For mixed case English data, name identification is relatively easy. Thus some researchers have focused on the more challenging task – classifying names into correct types. In (Roth and

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 420–427,
Sydney, July 2006.

Yi 2002, 2004), given name boundaries in the text, separate classifiers are first trained for name classification and semantic relation detection. Then, the output of the classifiers is used as a conditional distribution given the observed data. This information, along with the constraints among the relations and entities (specific relations require specific classes of names), is used to make global inferences by linear programming for the most probable assignment. They obtained significant improvements in both name classification and relation detection.

In (Ji and Grishman 2005) we generated N-best NE hypotheses and re-ranked them after coreference and semantic relation identification; we obtained a significant improvement in Chinese name tagging performance. In this paper we shall use a wider range of linguistic knowledge sources, and integrate cross-document techniques.

3 Baseline Name Tagger

We apply a multi-lingual (English / Chinese) bigram HMM tagger to identify four named entity types: Person, Organization, GPE ('geo-political entities' – locations which are also political units, such as countries, counties, and cities) and Location. The HMM tagger generally follows the Nymble model (Bikel et al, 1997), and uses best-first search to generate N-Best hypotheses for each input sentence.

In mixed-case English texts, most proper names are capitalized. So capitalization provides a crucial clue for name boundaries.

In contrast, a Chinese sentence is composed of a string of characters without any word boundaries or capitalization. Even after word segmentation there are still no obvious clues for the name boundaries. However, we can apply the following coarse "usable-character" restrictions to reduce the search space.

Standard Chinese family names are generally single characters drawn from a set of 437 family names (there are also 9 two-character family names, although they are quite infrequent) and given names can be one or two characters (Gao et al., 2005). Transliterated Chinese person names usually consist of characters in three relatively fixed character lists (Begin character list, Middle character list and End character list). Person abbreviation names and names including title words match a few patterns. The suffix words (if there are any) of Organization and GPE names belong to relatively fixed lists too.

However, this "usable-character" restriction is not as reliable as the capitalization information for English, since each of these special characters can also be part of common words.

3.1 Identification and Classification Errors

We begin our error analysis with an investigation of the English and Chinese baseline taggers, decomposing the errors into identification and classification errors. In Figure 1 we report the identification F-Measure for the baseline (the first hypothesis), and the N-best upper bound, the best of the N hypotheses[1], using different models: English MonoCase (EN-Mono, without capitalization), English Mixed Case (EN-Mix, with capitalization), Chinese without the usable character restriction (CH-NoRes) and Chinese with the usable character restriction (CH-WithRes).

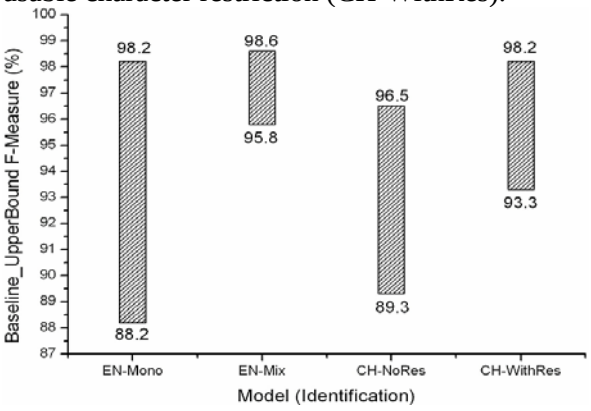


Figure 1. Baseline and Upper Bound of Name Identification

Figure 1 shows that capitalization is a crucial clue in English name identification (increasing the F measure by 7.6% over the monocase score). We can also see that the best of the top N (N <= 30) hypotheses is very good, so reranking a small number of hypotheses has the potential of producing a very good tagger.

The "usable" character restriction plays a major role in Chinese name identification, increasing the F-measure 4%. With this restriction, the performance of the best-of-N-best is again very good. However, it is evident that, even with this restriction, identification is more challenging for Chinese, due to the absence of capitalization and word boundaries.

Figure 2 shows the classification accuracy of the above four models. We can see that capitalization does not help English name classification;

[1] These figures were obtained using training and test corpora described later in this paper, and a value of N ranging from 1 to 30 depending on the margin of the HMM tagger, as also described below. All figures are with respect to the official ACE keys prepared by the Linguistic Data Consortium.

and the difficulty of classification is similar for the two languages.

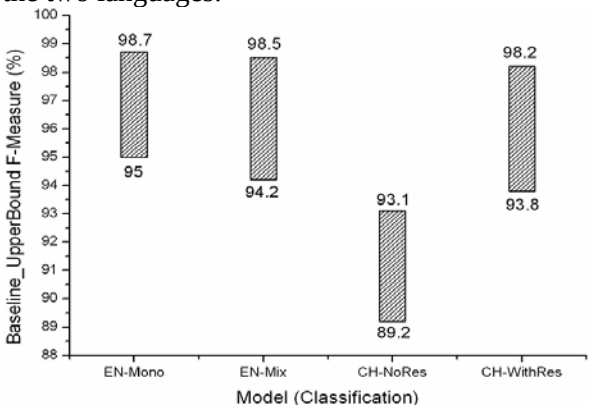


Figure 2. Baseline and Upper Bound of Name Classification

3.2 Identification Errors in Chinese

For the remainder of this paper we shall focus on the more difficult problems of Chinese tagging, using the HMM system with character restrictions as our baseline. The name identification errors of this system can be divided into missed names (21%), spurious names (29%), and boundary errors, where there is a partial overlap between the names in the key and the system response (50%). Confusion between names and nominals (phrases headed by a common noun) is a major source of both missed and spurious names (56% of missed, 24% of spurious). In a language without capitalization, this is a hard task even for people; one must rely largely on world knowledge to decide whether a phrase (such as the "criminal-processing team") is an organization name or merely a description of an organization. The other major source of missed names is words not seen in the training data, generally representing minor cities or other locations in China (28%). For spurious names, the largest source of error is names of a type not included in the key (44%) which are mistakenly tagged as one of the known name types.[2] As we shall see, different types of knowledge are required for correcting different types of errors.

4 Mutual Inferences between Information Extraction Stages

4.1 Extraction Pipeline

Name tagging is typically one of the first stages in an information extraction pipeline. Specifically, we will consider a system which was developed for the ACE (Automatic Content Extraction) task[3] and includes the following stages: name structure parsing, coreference, semantic relation extraction and event extraction (Ji et al., 2006).

All these stages are performed after name tagging since they take names as input "objects". However, the inferences from these subsequent stages can also provide valuable constraints to identify and classify names.

Each of these stages connects the name candidate to other linguistic elements in the sentence, document, or corpus, as shown in Figure 3.

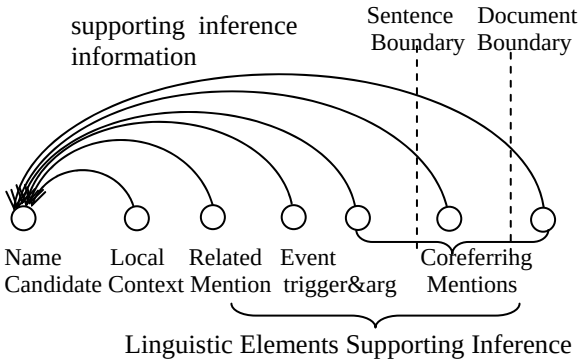


Figure 3. Name candidate and its global context

The baseline name tagger (HMM) uses very local information; feedback from later extraction stages allows us to draw from a wider context in making final name tagging decisions.

In the following we use two related (translated) texts as examples, to give some intuition of how these different types of linguistic evidence improve name tagging.[4]

Document 1: Yugoslav election

[…] More than 300,000 people rushed the *<bei er ge le>*$_0$ congress building, forcing *<yugoslav>*$_1$ president *<mi lo se vi c>*$_2$ to admit frankly that in the Sept. 24 election he was beaten by his opponent *<ke shi tu ni cha>*$_3$.

<mi lo se vi c>$_4$ was forced to flee *<bei er ge le>*$_5$; the winning opposition party's *<sai er wei ya>*$_6$ *<anti-democracy committee>*$_7$ on the morning of the 6th formed a *<crisis-handling*

[2] If the key included an 'other' class of names, these would be classification errors; since it does not -- since these names are not tagged in the key -- the automatic scorer treats them as spurious names.

[3] The ACE task description can be found at http://www.itl.nist.gov/iad/894.01/tests/ace/ and the ACE guidelines at http://www.ldc.upenn.edu/Projects/ACE/

[4] Rather than offer the most fluent translation, we have provided one that more closely corresponds to the Chinese text in order to more clearly illustrate the linguistic issues. Transliterated names are rendered phonetically, character by character.

committee>$_8$, to deal with transfer-of-power issues.

This crisis committee includes police, supply, economics and other important departments.

In such a crisis, people cannot think through this question: has the *<yugoslav>$_9$* president *<mi lo se vi c>$_{10}$* used up his skills?

According to the official voting results in the first round of elections, *<mi lo se vi c>$_{11}$* was beaten by *<18 party opposition committee>$_{12}$* candidate *<ke shi tu ni cha>$_{13}$*. […]

Document 2: Biography of these two leaders

[…]*<ke shi tu ni cha>$_{14}$* used to pursue an academic career, until 1974, when due to his opposition position he was fired by *<bei er ge le>$_{15}$ <law school>$_{16}$* and left the academic community. *<ke shi tu ni cha>$_{17}$* also at the beginning of the 1990s joined the opposition activity, and in 1992 founded *<sai er wei ya>$_{18}$ <opposition party>$_{19}$*.

This famous new leader and his previous classmate at law school, namely his wife *<zuo li ka>$_{20}$* live in an apartment in *<bei er ge le>$_{21}$*.

The vanished *<mi lo se vi c>$_{22}$* was born in *<sai er wei ya>$_{23}$* 's central industrial city. […]

4.1 Inferences for Correcting Name Errors

4.2.1 Internal Name Structure

Constraints and preferences on the structure of individual names can capture local information missed by the baseline name tagger. They can correct several types of identification errors, including in particular boundary errors. For example, "*<ke shi tu ni cha>$_3$*" is more likely to be correct than "*<shi tu ni cha>$_3$*" since "*shi*" (什) cannot be the first character of a transliterated name.

Name structures help to classify names too. For example, "*anti-democracy committee$_7$*" is parsed as "[Org-Modifier anti-democracy] [Org-Suffix committee]", and the first character is not a person last name or the first character of a transliterated person name, so it is more likely to be an organization than a person name.

4.2.2 Patterns

Information about expected sequences of constituents surrounding a name can be used to correct name *boundary errors*. In particular, event extraction is performed by matching patterns involving a "trigger word" (typically, the main verb or nominalization representing the event) and a set of arguments. When a name candidate is involved in an event, the trigger word and other arguments of the event can help to determine the name boundaries. For example, in the sentence "*The vanished mi lo se vi c was born in sai er wei ya 's central industrial city*", "*mi lo se vi c*" is more likely to be a name than "*mi lo se*", "*sai er wei ya"* is more likely be a name than *"er wei"*, because these boundaries will allow us to match the event pattern "[Adj] [PER-NAME] [Trigger word for 'born' event] in [GPE-NAME]'s [GPE-Nominal]".

4.2.3 Selection

Any context which can provide selectional constraints or preferences for a name can be used to correct name *classification* errors. Both semantic relations and events carry selectional constraints and so can be used in this way.

For instance, if the "*Personal-Social/Business*" relation ("*opponent*") between "*his*" and "*<ke shi tu ni cha>$_3$*" is correctly identified, it can help to classify "*<ke shi tu ni cha>$_3$*" as a person name. Relation information is sometimes crucial to classifying names. "*<mi lo se vi c>$_{10}$*" and "*<ke shi tu ni cha>$_{13}$*" are likely person names because they are "*employees*" of "*<yugoslav>$_9$*" and "*<18 party opponent committee>$_{12}$*". Also the "*Personal-Social/Family*" relation ("*wife*") between "*his*" and "*<zuo li ka>$_{20}$*" helps to classify *<zuo li ka>$_{20}$* as a person name.

Events, like relations, can provide effective selectional preferences to correctly classify names. For example, "*<mi lo se vi c>$_{2,4,10,11,22}$*" are likely person names because they are involved in the following events: "*claim*", "*escape*", "*built*", "*beat*", "*born*", while "*<sai er wei ya>$_{23}$*"can be easily tagged as GPE because it's a "*birth-place*" in the event "*born*".

4.2.4 Coreference

Names which are introduced in an article are likely to be referred to again, either by repeating the same name or describing it with nominal mentions (phrases headed by common nouns). These mentions will have the same spelling (though if a name has several parts, some may be dropped) and same semantic type. So if the boundary or type of one mention can be determined with some confidence, coreference can be used to disambiguate other mentions.

For example, if "*< mi lo se vi c>$_2$*" is confirmed as a name, then "*< mi lo se vi c>$_{10}$*" is more likely to be a name than "*< mi lo se>$_{10}$*", by

refering to "< *mi lo se vi c*>$_2$". Also "This crisis committee" supports the analysis of "<*crisis-handling committee*>$_8$" as an organization name in preference to the alternative name candidate "<*crisis-handling*>$_8$".

For a name candidate, high-confidence information about the type of one mention can be used to determine the type of other mentions. For example, for the repeated person name "< *mi lo se vi c*>$_{2,4,10,11,22}$" type information based on the event context of one mention can be used to classify or confirm the type of the others. The person nominal "This famous new leader" confirms "<*ke shi tu ni cha*>$_{17}$" as a person name.

5 Incremental Re-Ranking Algorithm

5.1 Overall Architecture

In this section we will present the algorithms to capture the intuitions described in Section 4. The overall system pipeline is presented in Figure 4.

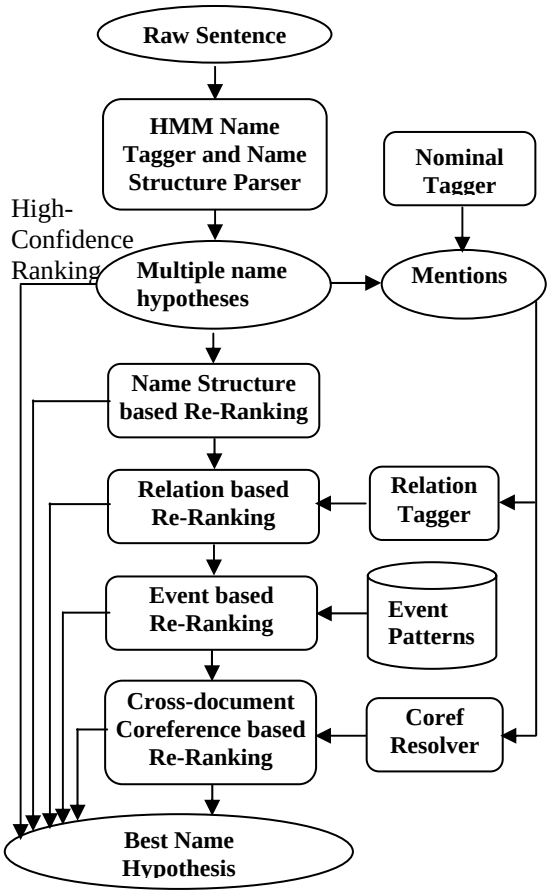


Figure 4. System Architecture

The baseline name tagger generates N-Best multiple hypotheses for each sentence, and also computes the margin – the difference between the log probabilities of the top two hypotheses. This is used as a rough measure of confidence in the top hypothesis. A large margin indicates greater confidence that the first hypothesis is correct.[5] It generates name structure parsing results too, such as the family name and given name of person, the prefixes of the abbreviation names, the modifiers and suffixes of organization names.

Then the results from subsequent components are exploited in four incremental re-rankers. From each re-ranking step we output the best name hypothesis directly if the re-ranker has high confidence in its decisions. Otherwise the sentence is forwarded to the next re-ranker, based on other features. In this way we can adjust the ranking of multiple hypotheses and select the best tagging for each sentence gradually.

The nominal mention tagger (noun phrase chunker) uses a maximum entropy model. Entity type assignment for the nominal heads is done by table look-up. The coreference resolver is a combination of high-precision heuristic rules and maximum entropy models. In order to incorporate wider context we use cross-document coreference for the test set. We cluster the documents using a cross-entropy metric and then treat the entire cluster as a single document.

The relation tagger uses a K-nearest-neighbor algorithm.

We extract event patterns from the ACE05 training corpus for personnel, contact, life, business, and conflict events. We also collect additional event trigger words that appear frequently in name contexts, from a syntactic dictionary, a synonym dictionary and Chinese PropBank V1.0. Then the patterns are generalized and tested semi-automatically.

5.2 Supervised Re-Ranking Model

In our name re-ranking model, each hypothesis is an NE tagging of *the entire sentence*, for example, "*The vanished <PER>mi lo se vi c</PER> was born in <GPE>sai er wei ya</GPE>'s central industrial city*"; and each pair of hypotheses *(h_i, h_j)* is called a "sample".

[5] The margin also determines the number of hypotheses (N) generated by the baseline tagger. Using cross-validation on the training data, we determine the value of N required to include the best hypothesis, as a function of the margin. We then divide the margin into ranges of values, and set a value of N for each range, with a maximum of 30.

Re-Ranker	Property for comparing names N_{ik} and N_{jk}	
Name Structure Based	*HMMMargin*	scaled margin value from HMM
	$Idiom_{ik}$	-1 if N_{ik} is part of an idiom; otherwise 0
	$PERContext_{ik}$	the number of PER context words if N_{ik} and N_{jk} are both PER; otherwise 0
	$ORGSuffix_{ik}$	1 if N_{ik} is tagged as ORG and it includes a suffix word; otherwise 0
	$PERCharac\text{-}ter_{ik}$	-1 if N_{ik} is tagged as PER without family name, and it does not consist entirely of transliterated person name characters; otherwise 0
	$Titlestructure_{ik}$	-1 if N_{ik} = title word + family name while N_{jk} = title word + family name + given name; otherwise 0
	$Digit_{ik}$	-1 if N_{ik} is PER or GPE and it includes digits or punctuation; otherwise 0
	$AbbPER_{ik}$	-1 if N_{ik} = little/old + family name + given name while N_{jk} = little/old + family name; otherwise 0
	$SegmentPER_{ik}$	-1 if N_{ik} is GPE (PER)* GPE , while N_{jk} is PER*; otherwise 0
	$Voting_{ik}$	the voting rate among all the candidate hypotheses[6]
	$Famous\text{-}Name_{ik}$	1 if N_{ik} is tagged as the same type in one of the famous name lists[7]; otherwise 0
Relation Based	$Probability1_i$	scaled ranking probability for (h_i, h_j) from name structure based re-ranker
	$Relation\ Constraint_{ik}$	If N_{ik} is in relation R (N_{ik} = $EntityType_1$, M_2 = $EntityType_2$), compute Prob($EntityType_1$|$EntityType_2$, R) from training data and scale it; otherwise 0
	$Conjunction\ of\ InRelation_i$ & $Probability1_i$	$Inrelation_{ik}$ is 1 if N_{ik} and N_{jk} have different name types, and N_{ik} is in a definite relation while N_{jk} is not; otherwise 0. $Inrelation_i = \sum_k Inrelation_{ik}$
Event Based	$Probability2_i$	scaled ranking probability for (h_i, h_j) from relation based re-ranker
	$Event\ Constraint_i$	1 if all entity types in h_i match event pattern, -1 if some do not match, and 0 if the argument slots are empty
	EventSubType	Event subtype if the patterns are extracted from ACE data, otherwise"None"
Cross-document Coreference Based	$Probability3_i$	scaled ranking probability for (h_i, h_j) from event based re-ranker
	$Head_{ik}$	1 if N_{ik} includes the head word of name; otherwise 0
	$CorefNum_{ik}$	the number of mentions corefered to N_{ik}
	$WeightNum_{ik}$	the sum of all link weights between N_{ik} and its corefered mentions, 0.8 for name-name coreference; 0.5 for apposition; 0.3 for other name-nominal coreference
	$NumHigh\text{-}Coref_i$	the number of mentions which corefer to N_{ik} and output by previous re-rankers with high confidence

Table 3. Re-Ranking Properties

Component		Data
Training	Baseline name tagger	2978 texts from the People's Daily in 1998 and 1300 texts from ACE03, 04, 05 training data
	Nominal tagger	Chinese Penn TreeBank V5.1
	Coreference resolver	1300 texts from ACE03, 04, 05 training data
	Relation tagger	633 ACE 05 texts, and 546 ACE 04 texts with types/subtypes mapped into 05 set
	Event pattern	376 trigger words, 661 patterns
	Name structure, coreference and relation based re-rankers	1,071,285 samples (pairs of hypotheses) from ACE 03, 04 and 05 training data
	Event based re-ranker	325,126 samples from ACE sentences including event trigger words
Test		100 texts from ACE 04 training corpus, includes 2813 names: 1126 persons, 712 GPEs, 785 organizations and 190 locations.

Table 4. Data Description

[6] The method of counting the voting rate refers to (Zhai, 04) and (Ji and Grishman, 05)

[7] Extracted from the high-frequency name lists from the training corpus, and country/province/state/ city lists from Chinese wikipedia.

The goal of each re-ranker is to learn a ranking function f of the following form: for each pair of hypotheses (h_i, h_j), f : H × H → {-1, 1}, such that f(h_i, h_j) = 1 if h_i is better than h_j; f (h_i, h_j) = -1 if h_i is worse than h_j. In this way we are able to convert ranking into a classification problem. And then a maximum entropy model for re-ranking these hypotheses can be trained and applied.

During training we use F-measure to measure the quality of each name hypothesis against the key. During test we get from the MaxEnt classifier the probability (ranking confidence) for each pair: Prob (f (h_i, h_j) = 1). Then we apply a dynamic decoding algorithm to output the best hypothesis. More details about the re-ranking algorithm are presented in (Ji et al., 2006).

5.3 Re-Ranking Features

For each sample (h_i, h_j), we construct a feature set for assessing the ranking of h_i and h_j. Based on the information obtained from inferences, we compute (for each property) the property score PS_{ik} for each individual name candidate N_{ik} in h_i; some of these properties depend also on the corresponding name tags in h_j. Then we sum over all names in each hypothesis h_i:

$$PS_i = \sum_k PS_{ik}$$

Finally we use the quantity $(PS_i – PS_j)$ as the feature value for the sample (h_i, h_j). Table 3 summarizes the property scores PS_{ik} used in the different re-rankers; space limitations prevent us from describing them in further detail.

6 Experimental Results and Analysis

Table 4 shows the data used to train each stage, drawn from the ACE training data and other sources. The training samples of the re-rankers are obtained by running the name tagger in cross-validation. 100 ACE 04 documents were held out for use as test data.

In the following we evaluate the contributions of re-rankers in name identification and classification separately.

Model	Identification		
	Precision	Recall	F-Measure
Baseline	93.2	93.4	93.3
+name structure	94.0	93.5	93.7
+relation	93.9	93.7	93.8
+event	94.1	93.8	93.9
+cross-doc coreference	95.1	93.9	94.5

Table 5. Name Identification

Model	Classification Accuracy	Identification +Classification		
		P	R	F
Baseline	93.8	87.4	87.6	87.5
+name structure	94.3	88.7	88.2	88.4
+relation	95.2	89.4	89.2	89.3
+event	95.7	90.1	89.8	89.9
+cross-doc coreference	96.5	91.8	90.6	91.2

Table 6. Name Classification

Tables 5 and 6 show the performance on identification, classification, and the combined task as we add each re-ranker to the system.

The gain is greater for classification (2.7%) than for identification (1.2%). Furthermore, we can see that the gain in identification is produced primarily by the name structure and coreference components. As we noted earlier, the name structure analysis can correct boundary errors by preferring names with complete internal components, while coreference can resolve a boundary ambiguity for one mention of a name if another mention is unambiguous. The greatest gains were therefore obtained in boundary errors: the stages together eliminated over 1/3 of boundary errors and about 10% of spurious names; only a few missing names were corrected, and some correct names were deleted.

Both relations and events contribute substantially to classification performance through their selectional constraints. The lesser contribution of events is related to their lower frequency. Only 11% of the sentences in the test data contain instances of the original ACE event types. To increase the impact of the event patterns, we broadened their coverage to include additional frequent event types, so that finally 35% of sentences contain event "trigger words".

We used a simple cross-document coreference method in which the test documents were clustered based on their cross-entropy and documents in the same cluster were treated as a single document for coreference. This produced small gains in both identification (0.6% vs. 0.4%) and classification (0.8% vs. 0.4%) over single-document coreference.

7 Discussion

The use of 'feedback' from subsequent stages of analysis has yielded substantial improvements in name tagging accuracy, from F=87.5 with the baseline HMM to F=91.2. This performance compares quite favorably with the performance of the human annotators who prepared the ACE

2005 training data. The annotator scores (when measured against a final key produced by review and adjudication of the two annotations) were F=92.5 for one annotator and F=92.7 for the other.

As in the case of the automatic tagger, human classification accuracy (97.2 - 97.6%) was better than identification accuracy (F = 95.0 - 95.2%).

In Figure 5 we summarize the error rates for the baseline system, the improved system without coreference based re-ranker, the final system with re-ranking, and a single annotator.[8]

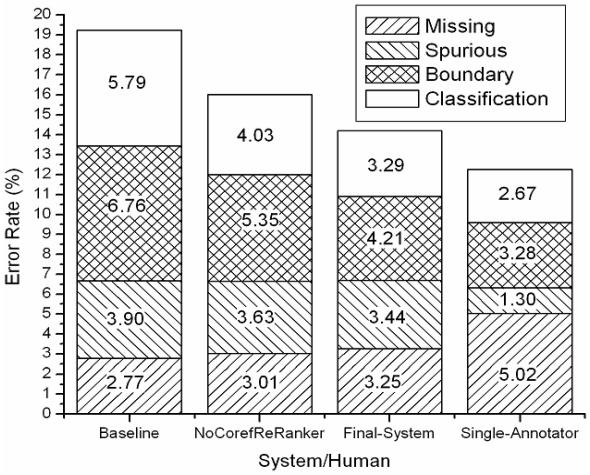


Figure 5. Error Distribution

Figure 5 shows that the performance improvement reflects a reduction in classification and boundary errors. Compared to the system, the human annotator's identification accuracy was much more skewed (52.3% missing, 13.5% spurious), suggesting that a major source of identification error was not difference in judgement but rather names which were simply overlooked by one annotator and picked up by the other. This further suggests that through an extension of our joint inference approach we may soon be able to exceed the performance of a single manual annotator.

Our analysis of the types of errors, and the performance of our knowledge sources, gives some indication of how these further gains may be achieved. The selectional force of event extraction was limited by the frequency of event patterns – only about 1/3 of sentences had a pattern instance. Even with this limitation, we obtained a gain of 0.5% in name classification. Capturing a broader range of selectional patterns should yield further improvements. Nearly 70% of the spurious names remaining in the final output were in fact instances of 'other' types of names, such as book titles and building names; creating explicit models of such names should improve performance. Finally, our cross-document coreference is currently performed only within the (small) test corpus. Retrieving related articles from a large collection should increase the likelihood of finding a name instance with a disambiguating context.

[8] Here *spurious* errors are names in the system response which do not overlap names in the key; *missing* errors are names in the key which do not overlap names in the system response; and *boundary* errors are names in the system response which *partially* overlap names in the key plus names in the key which partially overlap names in the system response.

Acknowledgment

This material is based upon work supported by the Defense Advanced Research Projects Agency under Contract No. HR0011-06-C-0023, and the National Science Foundation under Grant IIS-00325657. Any opinions, findings and conclusions expressed in this material are those of the authors and do not necessarily reflect the views of the U. S. Government.

References

Daniel M. Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. 1997. Nymble: a high-performance Learning Name-finder. *Proc. ANLP1997*. pp. 194-201., Washington, D.C.

Jianfeng Gao, Mu Li, Andi Wu and Chang-Ning Huang. 2005. Chinese Word Segmentation and Named Entity Recognition: A Pragmatic Approach. *Computational Linguistics 31(4)*. pp. 531-574

Heng Ji and Ralph Grishman. 2005. Improving Name Tagging by Reference Resolution and Relation Detection. *Proc. ACL2005*. pp. 411-418. Ann Arbor, USA.

Heng Ji, Cynthia Rudin and Ralph Grishman. 2006. Re-Ranking Algorithms for Name Tagging. *Proc. HLT/NAACL 06 Workshop on Computationally Hard Problems and Joint Inference in Speech and Language Processing*. New York, NY, USA

Dan Roth and Wen-tau Yih. 2004. A Linear Programming Formulation for Global Inference in Natural Language Tasks. *Proc. CONLL2004.*

Dan Roth and Wen-tau Yih. 2002. Probabilistic Reasoning for Entity & Relation Recognition. *Proc. COLING2002.*

Lufeng Zhai, Pascale Fung, Richard Schwartz, Marine Carpuat, and Dekai Wu. 2004. Using N-best Lists for Named Entity Recognition from Chinese Speech. *Proc. NAACL 2004 (Short Papers)*

Unsupervised Segmentation of Chinese Text by Use of Branching Entropy

Zhihui Jin and **Kumiko Tanaka-Ishii**
Graduate School of Information Science and Technology
University of Tokyo

Abstract

We propose an unsupervised segmentation method based on an assumption about language data: that the increasing point of entropy of successive characters is the location of a word boundary. A large-scale experiment was conducted by using 200 MB of unsegmented training data and 1 MB of test data, and precision of 90% was attained with recall being around 80%. Moreover, we found that the precision was stable at around 90% independently of the learning data size.

1 Introduction

The theme of this paper is the following assumption:

> The uncertainty of tokens coming after a sequence helps determine whether a given position is at a boundary. (A)

Intuitively, as illustrated in Figure 1, the variety of successive tokens at each character inside a word monotonically decreases according to the offset length, because the longer the preceding character n-gram, the longer the preceding context and the more it restricts the appearance of possible next tokens. For example, it is easier to guess which character comes after "natura" than after "na". On the other hand, the uncertainty at the position of a word border becomes greater, and the complexity increases, as the position is out of context. With the same example, it is difficult to guess which character comes after "natural ". This suggests that a word border can be detected by focusing on the differentials of the uncertainty of branching.

In this paper, we report our study on applying this assumption to Chinese word segmentation by formalizing the uncertainty of successive tokens via the branching entropy (which we mathematically define in the next section). Our intention in this paper is above all to study the fundamental and scientific statistical property underlying language data, so that it can be applied to language engineering.

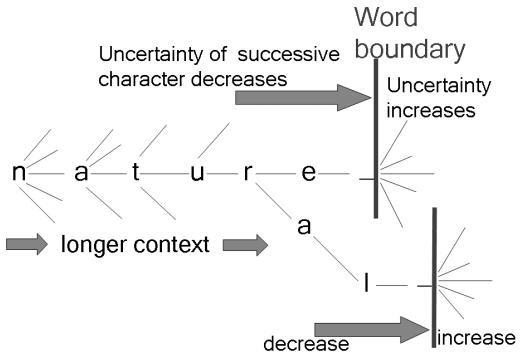


Figure 1: Intuitive illustration of a variety of successive tokens and a word boundary

The above assumption (A) dates back to the fundamental work done by Harris (Harris, 1955), where he says that when the number of different tokens coming after every prefix of a word marks the maximum value, then the location corresponds to the morpheme boundary. Recently, with the increasing availability of corpora, this property underlying language has been tested through segmentation into words and morphemes. Kempe (Kempe, 1999) reports a preliminary experiment to detect word borders in German and English texts by monitoring the entropy of successive characters for 4-grams. Also, the second author of this paper (Tanaka-Ishii, 2005) have shown how Japanese and Chinese can be segmented into words by formalizing the uncertainty with the branching entropy. Even though the test data was limited to a small amount in this work, the report suggested how assumption

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 428–435,
Sydney, July 2006.

(A) holds better when each of the sequence elements forms a semantic unit. This motivated our work to conduct a further, larger-scale test in the Chinese language, which is the only human language consisting entirely of ideograms (i.e., semantic units). In this sense, the choice of Chinese as the language in our work is essential.

If the assumption holds well, the most important and direct application is unsupervised text segmentation into words. Many works in unsupervised segmentation so far could be interpreted as formulating assumption (A) in a similar sense where branching stays low inside words but increases at a word or morpheme border. None of these works, however, is directly based on (A), and they introduce other factors within their overall methodologies. Some works are based on in-word branching frequencies formulated in an original evaluation function, as in (Ando and Lee, 2000) (boundary precision=84.5%,recall=78.0%, tested on 12500 Japanese ideogram words). Sun et al. (Sun et al., 1998) uses mutual information (boundary p=91.8%, no report for recall, 1588 Chinese characters), and Feng(Feng et al., 2004) incorporates branching counts in the evaluation function to be optimized for obtaining boundaries (word precision=76%, recall=78%, 2000 sentences). From the performance results listed here, we can see that unsupervised segmentation is more difficult, by far, than supervised segmentation; therefore, the algorithms are complex, and previous studies have tended to be limited in terms of both the test corpus size and the target.

In contrast, as assumption (A) is simple, we keep this simplicity in our formalization and directly test the assumption on a large-scale test corpus consisting of 1001 KB manually segmented data with the training corpus consisting of 200 MB of Chinese text.

Chinese is such an important language that supervised segmentation methods are already very mature. The current state-of-the-art segmentation software developed by (Low et al., 2005), which ranks as the best in the SIGHAN bakeoff (Emerson, 2005), attains word precision and recall of 96.9% and 96.8%, respectively, on the PKU track. There is also free

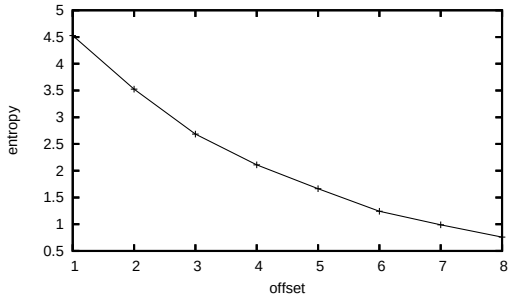


Figure 2: Decrease in $H(X|X_n)$ for Chinese characters when n is increased

software such as (Zhang et al., 2003) whose performance is also high. Even then, as most supervised methods learn on manually segmented newspaper data, when the input text is not from newspapers, the performance can be insufficient. Given that the construction of learning data is costly, we believe the performance can be raised by combining the supervised and unsupervised methods.

Consequently, this paper verifies assumption (A) in a fundamental manner for Chinese text and addresses the questions of why and to what extent (A) holds, when applying it to the Chinese word segmentation problem. We first formalize assumption (A) in a general manner.

2 The Assumption

Given a set of elements χ and a set of n-gram sequences χ_n formed of χ, the conditional entropy of an element occurring after an n-gram sequence X_n is defined as

$$H(X|X_n) = -\sum_{x_n \in \chi_n} P(x_n) \sum_{x \in \chi} P(x|x_n) \log P(x|x_n), \quad (1)$$

where $P(x) = P(X = x)$, $P(x|x_n) = P(X = x|X_n = x_n)$, and $P(X = x)$ indicates the probability of occurrence of x.

A well-known observation on language data states that $H(X|X_n)$ decreases as n increases (Bell et al., 1990). For example, Figure 2 shows how $H(X|X_n)$ shifts when n increases from 1 to 8 characters, where n is the length of a word prefix. This is calculated for all words existing in the test corpus, with the entropy being measured in the learning data (the learning and test data are defined in §4).

This phenomenon indicates that X will become easier to estimate as the context of X_n

gets longer. This can be intuitively understood: it is easy to guess that "e" will follow after "Hello! How ar", but it is difficult to guess what comes after the short string "He". The last term $-\log P(x|x_n)$ in the above formula indicates the information of a token of x coming after x_n, and thus the branching after x_n. The latter half of the formula, the local entropy value for a given x_n,

$$H(X|X_n = x_n) = -\sum_{x \in \chi} P(x|x_n) \log P(x|x_n), \quad (2)$$

indicates the average information of branching for a *specific* n-gram sequence x_n. As our interest in this paper is this local entropy, we denote $H(X|X_n = x_n)$ simply as $h(x_n)$ in the rest of this paper.

The decrease in $H(X|X_n)$ globally indicates that given an n-length sequence x_n and another $(n+1)$-length sequence y_{n+1}, the following inequality holds *on average*:

$$h(x_n) > h(y_{n+1}). \quad (3)$$

One reason why inequality (3) holds for language data is that there is *context* in language, and y_{n+1} carries a *longer context* as compared with x_n. Therefore, if we suppose that x_n is the prefix of x_{n+1}, then it is very likely that

$$h(x_n) > h(x_{n+1}) \quad (4)$$

holds, because the longer the preceding n-gram, the longer the *same* context. For example, it is easier to guess what comes after x_6="natura" than what comes after x_5 = "natur". Therefore, the decrease in $H(X|X_n)$ can be expressed as the concept that if the context is longer, the uncertainty of the branching decreases on average. Then, taking the logical contraposition, if the uncertainty does not decrease, the context is not longer, which can be interpreted as the following:

> If the entropy of successive tokens increases, the location is at a context border. (B)

For example, in the case of x_7 = "natural", the entropy h("natural") should be larger than h("natura"), because it is uncertain what character will allow x_7 to succeed. In the next section, we utilize assumption (B) to detect context boundaries.

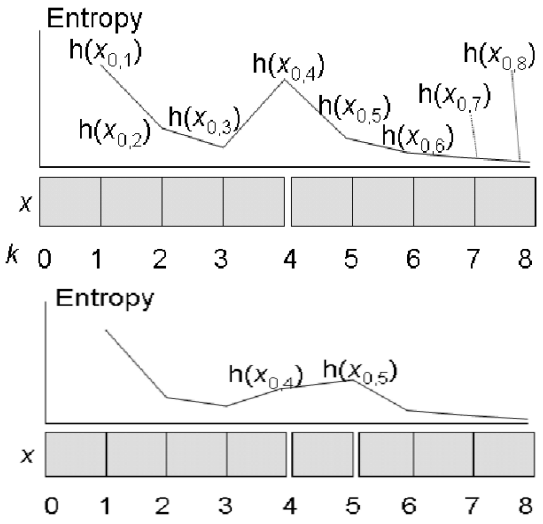


Figure 3: Our model for boundary detection based on the entropy of branching

3 Boundary Detection Using the Entropy of Branching

Assumption (B) gives a hint on how to utilize the branching entropy as an indicator of the context boundary. When two semantic units, both longer than 1, are put together, the entropy would appear as in the first figure of Figure 3. The first semantic unit is from offsets 0 to 4, and the second is from 4 to 8, with each unit formed by elements of χ. In the figure, one possible transition of the branching degree is shown, where the plot at k on the horizontal axis denotes the entropy for $h(x_{0,k})$ and $x_{n,m}$ denotes the substring between offsets n and m.

Ideally, the entropy would take a maximum at 4, because it will decrease as k is increased in the ranges of $k < 4$ and $4 < k < 8$, and at $k = 4$, it will rise. Therefore, the position at $k = 4$ is detected as the "local maximum value" when monitoring $h(x_{0,k})$ over k. The boundary condition after such observation can be redefined as the following:

B$_{max}$ Boundaries are locations where the entropy is locally maximized.

A similar method is proposed by Harris (Harris, 1955), where morpheme borders can be detected by using the local maximum of the number of different tokens coming after a prefix.

This only holds, however, for semantic units longer than 1. Units often have a length of

1, especially in our case with Chinese characters as elements, so that there are many one-character words. If a unit has length 1, then the situation will look like the second graph in Figure 3, where three semantic units, $x_{0,4}$, $x_{4,5}$, and $x_{5,8}$, are present, with the middle unit having length 1. First, at $k = 4$, the value of h increases. At $k = 5$, the value may increase or decrease, because the longer context results in an uncertainty decrease, *though an uncertainty decrease does not necessarily mean a longer context.* When h increases at $k = 5$, the situation will look like the second graph. In this case, the condition B_{max} will not suffice, and we need a second boundary condition:

$\mathrm{B}_{increase}$ Boundaries are locations where the entropy is increased.

On the other hand, when h decreases at $k = 5$, then even $\mathrm{B}_{increase}$ cannot be applied to detect $k = 5$ as a boundary. We have other chances to detect $k = 5$, however, by considering $h(x_{i,k})$, where $0 < i < k$. According to inequality (3), then, a similar trend should be present for plots of $h(x_{i,k})$, assuming that $h(x_{0,n}) > h(x_{0,n+1})$; then, we have

$$h(x_{i,n}) > h(x_{i,n+1}), \quad \text{for} \quad 0 < i < n. \qquad (5)$$

The value $h(x_{i,k})$ would hopefully rise for some i if the boundary at $k = 5$ is important, although $h(x_{i,k})$ can increase or decrease at $k = 5$, just as in the case for $h(x_{0,n})$.

Therefore, when the target language consists of many one-element units, $\mathrm{B}_{increase}$ is crucial for collecting all boundaries. Note that the boundaries detected by B_{max} are included in those detected by the condition $\mathrm{B}_{increase}$, and also that $\mathrm{B}_{increase}$ is a boundary condition representing the assumption (B) more directly.

So far, we have considered only regular-order processing: the branching degree is calculated for *successive* elements of x_n. We can also consider the reverse order, which involves calculating h for the *previous* element of x_n. In the case of the previous element, the question is whether the head of x_n forms the *beginning* of a context boundary.

Next, we move on to explain how we actually applied the above formalization to the problem of Chinese segmentation.

4 Data

The whole data for training amounted to 200 MB, from the Contemporary Chinese Corpus of the Center of Chinese Linguistics at Peking University (Center for Chinese Linguistics, 2006). It consists of several years of Peoples' Daily newspapers, contemporary Chinese literature, and some popular Chinese magazines. Note that as our method is unsupervised, this learning corpus is just text without any segmentation.

The test data were constructed by selecting sentences from the manually segmented People's Daily corpus of Peking University. In total, the test data amounts to 1001 KB, consisting 147026 Chinese words. The word boundaries indicated in the corpus were used as our golden standard.

As punctuation is clear from text boundaries in Chinese text, we pre-processed the test data by segmenting sentences at punctuation locations to form text fragments. Then, from all fragments, n-grams of less than 6 characters were obtained. The branching entropies for all these n-grams existing within the test data were obtained from the 200 MB of data.

We used 6 as the maximum n-gram length because Chinese words with a length of more than 5 characters are rare. Therefore, scanning the n-grams up to a length of 6 was sufficient. Another reason is that we actually conducted the experiment up to 8-grams, but the performance did not improve from when we used 6-grams.

Using this list of words ranging from unigrams to 6-grams and their branching entropies, the test data were processed so as to obtain the word boundaries.

5 Analysis for Small Examples

Figure 4 shows an actual graph of the entropy shift for the input phrase 未来发展的目标和指导 方针 (*wei lai fa zhan de mu biao he zhi dao fang zhen*, the aim and guideline of future development). The upper figure shows the entropy shift for the forward case, and the lower figure shows the entropy shift for the backward case. Note that for the backward case, the branching entropy was calculated for characters *before* the x_n.

In the upper figure, there are two lines, one

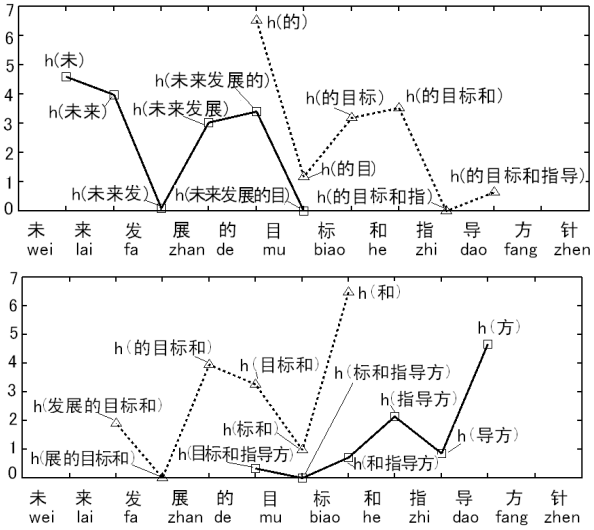


Figure 4: Entropy shift for a small example (forward and backward)

for the branching entropy after the substrings starting from 未. The leftmost line plots h(未), h(未来) ... h(未来发展的目). There are two increasing points, indicating that the phrase was segmented between 发展 and 的, and between 的 and 目标. The second line plots h(的) ...h(的目标和指导). The increasing locations are between 目标 and 和, between 和 and 指导, and after 指导.

The lower figure is the same. There are two lines, one for the branching entropy before the substring ending with suffix 方. The rightmost line plots h(方), h(导方) ...h(目标和指导方) running from back to front. We can see increasing points (as seen from back to front) between 和 and 指导, and between 的 and 目标. As for the last line, it also starts from 目 and runs from back to front, indicating boundaries between 的 and 目标, between 发展 and 的, and just before 发展.

If we consider all the increasing points in all four lines and take the set union of them, we obtain the correct segmentation as follows:

未来|发展|的| 目标|和|指导|方针,

which is the 100 % correct segmentation in terms of both recall and precision.

In fact, as there are 12 characters in this input, there should be 12 lines starting from each character for all substrings. For readability, however, we only show two lines each for the forward and backward cases. Also, the maximum length of a line is 6, because we only took 6-grams out of the learning data. If we consider all the increasing points in all 12 lines and take the set union, then we again obtain 100 % precision and recall. It is amazing how all 12 lines indicate only correct word boundaries.

Also, note how the correct full segmentation is obtained only with partial information from 4 lines taken from the 12 lines. Based on this observation, we next explain the algorithm that we used for a larger-scale experiment.

6 Algorithm for Segmentation

Having determined the entropy for all n-grams in the learning data, we could scan through each chunk of test data in both the forward order and the backward order to determine the locations of segmentation.

As our intention in this paper is above all to study the innate linguistic structure described by assumption (B), we do not want to add any artifacts other than this assumption. For such exact verification, we have to scan through all possible substrings of an input, which amounts to $O(n^2)$ computational complexity, where n indicates the input length of characters.

Usually, however, $h(x_{m,n})$ becomes impossible to measure when $n - m$ becomes large. Also, as noted in the previous section, words longer than 6 characters are very rare in Chinese text. Therefore, given a string x, all n-grams of no more than 6 grams are scanned, and the points where the boundary condition holds are output as boundaries.

As for the boundary conditions, we have B_{max} and $\mathrm{B}_{increase}$, and we also utilize $\mathrm{B}_{ordinary}$, where location n is considered as a boundary when the branching entropy $h(x_n)$ is simply above a given threshold. Precisely, there are three boundary conditions:

$\mathbf{B}_{max}$ $h(x_n) > valmax$,
where $h(x_n)$ takes a local maximum,
$\mathbf{B}_{increase}$ $h(x_{n+1}) - h(x_n) > valdelta$,
$\mathbf{B}_{ordinary}$ $h(x_n) > val$,

where $valmax$, $valdelta$, and val are arbitrary thresholds.

7 Large-Scale Experiments

7.1 Definition of Precision and Recall

Usually, when precision and recall are addressed in the Chinese word segmentation domain, they are calculated based on the number of *word*s. For example, consider a correctly segmented sequence "aaa|bbb|ccc|ddd", with a,b,c,d being characters and "|" indicating a word boundary. Suppose that the machine's result is "aaabbb|ccc|ddd"; then the correct words are only "ccc" and "ddd", giving a value of 2. Therefore, the precision is 2 divided by the number of words in the results (i.e., 3 for the words "aaabbb", "ccc", "ddd"), giving 67%, and the recall is 2 divided by the total number of words in the golden standard (i.e., 4 for the words "aaa","bbb", "ccc", "ddd") giving 50%. We call these values the word precision and recall, respectively, throughout this paper.

In our case, we use slightly different measures for the *boundary* precision and recall, which are based on the correct number of *boundaries*. These scores are also utilized especially in previous works on unsupervised segmentation (Ando and Lee, 2000) (Sun et al., 1998). Precisely,

$$Precision = \frac{N_{correct}}{N_{test}} \quad (6)$$

$$Recall = \frac{N_{correct}}{N_{true}}, \quad where \quad (7)$$

$N_{correct}$ is the number of correct boundaries in the result,

N_{test} is the number of boundaries in the test result, and,

N_{true} is the number of boundaries in the golden standard.

For example, in the case of the machine result being "aaabbb|ccc|ddd", the precision is 100% and the recall is 75%. Thus, we consider there to be no imprecise result as a boundary in the output of "aaabbb|ccc|ddd".

The crucial reason for using the boundary precision and recall is that boundary detection and word extraction are not exactly the same task. In this sense, assumption (A) or (B) is a general assumption about a *boundary* (of a sentence, phrase, word, morpheme). Therefore, the boundary precision and recall measure serves for directly measuring boundaries.

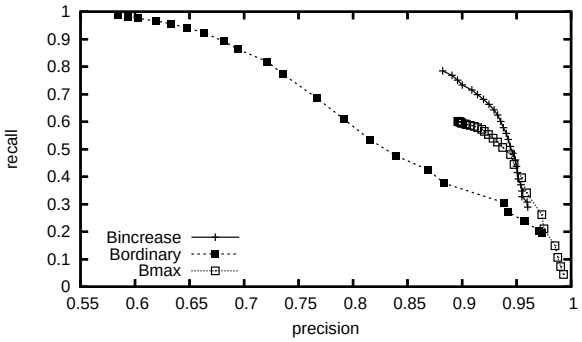


Figure 5: Precision and recall

Note that all precision and recall scores from now on in this paper are boundary precision and recall. Even in comparing the supervised methods with our unsupervised method later, the precision and recall values are all *recalculated* as boundary precision and recall.

7.2 Precision and Recall

The precision and recall graph is shown in Figure 5. The horizontal axis is the precision and the vertical axis is the recall. The three lines from right to left (top to bottom) correspond to $B_{increase}$ ($0.0 \leq valdelta \leq 2.4$), B_{max} ($4.0 \leq valmax \leq 6.2$), and $B_{ordinary}$ ($4.0 \leq val \leq 6.2$). All are plotted with an interval of 0.1. For every condition, the larger the threshold, the higher the precision and the lower the recall.

We can see how $B_{increase}$ and B_{max} keep high precision as compared with $B_{ordinary}$. We also can see that the boundary can be more easily detected if it is judged as comprising the proximity value of $h(x_n)$.

For $B_{increase}$, in particular, when $valdelta = 0.0$, the precision and recall are still at 0.88 and 0.79, respectively. Upon increasing the threshold to $valdelta = 2.4$, the precision is higher than 0.96 at the cost of a low recall of 0.29. As for B_{max}, we also observe a similar tendency but with low recall due to the smaller number of local maximum points as compared with the number of increasing points. Thus, we see how $B_{increase}$ attains a better performance among the three conditions. This shows the correctness of assumption (B).

From now on, we consider only $B_{increase}$ and proceed through our other experiments.

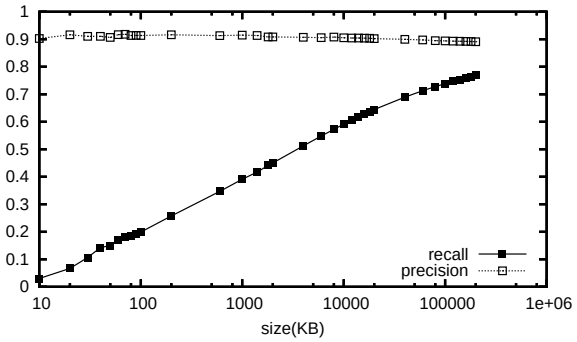


Figure 6: Precision and recall depending on training data size

Next, we investigated how the training data size affects the precision and recall. This time, the horizontal axis is the amount of learning data, varying from 10 KB up to 200 MB, on a log scale. The vertical axis shows the precision and recall. The boundary condition is $B_{increase}$ with $valdelta = 0.1$.

We can see how the precision always remains high, whereas the recall depends on the amount of data. The precision is stable at an amazingly high value, even when the branching entropy is obtained from a very small corpus of 10 KB. Also, the linear increase in the recall suggests that if we had more than 200 MB of data, we would expect to have an even higher recall. As the horizontal axis is in a log scale, however, we would have to have gigabytes of data to achieve the last several percent of recall.

7.3 Error Analysis

According to our manual error analysis, the top-most three errors were the following:

- Numbers: dates, years, quantities (example: 1998, written in Chinese number characters)
- One-character words (example:在(at) 又(again) 向(toward) 和(and))
- Compound Chinese words (example: 解放思想(open mind) being segmented into 解放(open) and 思想(mind))

The reason for the bad results with numbers is probably because the branching entropy for digits is less biased than for usual ideograms. Also, for one-character words, our method is limited, as we explained in §3. Both of these two problems, however, can be solved by applying special preprocessing for numbers and one-character words, given that many of the one-character words are functional characters, which are limited in number. Such improvements remain for our future work.

The third error type, in fact, is one that could be judged as correct segmentation. In the case of "open mind", it was not segmented into two words in the golden standard; therefore, our result was judged as incorrect. This could, however, be judged as correct.

The structures of Chinese words and phrases are very similar, and there are no clear criteria for distinguishing between a word and a phrase. The unsupervised method determines the structure and segments words and phrases into smaller pieces. Manual recalculation of the accuracy comprising such cases also remains for our future work.

8 Conclusion

We have reported an unsupervised Chinese segmentation method based on the branching entropy. This method is based on an assumption that "if the entropy of successive tokens increases, the location is at the context border." The entropies of n-grams were learned from an unsegmented 200-MB corpus, and the actual segmentation was conducted directly according to the above assumption, on 1 MB of test data. We found that the precision was as high as 90% with recall being around 80%. We also found an amazing tendency for the precision to always remain high, regardless of the size of the learning data.

There are two important considerations for our future work. The first is to figure out how to combine the supervised and unsupervised methods. In particular, as the performance of the supervised methods could be insufficient for data that are not from newspapers, there is the possibility of combining the supervised and unsupervised methods to achieve a higher accuracy for general data. The second future work is to verify our basic assumption in other languages. In particular, we should undertake experimental studies in languages written with phonogram characters.

References

R.K. Ando and L. Lee. 2000. Mostly-unsupervised statistical segmentation of Japanese: Applications to kanji. In *ANLP-NAACL*.

T.C. Bell, J.G. Cleary, and Witten. I.H. 1990. *Text Compression*. Prentice Hall.

Center for Chinese Linguistics. 2006. Chinese corpus. visited 2006, searchable from `http://ccl.pku.edu.cn/YuLiao_Contents.Asp`, part of it freely available from `http://www.icl.pku.edu.cn`.

T. Emerson. 2005. The second international chinese word segmentation bakeoff. In *SIGHAN*.

H.D. Feng, K. Chen, C.Y. Kit, and Deng. X.T. 2004. Unsupervised segmentation of chinese corpus using accessor variety. In *IJCNLP*, pages 255–261.

S.Z. Harris. 1955. From phoneme to morpheme. *Language*, pages 190–222.

A. Kempe. 1999. Experiments in unsupervised entropy-based corpus segmentation. In *Workshop of EACL in Computational Natural Language Learning*, pages 7–13.

J.K. Low, H.T Ng, and W. Guo. 2005. A maximum entropy approach to chinese word segmentation. In *SIGHAN*.

M. Sun, D. Shen, and B. K. Tsou. 1998. Chinese word segmentation without using lexicon and hand-crafted training data. In *COLING-ACL*.

K. Tanaka-Ishii. 2005. Entropy as an indicator of context boundaries —an experiment using a web search engine —. In *IJCNLP*, pages 93–105.

H.P. Zhang, Yu H.Y., Xiong D.Y., and Q Liu. 2003. Hhmm-based chinese lexical analyzer ictclas. In *SIGHAN*. visited 2006, available from `http://www.nlp.org.cn`.

A FrameNet-based Semantic Role Labeler for Swedish

Richard Johansson and **Pierre Nugues**
Department of Computer Science, LTH
Lund University, Sweden
{richard, pierre}@cs.lth.se

Abstract

We present a FrameNet-based semantic role labeling system for Swedish text. As training data for the system, we used an annotated corpus that we produced by transferring FrameNet annotation from the English side to the Swedish side in a parallel corpus. In addition, we describe two frame element bracketing algorithms that are suitable when no robust constituent parsers are available.

We evaluated the system on a part of the FrameNet example corpus that we translated manually, and obtained an accuracy score of 0.75 on the classification of pre-segmented frame elements, and precision and recall scores of 0.67 and 0.47 for the complete task.

1 Introduction

Semantic role labeling (SRL), the process of automatically identifying arguments of a predicate in a sentence and assigning them semantic roles, has received much attention during the recent years. SRL systems have been used in a number of projects in Information Extraction and Question Answering, and are believed to be applicable in other domains as well.

Building SRL systems for English has been studied widely (Gildea and Jurafsky, 2002; Litkowski, 2004), *inter alia*. However, all these works rely on corpora that have been produced at the cost of a large effort by human annotators. For instance, the current FrameNet corpus (Baker et al., 1998) consists of 130,000 manually annotated sentences. For smaller languages such as Swedish, such corpora are not available.

In this work, we describe a FrameNet-based semantic role labeler for Swedish text. Since there was no existing training corpus available — no FrameNet-annotated Swedish corpus of substantial size exists — we used an English-Swedish parallel corpus whose English part was annotated with semantic roles using the FrameNet annotation scheme. We then applied a *cross-language transfer* to derive an annotated Swedish part. To evaluate the performance of the Swedish SRL system, we applied it to a small portion of the FrameNet example corpus that we translated manually.

1.1 FrameNet: an Introduction

FrameNet (Baker et al., 1998) is a lexical database that describes English words using Frame Semantics (Fillmore, 1976). In this framework, predicates (or in FrameNet terminology, *target words*) and their arguments are linked by means of *semantic frames*. A frame can intuitively be thought of as a template that defines a set of slots, *frame elements* (FEs), that represent parts of the conceptual structure and typically correspond to prototypical participants or properties.

Figure 1 shows an example sentence annotated with FrameNet information. In this example, the target word *statements* belongs to ("evokes") the frame STATEMENT. Two constituents that fill slots of the frame (SPEAKER and TOPIC) are annotated as well.

As usual in these cases, [both parties]$_{\text{SPEAKER}}$ agreed to make no further **statements** [on the matter]$_{\text{TOPIC}}$.

Figure 1: A sentence from the FrameNet example corpus.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 436–443,
Sydney, July 2006.

The initial versions of FrameNet were focused on describing situations and events, i.e. typically verbs and their nominalizations. Currently, however, FrameNet defines frames for a wider range of semantic relations that can be thought of as predicate/argument structures, including descriptions of events, states, properties, and objects.

FrameNet consists of the following main parts:

- An *ontology* consisting of a set of frames, frame elements for each frame, and relations (such as inheritance and causative-of) between frames.
- A list of *lexical units*, that is word forms paired with their corresponding frames. The frame is used to distinguish between different senses of the word, although the treatment of polysemy in FrameNet is relatively coarse-grained.
- A collection of *example sentences* that provide lexical evidence for the frames and the corresponding lexical units. Although this corpus is not intended to be representative, it is typically used as a training corpus when contructing automatic FrameNet labelers.

1.2 Related Work

Since training data is often a scarce resource for most languages other than English, a wide range of methods have been proposed to reduce the need for manual annotation. Many of these have relied on existing resources for English and a transfer method based on word alignment in a parallel corpus to automatically create an annotated corpus in a new language. Although these data are typically quite noisy, they have been used to train automatic systems.

For the particular case of transfer of FrameNet annotation, there have been a few projects that have studied transfer methods and evaluated the quality of the automatically produced corpus. Johansson and Nugues (2005) applied the word-based methods of Yarowsky et al. (2001) and obtained promising results. Another recent effort (Padó and Lapata, 2005) demonstrates that deeper linguistic information, such as parse trees in the source and target language, is very beneficial for the process of FrameNet annotation transfer.

A rather different method to construct bilingual semantic role annotation is the approach taken by BiFrameNet (Fung and Chen, 2004). In that work, annotated structures in a new language (in that case Chinese) are produced by mining for similar structures rather than projecting them via parallel corpora.

2 Automatic Annotation of a Swedish Training Corpus

2.1 Training an English Semantic Role Labeler

We selected the 150 most frequent frames in FrameNet and applied the Collins parser (Collins, 1999) to the example sentences for these frames. We built a conventional FrameNet parser for English using 100,000 of these sentences as a training set and 8,000 as a development set. The classifiers were based on Support Vector Machines that we trained using LIBSVM (Chang and Lin, 2001) with the Gaussian kernel. When testing the system, we did not assume that the frame was known a priori. We used the available semantic roles for all senses of the target word as features for the classifier.

On a test set from FrameNet, we estimated that the system had a precision of 0.71 and a recall of 0.65 using a strict scoring method. The result is slightly lower than the best systems at Senseval-3 (Litkowski, 2004), possibly because we used a larger set of frames, and we did not assume that the frame was known a priori.

2.2 Transferring the Annotation

We produced a Swedish-language corpus annotated with FrameNet information by applying the SRL system to the English side of Europarl (Koehn, 2005), which is a parallel corpus that is derived from the proceedings of the European Parliament. We projected the bracketing of the target words and the frame elements onto the Swedish side of the corpus by using the Giza++ word aligner (Och and Ney, 2003). Each word on the English side was mapped by the aligner onto a (possibly empty) set of words on the Swedish side. We used the maximal span method to infer the bracketing on the Swedish side, which means that the span of a projected entity was set to the range from the leftmost projected token to the rightmost. Figure 2 shows an example of this process.

To make the brackets conform to the FrameNet annotation practices, we applied a small set of heuristics. The FrameNet conventions specify that linking words such as prepositions and subordinat-

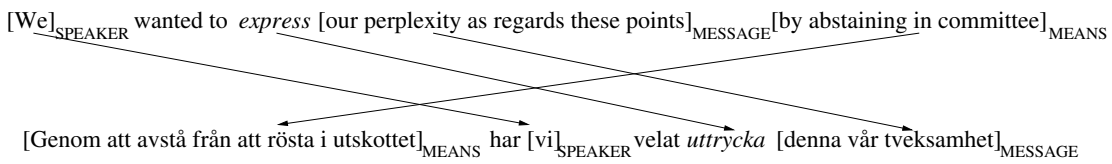


Figure 2: Example of projection of FrameNet annotation.

ing conjunctions should be included in the bracketing. However, since constructions are not isomorphic in the sentence pair, a linking word on the target side may be missed by the projection method since it is not present on the source side. For example, the sentence *the doctor was answering an emergency phone call* is translated into Swedish as *doktorn svarade **på** ett larmsamtal*, which uses a construction with a preposition *på* 'to/at/on' that has no counterpart in the English sentence. The heuristics that we used are specific for Swedish, although they would probably be very similar for any other language that uses a similar set of prepositions and connectives, i.e. most European languages.

We used the following heuristics:

- When there was only a linking word (preposition, subordinating conjunction, or infinitive marker) between the FE and the target word, it was merged with the FE.
- When a Swedish FE was preceded by a linking word, and the English FE starts with such a word, it was merged with the FE.
- We used a chunker and adjusted the FE brackets to include only complete chunks.
- When a Swedish FE crossed the target word, we used only the part of the FE that was on the right side of the target.

In addition, some bad annotation was discarded because we obviously could not use sentences where no counterpart for the target word could be found. Additionally, we used only the sentences where the target word was mapped to a noun, verb, or an adjective on the Swedish side.

Because of homonymy and polysemy problems, applying a SRL system without knowing target words and frames a priori necessarily introduces noise into the automatically created training corpus. There are two kinds of word sense ambiguity that are problematic in this case: the "internal" ambiguity, or the fact that there may be more than one frame for a given target word; and the "external" ambiguity, where frequently occurring word senses are not listed in FrameNet. To sidestep the problem of internal ambiguity, we used the available semantic roles for all senses of the target word as features for the classifier (as described above). Solving the problem of external ambiguity was outside the scope of this work.

Some potential target words had to be ignored since their sense ambiguity was too difficult to overcome. This category includes auxiliaries such as *be* and *have*, as well as verbs such as *take* and *make*, which frequently appear as support verbs for nominal predicates.

2.3 Motivation

Although the meaning of the two sentences in a sentence pair in a parallel corpus should be roughly the same, a fundamental question is whether it is meaningful to project semantic markup of text across languages. Equivalent words in two different languages sometimes exhibit subtle but significant semantic differences. However, we believe that a transfer makes sense, since the nature of FrameNet is rather coarse-grained. Even though the words that evoke a frame may not have exact counterparts, it is probable that the frame itself has.

For the projection method to be meaningful, we must make the following assumptions:

- The complete frame ontology in the English FrameNet is meaningful in Swedish as well, and each frame has the same set of semantic roles and the same relations to other frames.
- When a target word evokes a certain frame in English, it has a counterpart in Swedish that evokes the same frame.
- Some of the FEs on the English side have counterparts with the same semantic roles on the Swedish side.

In addition, we made the (obviously simplistic) assumption that the contiguous entities we project are also contiguous on the target side.

These assumptions may all be put into question. Above all, the second assumption will fail in many cases because the translations are not literal, which means that the sentences in the pair may express slightly different information. The third assumption may be invalid if the information expressed is realized by radically different constructions, which means that an argument may belong to another predicate or change its semantic role on the Swedish side. Padó and Lapata (2005) avoid this problem by using heuristics based on a target-language FrameNet to select sentences that are close in meaning. Since we have no such resource to rely on, we are forced to accept that this problem introduces a certain amount of noise into the automatically annotated corpus.

3 Training a Swedish SRL System

Using the transferred FrameNet annotation, we trained a SRL system for Swedish text. Like most previous systems, it consists of two parts: a FE bracketer and a classifier that assigns semantic roles to FEs. Both parts are implemented as SVM classifiers trained using LIBSVM. The semantic role classifier is rather conventional and is not described in this paper.

To construct the features used by the classifiers, we used the following tools:

- An HMM-based POS tagger,
- A rule-based chunker,
- A rule-based time expression detector,
- Two clause identifiers, of which one is rule-based and one is statistical,
- The MALTPARSER dependency parser (Nivre et al., 2004), trained on a 100,000-word Swedish treebank.

We constructed shallow parse trees using the clause trees and the chunks. Dependency and shallow parse trees for a fragment of a sentence from our test corpus are shown in Figures 3 and 4, respectively. This sentence, which was translated from an English sentence that read *the doctor was answering an emergency phone call*, comes from the English FrameNet example corpus.

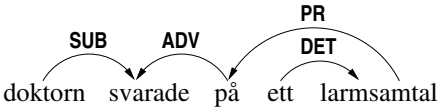


Figure 3: Example dependency parse tree.

[[doktorn]$_{NG\_nom}$ [svarade]$_{VG\_fin}$ [på]$_{PP}$ [ett larmsamtal]$_{NG\_nom}$]$_{Clause}$

Figure 4: Example shallow parse tree.

3.1 Frame Element Bracketing Methods

We created two redundancy-based FE bracketing algorithms based on binary classification of chunks as starting or ending the FE. This is somewhat similar to the chunk-based system described by Pradhan et al. (2005a), which uses a segmentation strategy based on IOB2 bracketing. However, our system still exploits the dependency parse tree during classification.

We first tried the conventional approach to the problem of FE bracketing: applying a parser to the sentence, and classifying each node in the parse tree as being an FE or not. We used a dependency parser since there is no constituent-based parser available for Swedish. This proved unsuccessful because the spans of the dependency subtrees frequently were incompatible with the spans defined by the FrameNet annotations. This was especially the case for non-verbal target words and when the head of the argument was above the target word in the dependency tree. To be usable, this approach would require some sort of transformation, possibly a conversion into a phrase-structure tree, to be applied to the dependency trees to align the spans with the FEs. Preliminary investigations were unsuccessful, and we left this to future work.

We believe that the methods we developed are more suitable in our case, since they base their decisions on several parse trees (in our case, two clause-chunk trees and one dependency tree). This redundancy is valuable because the dependency parsing model was trained on a treebank of just 100,000 words, which makes it less robust than Collins' or Charniak's parsers for English. In addition, the methods do not implicitly rely on the common assumption that every FE has a counterpart in a parse tree. Recent work in semantic role labeling, see for example Pradhan et al. (2005b), has focused on combining the results of SRL systems based on different types of syntax. Still, all

systems exploiting recursive parse trees are based on binary classification of nodes as being an argument or not.

The training sets used to train the final classifiers consisted of one million training instances for the start classifier, 500,000 for the end classifier, and 272,000 for the role classifier. The features used by the classifiers are described in Subsection 3.2, and the performance of the two FE bracketing algorithms compared in Subsection 4.2.

3.1.1 Greedy start-end

The first FE bracketing algorithm, the *greedy start-end* method, proceeds through the sequence of chunks in one pass from left to right. For each chunk opening bracket, a binary classifier decides if an FE starts there or not. Similarly, another binary classifier tests chunk end brackets for ends of FEs. To ensure compliance to the FrameNet annotation standard (bracket matching, and no FE crossing the target word), the algorithm inserts additional end brackets where appropriate. Pseudocode is given in Algorithm 1.

Algorithm 1 Greedy Bracketing

Input: A list L of chunks and a target word t
  Binary classifiers `starts` and `ends`
Output: The sets S and E of start and end brackets
  Split L into the sublists L_{before}, L_{target}, and L_{after}, which correspond to the parts of the list that is before, at, and after the target word, respectively.
  Initialize `chunk-open` to FALSE
  for L_{sub} **in** $\{L_{\text{before}}, L_{\text{target}}, L_{\text{after}}\}$ **do**
    for c **in** L_{sub} **do**
      if `starts`(c) **then**
        if `chunk-open` **then**
          Add an end bracket before c to E
        end if
        `chunk-open` $\leftarrow$ TRUE
        Add a start bracket before c to S
      end if
      if `chunk-open` $\wedge$ (`ends`$(c) \vee c$ is final in L_{sub}) **then**
        `chunk-open` $\leftarrow$ FALSE
        Add an end bracket after c to E
      end if
    end for
  end for

Figure 5 shows an example of this algorithm, applied to the example fragment. The small brackets correspond to chunk boundaries, and the large brackets to FE boundaries that the algorithm inserts. In the example, the algorithm inserts an end bracket after the word *doktorn* 'the doctor', since no end bracket was found before the target word *svarade* 'was answering'.

3.1.2 Globally optimized start-end

The second algorithm, the *globally optimized start-end* method, maximizes a global probability score over each sentence. For each chunk opening and closing bracket, probability models assign

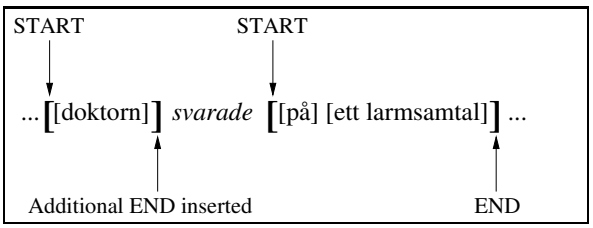


Figure 5: Illustration of the greedy start-end method.

the probability of an FE starting (or ending, respectively) at that chunk. The probabilities are estimated using the built-in sigmoid fitting methods of LIBSVM. Making the somewhat unrealistic assumption of independence of the brackets, the global probability score to maximize is defined as the product of all start and end probabilities. We added a set of constraints to ensure that the segmentation conforms to the FrameNet annotation standard. The constrained optimization problem is then solved using the JACOP finite domain constraint solver (Kuchcinski, 2003). We believe that an n-best beam search method would produce similar results. The pseudocode for the method can be seen in Algorithm 2. The definitions of the predicates `no-nesting` and `no-crossing`, which should be obvious, are omitted.

Algorithm 2 Globally Optimized Bracketing

Input: A list L of chunks and a target word t
  Probability models $\hat{P}_{\text{starts}}$ and $\hat{P}_{\text{ends}}$
Output: The sets $S_{\max}$ and $E_{\max}$ of start and end brackets

$$\begin{aligned}
\texttt{legal}(S,E) \quad &\leftarrow \quad |S| = |E| \\
&\wedge \quad \max(E) > \max(S) \wedge \min(S) < \min(E) \\
&\wedge \quad \texttt{no-nesting}(S,E) \wedge \texttt{no-crossing}(t,S,E) \\
\texttt{score}(S,E) \quad &\leftarrow \quad \prod_{c\in S} \hat{P}_{\text{starts}}(c) \cdot \prod_{c\in L\setminus S}(1-\hat{P}_{\text{starts}}(c)) \\
&\quad\ \cdot \prod_{c\in E} \hat{P}_{\text{ends}}(c) \cdot \prod_{c\in L\setminus E}(1-\hat{P}_{\text{ends}}(c))
\end{aligned}$$

$$(S_{\max}, E_{\max}) \leftarrow \operatorname{argmax}_{\{\texttt{legal}(S,E)\}} \texttt{score}(S,E)$$

Figure 6 shows an example of the globally optimized start-end method. In the example, the global probability score is maximized by a bracketing that is illegal because the FE starting at *doktorn* is not closed before the target ($0.8 \cdot 0.6 \cdot 0.6 \cdot 0.7 \cdot 0.8 \cdot 0.7 = 0.11$). The solution of the constrained problem is a bracketing that contains an end bracket before the target ($0.8 \cdot 0.4 \cdot 0.6 \cdot 0.7 \cdot 0.8 \cdot 0.7 = 0.075$)

3.2 Features Used by the Classifiers

Table 1 summarizes the feature sets used by the greedy start-end (GSE), optimized start-end (OSE), and semantic role classification (SRC).

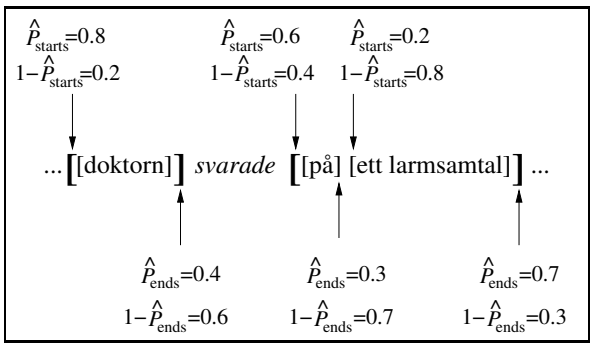


Figure 6: Illustration of the globally optimized start-end method.

	GSE	OSE	SRC
Target lemma	+	+	+
Target POS	+	+	+
Voice	+	+	+
Allowed role labels	+	+	+
Position	+	+	+
Head word (HW)	+	+	+
Head POS	+	+	+
Phrase/chunk type (PT)	+	+	+
HW/POS/PT, ±2 chunk window	+	+	-
Dep-tree & shallow path →target	+	+	+
Starting paths →target	+	+	-
Ending paths →target	+	+	-
Path →start	+	-	-

Table 1: Features used by the classifiers.

3.2.1 Conventional Features

Most of the features that we use have been used by almost every system since the first well-known description (Gildea and Jurafsky, 2002). The following of them are used by all classifiers:

- *Target word (predicate) lemma and POS*
- *Voice* (when the target word is a verb)
- *Position* (before or after the target)
- *Head word and POS*
- *Phrase or chunk type*

In addition, all classifiers use the set of allowed semantic role labels as a set of boolean features. This is needed to constrain the output to a label that is allowed by FrameNet for the current frame. In addition, this feature has proven useful for the FE bracketing classifiers to distinguish between event-type and object-type frames. For event-type frames, dependencies are often long-distance, while for object-type frames, they are typically restricted to chunks very near the target word. The part of speech of the target word alone is not enough to distinguish these two classes, since many nouns belong to event-type frames.

For the phrase/chunk type feature, we use slightly different values for the bracketing case and the role assignment case: for bracketing, the value of this feature is simply the type of the current chunk; for classification, it is the type of the largest chunk or clause that starts at the leftmost token of the FE. For prepositional phrases, the preposition is attached to the phrase type (for example, the second FE in the example fragment starts with the preposition *på* 'at/on', which causes the value of the phrase type feature to be PP-*på*).

3.2.2 Chunk Context Features

Similarly to the chunk-based PropBank argument bracketer described by Pradhan et al. (2005a), the start-end methods use the head word, head POS, and chunk type of chunks in a window of size 2 on both sides of the current chunk to classify it as being the start or end of an FE.

3.2.3 Parse Tree Path Features

Parse tree path features have been shown to be very important for argument bracketing in several studies. All classifiers used here use a set of such features:

- *Dependency tree path from the head to the target word.* In the example text, the first chunk (consisting of the word *doktorn*), has the value SUB-↑ for this feature. This means that to go from the head of the chunk to the target in the dependency graph (Figure 3), you traverse a SUB (subject) link upwards. Similarly, the last chunk (*ett larmsamtal*) has the value PR-↑-ADV-↑.

- *Shallow path from the chunk containing the head to the target word.* For the same chunks as above, these values are both NG_nom-↑-Clause-↓-VG_fin, which means that to traverse the shallow parse tree (Figure 4) from the chunk to the target, you start with a NG_nom node, go upwards to a Clause node, and finally down to the VG_fin node.

The start-end classifiers additionally use the full set of paths (dependency and shallow paths) to the target word from each node starting (or ending, respectively) at the current chunk, and the greedy end classifier also uses the path from the current chunk to the start chunk.

4 Evaluation of the System

4.1 Evaluation Corpus

To evaluate the system, we manually translated 150 sentences from the FrameNet example corpus. These sentences were selected randomly from the English development set. Some sentences were removed, typically because we found the annotation dubious or the meaning of the sentence difficult to comprehend precisely. The translation was mostly straightforward. Because of the extensive use of compounding in Swedish, some frame elements were merged with target words.

4.2 Comparison of FE Bracketing Methods

We compared the performance of the two methods for FE bracketing on the test set. Because of limited time, we used smaller training sets than for the full evaluation below (100,000 training instances for all classifiers). Table 2 shows the result of this comparison.

	Greedy	Optimized
Precision	0.70	0.76
Recall	0.50	0.44
$F_{\beta=1}$	0.58	0.55

Table 2: Comparison of FE bracketing methods.

As we can see from the Table 2, the globally optimized start-end method increased the precision somewhat, but decreased the recall and made the overall F-measure lower. We therefore used the greedy start-end method for our final evaluation that is described in the next section.

4.3 Final System Performance

We applied the Swedish semantic role labeler to the translated sentences and evaluated the result. We used the conventional experimental setting where the frame and the target word were given in advance. The results, with approximate 95% confidence intervals included, are presented in Table 3. The figures are precision and recall for the full task, classification accuracy of pre-segmented arguments, precision and recall for the bracketing task, full task precision and recall using the Senseval-3 scoring metrics, and finally the proportion of full sentences whose FEs were correctly bracketed and classified. The Senseval-3 method uses a more lenient scoring scheme that counts a FE as correctly identified if it overlaps with the gold standard FE and has the correct label. Although the strict measures are more interesting, we include these figures for comparison with the systems participating in the Senseval-3 Restricted task (Litkowski, 2004).

We include baseline scores for the argument bracketing and classification tasks, respectively. The bracketing baseline method considers non-punctuation subtrees dependent of the target word. When the target word is a verb, the baseline puts FE brackets around the words included in each of these subtrees[1]. When the target is a noun, we also bracket the target word token itself, and when it is an adjective, we additionally bracket its parent token. As a baseline for the argument classification task, every argument is assigned the most frequent semantic role in the frame. As can be seen from the table, all scores except the argument bracketing recall are well above the baselines.

Precision (Strict scoring method)	0.67 ± 0.064
Recall	0.47 ± 0.057
Argument Classification Accuracy	0.75 ± 0.050
Baseline	0.41 ± 0.056
Argument Bracketing Precision	0.80 ± 0.055
Baseline Precision	0.50 ± 0.055
Argument Bracketing Recall	0.57 ± 0.057
Baseline Recall	0.55 ± 0.057
Precision (Senseval-3 scoring method)	0.77 ± 0.057
Overlap	0.75 ± 0.039
Recall	0.55 ± 0.057
Complete Sentence Accuracy	0.29 ± 0.073

Table 3: Results on the Swedish test set with approximate 95% confidence intervals.

Although the performance figures are better than the baselines, they are still lower than for most English systems (although higher than some of the systems at Senseval-3). We believe that the main reason for the performance is the quality of the data that were used to train the system, since the results are consistent with the hypothesis that the quality of the transferred data was roughly equal to the performance of the English system multiplied by the figures for the transfer method (Johansson and Nugues, 2005). In that experiment, the transfer method had a precision of 0.84, a recall of 0.81, and an F-measure of 0.82. If we assume that the transfer performance is similar for Swedish, we arrive at a precision of $0.71 \cdot 0.84 = 0.60$, a recall of $0.65 \cdot 0.81 = 0.53$,

[1] This is possible because MALTPARSER produces projective trees, i.e. the words in each subtree form a contiguous substring of the sentence.

and an F-measure of 0.56. For the F-measure, 0.55 for the system and 0.56 for the product, the figures match closely. For the precision, the system performance (0.67) is significantly higher than the product (0.60), which suggests that the SVM learning method handles the noisy training set rather well for this task. The recall (0.47) is lower than the corresponding product (0.53), but the difference is not statistically significant at the 95% level. These figures suggest that the main effort towards improving the system should be spent on improving the training data.

5 Conclusion

We have described the design and implementation of a Swedish FrameNet-based SRL system that was trained using a corpus that was annotated using cross-language transfer from English to Swedish. With no manual effort except for translating sentences for evaluation, we were able to reach promising results. To our knowledge, the system is the first SRL system for Swedish in literature. We believe that the methods described could be applied to any language, as long as there exists a parallel corpus where one of the languages is English. However, the relatively close relationship between English and Swedish probably made the task comparatively easy in our case.

As we can see, the figures (especially the FE bracketing recall) leave room for improvement for the system to be useful in a fully automatic setting. Apart from the noisy training set, probable reasons for this include the lower robustness of the Swedish parsers compared to those available for English. In addition, we have noticed that the European Parliament corpus is somewhat biased. For instance, a very large proportion of the target words evoke the STATEMENT or DISCUSSION frames, but there are very few instances of the BEING_WET and MAKING_FACES frames. While training, we tried to balance the selection somewhat, but applying the projection methods on other types of parallel corpora (such as novels available in both languages) may produce a better training corpus.

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet Project. In *Proceedings of COLING-ACL'98*, pages 86–90, Montréal, Canada.

Chih-Chung Chang and Chih-Jen Lin, 2001. *LIBSVM: a library for support vector machines*.

Michael J. Collins. 1999. Head-driven statistical models for natural language parsing. Ph.D. thesis, University of Pennsylvania, Philadelphia.

Charles J. Fillmore. 1976. Frame semantics and the nature of language. *Annals of the New York Academy of Sciences: Conference on the Origin and Development of Language*, 280:20–32.

Pascale Fung and Benfeng Chen. 2004. BiFrameNet: Bilingual frame semantics resource construction by cross-lingual induction. In *Proceedings of COLING-2004*.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Richard Johansson and Pierre Nugues. 2005. Using parallel corpora for automatic transfer of FrameNet annotation. In *Proceedings of the 1st ROMANCE FrameNet Workshop*, Cluj-Napoca, Romania, 26-28 July.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT Summit 2005*.

Krzysztof Kuchcinski. 2003. Constraints-driven scheduling and resource assignment. *ACM Transactions on Design Automation of Electronic Systems*, 8(3):355–383.

Ken Litkowski. 2004. Senseval-3 task: Automatic labeling of semantic roles. In *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 9–12.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2004. Memory-based dependency parsing. In *Proceedings of CoNLL-2004*, pages 49–56.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Sebastian Padó and Mirella Lapata. 2005. Cross-lingual projection of role-semantic information. In *Proceedings of HLT/EMNLP 2005*.

Sameer Pradhan, Kadri Hacioglu, Valerie Krugler, Wayne Ward, James Martin, and Dan Jurafsky. 2005a. Support vector learning for semantic argument classification. *Machine Learning*, 60(1):11–39.

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James Martin, and Daniel Jurafsky. 2005b. Semantic role labeling using different syntactic views. In *Proceedings of ACL-2005*.

David Yarowsky, Grace Ngai, and Richard Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of HLT 2001*.

Obfuscating Document Stylometry to Preserve Author Anonymity

Gary Kacmarcik **Michael Gamon**
Natural Language Processing Group
Microsoft Research
Redmond, WA USA
{garykac,mgamon}@microsoft.com

Abstract

This paper explores techniques for reducing the effectiveness of standard authorship attribution techniques so that an author *A* can preserve anonymity for a particular document *D*. We discuss feature selection and adjustment and show how this information can be fed back to the author to create a new document *D'* for which the calculated attribution moves away from *A*. Since it can be labor intensive to adjust the document in this fashion, we attempt to quantify the amount of effort required to produce the anonymized document and introduce two levels of anonymization: shallow and deep. In our test set, we show that shallow anonymization can be achieved by making 14 changes per 1000 words to reduce the likelihood of identifying *A* as the author by an average of more than 83%. For deep anonymization, we adapt the unmasking work of Koppel and Schler to provide feedback that allows the author to choose the level of anonymization.

1 Introduction

Authorship identification has been a long standing topic in the field of *stylometry*, the analysis of literary style (Holmes 1998). Issues of style, genre, and authorship are an interesting sub-area of text categorization. In authorship detection it is not the topic of a text but rather the stylistic properties that are of interest. The writing style of a particular author can be identified by analyzing the form of the writing, rather than the content. The analysis of style therefore needs to abstract away from the content and focus on the content-independent form of the linguistic expressions in a text.

Advances in authorship attribution have raised concerns about whether or not authors can truly maintain their anonymity (Rao and Rohatgi 2000). While there are clearly many reasons for wanting to unmask an anonymous author, notably law enforcement and historical scholarship, there are also many legitimate reasons for an author to wish to remain anonymous, chief among them the desire to avoid retribution from an employer or government agency. Beyond the issue of personal privacy, the public good is often served by whistle-blowers who expose wrongdoing in corporations and governments. The loss of an expectation of privacy can result in a chilling effect where individuals are too afraid to draw attention to a problem, because they fear being discovered and punished for their actions.

It is for this reason that we set out to investigate the feasibility of creating a tool to support anonymizing a particular document, given the assumption that the author is willing to expend a reasonable amount of effort in the process. More generally, we sought to investigate the sensitivity of current attribution techniques to manipulation.

For our experiments, we chose a standard data set, the Federalist Papers, since the variety of published results allows us to simulate authorship attribution "attacks" on the obfuscated document. This is important since there is no clear consensus as to which features should be used for authorship attribution.

2 Document Obfuscation

Our approach to document obfuscation is to identify the features that a typical authorship attribution technique will use as markers and then adjust the frequencies of these terms to render them less effective on the target document.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 444–451,
Sydney, July 2006.

While it is obvious that one can affect the attribution result by adjusting feature values, we were concerned with:

- How easy is it to identify and present the required changes to the author?
- How resilient are the current authorship detection techniques to obfuscation?
- How much work is involved for the author in the obfuscation process?

The only related work that we are aware of is (Rao and Rohatgi 2000) who identify the problem and suggest (somewhat facetiously, they admit) using a round-trip machine translation (MT) process (e.g., English → French → English) to obscure any traces of the original author's style. They note that the current quality of MT would be problematic, but this approach might serve as a useful starting point for someone who wants to scramble the words a bit before hand-correcting egregious errors (taking care not to re-introduce their style).

2.1 The Federalist Papers

One of the standard document sets used in authorship attribution is the Federalist Papers, a collection of 85 documents initially published anonymously, but now known to have been written by 3 authors: Alexander Hamilton, John Madison and John Jay. Due to illness, Jay only wrote 5 of the papers, and most of the remaining papers are of established authorship (Hamilton = 51; Madison = 14; and 3 of joint authorship between Hamilton and Madison). The 12 remaining papers are disputed between Hamilton and Madison. In this work we limit ourselves to the 65 known single-author papers and the 12 disputed papers.

While we refer to these 12 test documents as "disputed", it is generally agreed (since the work of Mosteller and Wallace (1964)) that all of the disputed papers were authored by Madison. In our model, we accept that Madison is the author of these papers and adopt the fiction that he is interested in obscuring his role in their creation.

2.2 Problem Statement

A more formal problem statement is as follows: We assume that an author *A* (in our case, Madison) has created a document *D* that needs to be anonymized. The author self-selects a set *K* of *N* authors (where $A \in K$) that some future agent (the "attacker" following the convention used in cryptography) will attempt to select between.

The goal is to use authorship attribution techniques to create a new document *D'* based on *D* but with features that identify *A* as the author suppressed.

3 Document Preparation

Before we can begin with the process of obfuscating the author style in *D*, we need to gather a training corpus and normalize all of the documents.

3.1 Training Corpus

While the training corpus for our example is trivially obtained, authors wishing to anonymize their documents would need to gather their own corpus specific for their use.

The first step is to identify the set of authors *K* (including *A*) that could have possibly written the document. This can be a set of co-workers or a set of authors who have published on the topic. Once the authors have been selected, a suitable corpus for each author needs to be gathered. This can be emails or newsgroup postings or other documents. In our experiments, we did not include *D* in the corpus for *A*, although it does not seem unreasonable to do so.

For our example of the Federalist Papers, *K* is known to be {Hamilton, Madison} and it is already neatly divided into separate documents of comparable length.

3.2 Document Cleanup

Traditional authorship attribution techniques rely primarily on associating idiosyncratic formatting, language usage and spelling (misspellings, typos, or region-specific spelling) with each author in the study. Rao and Rohatgi (2000) and Koppel and Schler (2003) both report that these words serve as powerful discriminators for author attribution. Thus, an important part of any obfuscation effort is to identify these idiosyncratic usage patterns and normalize them in the text.

Koppel and Schler (2003) also note that many of these patterns can be identified using the basic spelling and grammar checking tools available in most word processing applications. Correcting the issues identified by these tools is an easy first step in ensuring the document conforms to conventional norms. This is especially important for work that will not be reviewed or edited since these idiosyncrasies are more likely to go unnoticed.

However, there are distinctive usage patterns that are not simple grammar or spelling errors that also need to be identified. A well-known example of this is the usage of while/whilst by the authors of the Federalist Papers.

	Hamilton	Madison	Disputed
while	36	0	0
whilst	1	12	9

Table 1 : Occurrence counts of "while" and "whilst" in the Federalist Papers (excluding documents authored by Jay and those which were jointly authored).

In the disputed papers, "whilst" occurs in 6 of the documents (9 times total) and "while" occurs in none. To properly anonymize the disputed documents, "whilst" would need to be eliminated or normalized.

This is similar to the problem with idiosyncratic spelling in that there are two ways to apply this information. The first is to simply correct the term to conform to the norms as defined by the authors in *K*. The second approach is to incorporate characteristic forms associated with a particular author. While both approaches can serve to reduce the author's stylometric fingerprint, the latter approach carries the risk of attempted style forgery and if applied indiscriminately may also provide clues that the document has been anonymized (if strong characteristics of multiple authors can be detected).

For our experiments, we opted to leave these markers in place to see how they were handled by the system. We did, however, need to normalize the paragraph formatting, remove all capitalization and convert all footnote references to use square brackets (which are otherwise unused in the corpus).

3.3 Tokenization

To tokenize the documents, we separated sequences of letters using spaces, newlines and the following punctuation marks: .,()-:;`'?![]. No stemming or morphological analysis was performed. This process resulted in 8674 unique tokens for the 65 documents in the training set.

4 Feature Selection

The process of feature selection is one of the most crucial aspects of authorship attribution. By far the most common approach is to make use of the frequencies of common function words that are content neutral, but practitioners have also made use of other features such as letter metrics (e.g., bi-grams), word and sentence length metrics, word tags and parser rewrite rules. For this work, we opted to limit our study to word frequencies since these features are generally acknowledged to be effective for authorship attribution and are transparent, which allows the author to easily incorporate the information for document modification purposes.

We wanted to avoid depending on an initial list of candidate features since there is no guarantee that the attackers will limit themselves to any of the commonly used lists. Avoiding these lists makes this work more readily useful for non-English texts (although morphology or stemming may be required).

We desire two things from our feature selection process beyond the actual features. First, we need a ranking of the features so that the author can focus efforts on the most important features. The second requirement is that we need a threshold value so that the author knows how much the feature frequency needs to be adjusted.

To rank and threshold the features, we used decision trees (DTs) and made use of the readily available WinMine toolkit (Chickering 2002). DTs produced by WinMine for continuously valued features such as frequencies are useful since each node in the tree provides the required threshold value. For term-ranking, we created a Decision Tree Root (DTR) ranking metric to order the terms based on how discriminating they are. DTR Rank is computed by creating a series of DTs where we remove the root feature, i.e. the most discriminating feature, before creating the next DT. In this fashion we create a ranking based on the order in which the DT algorithm determined that the term was most discriminatory. The DTR ranking algorithm is as follows:

1) Start with a set of features
2) Build DT and record root feature
3) Remove root feature from list of features
4) Repeat from step 2

It is worth noting that the entire DT need not be calculated since only the root is of interest. The off-the-shelf DT toolkit could be replaced with a custom implementation[1] that returned only the root (also known as a *decision stump*). Since

[1] Many DT learners are information-gain based, but the WinMine toolkit uses a Bayesian scoring criterion described in Chickering et al. (1997) with normal-Wishart parameter priors used for continuously valued features.

Token	*DTR*	*Threshold*	*Occurrence #49*
upon	1	> 0.003111	0 → 6
whilst	2	< 0.000516	1 → 0
on	3	< 0.004312	16 → 7
powers	4	< 0.002012	2 → 2
there	5	> 0.002911	2 → 5
few	6	< 0.000699	1 → 2
kind	7	> 0.001001	0 → 2
consequently	8	< 0.000513	1 → 0
wished	9	> 0.000434	1 → 0
although	10	< 0.000470	0 → 0

Table 2 : Top 10 DTR Rank ordered terms with threshold and corresponding occurrence count (original document → obfuscated version) for one of the disputed documents (#49).

our work is exploratory, we did not pursue optimizations along these lines.

For our first set of experiments, we applied DTR ranking starting with all of the features (8674 tokens from the training set) and repeated until the DT was unable to create a tree that performed better than the baseline of p(Hamilton) = 78.46%. In this fashion, we obtained an ordered list of 2477 terms, the top 10 of which are shown in Table 2, along with the threshold and bias. The threshold value is read directly from the DT root node and the bias (which indicates whether we desire the feature value to be above or below the threshold) is determined by selecting the branch of the DT which has the highest ratio of non-*A* to *A* documents.

Initially, this list looks promising, especially since known discriminating words like “upon” and “whilst” are the top two ranked terms. However, when we applied the changes to our baseline attribution model (described in detail in the Evaluation section), we discovered that while it performed well on some test documents, others were left relatively unscathed. This is shown in Figure 1 which graphs the confidence in assigning the authorship to Madison for each disputed document as each feature is adjusted. We expect the confidence to start high on the left side and move downward as more features are adjusted. After adjusting all of the identified features, half of the documents were still assigned to Madison (i.e., confidence > 0.50).

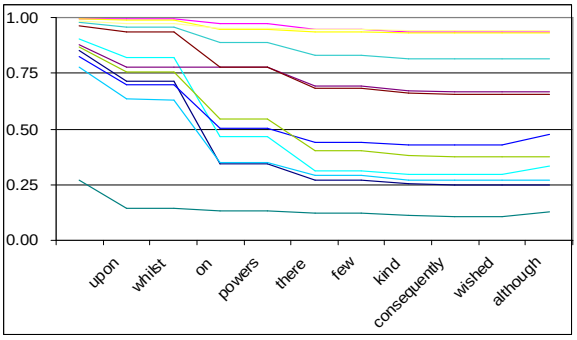


Figure 1 : Confidence in assigning disputed papers to Madison graphed as each feature is adjusted. Each line corresponds to one of the 12 disputed documents. Features are ordered by DTR Rank and the attribution model is SVM30. Values above 0.5 are assigned to Madison and those below 0.5 are assigned to Hamilton.

Choosing just the high-frequency terms was also problematic since most of them were not considered to be discriminating by DTR ranking (see Table 3). The lack of DTR rank not only means that these are poor discriminators, but it also means that we do not have a threshold value to drive the feature adjustment process.

Token	*DTR*	*Frequency*	*Token*	*DTR*	*Frequency*
the	-	0.094227	it	-	0.013404
,	595	0.068937	is	-	0.011873
of	-	0.063379	which	-	0.010933
to	39	0.038404	as	-	0.008811
.	-	0.027977	by	58	0.008614
and	185	0.025408	;	57	0.007773
in	119	0.023838	this	575	0.007701
a	515	0.021446	would	477	0.007149
be	-	0.020139	have	-	0.006873
that	-	0.014823	or	-	0.006459

Table 3 : Top 20 terms sorted by frequency.

We next combined the DTR and the term frequency approaches by computing DTR one the set of features whose frequency exceeds a specified threshold for any one of the authors. Selecting a frequency of 0.001 produces a list of 35 terms, the first 14 of which are shown in Table 4.

Token	*Frequency*	*Threshold*	*Δ 49*
upon	0.002503	> 0.003111	+6
on	0.004429	< 0.004312	-9
powers	0.001485	< 0.002012	0
there	0.002707	< 0.002911	+3
to	0.038404	> 0.039071	+7
men	0.001176	> 0.001531	+1
;	0.007773	< 0.007644	0
by	0.008614	< 0.008110	-2
less	0.001176	< 0.001384	-1
in	0.023838	> 0.023574	+6
at	0.002990	> 0.003083	0
those	0.002615	> 0.002742	+4
and	0.025408	< 0.025207	-1
any	0.002930	> 0.003005	+2

Table 4 : Top 14 DTR(0.001) ranked items. The last column is the number of changes required to achieve the threshold frequency for document #49.

Results for this list were much more promising and are shown in Figure 2. The confidence of attributing authorship to Madison is reduced by an average of 84.42% (σ = 12.51%) and all of the documents are now correctly misclassified as being written by Hamilton.

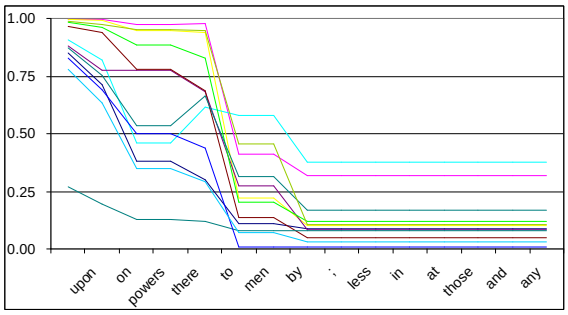


Figure 2 : Confidence in assigning disputed papers to Madison graphed as each feature is adjusted. Feature order is DTR(0.001) and the attribution model is SVM30.

5 Evaluation

Evaluating the effectiveness of any authorship obfuscation approach is made difficult by the fact that it is crucially dependent on the authorship detection method that is being utilized. An advantage of using the Federalist Papers as the test data set is that there are numerous papers documenting various methods that researchers have used to identify the authors of the disputed papers.

However, because of differences in the exact data set[2] and machine learning algorithm used, it is not reasonable to create an exact and complete implementation of each system. For our experiments, we used only the standard Federalist Papers documents and tested each feature set using linear-kernel SVMs, which have been shown to be effective in text categorization (Joachims 1998). To train our SVMs we used a sequential minimal optimization (SMO) implementation described in (Platt 1999).

The SVM feature sets that we used for the evaluation are summarized in Table 5.

For the early experiments described in the previous section we used SVM30, which incorporates the final set of 30 terms that Mosteller & Wallace used for their study. As noted earlier, they made use of a different data set than we did, so we did expect to see some differences in the results. The baseline model (plotted as the left-most column of points in Figure 1 and Figure 2) assigned all of the disputed papers to Madison except one[3].

SVM70	(Mosteller & Wallace 1964)	70 common function words.[4]
SVM30	(Mosteller & Wallace 1964)	Final 30 terms.[5]
SVM11	(Tweedie, Singh & Holmes 1996)	on, upon, there, any, an, every, his, from, may, can, do
SVM08	(Holmes & Forsyth 1995)	upon, both, on, there, whilst, kind, by, consequently
SVM03	(Bosch & Smith 1998)	upon, our, are

Table 5 : Summary of feature words used in other Federalist Papers studies.

5.1 Feature Modification

Rather than applying the suggested modifications to the original documents and regenerating the document feature vectors from scratch each time, we simplified the evaluation process by adjusting the feature vector directly and ignoring the impact of the edits on the overall document probabilities. The combination of insertions and deletions results in the total number of words in the document being increased by an average of 19.58 words (σ = 7.79), which is less than 0.5% of the document size. We considered this value to be small enough that we could safely ignore its impact.

Modifying the feature vector directly also allows us to consider each feature in isolation, without concern for how they might interact with each other (e.g. converting whilst→while or rewriting an entire sentence). It also allows us to avoid the problem of introducing rewrites into the document with our distinctive stylometric signature instead of a hypothetical Madison rewrite.

5.2 Experiments

We built SVMs for each feature set listed in Table 5 and applied the obfuscation technique described above by adjusting the values in the feature vector by increments of the single-word probability for each document. The results that we obtained were the same as observed with our test model – all of the models were coerced to prefer Hamilton for each of the disputed documents.

[2] Mosteller & Wallace and some others augmented the Federalist Papers with additional document samples (5 Hamilton and 36 Madison), but this has not been done universally by all researchers.

[3] Document #55. However, this is not inconsistent with Mosteller &Wallace's results: "Madison is extremely likely […] to have written all the disputed Federalists […] with the possible exception of No. 55. For No. 55 our evidence is relatively weak […]." (Mosteller & Wallace 1964) p.263.

[4] *ibid* p.38.

[5] *ibid* p.66.

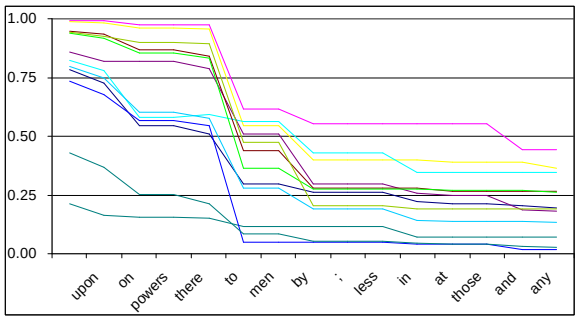


Figure 3 : Confidence in assigning disputed papers to Madison graphed as each feature is adjusted. Feature order is DTR(0.001) and the attribution model is SVM70.

Figure 3 shows the graph for SVM70, the model that was most resilient to our obfuscation techniques. The results for all models are summarized in Table 6. The overall reduction achieved across all models is 86.86%.

	% Reduction	σ
SVM70	74.66%	12.97%
SVM30	84.42%	12.51%
SVM11	82.65%	10.99%
SVM08	93.54%	4.44%
SVM03	99.01%	0.74%

Table 6 : Percent reduction in the confidence of assigning the disputed papers to Madison for each of the tested feature sets.

Of particular note in the results are those for SVM03, which proved to be the most fragile model because of its low dimension. If we consider this case an outlier and remove it from study, our overall reduction becomes 83.82%.

5.3 Feature Changes

As stated earlier, an important aspect of any obfuscation approach is the number of changes required to effect the mis-attribution. Table 7 summarizes the absolute number of changes (both insertions and deletions) and also expresses this value related to the original document size. The average number of changes required per 1000 words in the document is 14.2. While it is difficult to evaluate how much effort would be required to make each of these individual changes, this value seems to be within the range that a motivated person could reasonably undertake.

More detailed summaries of the number of feature changes required for single document (#49) are given in Table 2 and Table 4.

By calculating the overall number of changes required, we implicitly consider insertions and deletions to be equally weighted. However, while deletion sites in the document are easy to identify, proposing insertion sites can be more problematic. We do not address this difference in this paper, although it is clear that more investigation is required in this area.

Document	Changes	Doc Size	Changes/1000
49	42	3849	10.9
50	46	2364	19.5
51	67	4039	16.6
52	52	3913	13.3
53	62	4592	13.5
54	53	4246	12.5
55	52	4310	12.1
56	59	3316	17.8
57	60	4610	13.0
58	54	4398	12.3
62	78	5048	15.5
63	91	6429	14.2

Table 7 : Changes required per document

6 Deep Obfuscation

The techniques described above result in what we term *shallow* obfuscation since they focus on a small number of features and are only useful as a defense against standard attribution attacks. More advanced attribution techniques, such as that described in (Koppel and Schler 2004) look deeper into the author's stylometric profile and can identify documents that have been obfuscated in this manner.

Koppel and Schler introduce an approach they term "unmasking" which involves training a series of SVM classifiers where the most strongly weighted features are removed after each iteration. Their hypothesis is that two texts from different authors will result in a steady and relatively slow decline of classification accuracy as features are being removed. In contrast, two texts from the same author will produce a relatively fast decline in accuracy. According to the authors, a slow decline indicates deep and fundamental stylistic differences in style - beyond the "obvious" differences in the usage of a few frequent words. A fast decline indicates that there is an underlying similarity once the impact of a few superficial distinguishing markers has been removed.

We repeated their experiments using 3-fold cross-validation to compare Hamilton and Madison with each other and the original (D) and obfuscated (D') documents. The small number of documents required that we train the SVM using the 50 most frequent words. Using a larger pool of feature words resulted in unstable models, especially when comparing Madison (14 documents) with D and D' (12 documents). The results of this comparison are shown in Figure 4.

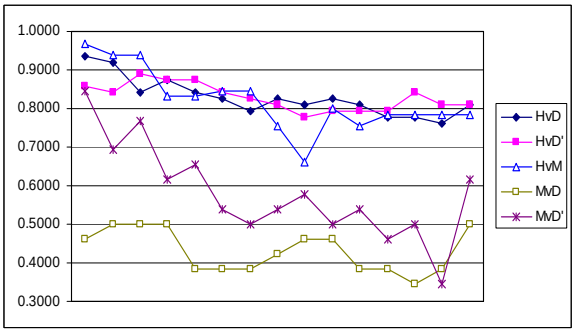


Figure 4 : Unmasking the obfuscated document. The y-axis plots the accuracy of a classifier trained to distinguish between two authors; the x-axis plots each iteration of the unmasking process. The top three lines compare Hamilton (H) versus Madison (M), the original document (D) and the obfuscated document (D'). The bottom line is M vs. D and the middle line is M vs. D'.

In this graph, the comparison of Hamilton and the modified document (MvD') exhibits the characteristic curve described by Koppel and Schler, which indicates that the original author can still be detected. However, the curve has been raised above the curve for the original document which suggests that our approach does help insulate against attacks that identify deep stylometric features.

Modifying additional features continues this trend and raises the curve further. Figure 5 summarizes this difference by plotting the difference between the accuracy of the HvD' and MvD' curves for documents at different levels of feature modification. An ideal curve in this graph would be one that hugged the x-axis since this would indicate that it was as difficult to train a classifier to distinguish between M and D' as it is to distinguish between H and D'. In this graph, the "0" curve corresponds to the original document, and the "14" curve to the modified document shown in Figure 4. The "35" curve uses all of the DTR(0.001) features.

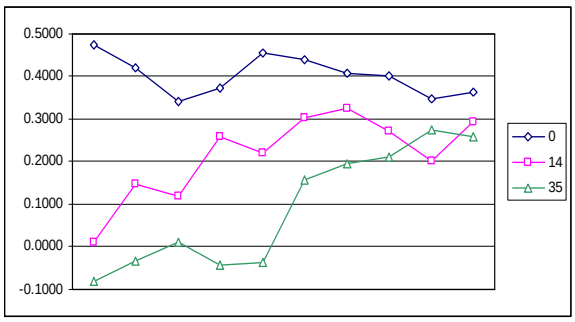


Figure 5 : Overall impact of feature modification for different levels of obfuscation. The y-axis plots the accuracy delta between the HvD' and MvD' curves; the x-axis plots each iteration of the unmasking process. The legend indicates the number of features modified for each curve.

This graph demonstrates that using DTR ranking to drive feature adjustment can produce documents that are increasingly harder to detect as being written by the author. While it is unsurprising that a deep level of obfuscation is not achieved when only a minimal number of features are modified, this graph can be used to measure progress so that the author can determine enough features have been modified to achieve the desired level of anonymization. Equally unsurprising is that this increased anonymization comes at an additional cost, summarized in Table 8.

Num Features	Changes/1000
7	9.9
14	14.2
21	18.3
28	22.5
35	25.1

Table 8 : Relationship between number of features modified and corresponding changes required per 1000 words.

While in this work we limited ourselves to the 35 DTR(0.001) features, further document modification can be driven by lowering the DTR probability threshold to identify additional terms in an orderly fashion.

7 Conclusion

In this paper, we have shown that the standard approaches to authorship attribution can be confounded by directing the author to selectively edit the test document. We have proposed a technique to automatically identify distinctive features and their frequency thresholds. By using a list of features that are both frequent and highly ranked according to this automatic technique, the amount of effort required to achieve reasonable authorship obfuscation seems to be well within the realm of a motivated author. While we make no claim that this is an easy task, and we make the assumption that the author has undertaken basic preventative measures (like spellchecking and grammar checking), it does not seem to be an onerous task for a motivated individual.

It not surprising that we can change the outcome by adjusting the values of features used in authorship detection. Our contribution, however, is that many of the important features can be determined by simultaneously considering term-frequency and DTR rank, and that this process results in a set of features and threshold values that are transparent and easy to control.

Given this result, it is not unreasonable to expect that a tool could be created to provide feedback to an author who desires to publish a document anonymously. A sophisticated paraphrase tool could theoretically use the function word change information to suggest rewrites that worked toward the desired term frequency in the document.

For our experiments, we used a simplified model of the document rewrite process by evaluating the impact of each term modification in isolation. However, modifying the document to increase or decrease the frequency of a term will necessarily impact the frequencies of other terms and thus affect the document's stylometric signature. Further experimentation is clearly needed in this area needs to address the impact of this interdependency.

One limitation to this approach is that it applies primarily to authors that have a reasonably-sized corpus readily available (or easily created). However, for situations where a large corpus is not available, automated authorship attribution techniques are likely to be less effective (and thus obfuscation is less necessary) since the number of possible features can easily exceed the number of available documents. An interesting experiment would be to explore how this approach applies to different types of corpora like email messages.

We also recognize that these techniques could be used to attempt to imitate another author's style. We do not address this issue other than to say that our thresholding approach is intended to push feature values just barely across the threshold away from *A* rather than to mimic any one particular author.

Finally, in these results, there is a message for those involved in authorship attribution: simple SVMs and low-dimensional models (like SVM03) may appear to work well, but are far less resilient to obfuscation attempts than Koppel and Schler's unmasking approach. Creating classifiers with the minimum number of features produces a model that is brittle and more susceptible to even simplistic obfuscation attempts.

8 Acknowledgements

Thanks are in order to the reviewers of earlier drafts of this document, notably Chris Brockett and our anonymous reviewers. In addition, Max Chickering provided useful information regarding his implementation of DTs in the WinMine toolkit.

References

R. A. Bosch and J. A. Smith. 1998. Separating Hyperplanes and the Authorship of the Federalist Papers. *American Mathematical Monthly*, Vol. 105 #7 pp. 601-608.

D. M. Chickering, D. Heckerman and C. Meek. 1997. A Bayesian Approach to Learning Bayesian Networks with Local Structure. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence* (UAI97 Providence, RI), pp. 80-89.

D. M. Chickering. 2002. The WinMine Toolkit. Technical Report MSR-TR-2002-103.

D. I. Holmes and R. S. Forsyth. 1995. The Federalist Revisited: New Directions in Authorship Attribution. *Literary and Linguistic Computing* 10(2), pp.111-127.

D. I. Holmes. 1998. The Evolution of Stylometry in Humanities Scholarship. *Literary and Linguistic Computing* 13(3), pp.111-117.

T. Joachims. 1998. Text Categorization with Support Vector Machines: Learning with many Relevant Features. In *Proceedings of the 10th European Conference on Machine Learning*, pp.137-142.

M. Koppel and J. Schler. 2003. Exploiting Stylistic Idiosyncrasies for Authorship Attribution. In *Proceedings of IJCAI'03 Workshop on Computational Approaches to Style Analysis and Synthesis* (Acapulco, Mexico). pp.69-72.

M. Koppel and J. Schler, 2004. Authorship Verification as a One-Class Classification Problem. In *Proceedings of the Twenty-First International Conference on Machine Learning* (ICML 04 Banff, Alberta, Canada), pp.489-495.

F. Mosteller and D. L. Wallace. 1964. *Inference and Disputed Authorship: The Federalist*. Addison-Wesley (Reading, Massachusetts, USA).

J. Platt. 1999. Fast Training of SVMs Using Sequential Minimal Optimization. In B. Schölkopf, C. Burges and A. Smola (eds.) *Advances in Kernel Methods: Support Vector Learning*. MIT Press (Cambridge, MA, USA), pp.185-208.

J. R. Rao and P. Rohatgi. 2000. Can Pseudonymity Really Guarantee Privacy?, In *Proceedings of the 9th USENIX Security Symposium* (Denver, Colorado, USA), pp.85-96.

F. J. Tweedie, S. Singh and D. I. Holmes. 1996. Neural Network Applications in Stylometry: The Federalist Papers. In *Computers and the Humanities* 30(1), pp.1-10.

Automatic Construction of Polarity-tagged Corpus from HTML Documents

Nobuhiro Kaji and **Masaru Kitsuregawa**
Institute of Industrial Science
the University of Tokyo
4-6-1 Komaba, Meguro-ku, Tokyo 153-8505 Japan
{kaji,kitsure}@tkl.iis.u-tokyo.ac.jp

Abstract

This paper proposes a novel method of building polarity-tagged corpus from HTML documents. The characteristics of this method is that it is fully automatic and can be applied to arbitrary HTML documents. The idea behind our method is to utilize certain layout structures and linguistic pattern. By using them, we can automatically extract such sentences that express opinion. In our experiment, the method could construct a corpus consisting of 126,610 sentences.

1 Introduction

Recently, there has been an increasing interest in such applications that deal with opinions (a.k.a. sentiment, reputation etc.). For instance, Morinaga et al. developed a system that extracts and analyzes reputations on the Internet (Morinaga et al., 2002). Pang et al. proposed a method of classifying movie reviews into positive and negative ones (Pang et al., 2002).

In these applications, one of the most important issue is how to determine the *polarity* (or *semantic orientation*) of a given text. In other words, it is necessary to decide whether a given text conveys positive or negative content.

In order to solve this problem, we intend to take statistical approach. More specifically, we plan to learn the polarity of texts from a corpus in which phrases, sentences or documents are tagged with labels expressing the polarity (*polarity-tagged corpus*).

So far, this approach has been taken by a lot of researchers (Pang et al., 2002; Dave et al., 2003; Wilson et al., 2005). In these previous works, polarity-tagged corpus was built in either of the following two ways. It is built manually, or created from review sites such as AMAZON.COM. In some review sites, the review is associated with metadata indicating its polarity. Those reviews can be used as polarity-tagged corpus. In case of AMAZON.COM, the review's polarity is represented by using 5-star scale.

However, both of the two approaches are not appropriate for building large polarity-tagged corpus. Since manual construction of tagged corpus is time-consuming and expensive, it is difficult to build large polarity-tagged corpus. The method that relies on review sites can not be applied to domains in which large amount of reviews are not available. In addition, the corpus created from reviews is often noisy as we discuss in Section 2.

This paper proposes a novel method of building polarity-tagged corpus from HTML documents. The idea behind our method is to utilize certain layout structures and linguistic pattern. By using them, we can automatically extract sentences that express opinion (opinion sentences) from HTML documents. Because this method is fully automatic and can be applied to arbitrary HTML documents, it does not suffer from the same problems as the previous methods.

In the experiment, we could construct a corpus consisting of 126,610 sentences. To validate the quality of the corpus, two human judges assessed a part of the corpus and found that 92% opinion sentences are appropriate ones. Furthermore, we applied our corpus to opinion sentence classification task. Naive Bayes classifier was trained on our corpus and tested on three data sets. The result demonstrated that the classifier achieved more than 80% accuracy in each data set.

The following of this paper is organized as fol-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 452–459,
Sydney, July 2006.

lows. Section 2 shows the design of the corpus constructed by our method. Section 3 gives an overview of our method, and the detail follows in Section 4. In Section 5, we discuss experimental results, and in Section 6 we examine related works. Finally we conclude in Section 7.

2 Corpus Design

This Section explains the design of our corpus that is built automatically. Table 1 represents a part of our corpus that was actually constructed in the experiment. Note that this paper treats Japanese. The sentences in the Table are translations, and the original sentences are in Japanese.

The followings are characteristics of our corpus:

- Our corpus uses two labels, $+$ and $-$. They denote positive and negative sentences respectively. Other labels such as 'neutral' are not used.
- Since we do not use 'neutral' label, such sentence that does not convey opinion is not stored in our corpus.
- The label is assigned to not multiple sentences (or document) but single sentence. Namely, our corpus is tagged at sentence level rather than document level.

It is important to discuss the reason that we intend to build a corpus tagged at sentence level rather than document level. The reason is that one document often includes both positive and negative sentences, and hence it is difficult to learn the polarity from the corpus tagged at document level. Consider the following example (Pang et al., 2002):

> This film should be brilliant. It sounds like a great plot, the actors are first grade, and the supporting cast is good as well, and Stallone is attempting to deliver a good performance. However, it can't hold up.

This document as a whole expresses negative opinion, and should be labeled 'negative' if it is tagged at document level. However, it includes several sentences that represent positive attitude.

We would like to point out that polarity-tagged corpus created from reviews prone to be tagged at document-level. This is because meta-data (e.g. stars in AMAZON.COM) is usually associated with one review rather than individual sentences in a review. This is one serious problem in previous works.

Table 1: A part of automatically constructed polarity-tagged corpus.

label	opinion sentence
$+$	It has high adaptability.
$-$	The cost is expensive.
$-$	The engine is powerless and noisy.
$+$	The usage is easy to understand.
$+$	Above all, the price is reasonable.

3 The Idea

This Section briefly explains our basic idea, and the detail of our corpus construction method is represented in the next Section.

Our idea is to use certain layout structures and linguistic pattern in order to extract opinion sentences from HTML documents. More specifically, we used two kinds of layout structures: the itemization and the table. In what follows, we explain examples where opinion sentences can be extracted by using the itemization, table and linguistic pattern.

3.1 Itemization

The first idea is to extract opinion sentences from the itemization (Figure 1). In this Figure, opinions about a music player are itemized and these itemizations have headers such as 'pros' and 'cons'. By using the headers, we can recognize that opinion sentences are described in these itemizations.

Pros:

- The sound is natural.
- Music is easy to find.
- Can enjoy creating my favorite play-lists.

Cons:

- The remote controller does not have an LCD display.
- The body gets scratched and fingerprinted easily.
- The battery drains quickly when using the backlight.

Figure 1: Opinion sentences in itemization.

Hereafter, such phrases that indicate the pres-

ence of opinion sentences are called *indicators*. Indicators for positive sentences are called *positive indicators*. 'Pros' is an example of positive indicator. Similarly, indicators for negative sentences are called *negative indicators*.

3.2 Table

The second idea is to use the table structure (Figure 2). In this Figure, a car review is summarized in the table.

Mileage(urban)	7.0km/litter
Mileage(highway)	9.0km/litter
Plus	This is a four door car, but it's so cool.
Minus	The seat is ragged and the light is dark.

Figure 2: Opinion sentences in table.

We can predict that there are opinion sentences in this table, because the left column acts as a header and there are indicators (plus and minus) in that column.

3.3 Linguistic pattern

The third idea is based on linguistic pattern. Because we treat Japanese, the pattern that is discussed in this paper depends on Japanese grammar although we think there are similar patterns in other languages including English.

Consider the Japanese sentences attached with English translations (Figure 3). Japanese sentences are written in italics and '-' denotes that the word is followed by postpositional particles. For example, '*software-no*' means that '*software*' is followed by postpositional particle '*no*'. Translations of each word and the entire sentence are represented below the original Japanese sentence. '-POST' means postpositional particle.

In the examples, we focused on the singly underlined phrases. Roughly speaking, they correspond to 'the advantage/weakness is to' in English. In these phrases, indicators ('*riten* (advantage)' and '*ketten* (weakness)') are followed by postpositional particle '*-ha*', which is topic marker. And hence, we can recognize that something good (or bad) is the topic of the sentence.

Based on this observation, we crafted a linguistic pattern that can detect the singly underlined phrases. And then, we extracted doubly underlined phrases as opinions. They correspond to 'run quickly' and 'take too much time'. The detail of this process is discussed in the next Section.

4 Automatic Corpus Construction

This Section represents the detail of the corpus construction procedure.

As shown in the previous Section, our idea utilizes the indicator, and it is important to recognize indicators in HTML documents. To do this, we manually crafted lexicon, in which positive and negative indicators are listed. This lexicon consists of 303 positive and 433 negative indicators.

Using this lexicon, the polarity-tagged corpus is constructed from HTML documents. The method consists of the following three steps:

1. Preprocessing

   Before extracting opinion sentences, HTML documents are preprocessed. This process involves separating texts form HTML tags, recognizing sentence boundary, and complementing omitted HTML tags etc.

2. Opinion sentence extraction

   Opinion sentences are extracted from HTML documents by using the itemization, table and linguistic pattern.

3. Filtering

   Since HTML documents are noisy, some of the extracted opinion sentences are not appropriate. They are removed in this step.

For the preprocessing, we implemented simple rule-based system. We cannot explain its detail for lack of space. In the remainder of this Section, we describe three extraction methods respectively, and then examine filtering technique.

4.1 Extraction based on itemization

The first method utilizes the itemization. In order to extract opinion sentences, first of all, we have to find such itemization as illustrated in Figure 1. They are detected by using indicator lexicon and HTML tags such as <h1> and <ul> etc.

After finding the itemizations, the sentences in the items are extracted as opinion sentences. Their polarity labels are assigned according to whether the header is positive or negative indicator. From the itemization in Figure 1, three positive sentences and three negative ones are extracted.

The problem here is how to treat such item that has more than one sentences (Figure 4). In this itemization, there are two sentences in each of the

(1) *kono* *software-no* *riten-ha* *hayaku* *ugoku* *koto*
this software-POST advantage-POST quickly run to
The advantage of this software is to run quickly.

(2) *ketten-ha* *jikan-ga* *kakarisugiru* *koto-desu*
weakness-POST time-POST take too much to-POST
The weakness is to take too much time.

Figure 3: Instances of the linguistic pattern.

third and fourth item. It is hard to precisely predict the polarity of each sentence in such items, because such item sometimes includes both positive and negative sentences. For example, in the third item of the Figure, there are two sentences. One ('Has high pixel...') is positive and the other ('I was not satisfied...') is negative.

To get around this problem, we did not use such items. From the itemization in Figure 4, only two positive sentences are extracted ('the color is really good' and 'this camera makes me happy while taking pictures').

Pros:

- The color is really good.
- This camera makes me happy while taking pictures.
- Has high pixel resolution with 4 million pixels. I was not satisfied with 2 million.
- EVF is easy to see. But, compared with SLR, it's hard to see.

Figure 4: Itemization where more than one sentences are written in one item.

4.2 Extraction based on table

The second method extracts opinion sentences from the table. Since the combination of <table> and other tags can represent various kinds of tables, it is difficult to craft precise rules that can deal with any table.

Therefore, we consider only two types of tables in which opinion sentences are described (Figure 5). Type A is a table in which the leftmost column acts as a header, and there are indicators in that column. Similarly, type B is a table in which the first row acts as a header. The table illustrated in Figure 2 is categorized into type A.

The type of the table is decided as follows. The table is categorized into type A if there are both positive and negative indicators in the leftmost column. The table is categorized into type B if it is not type A and there are both positive and negative indicators in the first row.

type A

I_+	+	+	+
I_-	−	−	−

type B

	I_+	I_-	
	+	−	
	+	−	
	+	−	

I_+:positive indicator +:positive sentence
I_-:negative indicator −:negative sentence

Figure 5: Two types of tables.

After the type of the table is decided, we can extract opinion sentences from the cells that correspond to + and − in the Figure 5. It is obvious which label (positive or negative) should be assigned to the extracted sentence.

We did not use such cell that contains more than one sentences, because it is difficult to reliably predict the polarity of each sentence. This is similar to the extraction from the itemization.

4.3 Extraction based on linguistic pattern

The third method uses linguistic pattern. The characteristic of this pattern is that it takes dependency structure into consideration.

First of all, we explain Japanese dependency structure. Figure 6 depicts the dependency representations of the sentences in the Figure 3. Japanese sentence is represented by a set of dependencies between phrasal units called *bunsetsu*-phrases. Broadly speaking, *bunsetsu*-phrase is an unit similar to baseNP in English. In the Figure, square brackets enclose *bunsetsu*-phrase and arrows show modifier → head dependencies between *bunsetsu*-phrases.

In order to extract opinion sentences from these dependency representations, we crafted the following dependency pattern.

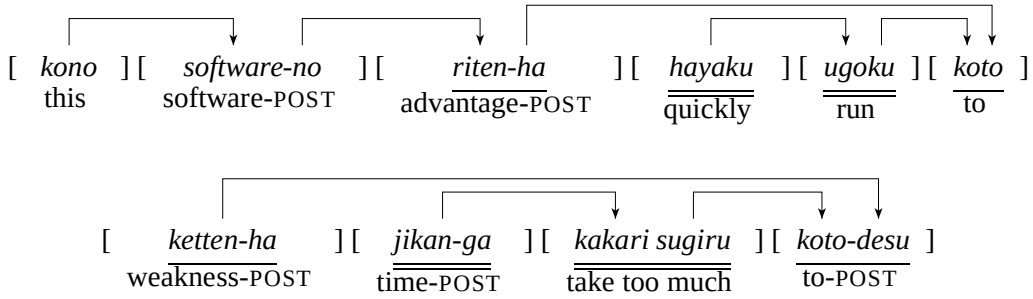


Figure 6: Dependency representations.

This pattern matches the singly underlined *bunsetsu*-phrases in the Figure 6. In the modifier part of this pattern, the indicator is followed by postpositional particle '*ha*', which is topic marker[1]. In the head part, '*koto* (to)' is followed by arbitrary numbers of postpositional particles.

If we find the dependency that matches this pattern, a phrase between the two *bunsetsu*-phrases is extracted as opinion sentence. In the Figure 6, the doubly underlined phrases are extracted. This heuristics is based on Japanese word order constraint.

4.4 Filtering

Sentences extracted by the above methods sometimes include noise text. Such texts have to be filtered out. There are two cases that need filtering process.

First, some of the extracted sentences do not express opinions. Instead, they represent objects to which the writer's opinion is directed (Table 7). From this table, 'the overall shape' and 'the shape of the taillight' are wrongly extracted as opinion sentences. Since most of the objects are noun phrases, we removed such sentences that have the noun as the head.

Mileage(urban)	10.0km/litter
Mileage(highway)	12.0km/litter
Plus	The overall shape.
Minus	The shape of the taillight.

Figure 7: A table describing only objects to which the opinion is directed.

Secondly, we have to treat duplicate opinion sentences because there are mirror sites in the HTML documents. When there are more than one sentences that are exactly the same, one of them is held and the others are removed.

[1]To be exact, some of the indicators such as 'strong point' consists of more than one *bunsetsu*-phrase, and the modifier part sometimes consists of more than one *bunsetsu*-phrase.

5 Experimental Results and Discussion

This Section examines the results of corpus construction experiment. To analyze Japanese sentence we used Juman and KNP[2].

5.1 Corpus Construction

About 120 millions HTML documents were processed, and 126,610 opinion sentences were extracted. Before the filtering, there were 224,002 sentences in our corpus. Table2 shows the statistics of our corpus. The first column represents the three extraction methods. The second and third column shows the number of positive and negative sentences by extracted each method. Some examples are illustrated in Table 3.

Table 2: # of sentences in the corpus.

	Positive	Negative	Total
Itemization	18,575	15,327	33,902
Table	12,103	11,016	23,119
Linguistic Pattern	34,282	35,307	69,589
Total	64,960	61,650	126,610

The result revealed that more than half of the sentences are extracted by linguistic pattern (see the fourth row). Our method turned out to be effective even in the case where only plain texts are available.

5.2 Quality assessment

In order to check the quality of our corpus, 500 sentences were randomly picked up and two judges manually assessed whether appropriate labels are assigned to the sentences.

The evaluation procedure is the followings.

[2]http://www.kc.t.u-tokyo.ac.jp/nl-resource/top.html

Table 3: Examples of opinion sentences.

label	opinion sentence			
	cost	*keisan-ga*	*yoininaru*	
+	cost	computation-POST	become easy	
	It becomes easy to compute cost.			
	kantan-de	*jikan-ga*	*setsuyakudekiru*	
+	easy-POST	time-POST	can save	
	It's easy and can save time.			
	soup-ha	*koku-ga ari*	*oishii*	
+	soup-POST	rich	flavorful	
	The soup is rich and flavorful.			
	HTML	*keishiki-no*	*mail-ni*	*taioshitenai*
−	HTML	format-POST	mail-POST	cannot use
	Cannot use mails in HTML format.			
	jugyo-ga	*hijoni*	*tsumaranai*	
−	lecture-POST	really	boring	
	The lecture is really boring.			
	kokoro-ni nokoru	*ongaku-ga*	*nai*	
−	impressive	music-POST	there is no	
	There is no impressive music.			

- Each of the 500 sentences are shown to the two judges. Throughout this evaluation, We did not present the label automatically tagged by our method. Similarly, we did not show HTML documents from which the opinion sentences are extracted.

- The two judges individually categorized each sentence into three groups: positive, negative and neutral/ambiguous. The sentence is classified into the third group, if it does not express opinion (neutral) or if its polarity depends on the context (ambiguous). Thus, two goldstandard sets were created.

- The precision is estimated using the goldstandard. In this evaluation, the precision refers to the ratio of sentences where correct labels are assigned by our method. Since we have two goldstandard sets, we can report two different precision values. A sentence that is categorized into neutral/ambiguous by the judge is interpreted as being assigned incorrect label by our method, since our corpus does not have a label that corresponds to neutral/ambiguous.

We investigated the two goldstandard sets, and found that the judges agree with each other in 467 out of 500 sentences (93.4%). The Kappa value was 0.901. From this result, we can say that the goldstandard was reliably created by the judges.

Then, we estimated the precision. The precision was 459/500 (91.5%) when one goldstandard was used, and 460/500 (92%) when the other was used. Since these values are nearly equal to the agreement between humans (467/500), we can conclude that our method successfully constructed polarity-tagged corpus.

After the evaluation, we analyzed errors and found that most of them were caused by the lack of context. The following is a typical example.

> You see, there is much information.

In our corpus this sentence is categorized into positive one. The below is a part of the original document from which this sentence was extracted.

> I recommend this guide book. The Pros. of this book is that, *you see, there is much information.*

On the other hand, both of the two judges categorized the above sentence into neutral/ambiguous, probably because they can easily assume context where much information is not desirable.

> *You see, there is much information.* But, it is not at all arranged, and makes me confused.

In order to precisely treat this kind of sentences, we think discourse analysis is inevitable.

5.3 Application to opinion classification

Next, we applied our corpus to opinion sentence classification. This is a task of classifying sentences into positive and negative. We trained a classifier on our corpus and investigated the result.

Classifier and data sets As a classifier, we chose Naive Bayes with bag-of-words features, because it is one of the most popular one in this task. Negation was processed in a similar way as previous works (Pang et al., 2002).

To validate the accuracy of the classifier, three data sets were created from review pages in which the review is associated with meta-data. To build data sets tagged at sentence level, we used such reviews that contain only one sentence. Table 4 represents the domains and the number of sentences in each data set. Note that we confirmed there is no duplicate between our corpus and the these data sets.

The result and discussion Naive Bayes classifier was trained on our corpus and tested on the three data sets (Table 5). In the Table, the second column represents the accuracy of the classification in each data set. The third and fourth

Table 5: Classification result.

	Accuracy	Positive		Negative	
		Precision	Recall	Precision	Recall
Computer	0.831	0.856	0.804	0.804	0.859
Restaurant	0.849	0.905	0.859	0.759	0.832
Car	0.833	0.860	0.844	0.799	0.819

Table 4: The data sets.

Domain	# of sentences	
	Positive	Negative
Computer	933	910
Restaurant	753	409
Car	1,056	800

columns represent precision and recall of positive sentences. The remaining two columns show those of negative sentences. Naive Bayes achieved over 80% accuracy in all the three domains.

In order to compare our corpus with a small domain specific corpus, we estimated accuracy in each data set using 10 fold crossvalidation (Table 6). In two domains, the result of our corpus outperformed that of the crossvalidation. In the other domain, our corpus is slightly better than the crossvalidation.

Table 6: Accuracy comparison.

	Our corpus	Crossvalidation
Computer	0.831	0.821
Restaurant	0.849	0.848
Car	0.833	0.808

One finding is that our corpus achieved good accuracy, although it includes various domains and is not accustomed to the target domain. Turney also reported good result without domain customization (Turney, 2002). We think these results can be further improved by domain adaptation technique, and it is one future work.

Furthermore, we examined the variance of the accuracy between different domains. We trained Naive Bayes on each data set and investigate the accuracy in the other data sets (Table 7). For example, when the classifier is trained on Computer and tested on Restaurant, the accuracy was 0.757. This result revealed that the accuracy is quite poor when the training and test sets are in different domains. On the other hand, when Naive Bayes is trained on our corpus, there are little variance in different domains (Table 5). This experiment indicates that our corpus is relatively robust against the change of the domain compared with small domain specific corpus. We think this is because our corpus is large and balanced. Since we cannot always get domain specific corpus in real application, this is the strength of our corpus.

Table 7: Cross domain evaluation.

		Training		
		Computer	Restaurant	Car
Test	Computer	—	0.701	0.773
	Restaurant	0.757	—	0.755
	Car	0.751	0.711	—

6 Related Works

6.1 Learning the polarity of words

There are some works that discuss learning the polarity of words instead of sentences.

Hatzivassiloglou and McKeown proposed a method of learning the polarity of adjectives from corpus (Hatzivassiloglou and McKeown, 1997). They hypothesized that if two adjectives are connected with conjunctions such as 'and/but', they have the same/opposite polarity. Based on this hypothesis, their method predicts the polarity of adjectives by using a small set of adjectives labeled with the polarity.

Other works rely on linguistic resources such as WordNet (Kamps et al., 2004; Hu and Liu, 2004; Esuli and Sebastiani, 2005; Takamura et al., 2005). For example, Kamps et al. used a graph where nodes correspond to words in the WordNet, and edges connect synonymous words in the WordNet. The polarity of an adjective is defined by its shortest paths from the node corresponding to 'good' and 'bad'.

Although those researches are closely related to our work, there is a striking difference. In those researches, the target is limited to the polarity of words and none of them discussed sentences. In addition, most of the works rely on external resources such as the WordNet, and cannot treat words that are not in the resources.

6.2 Learning subjective phrases

Some researchers examined the acquisition of subjective phrases. The subjective phrase is more general concept than opinion and includes both positive and negative expressions.

Wiebe learned subjective adjectives from a set of seed adjectives. The idea is to automatically identify the synonyms of the seed and to add them to the seed adjectives (Wiebe, 2000). Riloff et al. proposed a bootstrapping approach for learning subjective nouns (Riloff et al., 2003). Their method learns subjective nouns and extraction patterns in turn. First, given seed subjective nouns, the method learns patterns that can extract subjective nouns from corpus. And then, the patterns extract new subjective nouns from corpus, and they are added to the seed nouns. Although this work aims at learning only nouns, in the subsequent work, they also proposed a bootstrapping method that can deal with phrases (Riloff and Wiebe, 2003). Similarly, Wiebe also proposes a bootstrapping approach to create subjective and objective classifier (Wiebe and Riloff, 2005).

These works are different from ours in a sense that they did not discuss how to determine the polarity of subjective words or phrases.

6.3 Unsupervised sentiment classification

Turney proposed the unsupervised method for sentiment classification (Turney, 2002), and similar method is utilized by many other researchers (Yu and Hatzivassiloglou, 2003). The concept behind Turney's model is that positive/negative phrases co-occur with words like 'excellent/poor'. The co-occurrence statistic is measured by the result of search engine. Since his method relies on search engine, it is difficult to use rich linguistic information such as dependencies.

7 Conclusion

This paper proposed a fully automatic method of building polarity-tagged corpus from HTML documents. In the experiment, we could build a corpus consisting of 126,610 sentences.

As a future work, we intend to extract more opinion sentences by applying this method to larger HTML document sets and enhancing extraction rules. Another important direction is to investigate more precise model that can classify or extract opinions, and learn its parameters from our corpus.

References

Kushal Dave, Steve Lawrence, and David M.Pennock. 2003. Mining the peanut gallery: Opinion extraction and semantic classification of product revews. In *Proceedings of the WWW*, pages 519–528.

Andrea Esuli and Fabrizio Sebastiani. 2005. Determining the semantic orientation of terms throush gloss classification. In *Proceedings of the CIKM*.

Vasileios Hatzivassiloglou and Katheleen R. McKeown. 1997. Predicting the semantic orientation of adjectives. In *Proceedings of the ACL*, pages 174–181.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the KDD*, pages 168–177.

Jaap Kamps, Maarten Marx, Robert J. Mokken, and Maarten de Rijke. 2004. Using wordnet to measure semantic orientations of adjectives. In *Proceedings of the LREC*.

Satoshi Morinaga, Kenji Yamanishi, Kenji Tateishi, and Toshikazu Fukushima. 2002. Mining product reputations on the web. In *Proceedings of the KDD*.

Bo Pang, Lillian Lee, and Shivakumar Vaihyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the EMNLP*.

Ellen Riloff and Janyce Wiebe. 2003. Learning extraction patterns for subjective expressions. In *Proceedings of the EMNLP*.

Ellen Riloff, Janyce Wiebe, and Theresa Wilson. 2003. Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the CoNLL*.

Hiroya Takamura, Takashi Inui, and Manabu Okumura. 2005. Extracting semantic orientation of words using spin model. In *Proceedings of the ACL*, pages 133–140.

Peter D. Turney. 2002. Thumbs up or thumbs down? senmantic orientation applied to unsupervised classification of reviews. In *Proceedings of the ACL*, pages 417–424.

Janyce Wiebe and Ellen Riloff. 2005. Creating subjective and objective sentence classifiers from unannotated texts. In *Proceedings of the CICLing*.

Janyce M. Wiebe. 2000. Learning subjective adjectives from corpora. In *Proceedings of the AAAI*.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the HLT/EMNLP*.

Hong Yu and Yasileios Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the EMNLP*.

Minority Vote: At-Least-N Voting Improves Recall for Extracting Relations

Nanda Kambhatla
IBM T.J. Watson Research Center
1101 Kitchawan Road Rt 134
Yorktown, NY 10598
nanda@us.ibm.com

Abstract

Several NLP tasks are characterized by asymmetric data where one class label *NONE*, signifying the absence of any structure (named entity, coreference, relation, etc.) dominates all other classes. Classifiers built on such data typically have a higher precision and a lower recall and tend to overproduce the *NONE* class. We present a novel scheme for voting among a committee of classifiers that can significantly boost the recall in such situations. We demonstrate results showing up to a 16% relative improvement in ACE value for the 2004 ACE relation extraction task for English, Arabic and Chinese.

1 Introduction

Statistical classifiers are widely used for diverse NLP applications such as part of speech tagging (Ratnaparkhi, 1999), chunking (Zhang et al., 2002), semantic parsing (Magerman, 1993), named entity extraction (Borthwick, 1999; Bikel et al., 1997; Florian et al., 2004), coreference resolution (Soon et al., 2001), relation extraction (Kambhatla, 2004), etc. A number of these applications are characterized by a dominance of a *NONE* class in the training examples. For example, for coreference resolution, classifiers might classify whether a given pair of mentions are references to the same entity or not. In this case, we typically have a lot more examples of mention pairs that are not coreferential (i.e. the *NONE* class) than otherwise. Similarly, if a classifier is predicting the presence/absence of a semantic relation between two mentions, there are typically far more examples signifying an absence of a relation.

Classifiers built with asymmetric data dominated by one class (a *NONE* class donating absence of a relation or coreference or a named entity etc.) can overgenerate the *NONE* class. This often results in a unbalanced classifier where precision is higher than recall.

In this paper, we present a novel approach for improving the recall of such classifiers by using a new voting scheme from a committee of classifiers. There are a plethora of algorithms for combining classifiers (e.g. see (Xu et al., 1992)). A widely used approach is a **majority voting** scheme, where each classifier in the committee gets a vote and the class with the largest number of votes 'wins' (i.e. the corresponding class is output as the prediction of the committee).

We are interested in improving overall recall and reduce the overproduction of the class *NONE*. Our scheme predicts the class label C obtaining the second highest number of votes when *NONE* gets the highest number of votes, provided C gets **at least** N votes. Thus, we predict a label other than *NONE* when there is some evidence of the presense of the structure we are looking for (relations, coreference, named entities, etc.) even in the absense of a clear majority.

This paper is organized as follows. In section 2, we give an overview of the various schemes for combining classifiers. In section 3, we present our vot-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 460–466,
Sydney, July 2006.

ing algorithm. In section 4, we describe the ACE relation extraction task. In section 5, we present empirical results for relation extraction and we discuss our results and conclude in section 6.

2 Combining Classifiers

Numerous methods for combining classifiers have been proposed and utlized to improve the performance of different NLP tasks such as part of speech tagging (Brill and Wu, 1998), identifying base noun phrases (Tjong Kim Sang et al., 2000), named entity extraction (Florian et al., 2003), etc. Ho *et al* (1994) investigated different approaches for reranking the outputs of a committee of classifiers and also explored union and intersection methods for reducing the set of predicted categories. Florian *et al* (2002) give a broad overview of methods for combining classifiers and present empirical results for word sense disambiguation.

Xu *et al* (1992) and Florian *et al* (2002) consider three approaches for combining classifiers. In the first approach, individual classifiers output posterior probabilities that are merged (e.g. by taking an average) to arrive at a composite posterior probability of each class. In the second scheme, each classifier outputs a ranked list of classes instead of a probability distribution and the different ranked lists are merged to arrive at a final ranking. Methods using the third approach, often called *voting methods*, treat each classifier as a black box that outputs only the top ranked class and combines these to arrive at the final decision (class). The choice of approach and the specific method of combination may be constrained by the specific classification algorithms in use.

In this paper, we focus on voting methods, since for small data sets, it is hard to reliably estimate probability distributions or even a complete ordering of classes especially when the number of classes is large.

A widely used voting method for combining classifiers is a **Majority Vote** scheme (e.g. (Brill and Wu, 1998; Tjong Kim Sang et al., 2000)). Each classifier gets to vote for its top ranked class and the class with the highest number of votes 'wins'. Henderson *et al* (1999) use a Majority Vote scheme where different parsers vote on constituents' membership in a hypothesized parse. Halteren *et al* (1998) compare a number of voting methods including a Majority Vote scheme with other combination methods for part of speech tagging.

In this paper, we induce multiple classifiers by using **bagging** (Breiman, 1996). Following Breiman's approach, we obtain multiple classifiers by first making bootstrap replicates of the training data and training different classifiers on each of the replicates. The bootstrap replicates are induced by repeatedly *sampling with replacement* training events from the original training data to arrive at replicate data sets of the same size as the training data set. Breiman (1996) uses a Majority Vote scheme for combining the output of the classifiers. In the next section, we will describe the different voting schemes we explored in our work.

3 At-Least-N Voting

We are specifically interested in NLP tasks characterized by asymmetric data where, typically, we have far more occurances of a *NONE* class that signifies the absense of structure (e.g. a named entity, or a coreference relation or a semantic relation). Classifiers trained on such data sets can overgenerate the *NONE* class, and thus have a higher precision and lower recall in discovering the underlying structure (i.e. the named entities or coreference links etc.). With such tasks, the benefits yielded by a Majority Vote is limited, since, because of the asymmetry in the data, a majority of the classifiers might predict *NONE* most of the time.

We propose alternative voting schemes, dubbed **At-Least-N Voting**, to deal with the overproduction of *NONE*. Given a committee of classifiers (obtained by bagging or some other mechanism), the classifiers first cast their vote. If the majority vote is for a class C other than *NONE*, we simply output C as the prediction. If the majority vote is for *NONE*, we output the class label obtaining the second highest number of votes, *provided* it has at least N votes. Thus, we choose to defer to the minority vote of classifiers which agree on finding some structure even when the majority of classifiers vote for *NONE*. We expect this voting scheme to increase recall at the expense of precision.

At-Least-N Voting induces a spectrum of combi-

nation methods ranging from a Majority Vote (when N is more than half of the total number of classifiers) to a scheme, where the evidence of any structure by even one classifier is believed (At-Least-1 Voting). The exact choice of N is an empirical one and depends on the amount of asymmetry in the data and the imbalance between precision and recall in the classifiers.

4 The ACE Relation Extraction Task

Automatic Content Extraction (ACE) is an annual evaluation conducted by NIST (NIST, 2004) on information extraction, focusing on extraction of entities, events, and relations. The Entity Detection and Recognition task entails detection of mentions of entities and grouping together the mentions that are references to the same entity. In ACE terminology, *mentions* are references in text (or audio, chats, ...) to real world *entities*. Similarly *relation mentions* are references in text to semantic relations between entity mentions and *relations* group together all relation mentions that identify the same semantic relation between the same entities.

In the frament of text:

> John's son, Jim went for a walk. Jim liked his father.

all the underlined words are mentions referring to two entities, John, and Jim. Morover, John and Jim have a *family* relation evidenced as two relation mentions "John's son" between the entity mentions "John" and "son" and "his father" between the entity mentions "his" and "father".

In the relation extraction task, systems must predict the presence of a predetermined set of binary relations among mentions of entities, label the relation, and identify the two arguments. In the 2004 ACE evaluation, systems were evaluated on their efficacy in correctly identifying relations among both system output entities and with 'true' entities (i.e. as annotated by human annotators as opposed to system output). In this paper, we present results for extracting relations between 'true' entities.

Table 1 shows the set of relation types, subtypes, and their frequency counts in the training data for the 2004 ACE evaluation. For training classifiers, the great paucity of positive training events (where relations exist) compared to the negative events (where relations do not exist) suggest that schemes for improving recall might benefit this task.

Type	Subtype	Count
ART	*user-or-owner*	*140*
(agent artifact)	*inventor/manufacturer*	*3*
	other	*6*
EMP-ORG	*employ-executive*	*420*
	employ-staff	*416*
	employ-undetermined	*62*
	member-of-group	*126*
	partner	*11*
	subsidiary	*213*
	other	*37*
GPE-AFF	*citizen-or-resident*	*173*
(GPE affiliation)	*based-in*	*225*
	other	*63*
DISCOURSE	*-none-*	*122*
PHYSICAL	*located*	*516*
	near	*81*
	part-whole	*333*
PER-SOC	*business*	*119*
(personal/social)	*family*	*115*
	other	*28*
OTHER-AFF	*ethnic*	*28*
(PER/ORG	*ideology*	*26*
affiliation)	*other*	*27*

Table 1: The set of types and subtypes of relations used in the 2004 ACE evaluation.

5 Experimental Results

In this section, we present results of experiments comparing three different methods of combining classifiers for ACE relation extraction:

- *At-Least-N* for different values of N,
- Majority Voting, and
- a simple algorithm, called **summing**, where we add the posterior scores for each class from all the classifiers and select the class with the maximum summed score.

Since the official ACE evaluation set is not publicly available, to facilitate comparison with our results and for internal testing of our algorithms, for each language (English, Arabic, and Chinese), we

	En	Ar	Ch
Training Set (documents)	*227*	*511*	*480*
Training Set (rel-mentions)	*3290*	*4126*	*4347*
Test Set (documents)	*114*	*178*	*166*
Test Set (rel-mentions)	*1381*	*1894*	*1774*

Table 2: The Division of LDC annotated data into training and development test sets.

divided the ACE 2004 training data provided by LDC in a roughly 75%:25% ratio into a training set and a test set. Table 2 summarizes the number of documents and the number of relation mentions in each data set. The test sets were deliberately chosen to be the most recent 25% of documents in chronological order, since entities and relations in news tend to repeat and random shuffles can greatly reduce the out-of-vocabulary problem.

5.1 Maximum Entropy Classifiers

We used bagging (Breiman, 1996) to create replicate training sets of the same size as the original training set by repeatedly sampling with replacement from the training set. We created 25 replicate training sets (bags) for each language (Arabic, Chinese, English) and trained separate maximum entropy classifiers on each bag. We then applied At-Least-N ($N = 1,2,5$), Majority Vote, and Summing algorithms with the trained classifiers and measured the resulting performance on our development set.

For each bag, we built maximum entropy models to predict the presence of relation mentions and the type and subtype of relations, when their presence is predicted. Our models operate on every pair of mentions in a document that are not references to the same entity, to extract relation mentions. Since there are 23 unique type-subtype pairs in Table 1, our classifiers have 47 classes: two classes for each pair corresponding to the two argument orderings (e.g. "John's son" vs. "his father") and a *NONE* class signifying no relation.

Similar to our earlier work (Kambhatla, 2004), we used a combination of lexical, syntactic, and semantic features including all the words in between the two mentions, the entity types and subtypes of the two mentions, the number of words in between the two mentions, features derived from the smallest parse fragment connecting the two mentions, etc. These features were held constant throughout these experiments.

5.2 Results

We report the F-measure, precision and recall for extracting relation mentions for all three languages. We also report *ACE value*[1], the official metric used by NIST that assigns 0% value to a system that produces no output and a 100% value to a system that extracts all relations without generating any false alarms. Note that the ACE value counts each relation only once even if it is expressed in text many times as different relation mentions. The reader is referred to the NIST web site (NIST, 2004) for more details on the ACE value computation.

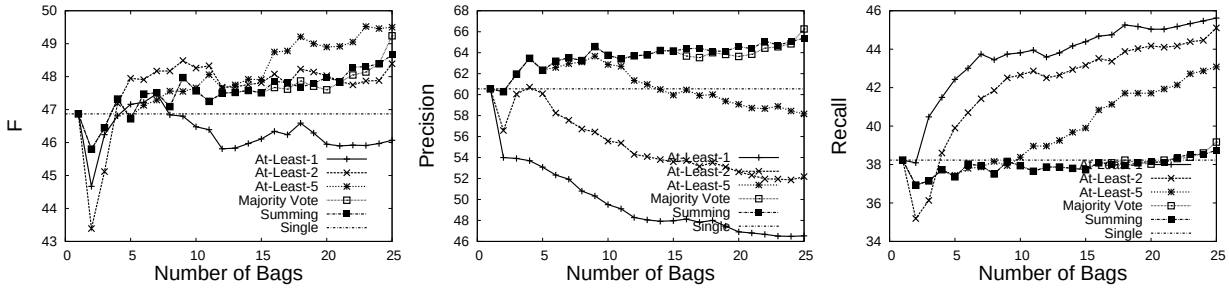
Figures 1(a), 1(b), and 1(c) show the F-measure, precision, and recall respectively for the English test set obtained by different classifier combination techniques as we vary the number of bags. Figures 2(a), 2(b), and 2(c) show similar curves for Chinese, and Figures 3(a), 3(b), and 3(c) show similar curves for Arabic. All these figures show the performance of a single classifier as a straight line.

From the plots, it is clear that our hope of increasing recall by combining classifiers is realized for all three languages. As expected, the recall rises fastest for At-Least-N when N is small, i.e when small minority opinion or even a single dissenting opinion is being trusted. Of course, the rise in recall is at the expense of a loss of precision. Overall, At-Least-N for intermediate ranges of N (N=5 for English and Chinese and N=2 for Arabic) performs best where the moderate loss in precision is more than offset by a rise in recall.

Both the Majority Vote method and the Summing method succeed in avoiding a sharp loss of precision. However, they fail to increase the recall significantly either.

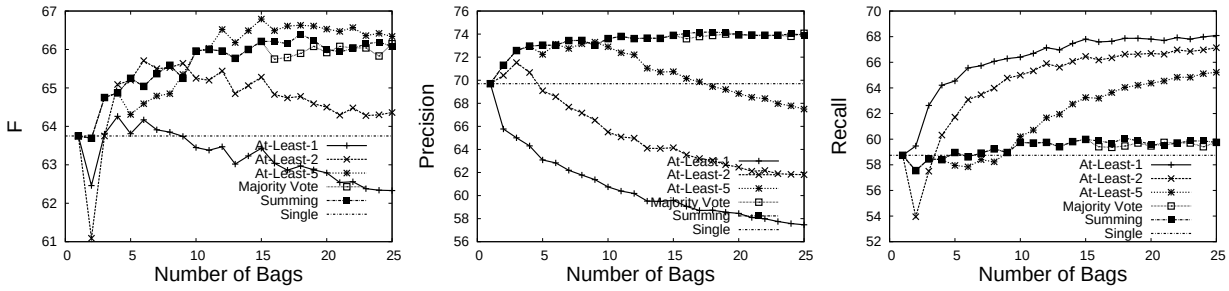
Table 3 summarizes the best results (F-measure) for each classifier combination method for all three languages compared with the result for a single classifier. At their best operating points, all three combination methods handily outperform the single classifier. At-Least-N seems to have a slight edge over the other two methods, but the difference is small.

[1]Here we use the ACE value metric used for the ACE 2004 evaluation



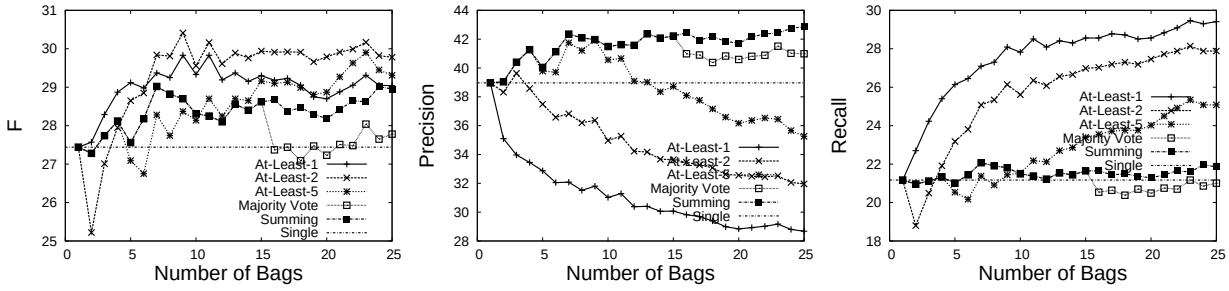
(a) F-measure (b) Precision (c) Recall

Figure 1: Comparing F-measure, precision, and recall of different voting schemes for **English** relation extraction.



(a) F-measure (b) Precision (c) Recall

Figure 2: Comparing F-measure, precision, and recall of different voting schemes for **Chinese** relation extraction.



(a) F-measure (b) Precision (c) Recall

Figure 3: Comparing F-measure, precision, and recall of different voting schemes for **Arabic** relation extraction.

	English	Arabic	Chinese
Single	*46.87*	*27.47*	*63.75*
At-Least-N	**49.52**	**30.41**	**66.79**
Majority Vote	*49.24*	*29.02*	*66.21*
Summing	*48.66*	*29.02*	*66.40*

Table 3: Comparing the best F-measure obtained by At-Least-N Voting with Majority Voting, Summing and the single best classifier.

	English	Arabic	Chinese
Single	*59.6*	*37.3*	*69.6*
At-Least-N	**63.9**	**43.5**	**71.0**

Table 4: Comparing the ACE Value obtained by At-Least-N Voting with the single best classifier for the operating points used in Table 3.

Table 4 shows the ACE value obtained by our best performing classifier combination method (At-Least-N at the operating points in Table 3) compared with a single classifier. Note that while the improvement for Chinese is slight, for Arabic performance improves by over 16% relative and for English, the improvement is over 7% relative over the single classifier[2]. Since the ACE value collapses relation mentions referring to the same relation, finding new relations (i.e. recall) is more important. This might explain the relatively larger difference in ACE value between the single classifier performance and At-Least-N.

The rules of the ACE evaluation prohibit us from presenting a detailed comparison of our relation extraction system with the other participants. However, our relation extraction system (using the At-Least-N classifier combination scheme as described here) performed very competitively in 2004 ACE evaluation both in the system output relation extraction task (RDR) and the relation extraction task where the 'true' mentions and entities are given.

Due to time limitations, we did not try At-Least-N with $N > 5$. From the plots, there is a potential for getting greater gains by experimenting with a larger number of bags and with a larger N.

[2]Note that ACE value metric used in the ACE 2004 evaluation weights entitites differently based on their type. Thus, relations with PERSON-NAME arguments end up contributing a lot more the overall score than relations with FACILITY-PRONOUN arguments.

6 Discussion

Several NLP problems exhibit a dominance of a *NONE* class that typically signifies a lack of structure like a named entity, coreference, etc. Especially when coupled with small training sets, this results in classifiers with unbalanced precision and recall. We have argued that a classifier voting scheme that is focused on improving recall can help increase overall performance in such situations.

We have presented a class of voting methods, dubbed *At-Least-N* that defer to the opinion of a minority of classifiers (consisting of N members) even when the majority predicts *NONE*. This can boost recall at the expense of precision. However, by varying N and the number of classifiers, we can pick an operating point that improves the overall F-measure.

We have presented results for ACE relation extraction for three languages comparing At-Least-N with Majority Vote and Summing methods for combining classifiers. All three classifier combination methods significantly outperform a single classifier. Also, At-Least-N consistently gave us the best performance across different languages.

We used bagging to induce multiple classifiers for our task. Because of the random bootstrap sampling, different replicate training sets might tilt towards one class or another. Thus, if we have many classifiers trained on the replicate training sets, some of them are likely to be better at predicting certain classes than others. In future, we plan to experiment with other methods for collecting a committee of classifiers.

References

D. M. Bikel, S. Miller, R. Schwartz, and R. Weischedel. 1997. Nymble: a high-performance learning name-finder. In *Proceedings of ANLP-97*, pages 194–201.

A. Borthwick. 1999. *A Maximum Entropy Approach to Named Entity Recognition*. Ph.D. thesis, New York University.

L. Breiman. 1996. Bagging predictors. In *Machine Learning*, volume 24, page 123.

E. Brill and J. Wu. 1998. Classifier combination for improved lexical disambiguation. *Proceedings of COLING-ACL'98*, pages 191–195, August.

Radu Florian and David Yarowsky. 2002. Modeling consensus: Classifier combination for word sense disambiguation. In *Proceedings of EMNLP'02*, pages 25–32.

R. Florian, A. Ittycheriah, H. Jing, and T. Zhang. 2003. Named entity recognition through classifier combination. In *Proceedings of CoNNL'03*, pages 168–171.

R. Florian, H. Hassan, A. Ittycheriah, H. Jing, N. Kambhatla, X. Luo, N Nicolov, and S Roukos. 2004. A statistical model for multilingual entity detection and tracking. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 1–8.

J. Henderson and E. Brill. 1999. Exploiting diversity in natural language processing: Combining parsers. In *Proceedings on EMNLP99*, pages 187–194.

T. K. Ho, J. J. Hull, and S. N. Srihari. 1994. Decision combination in multiple classifier systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(1):66–75, January.

Nanda Kambhatla. 2004. Combining lexical, syntactic, and semantic features with maximum entropy models for information extraction. In *The Proceedings of 42st Annual Meeting of the Association for Computational Linguistics*, pages 178–181, Barcelona, Spain, July. Association for Computational Linguistics.

D. Magerman. 1993. Parsing as statistical pattern recognition.

NIST. 2004. The ACE evaluation plan. www.nist.gov/speech/tests/ace/index.htm.

Adwait Ratnaparkhi. 1999. Learning to parse natural language with maximum entropy models. *Machine Learning*, 34:151–178.

W. M. Soon, H. T. Ng, and C. Y. Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.

E. F. Tjong Kim Sang, W. Daelemans, H. Dejean, R. Koeling, Y. Krymolowsky, V. Punyakanok, and D. Roth. 2000. Applying system combination to base noun phrase identification. In *Proceedings of COLING 2000*, pages 857–863.

H. van Halteren, J. Zavrel, and W. Daelemans. 1998. Improving data driven wordclass tagging by system combination. In *Proceedings of COLING-ACL'98*, pages 491–497.

L. Xu, A. Krzyzak, and C. Suen. 1992. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Trans. on Systems, Man. Cybernet*, 22(3):418–435.

T. Zhang, F. Damerau, and D. E. Johnson. 2002. Text chunking based on a generalization of Winnow. *Journal of Machine Learning Research*, 2:615–637.

Integration of Speech to Computer-Assisted Translation Using Finite-State Automata

Shahram Khadivi **Richard Zens** **Hermann Ney**
Lehrstuhl für Informatik 6 – Computer Science Department
RWTH Aachen University, D-52056 Aachen, Germany
{khadivi,zens,ney}@cs.rwth-aachen.de

Abstract

State-of-the-art computer-assisted translation engines are based on a statistical prediction engine, which interactively provides completions to what a human translator types. The integration of human speech into a computer-assisted system is also a challenging area and is the aim of this paper. So far, only a few methods for integrating statistical machine translation (MT) models with automatic speech recognition (ASR) models have been studied. They were mainly based on N-best rescoring approach. N-best rescoring is not an appropriate search method for building a real-time prediction engine. In this paper, we study the incorporation of MT models and ASR models using finite-state automata. We also propose some transducers based on MT models for rescoring the ASR word graphs.

1 Introduction

A desired feature of computer-assisted translation (CAT) systems is the integration of the human speech into the system, as skilled human translators are faster at dictating than typing the translations (Brown et al., 1994). Additionally, incorporation of a statistical prediction engine, i.e. a statistical interactive machine translation system, to the CAT system is another useful feature. A statistical prediction engine provides the completions to what a human translator types (Foster et al., 1997; Och et al., 2003). Then, one possible procedure for skilled human translators is to provide the oral translation of a given source text and then to post-edit the recognized text. In the post-editing step, a prediction engine helps to decrease the amount of human interaction (Och et al., 2003).

In a CAT system with integrated speech, two sources of information are available to recognize the speech input: the target language speech and the given source language text. The target language speech is a human-produced translation of the source language text. Statistical machine translation (MT) models are employed to take into account the source text for increasing the accuracy of automatic speech recognition (ASR) models.

Related Work

The idea of incorporating ASR and MT models was independently initiated by two groups: researchers at IBM (Brown et al., 1994), and researchers involved in the TransTalk project (Dymetman et al., 1994; Brousseau et al., 1995). In (Brown et al., 1994), the authors proposed a method to integrate the IBM translation model 2 (Brown et al., 1993) with an ASR system. The main idea was to design a language model (LM) to combine the trigram language model probability with the translation probability for each target word. They reported a perplexity reduction, but no recognition results. In the TransTalk project, the authors improved the ASR performance by rescoring the ASR N-best lists with a translation model. They also introduced the idea of a dynamic vocabulary for a speech recognition system where translation models were generated for each source language sentence. The better performing of the two is the N-best rescoring.

Recently, (Khadivi et al., 2005) and (Paulik et al., 2005a; Paulik et al., 2005b) have studied the integration of ASR and MT models. The first work showed a detailed analysis of the effect of different MT models on rescoring the ASR N-best lists. The other two works considered two parallel N-best lists, generated by MT and ASR systems,

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 467–474,
Sydney, July 2006.

respectively. They showed improvement in the ASR N-best rescoring when some proposed features are extracted from the MT N-best list. The main concept among all features was to generate different kinds of language models from the MT N-best list.

All of the above methods are based on an N-best rescoring approach. In this paper, we study different methods for integrating MT models to ASR word graphs instead of N-best list. We consider ASR word graphs as finite-state automata (FSA), then the integration of MT models to ASR word graphs can benefit from FSA algorithms. The ASR word graphs are a compact representation of possible recognition hypotheses. Thus, the integration of MT models to ASR word graphs can be considered as an N-best rescoring but with very large value for N. Another advantage of working with ASR word graphs is the capability to pass on the word graphs for further processing. For instance, the resulting word graph can be used in the prediction engine of a CAT system (Och et al., 2003).

The remaining part is structured as follows: in Section 2, a general model for an automatic text dictation system in the computer-assisted translation framework will be described. In Section 3, the details of the machine translation system and the speech recognition system along with the language model will be explained. In Section 4, different methods for integrating MT models into ASR models will be described, and also the experimental results will be shown in the same section.

2 Speech-Enabled CAT Models

In a speech-enabled computer-assisted translation system, we are given a source language sentence $f_1^J = f_1 \dots f_j \dots f_J$, which is to be translated into a target language sentence $e_1^I = e_1 \dots e_i \dots e_I$, and an acoustic signal $x_1^T = x_1 \dots x_t \dots x_T$, which is the spoken target language sentence. Among all possible target language sentences, we will choose the sentence with the highest probability:

$$\hat{e}_1^{\hat{I}} = \underset{I,e_1^I}{\text{argmax}} \{Pr(e_1^I|f_1^J, x_1^T)\} \quad (1)$$

$$\cong \underset{I,e_1^I}{\text{argmax}} \{Pr(e_1^I) Pr(f_1^J|e_1^I) Pr(x_1^T|e_1^I)\} \quad (2)$$

Eq. 1 is decomposed into Eq. 2 by assuming conditional independency between x_1^T and f_1^J. The decomposition into three knowledge sources allows for an independent modeling of the target language model $Pr(e_1^I)$, the translation model $Pr(f_1^J|e_1^I)$ and the acoustic model $Pr(x_1^T|e_1^I)$.

Another approach for modeling the posterior probability $Pr(e_1^I|f_1^J, x_1^T)$ is direct modeling using a log-linear model. The decision rule is given by:

$$\hat{e}_1^{\hat{I}} = \underset{I,e_1^I}{\text{argmax}} \left\{ \sum_{m=1}^{M} \lambda_m h_m(e_1^I, f_1^J, x_1^T) \right\} \quad (3)$$

Each of the terms $h_m(e_1^I, f_1^J, x_1^T)$ denotes one of the various models which are involved in the recognition procedure. Each individual model is weighted by its scaling factor λ_m. As there is no direct dependence between f_1^J and x_1^T, the $h_m(e_1^I, f_1^J, x_1^T)$ is in one of these two forms: $h_m(e_1^I, x_1^T)$ and $h_m(e_1^I, f_1^J)$. Due to the argmax operator which denotes the search, no renormalization is considered in Eq. 3. This approach has been suggested by (Papineni et al., 1997; Papineni et al., 1998) for a natural language understanding task, by (Beyerlein, 1998) for an ASR task, and by (Och and Ney, 2002) for an MT task. This approach is a generalization of Eq. 2. The direct modeling has the advantage that additional models can be easily integrated into the overall system. The model scaling factors λ_1^M are trained on a development corpus according to the final recognition quality measured by the word error rate (WER)(Och, 2003).

Search

The search in the MT and the ASR systems is already very complex, therefore a fully integrated search to combine ASR and MT models will considerably increase the complexity. To reduce the complexity of the search, we perform two independent searches with the MT and the ASR systems, the search result of each system will be represented as a large word graph. We consider MT and ASR word graphs as FSA. Then, we are able to use FSA algorithms to integrate MT and ASR word graphs. The FSA implementation of the search allows us to use standard optimized algorithms, e.g. available from an open source toolkit (Kanthak and Ney, 2004).

The recognition process is performed in two steps. First, the baseline ASR system generates a word graph in the FSA format for a given utterance x_1^T. Second, the translation models rescore each word graph based on the corresponding source language sentence. For each utterance, the decision about the best sentence is made according to the recognition and the translation models.

3 Baseline Components

In this section, we briefly describe the basic system components, namely the MT and the ASR systems.

3.1 Machine Translation System

We make use of the RWTH phrase-based statistical machine translation system for the English to German automatic translation. The system includes the following models: an n-gram language model, a phrase translation model and a word-based lexicon model. The latter two models are used for both directions: German to English and English to German. Additionally, a word penalty and a phrase penalty are included. The reordering model of the baseline system is distance-based, i.e. it assigns costs based on the distance from the end position of a phrase to the start position of the next phrase. More details about the baseline system can be found in (Zens and Ney, 2004; Zens et al., 2005).

3.2 Automatic Speech Recognition System

The acoustic model of the ASR system is trained on the VerbMobil II corpus (Sixtus et al., 2000). The corpus consists of German large-vocabulary conversational speech: 36k training sentences (61.5h) from 857 speakers. The test corpus is created from the German part of the bilingual English-German XEROX corpus (Khadivi et al., 2005): 1562 sentences including 18k running words (2.6h) from 10 speakers. The test corpus contains 114 out-of-vocabulary (OOV) words. The remaining part of the XEROX corpus is used to train a back off trigram language model using the SRI language modeling toolkit (Stolcke, 2002). The LM perplexity of the speech recognition test corpus is about 83. The acoustic model of the ASR system can be characterized as follows:

- recognition vocabulary of 16716 words;
- 3-state-HMM topology with skip;
- 2500 decision tree based generalized within-word triphone states including noise plus one state for silence;
- 237k gender independent Gaussian densities with global pooled diagonal covariance;
- 16 MFCC features;
- 33 acoustic features after applying LDA;
- LDA is fed with 11 subsequent MFCC vectors;
- maximum likelihood training using Viterbi approximation.

Table 1: Statistics of the machine translation corpus.

		English	German
Train:	Sentences	47 619	
	Running Words	528 779	467 633
	Vocabulary	9 816	16 716
	Singletons	2 302	6 064
Dev:	Sentences	700	
	Running Words	8 823	8 050
	Unknown words	56	108
Eval:	Sentences	862	
	Running Words	11 019	10 094
	Unknown words	58	100

The test corpus recognition word error rate is 20.4%. Compared to the previous system (Khadivi et al., 2005), which has a WER of 21.2%, we obtain a 3.8% relative improvement in WER. This improvement is due to a better and complete optimization of the overall ASR system.

4 Integration Approaches

In this section, we will introduce several approaches to integrate the MT models with the ASR models. To present the content of this section in a more reader-friendly way, we will first explain the task and corpus statistics, then we will present the results of N-best rescoring. Afterwards, we will describe the new methods for integrating the MT models with the ASR models. In each sub-section, we will also present the recognition results.

4.1 Task

The translation models are trained on the part of the English-German XEROX corpus which was not used in the speech recognition test corpus. We divide the speech recognition test corpus into two parts, the first 700 utterances as the development corpus and the rest as the evaluation corpus. The development corpus is used to optimize the scaling factors of different models (explained in Section 2). The statistics of the corpus are depicted in Table 1. The German part of the training corpus is also used to train the language model.

4.2 N-best Rescoring

To rescore the N-best lists, we use the method of (Khadivi et al., 2005). But the results shown here are different from that work due to a better optimization of the overall ASR system, using a

Table 2: Recognition WER [%] using N-best rescoring method.

Models		Dev	Eval
MT		47.1	50.5
ASR		19.3	21.3
ASR+MT	IBM-1	17.8	19.0
	HMM	18.2	19.2
	IBM-3	17.1	18.4
	IBM-4	17.1	18.3
	IBM-5	16.6	18.2
	Phrase -based	18.8	20.3

better MT system, and generating a larger N-best list from the ASR word graphs. We rescore the ASR N-best lists with the standard HMM (Vogel et al., 1996) and IBM (Brown et al., 1993) MT models. The development and evaluation sets N-best lists sizes are sufficiently large to achieve almost the best possible results, on average 1738 hypotheses per each source sentence are extracted from the ASR word graphs.

The recognition results are summarized in Table 2. In this table, the translation results of the MT system are shown first, which are obtained using the phrase-based approach. Then the recognition results of the ASR system are shown. Afterwards, the results of combined speech recognition and translation models are presented.

For each translation model, the N-best lists are rescored based on the translation probability $p(e_1^I|f_1^J)$ of that model and the probabilities of speech recognition and language models. In the last row of Table 2, the N-best lists are rescored based on the full machine translation system explained in Section 3.1.

The best possible hypothesis achievable from the N-best list has the WER (oracle WER) of 11.2% and 12.4% for development and test sets, respectively.

4.3 Direct Integration

At the first glance, an obvious method to combine the ASR and MT systems is the integration at the level of word graphs. This means the ASR system generates a large word graph for the input target language speech, and the MT system also generates a large word graph for the source language text. Both MT and ASR word graphs are in the target language. These two word graphs can be considered as two FSA, then using FSA theory, we can integrate two word graphs by applying the composition algorithm.

We conducted a set of experiments to integrate the ASR and MT systems using this method. We obtain a WER of 19.0% and 20.9% for development and evaluation sets, respectively. The results are comparable to N-best rescoring results for the phrase-based model which is presented in Table 2. The achieved improvements over the ASR baseline are statistically significant at the 99% level (Bisani and Ney, 2004). However, the results are not promising compared to the results of the rescoring method presented in Table 2 for HMM and IBM translation models. A detailed analysis revealed that only 31.8% and 26.7% of sentences in the development and evaluation sets have identical paths in both FSA, respectively. In other words, the search algorithm was not able to find any identical paths in two given FSA for the remaining sentences. Thus, the two FSA are very different from each other. One explanation for the failure of this method is the large difference between the WERs of two systems, as shown in Table 2 the WER for the MT system is more than twice as high as for the ASR system.

4.4 Integrated Search

In Section 4.3, two separate word graphs are generated using the MT and the ASR systems. Another explanation for the failure of the direct integration method is the independent search to generate the word graphs. The search in the MT and the ASR systems is already very complex, therefore a full integrated search to combine ASR and MT models will considerably increase the complexity.

However, it is possible to reduce this problem by integrating the ASR word graphs into the generation process of the MT word graphs. This means, the ASR word graph is used in addition to the usual language model. This kind of integration forces the MT system to generate identical paths to those in the ASR word graph. Using this approach, the number of identical paths in MT and ASR word graphs are increased to 39.7% and 34.4% of the sentences in development and evaluation sets, respectively. The WER of the integrated system are 19.0% and 20.7% for development and evaluation sets.

4.5 Lexicon-Based Transducer

The idea of a dynamic vocabulary, restricting and weighting the word lexicon of the ASR was first

introduced in (Brousseau et al., 1995). The idea was also seen later in (Paulik et al., 2005b), they extract the words of the MT N-best list to restrict the vocabulary of the ASR system. But they both reported a negative effect from this method on the recognition accuracy. Here, we extend the dynamic vocabulary idea by weighting the ASR vocabulary based on the source language text and the translation models. We use the lexicon model of the HMM and the IBM MT models. Based on these lexicon models, we assign to each possible target word e the probability $Pr(e|f_1^J)$. One way to compute this probability is inspired by IBM Model 1:

$$Pr(e|f_1^J) = \frac{1}{J+1}\sum_{j=0}^{J} p(e|f_j)$$

We can design a simple transducer (or more precisely an acceptor) using probability in Eq. 4 to efficiently rescore all paths (hypotheses) in the word graph with IBM Model 1:

$$\begin{aligned} P_{\text{IBM-1}}(e_1^I|f_1^J) &= \frac{1}{(J+1)^I}\prod_{i=1}^{I}\sum_{j=0}^{J} p(e_i|f_j) \\ &= \prod_{i=1}^{I}\frac{1}{(J+1)}\cdot p(e_i|f_1^J) \end{aligned}$$

The transducer is formed by one node and a number of self loops for each target language word. In each arc of this transducer, the input label is target word e and the weight is $-\log\frac{1}{J+1}\cdot p(e|f_1^J)$.

We conducted experiments using the proposed transducer. We built different transducers with the lexicons of HMM and IBM translation models. In Table 3, the recognition results of the rescored word graphs are shown. The results are very promising compared to the N-best list rescoring, especially as the designed transducer is very simple. Similar to the results for the N-best rescoring approach, these experiments also show the benefit of using HMM and IBM Models to rescore the ASR word graphs.

Due to its simplicity, this model can be easily integrated into the ASR search. It is a sentence specific unigram LM.

4.6 Phrase-Based Transducer

The phrase-based translation model is the main component of our translation system. The pairs of source and corresponding target phrases are extracted from the word-aligned bilingual training corpus (Zens and Ney, 2004). In this section, we design a transducer to rescore the ASR word graph using the phrase-based model of the MT system. For each source language sentence, we extract all possible phrases from the word-aligned training corpus. Using the target part of these phrases we build a transducer similar to the lexicon-based transducer. But instead of a target word on each arc, we have the target part of a phrase. The weight of each arc is the negative logarithm of the phrase translation probability.

Table 3: Recognition WER [%] using lexicon-based transducer to rescore ASR word graphs.

Models		Dev	Eval
ASR		19.3	21.3
ASR+MT	IBM-1	17.5	19.0
	HMM	17.8	19.2
	IBM-3	17.7	18.8
	IBM-4	17.8	18.8
	IBM-5	17.6	18.9

This transducer is a good approximation of non-monotone phrase-based-lexicon score. Using the designed transducer it is possible that some parts of the source texts are not covered or covered more than once. Then, this model can be compared to the IBM-3 and IBM-4 models, as they also have the same characteristic in covering the source words. The above assumption is not critical for rescoring the ASR word graphs, as we are confident that the word order is correct in the ASR output. In addition, we assume low probability for the existence of phrase pairs that have the same target phrase but different source phrases within a particular source language sentence.

Using the phrase-based transducer to rescore the ASR word graph results in WER of 18.8% and 20.2% for development and evaluation sets, respectively. The improvements are statistically significant at the 99% level compared to the ASR system. The results are very similar to the results obtained using N-best rescoring method. But the transducer implementation is much simpler because it does not consider the word-based lexicon, the word penalty, the phrase penalty, and the reordering models, it just makes use of phrase translation model. The designed transducer is much faster in rescoring the word graph than the MT system in rescoring the N-best list. The average speed to rescore the ASR word graphs with this transducer is 49.4 words/sec (source language

text words), while the average speed to translate the source language text using the MT system is 8.3 words/sec. The average speed for rescoring the N-best list is even slower and it depends on the size of N-best list.

A surprising result of the experiments as has also been observed in (Khadivi et al., 2005), is that the phrase-based model, which performs the best in MT, has the least contribution in improving the recognition results. The phrase-based model uses more context in the source language to generate better translations by means of better word selection and better word order. In a CAT system, the ASR system has much better recognition quality than MT system, and the word order of the ASR output is correct. On the other hand, the ASR recognition errors are usually single word errors and they are independent from the context. Therefore, the task of the MT models in a CAT system is to enhance the confidence of the recognized words based on the source language text, and it seems that the single word based MT models are more suitable than phrase-based model in this task.

4.7 Fertility-Based Transducer

In (Brown et al., 1993), three alignment models are described that include fertility models, these are IBM Models 3, 4, and 5. The fertility-based alignment models have a more complicated structure than the simple IBM Model 1. The fertility model estimates the probability distribution for aligning multiple source words to a single target word. The fertility model provides the probabilities $p(\phi|e)$ for aligning a target word e to ϕ source words. In this section, we propose a method for rescoring ASR word graphs based on the lexicon and fertility models.

In (Knight and Al-Onaizan, 1998), some transducers are described to build a finite-state based translation system. We use the same transducers for rescoring ASR word graphs. Here, we have three transducers: lexicon, null-emitter, and fertility. The lexicon transducer is formed by one node and a number of self loops for each target language word, similar to IBM Model 1 transducer in Section 4.5. On each arc of the lexicon transducer, there is a lexicon entry: the input label is a target word e, the output label is a source word f, and the weight is $-\log p(f|e)$. The null-emitter transducer, as its name states, emits the null word with a pre-defined probability after each input word. The fertility transducer is also a simple transducer to map zero or several instances of a source word to one instance of the source word.

The ASR word graphs are composed successively with the lexicon, null-emitter, fertility transducers and finally with the source language sentence. In the resulting transducer, the input labels of the best path represent the best hypothesis.

The mathematical description of the proposed method is as follows. We can decompose Eq. 1 using Bayes' decision rule:

$$\hat{e}_1^{\hat{I}} = \underset{I,e_1^I}{\text{argmax}}\{Pr(e_1^I|f_1^J, x_1^T)\} \quad (4)$$

$$\cong \underset{I,e_1^I}{\text{argmax}}\{Pr(f_1^J)Pr(e_1^I|f_1^J)Pr(x_1^T|e_1^I)\} \quad (5)$$

In Eq. 5, the term $Pr(x_1^T|e_1^I)$ is the acoustic model and can be represented with the ASR word graph[1], the term $Pr(e_1^I|f_1^J)$ is the translation model of the target language text to the source language text. The translation model can be represented by lexicon, fertility, and null-emitter transducers. Finally, the term $Pr(f_1^J)$ is a very simple language model, it is the source language sentence.

The source language model in Eq. 5 can be formed into the acceptor form in two different ways:

1. a linear acceptor, i.e. a sequence of nodes with one incoming arc and one outgoing arc, the words of source language text are placed consecutively in the arcs of the acceptor,

2. an acceptor containing possible permutations. To limit the permutations, we used an approach as in (Kanthak et al., 2005).

Each of these two acceptors results in different constraints for the generation of the hypotheses. The first acceptor restricts the system to generate exactly the same source language sentence, while the second acceptor forces the system to generate the hypotheses that are a reordered variant of the source language sentence. The experiments conducted do not show any significant difference in the recognition results among the two source language acceptors, except that the second acceptor is much slower than the first acceptor. Therefore, we use the first model in our experiments. Table 4 shows the results of rescoring the ASR word graphs using the fertility-based transducers.

[1]Actually, the ASR word graph is obtained by using $Pr(x_1^T|e_1^I)$ and $Pr(e_1^I)$ models. However, It does not cause any problem in the modeling, especially when we make use of the direct modeling, Eq. 3

Table 4: Recognition WER [%] using fertility-based transducer to rescore ASR word graphs.

Models		Dev	Eval
ASR		19.3	21.3
ASR+MT	IBM-3	17.4	18.6
	IBM-4	17.4	18.5
	IBM-5	17.6	18.7

As Table 4 shows, we get almost the same or slightly better results when compared to the lexicon-based transducers.

Another interesting point about Eq. 5 is its similarity to speech translation (translation from target spoken language to source language text). Then, we can describe a speech-enabled CAT system as similar to a speech translation system, except that we aim to get the best ASR output (the best path in the ASR word graph) rather than the best translation. This is because the best translation, which is the source language sentence, is already given.

5 Conclusion

We have studied different approaches to integrate MT with ASR models, mainly using finite-state automata. We have proposed three types of transducers to rescore the ASR word graphs: lexicon-based, phrase-based and fertility-based transducers. All improvements of the combined models are statistically significant at the 99% level with respect to the baseline system, i.e. ASR only.

In general, N-best rescoring is a simplification of word graph rescoring. As the size of N-best list is increased, the results obtained by N-best list rescoring approach the results of the word graph rescoring. But we should consider that the statement is correct when we use exactly the same model and the same implementation to rescore the N-best list and word graph. Figure 1 shows the effect of the N-best list size on the recognition WER of the evaluation set. As we expected, the recognition results of N-best rescoring improve as N becomes larger, until the point that the recognition result converges to its optimum value. As shown in Figure 1, we should not expect that word graph rescoring methods outperform the N-best rescoring method, when the size of N-best lists are large enough. In Table 2, the recognition results are calculated using a large enough size for N-best lists, a maximum of 5,000 per sentence, which results in the average of 1738 hypotheses

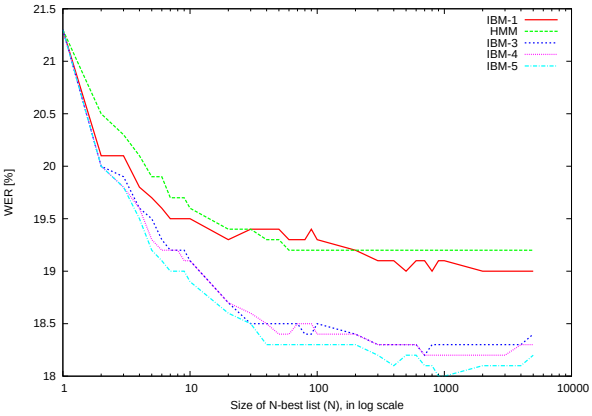


Figure 1: The N-best rescoring results for different N-best sizes on the evaluation set.

per sentence. An advantage of the word graph rescoring is the confidence of achieving the best possible results based on a given rescoring model.

The word graph rescoring methods presented in this paper improve the baseline ASR system with statistical significance. The results are competitive with the best results of N-best rescoring. For the simple models like IBM-1, the transducer-based integration generates similar or better results than N-best rescoring approach. For the more complex translation models, IBM-3 to IBM-5, the N-best rescoring produces better results than the transducer-based approach, especially for IBM-5. The main reason is due to exact estimation of IBM-5 model scores on the N-best list, while the transducer-based implementation of IBM-3 to IBM-5 is not exact and simplified. However, we observe that the fertility-based transducer which can be considered as a simplified version of IBM-3 to IBM-5 models can still obtain good results, especially if we compare the results on the evaluation set.

Acknowledgement

This work has been funded by the European Union under the RTD project TransType2 (IST 2001 32091) and the integrated project TC-STAR - Technology and Corpora for Speech to Speech Translation -(IST-2002-FP6-506738, http://www.tc-star.org).

References

P. Beyerlein. 1998. Discriminative model combination. In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 481 – 484, Seattle, WA, May.

M. Bisani and H. Ney. 2004. Bootstrap estimates for confidence intervals in ASR performance evaluationx. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 409–412, Montreal, Canada, May.

J. Brousseau, C. Drouin, G. Foster, P. Isabelle, R. Kuhn, Y. Normandin, and P. Plamondon. 1995. French speech recognition in an automatic dictation system for translators: the transtalk project. In *Proceedings of Eurospeech*, pages 193–196, Madrid, Spain.

P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, June.

P. F. Brown, S. F. Chen, S. A. Della Pietra, V. J. Della Pietra, A. S. Kehler, and R. L. Mercer. 1994. Automatic speech recognition in machine-aided translation. *Computer Speech and Language*, 8(3):177–187, July.

M. Dymetman, J. Brousseau, G. Foster, P. Isabelle, Y. Normandin, and P. Plamondon. 1994. Towards an automatic dictation system for translators: the TransTalk project. In *Proceedings of ICSLP-94*, pages 193–196, Yokohama, Japan.

G. Foster, P. Isabelle, and P. Plamondon. 1997. Target-text mediated interactive machine translation. *Machine Translation*, 12(1):175–194.

S. Kanthak and H. Ney. 2004. FSA: An efficient and flexible C++ toolkit for finite state automata using on-demand computation. In *Proc. of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 510–517, Barcelona, Spain, July.

S. Kanthak, D. Vilar, E. Matusov, R. Zens, and H. Ney. 2005. Novel reordering approaches in phrase-based statistical machine translation. In *43rd Annual Meeting of the Assoc. for Computational Linguistics: Proc. Workshop on Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond*, pages 167–174, Ann Arbor, Michigan, June.

S. Khadivi, A. Zolnay, and H. Ney. 2005. Automatic text dictation in computer-assisted translation. In *Interspeech'2005 - Eurospeech, 9th European Conference on Speech Communication and Technology*, pages 2265–2268, Portugal, Lisbon.

K. Knight and Y. Al-Onaizan. 1998. Translation with finite-state devices. In D. Farwell, L. Gerber, and E. H. Hovy, editors, *AMTA*, volume 1529 of *Lecture Notes in Computer Science*, pages 421–437. Springer Verlag.

F. J. Och and H. Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 295–302, Philadelphia, PA, July.

F. J. Och, R. Zens, and H. Ney. 2003. Efficient search for interactive statistical machine translation. In *EACL03: 10th Conf. of the Europ. Chapter of the Association for Computational Linguistics*, pages 387–393, Budapest, Hungary, April.

F. J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of the 41th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 160–167, Sapporo, Japan, July.

K. A. Papineni, S. Roukos, and R. T. Ward. 1997. Feature-based language understanding. In *EUROSPEECH*, pages 1435–1438, Rhodes, Greece, September.

K. A. Papineni, S. Roukos, and R. T. Ward. 1998. Maximum likelihood and discriminative training of direct translation models. In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 189–192, Seattle, WA, May.

M. Paulik, S. Stüker, C. Fügen, , T. Schultz, T. Schaaf, and A. Waibel. 2005a. Speech translation enhanced automatic speech recognition. In *Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 121–126, Puerto Rico, San Juan.

M. Paulik, C. Fügen, S. Stüker, T. Schultz, T. Schaaf, and A. Waibel. 2005b. Document driven machine translation enhanced ASR. In *Interspeech'2005 - Eurospeech, 9th European Conference on Speech Communication and Technology*, pages 2261–2264, Portugal, Lisbon.

A. Sixtus, S. Molau, S.Kanthak, R. Schlüter, and H. Ney. 2000. Recent improvements of the RWTH large vocabulary speech recognition system on spontaneous speech. In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1671 – 1674, Istanbul, Turkey, June.

A. Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proc. of the Int. Conf. on Speech and Language Processing (ICSLP)*, volume 2, pages 901–904, Denver, CO, September.

S. Vogel, H. Ney, and C. Tillmann. 1996. HMM-based word alignment in statistical translation. In *COLING '96: The 16th Int. Conf. on Computational Linguistics*, pages 836–841, Copenhagen, Denmark, August.

R. Zens and H. Ney. 2004. Improvements in phrase-based statistical machine translation. In *Proc. of the Human Language Technology Conf. (HLT-NAACL)*, pages 257–264, Boston, MA, May.

R. Zens, O. Bender, S. Hasan, S. Khadivi, E. Matusov, J. Xu, Y. Zhang, and H. Ney. 2005. The RWTH phrase-based statistical machine translation system. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, pages 155–162, Pittsburgh, PA, October.

GF Parallel Resource Grammars and Russian

Janna Khegai
Department of Computer Science
Chalmers University of Technology
SE-41296 Gothenburg, Sweden
janna@cs.chalmers.se

Abstract

A resource grammar is a standard library for the GF grammar formalism. It raises the abstraction level of writing domain-specific grammars by taking care of the general grammatical rules of a language. GF resource grammars have been built in parallel for eleven languages and share a common interface, which simplifies multilingual applications. We reflect on our experience with the Russian resource grammar trying to answer the questions: how well Russian fits into the common interface and where the line between language-independent and language-specific should be drawn.

1 Introduction

Grammatical Framework (GF) (Ranta, 2004) is a grammar formalism designed in particular to serve as an interlingua platform for natural language applications in sublanguage domains. A domain can be described using the GF grammar formalism and then processed by GF. Such descriptions are called **application grammars**.

A **resource grammar** (Ranta, to appear) is a general-purpose grammar that forms a basis for application grammars. Resource grammars have so far been implemented for eleven languages in parallel. The structural division into **abstract** and **concrete** descriptions, advocated in GF, is used to separate the language-independent common interface or **Application Programming Interface** (**API**) from corresponding language-specific implementations. Consulting the abstract part is sufficient for writing an application grammar without descending to implementation details. This approach raises the level of application grammar development and supports multilinguality, thus, providing both linguistic and computational advantages.

The current coverage is comparable with the Core Language Engine (CLE) project (Rayner et al., 2000). Other well-known multilingual general-purpose grammar projects that GF can be related to, are LFG grammars (Butt et al., 1999) and HPSG grammars (Pollard and Sag, 1994), although their parsing-oriented unification-based formalisms are very different from the GF generation-oriented type-theoretical formalism (Ranta, 2004).

A Russian resource grammar was added after similar grammars for English, Swedish, French and German (Arabic, Italian, Finnish, Norwegian, Danish and Spanish are also supported in GF). A language-independent API representing the coverage of the resource library, therefore, was already available. The task was to localize modules for Russian.

A resource grammar has morphological and syntactic modules. Morphological modules include a description of word classes, inflectional paradigms and a lexicon. Syntactic modules comprise a description of phrasal structures for analyzing bigger than one-word entities and various combination rules. Note, that semantics, defining the meanings of words and syntactic structures, is constructed in application grammars. This is because semantics is rather domain-specific, and, thus, it is much easier to construct a language-independent semantic model for a particular domain than a general-purpose resource semantics.

In the following sections we consider typical definitions from different resource modules focusing on aspects specific to Russian. We will also

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 475–482,
Sydney, July 2006.

demonstrate the library usage in a sample application grammar.

2 Word Classes

Every resource grammar starts with a description of word classes. Their names belong to the language-independent API, although their implementations are language-specific. Russian fits quite well into the common API here, since like all other languages it has nouns, verbs, adjectives etc. The type system for word classes of a language is the most stable part of the resource grammar library, since it follows traditional linguistic descriptions (Shelyakin, 2000; Wade, 2000; Starostin, 2005). For example, let us look at the implementation of the Russian adjective type `AdjDegree`:

```
param
 Degree  = Pos | Comp | Super;
 Case = Nom|Gen|Dat|Acc|Inst|Prep;
 Animacy = Animate | Inanimate;
 Gender = Masc | Fem | Neut;
 GenNum = ASingular Gender|APlural;
 AdjForm = AF Case Animacy GenNum;

oper
 AdjDegree : Type =
  {s : Degree => AdjForm => Str};
```

First, we need to specify parameters (`param`) on which inflection forms depend. A vertical slash (`|`) separates different parameter values. While in English the only parameter would be comparison degree (`Degree`), in Russian we have many more parameters:

- `Case`, for example: *большие дома – больших домов* (*big houses – big houses'*).
- `Animacy` only plays a role in the accusative case (`Acc`) in masculine (`Masc`) singular (`ASingular`) and in plural forms (`APlural`), namely, accusative animate form is the same as genitive (`Gen`) form, while accusative inanimate form is the same as nominative (`Nom`): *Я люблю большие дома – я люблю больших мужчин* (*I love big houses – I love big men*).
- `Gender` only plays role in singular: *большой дом – большая машина* (*big house – big car*). The plural never makes a gender distinction, thus, `Gender` and number are combined in the `GenNum` parameter to reduce redundant inflection table items. The possible values of `GenNum` are `ASingular Masc`, `ASingular Fem`, `ASingular Neut` and `APlural`.
- `Number`, for instance: *большой дом – большие дома* (*a big house – big houses*).
- `Degree` can be more complex, since most Russian adjectives have two comparative (`Comp`) forms: declinable attributive and indeclinable predicative[1]: *более высокий* (*more high*) – *выше* (*higher*), and more than one superlative (`Super`) forms: *самый высокий* (*the most high*) – *наивысший* (*the highest*).

Even another parameter can be added, since Russian adjectives in the positive (`Pos`) degree have long and short forms: *спокойная река* (*the calm river*) – *река – спокойна* (*the river is calm*). The short form has no case declension, thus, it can be considered as an additional case (Starostin, 2005). Note, that although the predicative usage of the long form is perfectly grammatical, it can have a slightly different meaning compared to the short form. For example: long, predicative *он – больной* (*"he is crazy"*) vs. short, predicative *он – болен* (*"he is ill"*).

An `oper` judgement combines the name of the defined operation, its type, and an expression defining it. The type for degree adjective (`AdjDegree`) is a table of strings (`s: .. => ..=> Str`) that has two main dimensions: `Degree` and `AdjForm`, where the last one is a combination of the parameters listed above. The reason to have the `Degree` parameter as a separate dimension is that a special type of adjectives `Adj` that just have positive forms is useful. It includes both non-degree adjective classes: possessive, like *мамин* (*mother's*), *лисий* (*fox'es*), and relative, like *русский* (*Russian*).

As a part of the language-independent API, the name `AdjDegree` denotes the adjective degree type for all languages, although each language has its own implementation. Maintaining parallelism among languages is rather straightforward at this stage, since the only thing shared is the name of

[1]The English *-er*/*more* and *-est*/*most* variations are exclusive, while in Russian both forms are valid.

a part of speech. A possible complication is that parsing with inflectionally rich languages can be less efficient compared to, for instance, English. This is because in GF all forms of a word are kept in the same declension table, which is convenient for generation, since GF is a generation-oriented grammar formalism. Therefore, the more forms there are, the bigger tables we have to store in memory, which can become an issue as the grammars grow and more languages are added (Dada and Ranta, 2006).

3 Inflection Paradigms and Lexicon

Besides word class declarations, morphology modules also contain functions defining common inflectional patterns (**paradigms**) and a lexicon. This information is language-specific, so fitting into the common API is not a consideration here. Paradigms are used to build the lexicon incrementally as new words are used in applications. A lexicon can also be extracted from other sources.

Unlike syntactic descriptions, morphological descriptions for many languages have been already developed in other projects. Thus, considerable efforts can be saved by reusing existing code. How easy we can perform the transformation depends on how similar the input and output formats are. For example, the Swedish morphology module is generated automatically from the code of another project, called Functional Morphology (Forsberg and Ranta, 2004). In this case the formats are very similar, so extracting is rather straightforward. However, this might not be the case if we build the lexicon from a very different representation or even from corpora, where post-modification by hand is simply inevitable.

A paradigm function usually takes one or more string arguments and forms a lexical entry. For example, the function `nGolova` describes the inflectional pattern for feminine inanimate nouns ending with *-a* in Russian. It takes the basic form of a word as a string (`Str`) and returns a noun (`CN` stands for Common Noun, see definition in section 4). Six cases times two numbers gives twelve forms, plus two inherent parameters `Animacy` and `Gender` (defined in section 2):

```
oper
 nGolova:  Str -> CN = \golova ->
   let golov = init golova in {
   s = table {
    SF Sg Nom => golov+"a";
    SF Sg Gen => golov+"ы";
    SF Sg Dat => golov+"е";
    SF Sg Acc => golov+"у";
    SF Sg Inst => golov+"ой";
    SF Sg Prepos => golov+"е";
    SF Pl Nom => golov+"ы";
    SF Pl Gen => golov;
    SF Pl Dat => golov+"ам";
    SF Pl Acc => golov+"ы";
    SF Pl Inst => golov+"ами";
    SF Pl Prepos => golov+"ах" };
   g = Fem;
   anim = Inanimate };
```

where `\golova` is a λ-abstraction, which means that the function argument of the type `Str` will be denoted as `golova` in the definition. The construction `let...in` is used to extract the word stem (`golov`), in this case, by cutting off the last letter (`init`). Of course, one could supply the stem directly, however, it is easier for the grammarian to just write the whole word without worrying what stem it has and let the function take care of the stem automatically. The table structure is simple – each line corresponds to one parameter value. The sign `=>` separates parameter values from corresponding inflection forms. Plus sign denotes string concatenation.

The **type signature** (`nGolova: Str -> CN`) and maybe a comment telling that the paradigm describes feminine inanimate nouns ending with *-a* are the only things the grammarian needs to know, in order to use the function `nGolova`. Implementation details (the inflection table) are hidden. The name `nGolova` is actually a transliteration of the Russian word *голова* (*head*) that represents nouns conforming to the pattern. Therefore, the grammarian can just compare a new word to the word *голова* in order to decide whether `nGolova` is appropriate. For example, we can define the word `mashina` (*машина*) corresponding to the English word *car*. Машина is a feminine, inanimate noun ending with *-a*. Therefore, a new lexical entry for the word *машина* can be defined by:

```
oper mashina = nGolova "машина" ;
```

Access via type signature becomes especially helpful with more complex parts of speech like verbs.

Lexicon and inflectional paradigms are language-specific, although, an attempt to build

a general-purpose interlingua lexicon in GF has been made. Multilingual dictionary can work for words denoting unique objects like *the sun* etc., but otherwise, having a common lexicon interface does not sound like a very good idea or at least something one would like to start with. Normally, multilingual dictionaries have bilingual organization (Kellogg, 2005).

At the moment the resource grammar has an interlingua dictionary for, so called, closed word classes like pronouns, prepositions, conjunctions and numerals. But even there, a number of discrepancies occurs. For example, the impersonal pronoun *one* (`OnePron`) has no direct correspondence in Russian. Instead, to express the same meaning Russian uses the infinitive: *если очень захотеть, можно в космос улететь* (*if one really wants, one can fly into the space*). Note, that the modal verb *can* is transformed into the adverb *можно* (*it is possible*). The closest pronoun to *one* is the personal pronoun *ты* (*you*), which is omitted in the final sentence: *если очень захочешь, можешь в космос улететь*. The Russian implementation of `OnePron` uses the later construction, skipping the string (`s`), but preserving number (`n`), person (`p`) and animacy (`anim`) parameters, which are necessary for agreement:

```
oper OnePron: Pronoun = {
  s = "";
  n = Singular;
  p = P2;
  anim = Animate };
```

4 Syntax

Syntax modules describe rules for combining words into phrases and sentences. Designing a language-independent syntax API is the most difficult part: several revisions have been made as the resource coverage has grown. Russian is very different from other resource languages, therefore, it sometimes fits poorly into the common API.

Several factors have influenced the API structure so far: application domains, parsing algorithms and supported languages. In general, the resource syntax is built bottom-up, starting with rules for forming noun phrases and verb phrases, continuing with relative clauses, questions, imperatives, and coordination. Some textual and dialogue features might be added, such as contrasting, topicalization, and question-answer relations.

On the way from dictionary entries towards complete sentences, categories loose declension forms and, consequently, get more parameters that "memorize" what forms are kept, which is necessary to arrange agreement later on. Closer to the end of the journey string fields are getting longer as types contain more complex phrases, while parameters are used for agreement and then left behind. Sentence types are the ultimate types that just contain one string and no parameters, since everything is decided and agreed on by that point.

Let us take a look at Russian nouns as an example. A noun lexicon entry type (`CN`) mentioned in section 3 is defined like the following:

```
param
 SubstForm = SF Number Case;
oper
 CN: Type = {
  s:  SubstForm => Str;
  g: Gender;
  anim: Animacy  };
```

As we have seen in section 3, the string table field `s` contains twelve forms. On the other hand, to use a noun in a sentence we need only one form and several parameters for agreement. Thus, the ultimate noun type to be used in a sentence as an object or a subject looks more like Noun Phrase (`NP`):

```
oper NP : Type = {
  s: Case => Str;
  Agreement: {
    n: Number;
    p: Person;
    g: Gender;
    anim: Animacy} };
```

which besides `Gender` and `Animacy` also contains `Number` and `Person` parameters (defined in section 2), while the table field `s` only contains six forms: one for each `Case` value.

The transition from `CN` to `NP` can be done via various intermediate types. A noun can get modifiers like adjectives – *красная комната* (*the red room)*, determiners – *много шума* (*much ado)*, genitive constructions – *герой нашего времени* (*a hero of our time)*, relative phrases – *человек, который смеётся* (*the man who laughs)*. Thus, the string field (`s`) can eventually contain more than one word. A noun can become a part of other phrases, e.g. a predicate in a verb phrase – *знание – сила* (*knowledge is power)* or a complement

in a prepositional phrase – *за рекой, в тени деревьев* (*across the river and into the trees)*.

The language-independent API has an hierarchy of intermediate types all the way from dictionary entries to sentences. All supported languages follow this structure, although in some cases this does not happen naturally. For example, the division between definite and indefinite noun phrases is not relevant for Russian, since Russian does not have any articles, while being an important issue about nouns in many European languages. The common API contains functions supporting such division, which are all conflated into one in the Russian implementation. This is a simple case, where Russian easily fits into the common API, although a corresponding phenomenon does not really exist.

Sometimes, a problem does not arise until the joining point, where agreement has to be made. For instance, in Russian, numeral modification uses different cases to form a noun phrase in nominative case: *три товарища* (*three comrades*), where the noun is in nominative, but *пять товарищей* (*five comrades*), where the noun is in genitive! Two solutions are possible. An extra non-linguistic parameter bearing the semantics of a numeral can be included in the `Numeral` type. Alternatively, an extra argument (`NumberVal`), denoting the actual number value, can be introduced into the numeral modification function (`IndefNumNP`) to tell apart numbers with the last digit between 2 and 4 from other natural numbers:

```
oper IndefNumNP: NumberVal ->
        Numeral -> CN -> NP;
```

Unfortunately, this would require changing the language-independent API (adding the `NumberVal` argument) and consequent adjustments in all other languages that do not need this information. Note, that `IndefNumNP`, `Numeral`, `CN` (Common Noun) and `NP` (Noun Phrase) belong to the language-independent API, i.e. they have different implementations in different languages. We prefer the encapsulation version, since the other option will make the function more error-prone.

Nevertheless, one can argue for both solutions, which is rather typical while designing a common interface. One has to decide what should be kept language-specific and what belongs to the language-independent API. Often this decision is more or less a matter of taste. Since Russian is not the main language in the GF resource library, the tendency is to keep things language-specific at least until the common API becomes too restrictive for a representative number of languages.

The example above demonstrates a syntactic construction, which exist both in the language-independent API and in Russian although the common version is not as universal as expected. There are also cases, where Russian structures are not present in the common interface at all, since there is no direct analogy in other supported languages. For instance, a short adjective form is used in phrases like *мне нужна помощь* (*I need help*) and *ей интересно искусство* (*she is interested in art*). In Russian, the expressions do not have any verb, so they sound like *to me needed help* and *to her interesting art*, respectively. Here is the function `predShortAdj` describing such adjective predication[2] specific to Russian:

```
oper predShortAdj: NP -> Adj ->
 NP -> S = \I, Needed, Help -> {
   s = let {
     toMe  = I.s ! Dat;
     needed  = Needed.s !
      AF Short Help.g Help.n;
     help = Help.s ! Nom
    } in
    toMe ++ needed ++ help };
```

`predShortAdj` takes three arguments: a non-degree adjective (`Adj`) and two noun phrases (`NP`) that work as a predicate, a subject and an object in the returned sentence (`S`). The third line indicates that the arguments will be denoted as `Needed`, `I` and `Help`, respectively (λ-abstraction). The sentence type (`S`) only contains one string field `s`. The construction `let...in` is used to first form the individual words (`toMe`, `needed` and `help`) to put them later into a sentence. Each word is produced by taking appropriate forms from inflection tables of corresponding arguments (`Needed.s`, `Help.s` and `I.s`). In the noun arguments `I` and `Help` dative and nominative cases, respectively, are taken (`!`-sign denotes the selection operation). The adjective `Needed` agrees with the noun `Help`, so `Help`'s gender (`g`) and number (`n`) are used to build an appropriate adjective form (`AF Short Help.g Help.n`). This is exactly where we finally use the parameters from `Help` argument of the type `NP` defined above. We only use the declension tables from the argu-

[2] In this example we disregard adjective past/future tense markers *было/будет*.

ments `I` and `Needed` – other parameters are just thrown away. Note, that `predShortAdj` uses the type `Adj` for non-degree adjectives instead of `AdjDegree` presented in section 2. We also use the `Short` adjective form as an extra `Case`-value.

5 An Example Application Grammar

The purpose of the example is to show similarities between the same grammar written for different languages using the resource library. Such similarities increase the reuse of previously written code across languages: once written for one language a grammar can be ported to another language relatively easy and fast. The more language-independent API functions (names conventionally starting with a capital letter) a grammar contains, the more efficient the porting becomes.

We will consider a fragment of `Health` – a small phrase-book grammar written using the resource grammar library in English, French, Italian, Swedish and Russian. It can form phrases like *she has a cold and she needs a painkiller*. The following categories (`cat`) and functions (`fun`) constitute language-independent abstract syntax (domain semantics):

```
cat
 Patient; Condition;
 Medicine; Prop;
fun
 ShePatient:    Patient;
 CatchCold:     Condition;
 PainKiller:    Medicine;
 BeInCondition: Patient ->
   Condition -> Prop;
 NeedMedicine:  Patient ->
   Medicine  -> Prop;
 And:   Prop -> Prop -> Prop;
```

Abstract syntax determines the class of statements we are able to build with the grammar. The category `Prop` denotes complete propositions like *she has a cold*. We also have separate categories of smaller units like `Patient`, `Medicine` and `Condition`. To produce a proposition one can, for instance, use the function `BeInCondition`, which takes two arguments of the types `Patient` and `Condition` and returns the result of the type `Prop`. For example, we can form the phrase *she has a cold* by combining three functions above:

```
BeInCondition
  ShePatient CatchCold
```

where `ShePatient` and `CatchCold` are constants used as arguments to the function `BeInCondition`.

Concrete syntax translates abstract syntax into natural language strings. Thus, concrete syntax is language-specific. However, having the language-independent resource API helps to make even a part of concrete syntax shared among the languages:

```
lincat
 Patient       = NP;
 Condition     = VP;
 Medicine      = CN;
 Prop          = S;
lin
 And           = ConjS;
 ShePatient    = SheNP;
 BeInCondition = PredVP;
```

The first group (`lincat`) tells that the semantic categories `Patient`, `Condition`, `Medicine` and `Prop` are expressed by the resource linguistic categories: noun phrase (`NP`), verb phrase (`VP`), common noun (`CN`) and sentence (`S`), respectively. The second group (`lin`) tells that the function `And` is the same as the resource coordination function `ConjS`, the function `ShePatient` is expressed by the resource pronoun `SheNP` and the function `BeInCondition` is expressed by the resource function `PredVP` (the classic `NP_VP->S` rule). Exactly the same rules work for all five languages, which makes the porting trivial[3]. However, this is not always the case.

Writing even a small grammar in an inflectionally rich language like Russian requires a lot of work on morphology. This is the part where using the resource grammar library may help, since resource functions for adding new lexical entries are relatively easy to use. For instance, the word *painkiller* is defined similarly in five languages by taking a corresponding basic word form as an argument to an inflection paradigm function:

```
-- English:
PainKiller = regN "painkiller";

-- French:
PainKiller = regN "calmant";

-- Italian:
PainKiller = regN "calmante";
```

[3] Different languages can actually share the same code using GF parameterized modules (Ranta, to appear)

```
-- Swedish:
PainKiller = regGenN
  "smärtstillande" Neut;

-- Russian:
PainKiller = nEe "обезболивающее";
```

The `Gender` parameter (`Neut`) is provided for Swedish.

In the remaining functions we see bigger differences: the idiomatic expressions *I have a cold* in French, Swedish and Russian is formed by adjective predication, while a transitive verb construction is used in English and Italian. Therefore, different functions (`PosA` and `PosTV`) are applied. `tvHave` and `tvAvere` denote transitive verb *to have* in English and Italian, respectively. `IndefOneNP` is used for forming an indefinite noun phrase from a noun in English and Italian:

```
-- English:
CatchCold = PosTV tvHave
 (IndefOneNP (regN "cold"));

-- Italian:
CatchCold = PosTV tvAvere
 (IndefOneNP (regN "raffreddore"));

-- French:
CatchCold = PosA (regA "enrhumé")

-- Swedish:
CatchCold = PosA
(mk2A "förkyld" "förkylt");

-- Russian:
CatchCold = PosA
 (adj_yj "простужен");
```

In the next example the Russian version is rather different from the other languages. The phrase *I need a painkiller* is a transitive verb predication together with complementation rule in English and Swedish. In French and Italian we need to use the idiomatic expressions *avoir besoin* and *aver bisogno*. Therefore, a classic `NP_VP` rule (`PredVP`) is used. In Russian the same meaning is expressed by using adjective predication defined in section 4:

```
--English:
NeedMedicine pat med = predV2
 (dirV2 (regV "need"))
  pat (IndefOneNP med);

-- Swedish:
NeedMedicine pat med = predV2
  (dirV2 (regV "behöver"))
  pat (DetNP nullDet med);

-- French:
NeedMedicine pat med = PredVP
 pat (avoirBesoin med);

-- Italian:
NeedMedicine pat med = PredVP
 pat (averBisogno med);

-- Russian:
NeedMedicine pat med =
 predShortAdj pat
 (adj_yj "нужен") med;
```

Note, that the medicine argument (`med`) is used with indefinite article in the English version (`IndefOneNP`), but without articles in Swedish, French and Italian. As we have mentioned in section 4, Russian does not have any articles, although the corresponding operations exist for the sake of consistency with the language-independent API.

`Health` grammar shows that the more similar languages are, the easier porting will be. However, as with traditional translation the grammarian needs to know the target language, since it is not clear whether a particular construction is correct in both languages, especially, when the languages seem to be very similar in general.

6 Conclusion

GF resource grammars are general-purpose grammars used as a basis for building domain-specific application grammars. Among pluses of using such grammar library are guaranteed grammaticality, code reuse (both within and across languages) and higher abstraction level for writing application grammars. According to the ”division of labor” principle, resource grammars comprise the necessary linguistic knowledge allowing application grammarians to concentrate on domain semantics.

Following Chomsky’s universal grammar hypothesis (Chomsky, 1981), GF multilingual resource grammars maintain a common API for all supported languages. This is implemented using

GF's mechanism of separating between abstract and concrete syntax. Abstract syntax declares universal principles, while language-specific parameters are set in concrete syntax. We are not trying to answer the general question what constitutes universal grammar and what beyond universal grammar differentiates languages from one another. We look at GF parallel resource grammars as a way to simplify multilingual applications.

The implementation of the Russian resource grammar proves that GF grammar formalism allows us to use the language-independent API for describing sometimes rather peculiar grammatical variations in different languages. However, maintaining parallelism across languages has its limits. From the beginning we were trying to put as much as possible into a common interface, shared among all the supported languages. Word classes seem to be rather universal at least for the eleven supported languages. Syntactic types and some combination rules are more problematic. For example, some Russian rules only make sense as a part of language-specific modules while some rules that were considered universal at first are not directly applicable to Russian.

Having a universal resource API and grammars for other languages has made developing Russian grammar much easier comparing to doing it from scratch. The abstract syntax part was simply reused. Some concrete syntax implementations like adverb description, coordination and subordination required only minor changes. Even for more language-specific rules it helps a lot to have a template implementation that demonstrates what kind of phenomena should be taken into account.

The GF resource grammar development is mostly driven by application domains like software specifications (Burke and Johannisson, 2005), math problems (Caprotti, 2006) or transport network dialog systems (Bringert et al., 2005). The structure of the resource grammar library is continually influenced by new domains and languages. The possible direction of GF parallel resource grammars' development is extending the universal interface by domain-specific and language-specific parts. Such adaptation seems to be necessary as the coverage of GF resource grammars grows.

Acknowledgements

Thanks to Professor Arto Mustajoki for fruitful discussions and to Professor Robin Cooper for reading and editing the final version of the paper. Special thanks to Professor Aarne Ranta, my supervisor and the creator of GF.

References

B. Bringert, R. Cooper, P. Ljunglöf, and A. Ranta. 2005. Multimodal Dialogue System Grammars. In *DIALOR'05, Nancy, France*.

D.A. Burke and K. Johannisson. 2005. Translating Formal Software Specifications to Natural Language / A Grammar-Based Approach. In *LACL 2005*, *LNAI* 3402, pages 51–66. Springer.

M. Butt, T. H. King, M.-E. Ni no, and F. Segond, editors. 1999. *A Grammar Writer's Cookbook*. Stanford: CSLI Publications.

O. Caprotti. 2006. WebALT! Deliver Mathematics Everywhere. In *SITE 2006, Orlando, USA*.

N. Chomsky. 1981. *Lectures on Government and Binding: The Pisa Lectures.* Dordrecht, Holland: Foris Publications.

A. E. Dada and A. Ranta. 2006. Implementing an arabic resource grammar in grammatical framework. At 20th Arabic Linguistics Symposium, Kalamazoo, Michigan. URL: `www.md stud.chalmers.se/~eldada/paper.pdf`.

M. Forsberg and A. Ranta. 2004. Functional morphology. In *ICFP'04*, pages 213–223. ACM Press.

M. Kellogg. 2005. Online french, italian and spanish dictionary. URL: `www.wordreference.com`.

C. Pollard and I. Sag. 1994. *Head-Driven Phrase Structure Grammar*. University of Chicago Press.

A. Ranta. 2004. Grammatical Framework: A Type-theoretical Grammar Formalism. *The Journal of Functional Programming*, 14(2):145–189.

A. Ranta. to appear. Modular Grammar Engineering in GF. *Research in Language and Computation*. URL: `www.cs.chalmers.se/~aarne/articles/ar-multieng.pdf`

M. Rayner, D. Carter, P. Bouillon, V. Digalakis, and M. Wirén. 2000. *The spoken language translator*. Cambridge University Press.

M.A. Shelyakin. 2000. *Spravochnik po russkoj grammatike (in Russian)*. Russky Yazyk, Moscow.

S. Starostin. 2005. Russian morpho-engine on-line. URL: `starling.rinet.ru/morph.htm`.

T. Wade. 2000. *A Comprehensive Russian Grammar*. Blackwell Publishing.

Automatic Identification of Pro and Con Reasons in Online Reviews

Soo-Min Kim and Eduard Hovy
USC Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292-6695
{skim, hovy}@ISI.EDU

Abstract

In this paper, we present a system that automatically extracts the pros and cons from online reviews. Although many approaches have been developed for extracting opinions from text, our focus here is on extracting the reasons of the opinions, which may themselves be in the form of either fact or opinion. Leveraging online review sites with author-generated pros and cons, we propose a system for aligning the pros and cons to their sentences in review texts. A maximum entropy model is then trained on the resulting labeled set to subsequently extract pros and cons from online review sites that do not explicitly provide them. Our experimental results show that our resulting system identifies pros and cons with 66% precision and 76% recall.

1 Introduction

Many opinions are being expressed on the Web in such settings as product reviews, personal blogs, and news group message boards. People increasingly participate to express their opinions online. This trend has raised many interesting and challenging research topics such as subjectivity detection, semantic orientation classification, and review classification.

Subjectivity detection is the task of identifying subjective words, expressions, and sentences. (Wiebe et al., 1999; Hatzivassiloglou and Wiebe, 2000; Riloff et al, 2003). Identifying subjectivity helps separate opinions from fact, which may be useful in question answering, summarization, etc.

Semantic orientation classification is a task of determining positive or negative sentiment of words (Hatzivassiloglou and McKeown, 1997; Turney, 2002; Esuli and Sebastiani, 2005). Sentiment of phrases and sentences has also been studied in (Kim and Hovy, 2004; Wilson et al., 2005). Document level sentiment classification is mostly applied to reviews, where systems assign a positive or negative sentiment for a whole review document (Pang et al., 2002; Turney, 2002).

Building on this work, more sophisticated problems in the opinion domain have been studied by many researchers. (Bethard et al., 2004; Choi et al., 2005; Kim and Hovy, 2006) identified the holder (source) of opinions expressed in sentences using various techniques. (Wilson et al., 2004) focused on the strength of opinion clauses, finding strong and weak opinions. (Chklovski, 2006) presented a system that aggregates and quantifies degree assessment of opinions scattered throughout web pages.

Beyond document level sentiment classification in online product reviews, (Hu and Liu, 2004; Popescu and Etzioni, 2005) concentrated on mining and summarizing reviews by extracting opinion sentences regarding product features.

In this paper, we focus on another challenging yet critical problem of opinion analysis, identifying *reasons* for opinions, especially for opinions in online product reviews. The opinion reason identification problem in online reviews seeks to answer the question "*What are the reasons that the author of this review likes or dislikes the product?"* For example, in hotel reviews, information such as "found 189 positive reviews and 65 negative reviews" may not fully satisfy the information needs of different users. More useful information would be "This hotel is great for families with young infants" or "Elevators are grouped according to floors, which makes the wait short".

This work differs in important ways from studies in (Hu and Liu, 2004) and (Popescu and Etzioni, 2005). These approaches extract features

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 483–490,
Sydney, July 2006.

of products and identify sentences that contain opinions about those features by using opinion words and phrases. Here, we focus on extracting pros and cons which include not only sentences that contain opinion-bearing expressions about products and features but also sentences with reasons why an author of a review writes the review. Following are examples identified by our system.

```
It creates duplicate files.
Video drains battery.
It won't play music from all
music stores
```

Even though finding reasons in opinion-bearing texts is a critical part of in-depth opinion assessment, no study has been done in this particular vein partly because there is no annotated data. Labeling each sentence is a time-consuming and costly task. In this paper, we propose a framework for automatically identifying reasons in online reviews and introduce a novel technique to automatically label training data for this task. We assume reasons in an online review document are closely related to pros and cons represented in the text. We leverage the fact that reviews on some websites such as epinions.com already contain pros and cons written by the same author as the reviews. We use those pros and cons to automatically label sentences in the reviews on which we subsequently train our classification system. We then apply the resulting system to extract pros and cons from reviews in other websites which do not have specified pros and cons.

This paper is organized as follows: Section 2 describes a definition of reasons in online reviews in terms of pros and cons. Section 3 presents our approach to identify them and Section 4 explains our automatic data labeling process. Section 5 describes experimental and results and finally, in Section 6, we conclude with future work.

2 Pros and Cons in Online Reviews

This section describes how we define reasons in online reviews for our study. First, we take a look at how researchers in Computational Linguistics define an opinion for their studies. It is difficult to define what an opinion means in a computational model because of the difficulty of determining the unit of an opinion. In general, researchers study opinion at three different levels: *word level*, *sentence level*, and *document level*.

Word level opinion analysis includes word sentiment classification, which views single lexical items (such as *good* or *bad*) as sentiment carriers, allowing one to classify words into positive and negative semantic categories. Studies in sentence level opinion regard the sentence as a minimum unit of opinion. Researchers try to identify opinion-bearing sentences, classify their sentiment, and identify opinion holders and topics of opinion sentences. Document level opinion analysis has been mostly applied to review classification, in which a whole document written for a review is judged as carrying either positive or negative sentiment. Many researchers, however, consider a whole document as the unit of an opinion to be too coarse.

In our study, we take the approach that a review text has a main opinion (recommendation or not) about a given product, but also includes various reasons for recommendation or non-recommendation, which are valuable to identify. Therefore, we focus on detecting those reasons in online product review. We also assume that reasons in a review are closely related to pros and cons expressed in the review. Pros in a product review are sentences that describe reasons why an author of the review likes the product. Cons are reasons why the author doesn't like the product. Based on our observation in online reviews, most reviews have both pros and cons even if sometimes one of them dominates.

3 Finding Pros and Cons

This section describes our approach for finding pro and con sentences given a review text. We first collect data from epinions.com and automatically label each sentences in the data set. We then model our system using one of the machine learning techniques that have been successfully applied to various problems in Natural Language Processing. This section also describes features we used for our model.

3.1 Automatically Labeling Pro and Con Sentences

Among many web sites that have product reviews such as amazon.com and epinions.com, some of them (e.g. epinions.com) explicitly state pros and cons phrases in their respective categories by each review's author along with the review text. First, we collected a large set of <review text, pros, cons> triplets from epin-

ions.com. A review document in epinions.com consists of a topic (a product model, restaurant name, travel destination, etc.), pros and cons (mostly a few keywords but sometimes complete sentences), and the review text. Our automatic labeling system first collects phrases in pro and con fields and then searches the main review text in order to collect sentences corresponding to those phrases. Figure 1 illustrates the automatic labeling process.

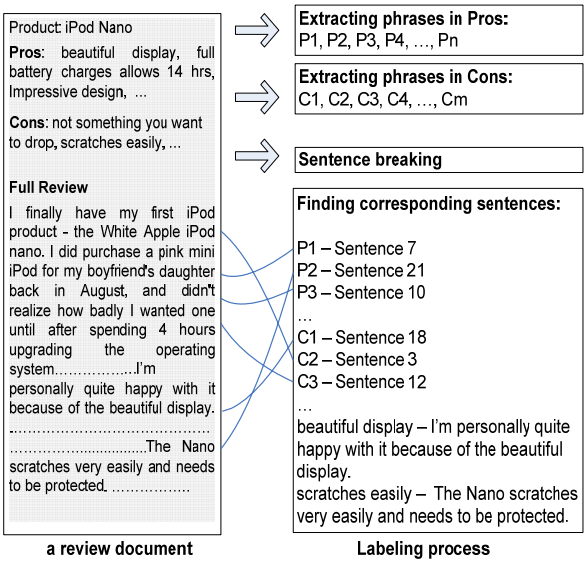


Figure 1. The automatic labeling process of pros and cons sentences in a review.

The system first extracts comma-delimited phrases from each pro and con field, generating two sets of phrases: {*P1, P2, …, Pn*} for pros and {*C1, C2, …, Cm*} for cons. In the example in Figure 1, "beautiful display" can be P_i and "not something you want to drop" can be C_j. Then the system compares these phrases to the sentences in the text in the "Full Review". For each phrase in {*P1, P2, …, Pn*} and {*C1, C2, …, Cm*}, the system checks each sentence to find a sentence that covers most of the words in the phrase. Then the system annotates this sentence with the appropriate "pro" or "con" label. All remaining sentences with neither label are marked as "neither". After labeling all the epinion data, we use it to train our pro and con sentence recognition system.

3.2 Modeling with Maximum Entropy Classification

We use Maximum Entropy classification for the task of finding pro and con sentences in a given review. Maximum Entropy classification has been successfully applied in many tasks in natural language processing, such as Semantic Role labeling, Question Answering, and Information Extraction.

Table 1: Classes defined for the classification tasks.

Class symbol	Description
PR	Sentences related to pros in a review
CR	Sentences related to cons in a review
NR	Sentences related to neither PR nor CR

Maximum Entropy models implement the intuition that the best model is the one that is consistent with the set of constraints imposed by the evidence but otherwise is as uniform as possible (Berger et al., 1996). We modeled the conditional probability of a class c given a feature vector x as follows:

$$p(c \mid x) = \frac{1}{Z_x} \exp(\sum_i \lambda_i f_i(c, x))$$

where Z_x is a normalization factor which can be calculated by the following:

$$Z_x = \sum_c \exp(\sum_i \lambda_i f_i(c, x))$$

In the first equation, $f_i(c, x)$ is a feature function which has a binary value, 0 or 1. λ_i is a weight parameter for the feature function $f_i(c, x)$ and higher value of the weight indicates that $f_i(c, x)$ is an important feature for a class c. For our system development, we used MegaM toolkit [1] which implements the above intuition.

In order to build an efficient model, we separated the task of finding pro and con sentences into two phases, each being a binary classification. The first is an identification phase and the second is a classification phase. For this 2-phase model, we defined the 3 classes of c listed in Table 1. The identification task separates pro and con candidate sentences (CR and PR in Table 1) from sentences irrelevant to either of them (NR). The classification task then classifies candidates into pros (PR) and cons (CR). Section 5 reports system results of both phases.

[1] http://www.isi.edu/~hdaume/megam/index.html

3.3 Features

The classification uses three types of features: lexical features, positional features, and opinion-bearing word features.

For **lexical features**, we use unigrams, bigrams, and trigrams collected from the training set. They investigate the intuition that there are certain words that are frequently used in pro and con sentences which are likely to represent reasons why an author writes a review. Examples of such words and phrases are: *"because"* and *"that's why"*.

For **positional features**, we first find paragraph boundaries in review texts using html tags such as <*br*> and <*p*>. After finding paragraph boundaries, we add features indicating the first, the second, the last, and the second last sentence in a paragraph. These features test the intuition used in document summarization that important sentences that contain topics in a text have certain positional patterns in a paragraph (Lin and Hovy, 1997), which may apply because reasons like pros and cons in a review document are most important sentences that summarize the whole point of the review.

For **opinion-bearing word features**, we used pre-selected opinion-bearing words produced by a combination of two methods. The first method derived a list of opinion-bearing words from a large news corpus by separating opinion articles such as letters or editorials from news articles which simply reported news or events. The second method calculated semantic orientations of words based on WordNet[2] synonyms. In our previous work (Kim and Hovy, 2005), we demonstrated that the list of words produced by a combination of those two methods performed very well in detecting opinion bearing sentences. Both algorithms are described in that paper.

The motivation for including the list of opinion-bearing words as one of our features is that pro and con sentences are quite likely to contain opinion-bearing expressions (even though some of them are only facts), such as "The waiting time was horrible" and "Their portion size of food was extremely generous!" in restaurant reviews. We presumed pro and con sentences containing only facts, such as "*The battery lasted 3 hours, not 5 hours like they advertised*", would be captured by lexical or positional features.

In Section 5, we report experimental results with different combinations of these features.

Table 2: Feature summary.

Feature category	Description	Symbol
Lexical Features	unigrams bigrams trigrams	***Lex***
Positional Features	the first, the second, the last, the second to last sentence in a paragraph	***Pos***
Opinion-bearing word features	pre-selected opinion-bearing words	***Op***

Table 2 summarizes the features we used for our model and the symbols we will use in the rest of this paper.

4 Data

We collected data from two different sources: epinions.com and complaints.com[3] (see Section 3.1 for details about review data in epinion.com). Data from epinions.com is mostly used to train the system whereas data from complaints.com is to test how the trained model performs on new data.

Complaints.com includes a large database of publicized consumer complaints about diverse products, services, and companies collected for over 6 years. Interestingly, reviews in complaint.com are somewhat different from many other web sites which are directly or indirectly linked to Internet shopping malls such as amazon.com and epinions.com. The purpose of reviews in complaints.com is to share consumers' mostly negative experiences and alert businesses to customers feedback. However, many reviews in Internet shopping mall related reviews are positive and sometimes encourage people to buy more products or to use more services.

Despite its significance, however, there is no hand-annotated data that we can use to build a system to identify reasons of complaints.com. In order to solve this problem, we assume that reasons in complaints reviews are similar to cons in other reviews and therefore if we are, somehow, able to build a system that can identify cons from

[2] http://wordnet.princeton.edu/

[3] http://www.complaints.com/

reviews, we can apply it to identify reasons in complaints reviews. Based on this assumption, we learn a system using the data from epinions.com, to which we can apply our automatic data labeling technique, and employ the resulting system to identify reasons from reviews in complaint.com. The following sections describe each data set.

4.1 Dataset 1: Automatically Labeled Data

We collected two different domains of reviews from epinions.com: product reviews and restaurant reviews. As for the product reviews, we collected 3241 reviews (115029 sentences) about mp3 players made by various manufacturers such as Apple, iRiver, Creative Lab, and Samsung. We also collected 7524 reviews (194393 sentences) about various types of restaurants such as family restaurants, Mexican restaurants, fast food chains, steak houses, and Asian restaurants. The average numbers of sentences in a review document are 35.49 and 25.89 respectively.

The purpose of selecting one of electronics products and restaurants as topics of reviews for our study is to test our approach in two extremely different situations. Reasons why consumers like or dislike a product in electronics' reviews are mostly about specific and tangible features. Also, there are somewhat a fixed set of features of a specific type of product, for example, ease of use, durability, battery life, photo quality, and shutter lag for digital cameras. Consequently, we can expect that reasons in electronics' reviews may share those product feature words and words that describe aspects of features such as *short* or *long* for *battery life*. This fact might make the reason identification task easy.

On the other hand, restaurant reviewers talk about very diverse aspects and abstract features as reasons. For example, reasons such as "You feel like you are in a train station or a busy amusement park that is ill-staffed to meet demand!", "preferential treatment given to large groups", and "they don't offer salads of any kind" are hard to predict. Also, they seem rarely share common keyword features.

We first automatically labeled each sentence in those reviews collected from each domain with the features described in Section 3.1. We divided the data for training and testing. We then trained our model using the training set and tested it to see if the system can successfully label sentences in the test set.

4.2 Dataset 2: Complaints.com Data

From the database [4] in complaints.com, we searched for the same topics of reviews as Dataset 1: 59 complaints reviews about mp3 players and 322 reviews about restaurants[5]. We tested our system on this dataset and compare the results against human judges' annotation results. Subsection 5.2 reports the evaluation results.

5 Experiments and Results

We describe two goals in our experiments in this section. The first is to investigate how well our pro and con detection model with different feature combinations performs on the data we collected from epinions.com. The second is to see how well the trained model performs on new data from a different source, complaint.com.

For both datasets, we carried out two separate sets of experiments, for the domains of mp3 players and restaurant reviews. We divided data into 80% for training, 10% for development, and 10% for test for our experiments.

5.1 Experiments on Dataset 1

Identification step: Table 3 and 4 show pros and cons sentences identification results of our system for mp3 player and restaurant reviews respectively. The first column indicates which combination of features was used for our model (see Table 2 for the meaning of Op, Lex, and Pos feature categories). We measure the performance with accuracy (*Acc*), precision (*Prec*), recall (*Recl*), and F-score [6].

The baseline system assigned all sentences as reason and achieved 57.75% and 54.82% of accuracy. The system performed well when it only used lexical features in mp3 player reviews (76.27% of accuracy in Lex), whereas it performed well with the combination of lexical and opinion features in restaurant reviews (Lex+Op row in Table 4).

It was very interesting to see that the system achieved a very low score when it only used opinion word features. We can interpret this phenomenon as supporting our hypothesis that pro and con sentences in reviews are often purely

[4] At the time (December 2005), there were total 42593 complaint reviews available in the database.

[5] Average numbers of sentences in a complaint is 19.57 for mp3 player reviews and 21.38 for restaurant reviews.

[6] We calculated F-score by $\frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

Table 3: Pros and cons sentences identification results on mp3 player reviews.

Features used	**Acc** (%)	**Prec** (%)	**Recl** (%)	**F-score** (%)
Op	60.15	65.84	57.31	61.28
Lex	**76.27**	66.18	**76.42**	**70.93**
Lex+Pos	63.10	**71.14**	60.72	65.52
Lex+Op	62.75	70.64	60.07	64.93
Lex+Pos+Op	62.23	70.58	59.35	64.48
Baseline	**57.75**			

Table 4: Reason sentence identification results on restaurant reviews.

Features used	**Acc** (%)	**Prec** (%)	**Recl** (%)	**F-score** (%)
Op	61.64	60.76	47.48	53.31
Lex	63.77	67.10	51.20	58.08
Lex+Pos	**63.89**	67.62	51.70	58.60
Lex+Op	61.66	**69.13**	**54.30**	**60.83**
Lex+Pos+Op	63.13	66.80	50.41	57.46
Baseline	**54.82**			

Table 5: Pros and cons sentences classification results for mp3 player reviews.

Features used	**Acc** (%)	Cons			Pros		
		Prec (%)	**Recl** (%)	**F-score** (%)	**Prec** (%)	**Recl** (%)	**F-score** (%)
Op	**57.18**	54.43	67.10	60.10	**61.18**	48.00	**53.80**
Lex	55.88	55.49	67.45	60.89	56.52	43.88	49.40
Lex+Pos	55.62	55.26	**68.12**	**61.02**	56.24	42.62	48.49
Lex+Op	55.60	55.46	64.63	59.70	55.81	46.26	50.59
Lex+Pos+Op	56.68	**56.70**	62.45	59.44	56.65	**50.71**	53.52
baseline	**53.87**	(mark all as pros)					

Table 6: Pros and cons sentences classification results for restaurant reviews.

Features used	**Acc** (%)	Cons			Pros		
		Prec (%)	**Recl** (%)	**F-score** (%)	**Prec** (%)	**Recl** (%)	**F-score** (%)
Op	**57.32**	54.78	51.62	53.15	**59.32**	**62.35**	**60.80**
Lex	55.76	55.94	52.52	54.18	55.60	58.97	57.24
Lex+Pos	56.07	**56.20**	**53.33**	**54.73**	55.94	58.78	57.33
Lex+Op	55.88	56.10	52.39	54.18	55.68	59.34	57.45
Lex+Pos+Op	55.79	55.89	53.17	54.50	55.70	58.38	57.01
baseline	**50.71**	**(**mark all as pros)					

factual. However, opinion features improved both precision and recall when combined with lexical features in restaurant reviews. It was also interesting that experiments on mp3 players reviews achieved mostly higher scores than restaurants. Like the observation we described in Subsection 4.1, frequently mentioned keywords of product features (e.g. durability) may have helped performance, especially with lexical features. Another interesting observation is that the positional features that helped in topic sentence identification did not help much for our task.

Classification step: Tables 5 and 6 show the system results of the pro and con classification task. The baseline system marked all sentences as pros and achieved 53.87% and 50.71% accuracy for each domain. All features performed better than the baseline but the results are not as good as in the identification task. Unlike the identification task, opinion words by themselves achieved the best accuracy in both mp3 player and restaurant domains. We think opinion words played more important roles in classifying pros and cons than identifying them. Position features helped recognizing con sentences in mp3 player reviews.

5.2 Experiments on Dataset 2

This subsection reports the evaluation results of our system on Dataset 2. Since Dataset 2 from complaints.com has no training data, we trained a system on Dataset 1 and applied it to Dataset 2.

A tough question, however, is how to evaluate the system results. Since it seemed impossible to evaluate the system without involving a human judge, we annotated a small set of data manually for evaluation purposes.

Gold Standard Annotation: Four humans annotated 3 sets of test sets: Testset 1 with 5 complaints (73 sentences), Testset 2 with 7 complaints (105 sentences), and Testset 3 with 6 complaints (85 sentences). Testset 1 and 2 are from mp3 player complaints and Testset 3 is from restaurant reviews. Annotators marked sentences if they describe specific reasons of the complaint. Each test set was annotated by 2 humans. The average pair-wise human agreement was 82.1%[7].

System Performance: Like the human annotators, our system also labeled reason sentences. Since our goal is to identify reason sentences in complaints, we applied a system modeled as in the identification phase described in Subsection 3.2 instead of the classification phase[8]. Table 7 reports the accuracy, precision, and recall of the system on each test set. We calculated numbers in each A and B column by assuming each annotator's answers separately as a gold standard.

Table 7: System results on Complaint.com reviews (A, B: The first and the second annotator of each set)

	Testset 1		**Testset 2**		**Testset 3**		Avg
	A	B	A	B	A	B	
Acc(%)	65.8	63.0	67.6	61.0	**77.6**	**72.9**	68.0
Prec(%)	50.0	60.7	68.6	62.9	67.9	60.7	61.8
Recl(%)	56.0	51.5	51.1	44.0	**65.5**	**58.6**	54.5

In Table 7, accuracies indicate the agreement between the system and human annotators. The average accuracy 68.0% is comparable with the pair-wise human agreement 82.1% even if there is still a lot of room for improvement[9]. It was interesting to see that Testset 3, which was from restaurant complaints, achieved higher accuracy and recall than the other test sets from mp3 player complaints, suggesting that it would be interesting to further investigate the performance of reason identification in various other review domains such as travel and beauty products in future work. Also, even though we were somewhat able to measure reason sentence identification in complaint reviews, we agree that we need more data annotation for more precise evaluation.

Finally, the followings are examples of sentences that our system identified as reasons of complaints.

```
(1) Unfortunately, I find that
I am no longer comfortable in
your establishment because of
the unprofessional, rude, ob-
noxious, and unsanitary treat-
ment from the employees.
(2) They never get my order
right the first time and what
really disgusts me is how they
handle the food.
(3) The kids play area at
Braum's in The Colony, Texas is
very dirty.
(4) The only complaint that I
have is that the French fries
are usually cold.
(5) The cashier there had short
changed me on the payment of my
bill.
```

As we can see from the examples, our system was able to detect con sentences which contained opinion-bearing expressions such as in (1), (2), and (3) as well as reason sentences that mostly described mere facts as in (4) and (5).

6 Conclusions and Future work

This paper proposes a framework for identifying one of the critical elements of online product reviews to answer the question, "What are reasons that the author of a review likes or dislikes the product?" We believe that pro and con sentences in reviews can be answers for this question. We present a novel technique that automatically labels a large set of pro and con sentences in online reviews using clue phrases for pros and cons in epinions.com in order to train our system. We applied it to label sentences both on epinions.com and complaints.com. To investigate the reliability of our system, we tested it on two extremely different review domains, mp3 player reviews and restaurant reviews. Our system with the best feature selection performs 71% F-score in the reason identification task and 61% F-score in the reason classification task.

[7] The kappa value was 0.63.

[8] In complaints reviews, we believe that it is more important to identify reason sentences than to classify because most reasons in complaints are likely to be cons.

[9] The baseline system which assigned the majority class to each sentence achieved 59.9% of average accuracy.

The experimental results further show that pro and con sentences are a mixture of opinions and facts, making identifying them in online reviews a distinct problem from opinion sentence identification. Finally, we also apply the resulting system to another review data in complaints.com in order to analyze reasons of consumers' complaints.

In the future, we plan to extend our pro and con identification system on other sorts of opinion texts, such as debates about political and social agenda that we can find on blogs or news group discussions, to analyze why people support a specific agenda and why people are against it.

Reference

Berger, Adam L., Stephen Della Pietra, and Vincent Della Pietra. 1996. A maximum entropy approach to natural language processing, *Computational Linguistics*, (22-1).

Bethard, Steven, Hong Yu, Ashley Thornton, Vasileios Hatzivassiloglou, and Dan Jurafsky. 2004. Automatic Extraction of Opinion Propositions and their Holders, *AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications*.

Chklovski, Timothy. 2006. Deriving Quantitative Overviews of Free Text Assessments on the Web. *Proceedings of 2006 International Conference on Intelligent User Interfaces (IUI06).* Sydney, Australia.

Choi, Y., Cardie, C., Riloff, E., and Patwardhan, S. 2005. Identifying Sources of Opinions with Conditional Random Fields and Extraction Patterns. *Proceedings of HLT/EMNLP-05.*

Esuli, Andrea and Fabrizio Sebastiani. 2005. Determining the semantic orientation of terms through gloss classification. *Proceedings of CIKM-05, 14th ACM International Conference on Information and Knowledge Management*, Bremen, DE, pp. 617-624.

Hatzivassiloglou, Vasileios and Kathleen McKeown. 1997. Predicting the Semantic Orientation of Adjectives. *Proceedings of 35th Annual Meeting of the Assoc. for Computational Linguistics* (ACL-97): 174-181

Hatzivassiloglou, Vasileios and Janyce Wiebe. 2000. Effects of Adjective Orientation and Gradability on Sentence Subjectivity. *Proceedings of International Conference on Computational Linguistics (COLING-2000).* Saarbrücken, Germany.

Hu, Minqing and Bing Liu. 2004. Mining and summarizing customer reviews". *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD-2004)*, Seattle, Washington, USA.

Kim, Soo-Min and Eduard Hovy. 2004. Determining the Sentiment of Opinions. *Proceedings of COLING-04*. pp. 1367-1373. Geneva, Switzerland.

Kim, Soo-Min and Eduard Hovy. 2005. Automatic Detection of Opinion Bearing Words and Sentences. *In the Companion Volume of the Proceedings of IJCNLP-05, Jeju Island, Republic of Korea.*

Kim, Soo-Min and Eduard Hovy. 2006. Identifying and Analyzing Judgment Opinions. *Proceedings of HLT/NAACL-2006, New York City, NY.*

Lin, Chin-Yew and Eduard Hovy. 1997. Identifying Topics by Position. *Proceedings of the 5th Conference on Applied Natural Language Processing (ANLP97).* Washington, D.C.

Pang, Bo, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment Classification using Machine Learning Techniques, *Proceedings of EMNLP 2002*.

Popescu, Ana-Maria, and Oren Etzioni. 2005. Extracting Product Features and Opinions from Reviews , *Proceedings of HLT-EMNLP 2005*.

Riloff, Ellen, Janyce Wiebe, and Theresa Wilson. 2003. Learning Subjective Nouns Using Extraction Pattern Bootstrapping. *Proceedings of Seventh Conference on Natural Language Learning (CoNLL-03). ACL SIGNLL*. Pages 25-32.

Turney, Peter D. 2002. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews, *Proceedings of ACL-02*, Philadelphia, Pennsylvania, 417-424

Wiebe, Janyce M., Bruce, Rebecca F., and O'Hara, Thomas P. 1999. Development and use of a gold standard data set for subjectivity classifications. *Proceedings of ACL-99.* University of Maryland, June, pp. 246-253.

Wilson, Theresa, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis. *Proceedings of HLT/EMNLP 2005*, Vancouver, Canada

Wilson, Theresa, Janyce Wiebe, and Rebecca Hwa. 2004. Just how mad are you? Finding strong and weak opinion clauses. *Proceedings of 19th National Conference on Artificial Intelligence* (AAAI-2004).

Interpreting Semantic Relations in Noun Compounds via Verb Semantics

Su Nam Kim[†] **and Timothy Baldwin**[†‡]
† Computer Science and Software Engineering
University of Melbourne, Victoria 3010 Australia
and
‡ NICTA Victoria Research Lab
University of Melbourne, Victoria 3010 Australia
{snkim,tim}@csse.unimelb.edu.au

Abstract

We propose a novel method for automatically interpreting compound nouns based on a predefined set of semantic relations. First we map verb tokens in sentential contexts to a fixed set of seed verbs using `WordNet::Similarity` and Moby's Thesaurus. We then match the sentences with semantic relations based on the semantics of the seed verbs and grammatical roles of the head noun and modifier. Based on the semantics of the matched sentences, we then build a classifier using `TiMBL`. The performance of our final system at interpreting NCs is 52.6%.

1 Introduction

The interpretation of **noun compounds** (hereafter, NCs) such as *apple pie* or *family car* is a well-established sub-task of language understanding. Conventionally, the NC interpretation task is defined in terms of unearthing the underspecified semantic relation between the head noun and modifier(s), e.g. *pie* and *apple* respectively in the case of *apple pie*.

NC interpretation has been studied in the context of applications including question-answering and machine translation (Moldovan et al., 2004; Cao and Li, 2002; Baldwin and Tanaka, 2004; Lauer, 1995). Recent work on the automatic/semi-automatic interpretation of NCs (e.g., Lapata (2002), Rosario and Marti (2001), Moldovan et al. (2004) and Kim and Baldwin (2005)) has made assumptions about the scope of semantic relations or restricted the domain of interpretation. This makes it difficult to gauge the general-purpose utility of the different methods. Our method avoids any such assumptions while outperforming previous methods.

In seminal work on NC interpretation, Finin (1980) tried to interpret NCs based on hand-coded rules. Vanderwende (1994) attempted the automatic interpretation of NCs using hand-written rules, with the obvious cost of manual intervention. Fan et al. (2003) estimated the knowledge required to interpret NCs and claimed that performance was closely tied to the volume of data acquired.

In more recent work, Barker and Szpakowicz (1998) used a semi-automatic method for NC interpretation in a fixed domain. Lapata (2002) developed a fully automatic method but focused on nominalizations, a proper subclass of NCs.[1] Rosario and Marti (2001) classified the nouns in medical texts by tagging hierarchical information using neural networks. Moldovan et al. (2004) used the word senses of nouns based on the domain or range of interpretation of an NC, leading to questions of scalability and portability to novel domains/NC types. Kim and Baldwin (2005) proposed a simplistic general-purpose method based on the lexical similarity of unseen NCs with training instances.

The aim of this paper is to develop an automatic method for interpreting NCs based on semantic relations. We interpret semantic relations relative to a fixed set of constructions involving the modifier and head noun and a set of **seed verbs** for each semantic relation: e.g. *(the) family owns (a) car* is taken as evidence for *family car* being an instance of the POSSESSOR relation. We then attempt to map all instances of the modifier and head noun as the heads of NPs in a transitive sentential context onto our set of constructions via lexical similarity over the verb, to arrive at an interpretation: e.g. we would hope to predict that *possess* is sufficiently similar to *own* that *(the) family possesses (a) car* would be recognised as support-

[1] With nominalizations, the head noun is deverbal, and in the case of Lapata (2002), nominalisations are assumed to be interpretable as the modifier being either the subject (e.g. *child behavior*) or object (e.g. *car lover*) of the base verb of the head noun.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 491–498,
Sydney, July 2006.

ing evidence for the POSSESSOR relation. We use a supervised classifier to combine together the evidence contributed by individual sentential contexts of a given modifier–head noun combination, and arrive at a final interpretation for a given NC.

Mapping the actual verbs in sentences to appropriate seed verbs is obviously crucial to the success of our method. This is particularly important as there is no guarantee that we will find large numbers of modifier–head noun pairings in the sorts of sentential contexts required by our method, nor that we will find attested instances based on the seed verbs. Thus an error in mapping an attested verb to the seed verbs could result in a wrong interpretation or no classification at all. In this paper, we experiment with the use of WordNet (Fellbaum, 1998) and word clusters (based on Moby's Thesaurus) in mapping attested verbs to the seed verbs. We also make use of CoreLex in dealing with the semantic relation TIME and the RASP parser (Briscoe and Carroll, 2002) to determine the dependency structure of corpus data.

The data source for our set of NCs is binary NCs (i.e. NCs with a single modifier) from the Wall Street Journal component of the Penn Treebank. We deliberately choose to ignore NCs with multiple modifiers on the grounds that: (a) 88.4% of NC types in the Wall Street Journal component of the Penn Treebank and 90.6% of NC types in the British National Corpus are binary; and (b) we expect to be able to interpret NCs with multiple modifiers by decomposing them into binary NCs. Another simplifying assumption we make is to remove NCs incorporating proper nouns since: (a) the lexical resources we employ in this research do not contain them in large numbers; and (b) there is some doubt as to whether the set of semantic relations required to interpret NCs incorporating proper nouns is that same as that for common nouns.

The paper is structured as follows. Section 2 takes a brief look at the semantics of NCs and the basic idea behind the work. Section 3 details the set of NC semantic relations that is used in our research, Section 4 presents an extended discussion of our approach, Section 5 briefly explains the tools we use, Section 6.1 describes how we gather and process the data, Section 6.2 explains how we map the verbs to seed verbs, and Section 7 and Section 8 present the results and analysis of our approach. Finally we conclude our work in Section 9.

2 Motivation

The semantic relation in NCs is the relation between the head noun (denoted "H") and the modifier(s) (denoted "M"). We can find this semantic relation expressed in certain sentential constructions involving the head noun and modifier.

(1) *family car*
CASE: *family* ***owns*** *the car.*
FORM: H [own] M
RELATION: POSSESSOR

(2) *student protest*
CASE: *protest is* ***performed*** *by student.*
FORM: M is [performed] by H
RELATION: AGENT

In the examples above, the semantic relation (e.g. POSSESSOR) provides an interpretation of how the head noun and modifiers relate to each other, and the seed verb (e.g. *own*) provides a paraphrase evidencing that relation. For example, in the case of *family car*, the *family* is the POSSESSOR of the *car*, and in *student protest*, *student(s)* are the AGENT of the *protest*. Note that voice is important in aligning sentential contexts with semantic relations. For instance, *family car* can be represented as *car is owned by family* (passive) and *student protest* as *student performs protest* (active).

The exact nature of the sentential context varies with different synonyms of the seed verbs.

(3) *family car*
CASE: *Synonym=have/possess/belong to*
FORM: H [own] M
RELATION: POSSESSOR

(4) *student protest*
CASE: *Synonym=act/execute/do*
FORM: M is [performed] by H
RELATION: AGENT

The verb *own* in the POSSESSOR relation has the synonyms *have, possess* and *belong to*. In the context of *have* and *possess*, the form of relation would be same as the form with verb, *own*. However, in the context of *belong to*, *family car*

would mean that the *car belongs to family*. Thus, even when the voice of the verb is the same (*voice=active*), the grammatical role of the head noun and modifier can change.

In our approach we map the actual verbs in sentences containing the head noun and modifiers to seed verbs corresponding to the relation forms. The set of seed verbs contains verbs representative of each semantic relation form. We chose two sets of seed verbs of size 57 and 84, to examine how the coverage of actual verbs by seed verbs affects the performance of our method. Initially, we manually chose a set of 60 seed verbs. We then added synonyms from Moby's thesaurus for some of the 60 verbs. Finally, we filtered verbs from the two expanded sets, since these verbs occur very frequently in the corpus (as this might skew the results). The set of seed verbs {*have, own, possess, belong_to*} are in the set of 57 seed verbs, and {*acquire, grab, occupy*} are added to the set of 84 seed verbs; all correspond to the POSSESSOR relation.

For each relation, we generate a set of constructional templates associating a subset of seed verbs with appropriate grammatical relations for the head noun and modifier. Examples for POSSESSOR are:

$$S(\{have, own, possess\}^{\text{V}}, M^{\text{SUBJ}}, H^{\text{OBJ}}) \quad (5)$$

$$S(\{belong\_to\}^{\text{V}}, H^{\text{SUBJ}}, M^{\text{OBJ}}) \quad (6)$$

where V is the set of seed verbs, M is the modifier and H is the head noun.

Two relations which do not map readily onto seed verbs are TIME (e.g. *winter semester*) and EQUATIVE (e.g. *composer arranger*). Here, we rely on an independent set of contextual evidence, as outlined in Section 6.1.

Through matching actual verbs attested in corpus data onto seed verbs, we can match sentences with relations (see Section 6.2). Using this method we can identify the matching relation forms of semantic relations to decide the semantic relation for NCs.

3 Semantic Relations in Compound Nouns

While there has been wide recognition of the need for a system of semantic relations with which to classify NCs, there is still active debate as to what the composition of that set should be, or indeed whether it is reasonable to expect that all NCs should be interpretable with a fixed set of semantic relations.

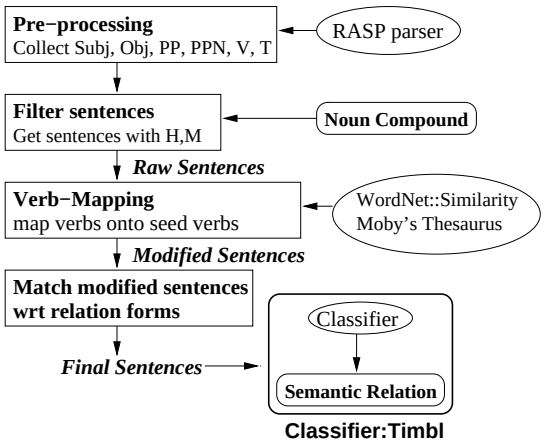


Figure 1: System Architecture

Based on the pioneering work on Levi (1979) and Finin (1980), there have been efforts in computational linguistics to arrive at largely task-specific sets of semantic relations, driven by the annotation of a representative sample of NCs from a given corpus type (Vanderwende, 1994; Barker and Szpakowicz, 1998; Rosario and Marti, 2001; Moldovan et al., 2004). In this paper, we use the set of 20 semantic relations defined by Barker and Szpakowicz (1998), rather than defining a new set of relations. The main reasons we chose this set are: (a) that it clearly distinguishes between the head noun and modifiers, and (b) there is clear documentation of each relation, which is vital for NC annotation effort. The one change we make to the original set of 20 semantic relations is to exclude the PROPERTY relation since it is too general and a more general form of several other relations including MATERIAL (e.g. *apple pie*).

4 Method

Figure 1 outlines the system architecture of our approach. We used three corpora: the Brown corpus (as contained in the Penn Treebank), the Wall Street Journal corpus (also taken from the Penn treebank), and the written component of the British National Corpus (BNC). We first parsed each of these corpora using RASP (Briscoe and Carroll, 2002), and identified for each verb token the voice, head nouns of the subject and object, and also, for each PP attached to that verb, the head preposition and head noun of the

NP (hereafter, **PPN**). Next, for our test NCs, we identified all verbs for which the modifier and head noun co-occur as subject, object, or PPN. We then mapped these verbs to seed verbs using `WordNet::Similarity` and Moby's Thesaurus (see Section 5 for details). Finally, we identified the corresponding relation for each seed verb and selected the best-fitting semantic relation using a classifier. To evaluate our approach, we built a classifier using `TiMBL` (Daelemans et al., 2004).

5 Resources

In this section, we outline the tools and resources employed in our method.

As our parser, we used RASP, generating a dependency representation for the most probable parse for each sentence. Note that RASP also lemmatises all words in a POS-sensitive manner.

To map actual verbs onto seed verbs, we experimented with two resources: `WordNet::Similarity` and Moby's thesaurus. `WordNet::Similarity`[2] is an open source software package that allows the user to measure the semantic similarity or relatedness between two words (Patwardhan et al., 2003). Of the many methods implemented in `WordNet::Similarity`, we report on results for one path-based method (WUP, Wu and Palmer (1994)), one content-information based method (JCN, Jiang and Conrath (1998)) and two semantic relatedness methods (LESK, Banerjee and Pedersen (2003), and VECTOR, (Patwardhan, 2003)). We also used a random similarity-generating method as a baseline (RANDOM).

The second semantic resource we use for verb-mapping method is Moby's thesaurus. Moby's thesaurus is based on Roget's thesaurus, and contains 30K root words, and 2.5M synonyms and related words. Since the direct synonyms of seed verbs have limited coverage over the set of sentences used in our experiment, we also experimented with using second-level indirect synonyms of seed verbs.

In order to deal with the TIME relation, we used CoreLex (Buitelaar, 1998). CoreLex is based on a unified approach to systematic polysemy and the semantic underspecification of nouns, and derives from WordNet 1.5. It contains 45 basic CoreLex types, systematic polysemous classes and 39,937 nouns with tags.

[2] www.d.umn.edu/ tpederse/similarity.html

As mentioned earlier, we built our supervised classifier using `TiMBL`.

6 Data Collection

6.1 Data Processing

To test our method, we extracted 2,166 NC types from the Wall Street Journal (WSJ) component of the Penn Treebank. We additionally extracted sentences containing the head noun and modifier in pre-defined constructional contexts from the amalgam of: (1) the Brown Corpus subset contained in the Penn Treebank, (2) the WSJ portion of the Penn Treebank, and (3) the British National Corpus (BNC). Note that while these pre-defined constructional contexts are based on the contexts in which our seed verbs are predicted to correlate with a given semantic relation, we instances of all verbs occurring in those contexts. For example, based on the construction in Equation 5, we extract all instances of $S(V_i, M_j^{\text{SUBJ}}, H_j^{\text{OBJ}})$ for all verbs V_i and all instances of $NC_j = (M_j, H_j)$ in our dataset.

Two annotators tagged the 2,166 NC types independently at 52.3% inter-annotator agreement, and then met to discus all contentious annotations and arrive at a mutually-acceptable gold-standard annotation for each NC. The Brown, WSJ and BNC data was pre-parsed with RASP, and sentences were extracted which contained the head noun and modifier of one of our 2,166 NCs in subject or object position, or as (head) noun within the NP of an PP. After extracting these sentences, we counted the frequencies of the different modifier–head noun pairs, and filtered out: (a) all constructional contexts not involving a verb contained in WordNet 2.0, and (b) all NCs for which the modifier and head noun did not co-occur in at least five sentential contexts. This left us with a total of 453 NCs for training and testing. The combined total number of sentential contexts for our 453 NCs was 7,714, containing 1,165 distinct main verbs.

We next randomly split the NC data into 80% training data and 20% test data. The final number of test NCs is 88; the final number of training NCs varies depending on the verb-mapping method.

As noted in Section 2, the relations TIME and EQUATIVE are not associated with seed verbs. For TIME, rather than using contextual evidence, we simply flag the possibility of a TIME if the modifier is found to occur in the *TIME* class of CoreLex. In the case of TIME, we consider coordinated occur-

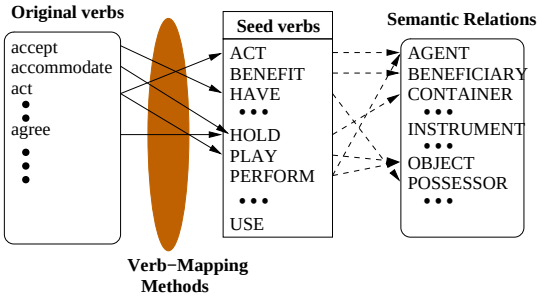


Figure 2: Verb mapping

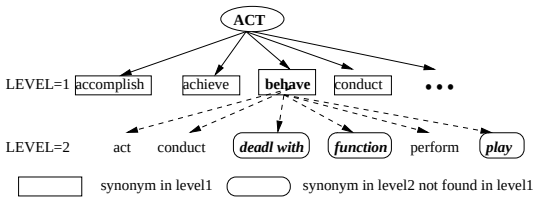


Figure 3: Expanding synonyms

# of SeedVB	D-Synonyms	D/I-Synonyms
57	6,755(87.6%)	7,388(95.8%)
84	6,987(90.6%)	7,389(95.8%)

Table 1: Coverage of D and D/I-Synonyms

rences of the modifier and head noun (e.g. *coach and player* for *player coach*) as evidence for the relation.[3] We thus separately collate statistics from coordinated NPs for each NC, and from this compute a weight for each NC based on mutual information:

$$TIME(NC_i) = -log_2 \frac{freq(coord(M_i, H_i))}{freqM_i \times freq(H_i)} \quad (7)$$

where M_i and H_i are the modifier and head of NC_i, respectively, and $freq(coord(M_i, H_i))$ is the frequency of occurrence of M_i and H_i in coordinated NPs.

Finally, we calculate a normalised weight for each seed verb by determining the proportion of head verbs each seed verb occurs with.

6.2 Verb Mapping

The sentential contexts gathered from corpus data contain a wide range of verbs, not just the seed verbs. To map the verbs onto seed verbs, and hence estimate which semantic relation(s) each is a predictor of, we experimented with two different methods. First we used the `WordNet::Similarity` package to calculate the similarity between a given verb and each of the seed verbs, experimenting with the 5 methods mentioned in Section 5. Second, we used Moby's thesaurus to extract both direct synonyms (D-SYNONYM) and a combination of direct and second-level indirect synonyms of verbs (I-SYNONYM), and from this, calculate the closest-matching seed verb(s) for a given verb.

Figure 2 depicts the procedure for mapping verbs in constructional contexts onto the seed verbs. Verbs found in the various contexts in the corpus (on the left side of the figure) map onto one or more seed verbs, which in turn map onto one or more semantic relations.[4] We replace all non-seed verbs in the corpus data with the seed verb(s) they map onto, potentially increasing the number of corpus instances.

Since direct (i.e. level 1) synonyms from Moby's thesaurus are not sufficient to map all verbs onto seed verbs, we also include second-level (i.e. level 2) synonyms, expanding from direct synonyms. Table 1 shows the coverage of sentences for test NCs, in which **D** indicates direct synonyms and **I** indicates indirect synonyms.

7 Evaluation

We evaluated our method over both 17 semantic relations (without EQUATIVE and TIME) and the full 19 semantic relations, due to the low frequency and lack of verb-based constructional contexts for EQUATIVE and TIME, as indicated in Table 2. Note that the test data set is the same for both sets of semantic relations, but that the training data in the case of 17 semantic relations will not contain any instances for the EQUATIVE and TIME relations, meaning that all such test instances will be misclassified. The baseline for all verb mapping methods is a simple majority-class classifier, which leads to an accuracy of 42.4% for the TOPIC relation. In evaluation, we use two different values for our method: *Count* and *Weight*. *Count* is based on the raw number of corpus instances, while *Weight* employs the seed verb weight described in Section 6.1.

[3] Note the order of the modifier and head in coordinated NPs is considered to be irrelevant, i.e. *player and coach* and *coach and player* are equally evidence for an EQUATIVE interpretation for *player coach* (and *coach player*).

[4] There is only one instance of a seed verb mapping to multiple semantic relations, namely *perform* which corresponds to the two relations AGENT and OBJECT.

# of SR	# SeedV	Method	WUP	JCN	RANDOM	LESK	VECTOR	D-SYNONYM	I-SYNONYM
17		Baseline	.423	.423	.423	.423	.423	.423	.423
	57	Count	.324	.408	.379	.416	**.466**	.337	.337
		Weight	.320	.408	.371	.416	**.466**	.337	.342
	84	Count	.406	**.470**	.184	.430	.413	.317	.333
		Weight	.424	.426	.259	.457	**.526**	.341	.406
19		Baseline	.409	.409	.409	.409	.409	.409	.409
	57	Count	.315	.420	.384	.440	**.466**	.350	.337
		Weight	.311	.420	.376	.440	**.466**	.350	.342
	84	Count	.413	**.470**	.200	.414	.413	.321	.333
		Weight	.439	.446	.280	.486	**.526**	.356	.393

Table 2: Results with 17 relations and with 19 relations

#of SR	# SeedVB	Method	WUP	JCN	RANDOM	LESK	VECTOR
17		Baseline	.423	.423	.423	.423	.423
	57	Count	.423	.385	.379	.413	**.466**
		Weight	.423	.385	.379	.413	**.466**
	84	Count	.325	.439	.420	**.484**	.466
		Weight	.281	.393	.317	**.476**	.466
19		Baseline	.409	.409	.409	.409	.409
	57	Count	**.423**	.397	.392	.413	.413
		Weight	.423	.397	.392	.413	**.500**
	84	Count	.333	.439	.425	**.484**	.413
		Weight	.290	.410	.317	**.484**	.413

Table 3: Results of combining the proposed method and with the method of Kim and Baldwin (2005)

As noted above, we excluded all NCs for which we were unable to find at least 5 instances of the modifier and head noun in an appropriate sentential context. This exclusion reduced the original set of 2,166 NCs to only 453, meaning that the proposed method is unable to classify up to 80% of NCs. For real-world applications, a method which is only able to arrive at a classification for 20% of instances is clearly of limited utility, and we need some way of expanding the coverage of the proposed method. This is achieved by adapting the similarity method proposed by Kim and Baldwin (2005) to our task, wherein we use lexical similarity to identify the nearest-neighbour NC for a given NC, and classify the given NC according to the classification for the nearest-neighbour. The results for the combined method are presented in Table 3.

8 Discussion

For the basic method, as presented in Table 2, the classifier produced similar results over the 17 semantic relations to the 19 semantic relations. Using data from *Weight* and *Count* for both 17 and 19 semantic relations, the classifier achieved the best performance with VECTOR (context vector-based distributional similarity), followed by JCN and LESK. The main reason is that VECTOR is more conservative than the other methods at mapping (original) verbs onto seed verbs, i.e. the average number of seed verbs a given verb maps onto is small. For the other methods, the semantics of the original sentences are often not preserved under verb mapping, introducing noise to the classification task.

Comparing the two sets of semantic relations (17 vs. 19), the set with more semantic relations achieved slightly better performance in most cases. A detailed breakdown of the results revealed that TIME both has an above-average classification accuracy and is associated with a relatively large number of test NCs, while EQUATIVE has a below-average classification accuracy but is associated with relatively few instances.

While an increased number of seed verbs generates more training instances under verb mapping, it is imperative that the choice of seed verbs be made carefully so that they not introduce noise into the classifier and reducing overall performance. Figure 4 is an alternate representation of the numbers from Table 2, with results for each individual method over 57 and 84 seed verbs juxtaposed for each of *Count* and *Weight*. From this, we get the intriguing result that *Count* generally performs better over fewer seed verbs, while *Weight* performs better over more seed verbs.

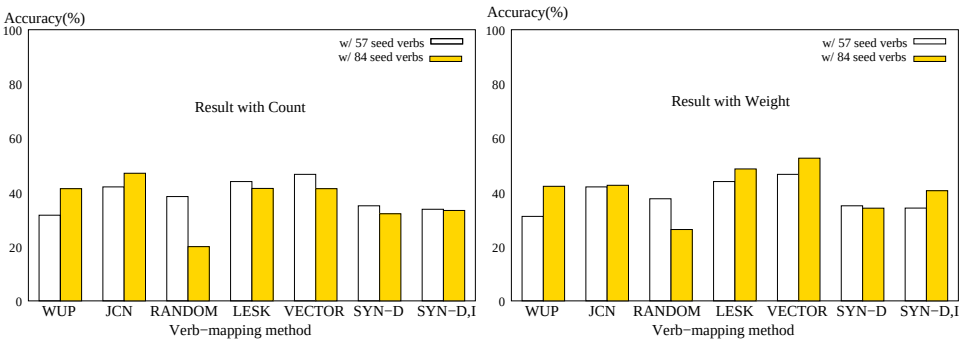


Figure 4: Performance with 57 vs. 84 seed verbs

#of SR	# SeedVB	WUP	LCH	JCN	LIN	RANDOM	LESK	VECTOR
17	Baseline	.433	.433	.441	.441	.433	.477	.428
	57	.449	.421	.415	.337	.409	.469	.344
	Baseline	.433	.433	.433	.433	.428	.438	.444
	84	**.476**	.416	.409	.349	.226	.465	.333
19	Baseline	.418	.418	.430	.430	.418	.477	.413
	57	.465	.418	.417	.341	.232	.462	.344
	Baseline	.413	.413	.418	.418	.413	.438	.426
	84	**.471**	.413	.407	.348	.218	.465	.320

Table 4: Results for the method of Kim and Baldwin (2005) over the test set used in this research

For the experiment where we combine our method with that of Kim and Baldwin (2005), as presented in Table 3, we find a similar pattern of results to the proposed method. Namely, VECTOR and LESK achieve the best performance, with minor variations in the absolute performance relative to the original method but the best results for each relation set actually dropping marginally over the original method. This drop is not surprising when we consider that we use an imperfect method to identify the nearest neighbour for an NC for which we are unable to find corpus instances in sufficient numbers, and then a second imperfect method to classify the instance.

Compared to previous work, our method produces reasonably stable performance when operated over the open-domain data with small amounts of training data. Rosario and Marti (2001) achieved about 60% using a neural network in a closed domain, Moldovan et al. (2004) achieved 43% using word sense disambiguation of the head noun and modifier over open domain data, and Kim and Baldwin (2005) produced 53% using lexical similarities of the head noun and modifier (using the same relation set, but evaluated over a different dataset). The best result achieved by our system was 52.6% over open-domain data, using a general-purpose relation set.

To get a better understanding of how our method compares with that of Kim and Baldwin (2005), we evaluated the method of Kim and Baldwin (2005) over the same data set as used in this research, the results of which are presented in Table 4. The relative results for the different similarity metrics mirror those reported in Kim and Baldwin (2005). WUP produced the best performance at 47-48% for the two relation sets, significantly below the accuracy of our method at 53.3%. Perhaps more encouraging is the result that the combined method—where we classify attested instances according to the proposed method, and classify unattested instances according to the nearest-neighbour method of Kim and Baldwin (2005) and the classifications from the proposed method—outperforms the method of Kim and Baldwin (2005). That is, the combined method has the coverage of the method of Kim and Baldwin (2005), but inherits the higher accuracy of the method proposed herein. Having said this, the performance of the Kim and Baldwin (2005) method over PRODUCT, TOPIC, LOCATION and SOURCE is superior to that of our method. In this sense, we believe that alternate methods of hybridisation may lead to even better results.

Finally, we wish to point out that the method as presented here is still relatively immature, with considerable scope for improvement. In its current form, we do not weight the different seed verbs

based on their relative similarity to the original verb. We also use the same weight and frequency for each seed verb relative to a given relation, despite seed verbs being more indicative of a given relation and also potentially occurring more often in the corpus. For instance, *possess* is more related to POSSESSOR than *occupy*. Also *possess* occurs more often in the corpus than *belong_to*. As future work, we intend to investigate whether allowances for these considerations can improve the performance of our method.

9 Conclusion

In this paper, we proposed a method for automatically interpreting noun compounds based on seed verbs indicative of each semantic relation. For a given modifier and head noun, our method extracted corpus instances of the two nouns in a range of constructional contexts, and then mapped the original verbs onto seed verbs based on lexical similarity derived from `WordNet::Similarity`, and Moby's Thesaurus. These instances were then fed into the `TiMBL` learner to build a classifier. The best-performing method was VECTOR, which is a context vector distributional similarity method. We also experimented with varying numbers of seed verbs, and found that generally the more seed verbs, the better the performance. Overall, the best-performing system achieved an accuracy of 52.6% with 84 seed verbs and the VECTOR verb-mapping method.

Acknowledgements

We would like to thank the members of the University of Melbourne LT group and the three anonymous reviewers for their valuable input on this research.

References

Timothy Baldwin and Takaaki Tanaka. 2004. Translation by machine of compound nominals: Getting it right. In *In Proceedings of the ACL 2004 Workshop on Multiword Expressions: Integrating Processing*, pages 24–31, Barcelona, Spain.

Satanjeev Banerjee and Ted Pedersen. 2003. Extended gloss overlaps as a measure of semantic relatedness. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, pages 805–810, Acapulco, Mexico.

Ken Barker and Stan Szpakowicz. 1998. Semi-automatic recognition of noun modifier relationships. In *Proceedings of the 17th international conference on Computational linguistics*, pages 96–102, Quebec, Canada.

Ted Briscoe and John Carroll. 2002. Robust accurate statistical annotation of general text. In *Proc. of the 3rd International Conference on Language Resources and Evaluation (LREC 2002)*, pages 1499–1504, Las Palmas, Canary Islands.

Paul Buitelaar. 1998. *CoreLex: Systematic Polysemy and Underspecification*. Ph.D. thesis, Brandeis University, USA.

Yunbo Cao and Hang Li. 2002. Base noun phrase translation using web data and the EM algorithm. In *19th International Conference on Computational Linguistics*, Taipei, Taiwan.

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2004. *TiMBL: Tilburg Memory Based Learner, version 5.1, Reference Guide*. ILK Technical Report 04-02.

James Fan, Ken Barker, and Bruce W. Porter. 2003. The knowledge required to interpret noun compounds. In *7th International Joint Conference on Artificial Intelligence*, pages 1483–1485, Acapulco, Mexico.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, USA.

Timothy W. Finin. 1980. *The semantic interpretation of compound nominals*. Ph.D. thesis, University of Illinois, Urbana, Illinois, USA.

Jay Jiang and David Conrath. 1998. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings on International Conference on Research in Computational Linguistics*, pages 19–33.

Su Nam Kim and Timothy Baldwin. 2005. Automatic interpretation of noun compounds using WordNet similarity. In *Second International Joint Conference On Natural Language Processing*, pages 945–956, JeJu, Korea.

Maria Lapata. 2002. The disambiguation of nominalizations. *Computational Linguistics*, 28(3):357–388.

Mark Lauer. 1995. *Designing Statistical Language Learners: Experiments on Noun Compounds*. Ph.D. thesis, Macquarie University.

Judith Levi. 1979. *The Syntax and Semantics of Complex Nominals*. New York: Academic Press.

Dan Moldovan, Adriana Badulescu, Marta Tatu, Daniel Antohe, and Roxana Girju. 2004. Models for the semantic classification of noun phrases. In *HLT-NAACL 2004: Workshop on Computational Lexical Semantics*, pages 60–67, Boston, USA.

Siddharth Patwardhan, Satanjeev Banerjee, and Ted Pedersen. 2003. Using measures of semantic relatedness for word sense disambiguation. In *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*.

Siddharth Patwardhan. 2003. Incorporating dictionary and corpus information into a context vector measure of semantic relatedness. Master's thesis, University of Minnesota, USA.

Barbara Rosario and Hearst Marti. 2001. Classifying the semantic relations in noun compounds via a domain-specific lexical hierarchy. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pages 82–90.

Lucy Vanderwende. 1994. Algorithm for automatic interpretation of noun sequences. In *Proceedings of the 15th Conference on Computational linguistics*, pages 782–788.

Zhibiao Wu and Martha Palmer. 1994. Verb semantics and lexical selection. In *32nd Annual Meeting of the Association for Computational Linguistics*, pages 133 –138.

Unsupervised Analysis for Decipherment Problems

Kevin Knight, Anish Nair, Nishit Rathod
Information Sciences Institute
and Computer Science Department
University of Southern California
knight@isi.edu, {anair,nrathod}@usc.edu

Kenji Yamada
Language Weaver, Inc.
4640 Admiralty Way, Suite 1210
Marina del Rey, CA 90292
kyamada@languageweaver.com

Abstract

We study a number of natural language decipherment problems using unsupervised learning. These include letter substitution ciphers, character code conversion, phonetic decipherment, and word-based ciphers with relevance to machine translation. Straightforward unsupervised learning techniques most often fail on the first try, so we describe techniques for understanding errors and significantly increasing performance.

1 Introduction

Unsupervised learning holds great promise for breakthroughs in natural language processing. In cases like (Yarowsky, 1995), unsupervised methods offer accuracy results than rival supervised methods (Yarowsky, 1994) while requiring only a fraction of the data preparation effort. Such methods have also been a key driver of progress in statistical machine translation, which depends heavily on unsupervised word alignments (Brown et al., 1993).

There are also interesting problems for which supervised learning is not an option. These include deciphering unknown writing systems, such as the Easter Island *rongorongo* script and the 20,000-word Voynich manuscript. Deciphering animal language is another case. Machine translation of human languages is another, when we consider language pairs where little or no parallel text is available. Ultimately, unsupervised learning also holds promise for scientific discovery in linguistics. At some point, our programs will begin finding novel, publishable regularities in vast amounts of linguistic data.

2 Decipherment

In this paper, we look at a particular type of unsupervised analysis problem in which we face a ciphertext stream and try to uncover the plaintext that lies behind it. We will investigate several applications that can be profitably analyzed this way. We will also apply the same technical solution these different problems.

The method follows the well-known noisy-channel framework. At the top level, we want to find the plaintext that maximizes the probability P(plaintext | ciphertext). We first build a probabilistic model P(p) of the plaintext source. We then build probabilistic channel model P(c | p) that explains how plaintext sequences (like p) become ciphertext sequences (like c). Some of the parameters in these models can be estimated with supervised training, but most cannot.

When we face a new ciphertext sequence c, we first use expectation-maximization (EM) (Dempster, Laird, and Rubin, 1977) to set all free parameters to maximize P(c), which is the same (by Bayes Rule) as maximizing the sum over all p of P(p) · P(c | p). We then use the Viterbi algorithm to choose the p maximizing P(p) · P(c | p), which is the same (by Bayes Rule) as our original goal of maximizing P(p | c), or plaintext given ciphertext.

Figures 1 and 2 show standard EM algorithms (Knight, 1999) for the case in which we have a bigram P(p) model (driven by a two-dimensional b table of bigram probabilities) and a one-for-one P(c | p) model (driven by a two-dimensional s table of substitution probabilities). This case covers Section 3, while more complex models are employed in later sections.

3 English Letter Substitution

An *informal substitution cipher* (Smith, 1943) disguises a text by substituting code letters for normal letters. This system is usually *exclusive*, meaning that each plaintext letter maps to only one ciphertext letter, and vice versa. There is surprisingly little published on this problem, e.g., (Peleg and Rosenfeld, 1979), because fast computers led to public-key cryptography before much computer analysis was done on such old-style ciphers. We study this problem first because it resembles many of the other problems we are interested in, and we can generate arbitrary amounts of test data.

We estimate unsmoothed parameter values for an English letter-bigram P(p) from news data. This is a 27x27 table that includes the space character. We then set up a uniform P(c | p), which also happens to be a

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 499–506,
Sydney, July 2006.

```
(a) ingcmpnqsnwf cv fpn owoktvcv hu ihgzsnwfv rqcffnw cw owgcnwf kowazoanv...
(b) wecitherkent is the analysis of wocoments pritten in ancient buncquges...
(c) decipherment is the analysis of documents written in ancient languages...
```

Figure 3: Letter substitution decipherment. (a) is the ciphertext, (b) is an automatic decipherment, and (c) is an improved decipherment.

Given a ciphertext c of length m, a plaintext vocabulary of v tokens, and a plaintext bigram model b:

1. set a s($c|p$) substitution table initially to be uniform
2. for several iterations do:
   a. set up a count table count(c, p) with zero entries
   b. P(c) = 0
   c. for all possible plaintexts $p_1 \cdots p_m$
      (each p_i drawn from plaintext vocabulary)
      compute P(p) = b(p_1 | boundary) · b(boundary | p_m) $\cdot \prod_{i=2}^{m}$ b($p_i|p_{i-1}$)
      compute P(c|p) = $\prod_{j=1}^{m}$ s($c_j|p_j$)
      P(c) += P(p) · P(c|p)
   d. for all plaintexts p of length m
      compute P(p|c) = $\frac{\text{P(p)} \cdot \text{P(c|p)}}{\text{P(c)}}$
      for j = 1 to m
         count(c_j, p_j) += P(p|c)
   e. normalize count(c, p) table to create a revised s($c|p$)

Figure 1: A naive application of the EM algorithm to break a substitution cipher. It runs in O(mv^m) time.

Given a ciphertext c of length m, a plaintext vocabulary of v tokens, and a plaintext bigram model b:

1. set the s($c|p$) substitution table initially to be uniform
2. for several iterations do:
   a. set up a count(c, p) table with zero entries
   b. for i = 1 to v
      Q[i,1] = b(p_i | boundary)
   c. for j = 2 to m
      for i = 1 to v
         Q[i,j] = 0
         for k = 1 to v
            Q[i,j] += Q[k,$j-1$] · b($p_i|p_k$) · s($c_{j-1}|p_k$)
   d. for i = 1 to v
      R[i,m] = b(boundary | p_i)
   e. for j = $m-1$ to 1
      for i = 1 to v
         R[i,j] = 0
         for k = 1 to v
            R[i,j] += R[k,$j+1$] · b($p_k|p_i$) · s($c_{j+1}|p_k$)
   f. for j = 1 to m
      for i = 1 to v
         count(c_j, p_i) += Q[i,j] · R[i,j] · P($c_j|p_i$)
   g. normalize count(c, p) table to create a revised s($c|p$)

Figure 2: An efficient O(mv^2) algorithm that accomplishes the same thing as Figure 1.

27x27 table. We set P(space | SPACE) = 1.0, and all other values to 1/26. We create our ciphertext by encrypting an out-of-domain encyclopedia article. This article contains 417 letters, some of which are shown in Figure 3(a).

The decipherment yielded by EM/Viterbi contains 68 errors—see Figure 3(b).

Can we do better? First, we are not taking advantage of the fact that the cipher system is exclusive. But, as we observe in the rest of this paper, most natural decipherment problems do not have this feature, so we do not take advantage of it in this case (and it is hard to model!).

We can certainly acquire vastly more data for estimating P(p). Using a 1.5-million character data set instead of a 70,000-character data set reduces the number of errors from 68 to 64. Next, we apply fixed-lambda interpolation smoothing to P(p). This reduces errors further to 62.

Next, we adjust our Viterbi search to maximize P(p) · P(c | p)^3 rather than P(p) · P(c | p). This cubing concept was introduced in another context by (Knight and Yamada, 1999). It serves to stretch out the P(c | p) probabilities, which tend to be too bunched up. This bunching is caused by incompatibilities between the n-gram frequencies used to train P(p) and the n-gram frequencies found in the correct decipherment of c. We find this technique extremely useful across decipherment applications. Here it reduces errors from 62 down to 42.

We also gain by using letter trigrams instead of bigrams. This reduces error from the original 68 to 57 (small source data) or 32 (large source data). Combining trigrams with cubing the channel probabilities reduces error to 15, which source-model smoothing further reduces to 10 (or 2.4%), as in Figure 3(c).

So far we have glossed over the number of EM iterations used. From the EM's point of view, the more iterations, the better, as these improve P(c). However, the decipherment error rate may jump around as iterations proceed. Figure 4 shows the effect of EM iterations on error rate. With the worse source models, it is better to stop the EM early. EM initially locks onto the correct theory, but task performance degrades as it tries to make the ciphertext decoding fit the expected bigram frequencies. Better source models do not suffer much.

If we give the system more knowledge about English vocabulary and grammar, it will further improve. We have also been able to get perfect performance by using the best-so-far decipherment in Figure 3 to pull down related English texts from the web, and using these to retrain P(p) to fuel a second decipherment. However, we only present the simple substitution cipher as a prototype of the kinds of applications we are really interested in, which we present in the following sections.

The experiments we have presented so far should not be viewed as tuning parameters for performance—

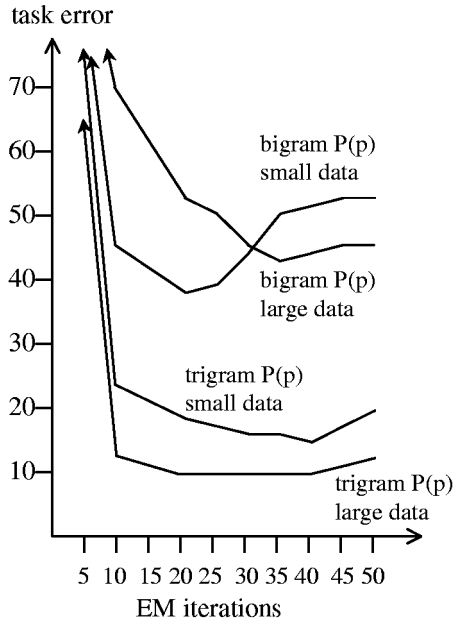


Figure 4: Decipherment error on letter substitution.

indeed, it is not correct to measure accuracy on a tuning/development data set. Rather, we have demonstrated some general strategies and observations (more data, larger n-grams, stability of good language models) that we can apply to other real decipherment situations. In many such situations, there is only a test set, and tuning is impossible even in principle—fortunately, we observe that the general strategies work robustly across a number of decipherment domains.

4 Character Code Conversion

Many human languages are straightforwardly represented at the character level by some widely-adopted standard (e.g., ASCII). In dealing with other languages (like Arabic), we must be equally prepared to process a few different standards. Documents in yet other languages (like Hindi) are found spread across the web in dozens if not hundreds of specialized encodings. These come with downloadable fonts for viewing. However, they are difficult to handle by computer, for example, to build a full-coverage Hindi web-search engine, or to pool Hindi corpora for training machine translation or speech recognition.

Character conversion tools exist for many pairs of major encoding systems, but it has been the experience of many researchers that these tools are flawed, despite the amount of work that goes into them. 100% accuracy is not to be found. Furthermore, nothing exists for most pairs. We believe that mild annotation techniques allow people to generate conversion tables quite quickly (and we show some results on this), but we follow here an unsupervised approach, as would be required to automatically generate a consistently-encoded Hindi web.

Our ciphertext c is a stream of bytes in an unknown encoding, with space separators; we use integers to represent these bytes, as in Figure 5(a). Our plaintext is a large collection of UTF8 standard Hindi. UTF8 builds complex Hindi character "chunks" out of up to 3 simple and combining characters. A Hindi word is a sequence of chunks, and words are separated by spaces.

We know that c is Hindi—we imagine that it was once UTF8, but that it somehow got enciphered.

Modeling is more complex than in the previous section. First, we have to decide what our plaintext tokens will be. Our first approach was to use chunks. Chunk boundaries are essentially those where we could draw a vertical line in written Hindi without disturbing any characters. We could then set up a model of how UTF8 is "encoded" to the mystery sequence in the putative channel—namely, we let each source chunk map to a particular target byte sequence. (By analogy, we would divide up English text into mostly letters, but would chunk ligatures like "fi" together. In fact, in extracting English text from pdf, we often find "fi" encoded by a single byte). This model is quite general and holds up across the encodings we have dealt with. However, there are over 900 chunks to contend with, and vast numbers of target byte sequences, so that the P(c | p) table is nearly unmanageable.

Therefore, we use a simpler model. We divide p into individual characters, and we set up a channel in which plaintext characters can map into either one or two ciphertext bytes. Instead of a table like P(c c | p), we set up two tables: P(f | p) for character fertility, and P(c | p) for character-to-byte substitution. This is similar to Model 3 of (Brown et al., 1993), but without null-generated elements or re-ordering.

Our actual ciphertext is an out-of-domain web page with 11,917 words of song lyrics in Hindi, in an idiosyncratic encoding. There is no known tool to convert from this encoding. In order to report error rates, we had to manually annotate a portion of this web page with correct UTF8. This was quite difficult. We were completely unable to do this manually by relying only on the ciphertext byte sequence—even though this is what we are asking our machine to do! But as Hindi readers, we also have access to the web-site rendering in Hindi glyphs, which helps us identify which byte sequences correspond to which Hindi glyphs, and then to UTF8. The labeled portion of our ciphertext consists of 59 running words (281 ciphertext bytes and 201 UTF8 characters).

Because the machine decipherment rarely consists of exactly 201 UTF8 characters, we report edit distance instead of error rate. An edit distance of 0 is perfect, while the edit distance for long incorrect decipherments may be greater than 201. With a source character bigram model, and the above channel, we obtain an edit distance of 161. With a trigram model, we get 127.

Now we introduce another idea that has worked across several decipherment problems. We use a fixed, uniform fertility model and allow EM only to manip-

```
(a) ... 13 5 14 . 16 2 25 26 2 25 . 17 2 13 . 15 2 8 . 7 2 4 2 9 2 2 ...
(b) ... 6 35    . 12 28 49 10 28  . 3 4 6   . 1 10 3 . 29 4 8 20 4   ...
(c) ... 6 35 24 . 12 28 21 4      . 11 6    . 12 25  . 29 8 22 4     ...
(d) ... 6/35/24 . 12/28 21/28     . 3/4 6   . 1/25   . 29 8 20/4     ... *
```

Figure 5: Hindi character code decipherment. (a) is the Hindi ciphertext byte sequence, (b) is an EM decipherment using a UTF8 trigram source model, (c) is a decipherment using a UTF8 word frequency model, and (d) is correct UTF8 (chunks joined with slash). Periods denote spaces between words; * denotes the correct answer.

```
P(13 | 6) = 0.66 *    P( 8|24) = 0.48
P(32 | 6) = 0.19      P(14|24) = 0.33 *
P( 2 | 6) = 0.13      P(17|24) = 0.14
P(16 | 6) = 0.02      P(25|24) = 0.04

P( 5 | 35) = 0.61 *   P(16|12) = 0.58 *
P(14 | 35) = 0.25     P( 2|12) = 0.32 *
P( 2 | 35) = 0.15     P(31|12) = 0.03
```

Figure 6: A portion of the learned P(c | p) substitution probabilities for Hindi decipherment. Correct mappings are marked with *.

ulate substitution probabilities. This prevents the algorithm from locking onto bad solutions. This gives an improved solution edit distance of 93, as in Figure 5(b), which can be compared to the correct decipherment in 5(d). Figure 6 shows a portion of the learned P(c | p) substitution table, with * indicating correct mappings.

15 out of 59 test words are deciphered exactly correctly. Another 16 out of 59 are perfect except for the addition of one extra UTF8 character (always "4" or "25"). Ours are the first results we know of with unsupervised techniques.

We also experimented with using a word-based source model in place of the character n-gram model. We built a word-unigram P(p) model out of only the top 5000 UTF8 words in our source corpus—it assigns probability zero to any word not in this list. This is a harsh model, considering that 16 out of 59 words in our UTF8-annotated test corpus do not even occur in the list, and are thus unreachable. On the plus side, EM considers only decipherments consisting of sequences of real Hindi words, and the Viterbi decoder only generates genuine Hindi words. The resulting decipherment edit distance is encouraging at 92, with the result shown in Figure 5(c). This model correctly deciphers 25 out of 59 words, with only some overlap to the previous 15 correct out of 59—one or other of the models is able to perfectly decipher 31 out of 59 words already, making a combination promising.

Our machine is also able to learn in a semi-supervised manner by aligning a cipher corpus with a manually-done translation into UTF8. EM searches for the parameter settings that maximize P(c | p), and a Viterbi alignment is a by-product. For the intuition, see Figure 5(a and d), in which plaintext character "6" occurs twice and may be guessed to correspond with ciphertext byte "13". EM does this perfectly, except for some regions where re-ordering indeed happens. We are able to move back to our chunk-based model in semi-supervised mode, which avoids the re-ordering problem, and we obtain near-perfect decipherment tables when we asked a human to re-type a few hundred words of mystery-encoded text in a UTF8 editor.

5 Phonetic Decipherment

This section expands previous work on phonetic decipherment (Knight and Yamada, 1999). Archaeologists are often faced with an unknown writing system that is believed to represent a known spoken language. That is, the written characters encode phonetic sequences (sometimes individual phonemes, and sometimes whole words), and the relationship between text and sound is to be discovered, followed by the meaning. Viewing text as a code for speech was radical some years ago. It is now the standard view of writing systems, and many even view written Chinese as a straightforward syllabary, albeit one that is much larger and complex than, say, Japanese kana. Both Linear B and Mayan writing were deciphered by viewing the observed text as a code/cipher for an approximately-known spoken language (Chadwick, 1958; Coe, 1993).

We follow (Knight and Yamada, 1999) in using Spanish as an example. The ciphertext is a 6980-character passage from Don Quixote, as in Figure 7(a). The plaintext is a very large out-of-domain Spanish phoneme sequence from which we compute only phoneme n-gram probabilities. We try deciphering without detailed knowledge of spoken Spanish words and grammar. The goal is for the decipherment to be understandable by modern Spanish speakers.

First, it is necessary to settle on the basic inventory of sounds and characters. Characters are easy; we simply tabulate the distinct ones observed in ciphertext. For sounds, we use a Spanish-relevant subset of the International Phonetic Alphabet (IPA), which seeks to capture all sounds in all languages; the implementation is SAMPA (Speech Assessment Methods Phonetic Alphabet). Here we show the sound and character inventories:

Sounds:

```
B, D, G, J (ny as in canyon), L (y as
in yarn), T (th as in thin), a, b, d,
e, f, g, i, k, l, m, n, o, p, r,
rr (trilled), s, t, tS (ch as in chin),
u, x (h as in hat)
```

(a) primera parte del ingenioso hidalgo don quijote de la mancha

```
(b) primera parte des intenioso liDasto don fuiLote de la manTia
(c) primera parte del inGenioso biDalGo don fuiLote de la manTia
(d) primera parte del inxenioso iDalGo  don kixote  de la manSa     *
```

Figure 7: Phonetic decipherment. (a) is written Spanish ciphertext, (b) is an initial decipherment, (c) is an improved decipherment, and (d) is the correct phonetic transcription.

Characters: ñ, á, é, í, ó, ú, a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z

The correct decipherment (Figure 7(d)) is a sequence of 6759 phonemes (here in SAMPA IPA).

We use a P(c | p) model that substitutes a single letter for each phoneme throughout the sequence. This considerably violates the rules of written Spanish (e.g., the K sound is often written with two letters q u, and the two K S sounds are often written x), so we do not expect a perfect decipherment. We do not enforce exclusivity; for example, the S sound may be written as c or s.

An unsmoothed phonetic bigram model gives an edit distance (error) of 805, as in Figure 7(b). Here we study smoothing techniques. A fixed-lambda interpolation smoothing yields 684 errors, while giving each phoneme its own trainable lambda yields a further reduction to 621. The corresponding edit distances for a trigram source model are 595, 703, and 492, the latter shown in Figure 7(c), an error of 7%. (This result is equivalent to Knight & Yamada [1999]'s 4% error, which did not count extra incorrect phonemes produced by decipherment, such as pronunciations of silent letters). Quality smoothing yields the best results. While even the best decipherment is flawed, it is perfectly understandable when synthesized, and it is very good with respect to the structure of the channel model.

6 Universal Phonetic Decipherment

What if the language behind the script is unknown? The next two sections address this question in two different ways.

One idea is to look for universal constraints on phoneme sequences. If we somehow know that P(K AE N UW L IY) is high, while P(R T M K T K) is low, that we may be able to exploit such knowledge in deciphering an alphabetic writing system. In fact, many universal constraints have been proposed by linguists. Two major camps include syllable theorists (who say that words are composed of syllables, and syllables have internal regular structure (Blevins, 1995)) and anti-syllable theorists (who say that words are composed of phonemes that often constrain each other even across putative syllable boundaries (Steriade, 1998)).

We use the same Don Quixote ciphertext as in the previous section. While the ultimate goal is to label each letter with a phoneme, we first attack a more tractable problem, that of labeling each letter as C (consonant) or V (vowel). Once we know which letters stand for consonant sounds, we can break them down further.

Our first approach is knowledge-free. We put together a fully-connected, uniform trigram source model P(p) over the tokens C, V, and SPACE. Our channel model P(c | p) is also fully-connected and uniform. We allow source as well as channel probabilities to float during training. This almost works, as shown in Figure 8(b). It correctly clusters letters into vowels and consonants, but assigns exactly the wrong labels! A complex cluster analysis (Finch and Chater, 1991) yields similar results.

Our second approach uses syllable theory. Our source model generates each source word in three phases. First, we probabilistically select the number of syllables to generate. Second, we probabilistically fill each slot with a syllable *type*. Every human language has a clear inventory of allowed syllable types, and many languages share the same inventory. Some examplars are (1995):

	V	CV	CVC	VC	CCV	CCVC	CVCC	VCC	CCVCC
Hua		✓							
Cayuvava	✓	✓							
Cairene		✓	✓						
Mazateco	✓	✓			✓				
Mokilese	✓	✓	✓	✓					
Sedang		✓	✓		✓	✓			
Klamath		✓	✓				✓		
Spanish	✓	✓	✓	✓	✓	✓			
Finnish	✓	✓	✓	✓			✓	✓	
Totonac		✓	✓		✓	✓	✓		✓
English	✓	✓	✓	✓	✓	✓	✓	✓	✓

For our purposes, we allow generation of V, VC, VCC, CV, CVC, CCV, CVCC, CCVC, or CCVCC. Elements of the syllable type sequence are chosen independently of each other, except that we disallow vowel-initial syllables following consonant-final syllables, following the phonetic universal tendency to "maximize the onset" (the initial consonant cluster of a syllable). Third, we spell out the chosen syllable types, so that the whole source model yields sequences over the tokens C, V, and SPACE, as before. This spelling-out is deterministic, except that we may turn a V into either one or two Vs, to account for dipthongs. The channel model again maps {C, V} onto {a, b, c, ...}, and we again run EM to learn both source and channel probabilities.

Figure 8(c) shows that this almost works. To make it work, 8(d), we force the number of syllables per word in the model to be fixed and uniform, rather than learned. This prevents the system from making analyses that are too short. We also execute several EM runs with randomly initialized P(c | p), and choose the run with the highest resulting P(c).

(a)	primera	parte	del	ingenioso	hidalgo	don	quijote	de	la	mancha
(b)	VVCVCVC	VCVVC	VCV	CVVCVVCVC	VCVCVVC	VCV	VCVVCVC	VC	VC	VCVVVC
(c)	CCV.CV.CV	CVC.CV	CVC	VC.CVC.CV.CV	CV.CVC.CV	CVC	CVC.CV.CV	CV	CV	CVC.CCV
(d)	CCV.CV.CV	CVC.CV	CVC	VC.CV.CV.V.CV	CV.CVC.CV	CVC	CV.V.CV.CV	CV	CV	CVC.CCV
(e)	NSV.NV.NV	NVS.NV	NVS	VS.NV.SV.V.NV	NV.NVS.NV	NVS	NV.V.NV.NV	NV	NV	NVS.NSV

Figure 8: Universal phonetic decipherment. The ciphertext (a) is the same as in the previous figure. (b) is an unsupervised consonant-vowel decipherment, (c) is a decipherment informed by syllable structure, (d) is an improved decipherment, and (e) is a decipherment that also attempts to distinguish sonorous (S) and non-sonorous (N) consonants.

We see that the Spanish letters are accurately divided into consonants and vowels, and it is also straightforward to ask about the learned syllable generation probabilities—they are CV (0.50), CVC (0.20), V (0.16), VC (0.11), CCV (0.02), CCVC (0.0002).

As a sanity check, we manually remove all P(c | p) parameters that match C with Spanish vowel-letters (a, e, i, o, u, y, and accented versions) and V with Spanish consonant-letters (b, c, d, etc), then re-run the same EM learning. We obtain the same P(c).

Exactly the same method works for Latin. Interestingly, the fully-connected P(c | p) model leads to a higher P(c) than the "correctly" constrained channel. We find that in the former, the letter i is sometimes treated as a vowel and other times as a consonant. The word "omnium" is analyzed by EM as VC.CV.VC, while "iurium" is analyzed as CVC.CVC.

We went a step further to see if EM could identify which letters encode *sonorous* versus *non-sonorous* consonants. Sonorous consonants are taken to be perceptually louder, and include n, m, l, and r. Additionally, vowels are more sonorous than consonants. A universal tendency (the *sonority hierarchy*) is that syllables have a sonority peak in the middle, which falls off to the left and right. This captures why the syllable G R A R G sounds more typical than R G A G R. There are exceptions, but the tendency is strong.

We modify our source model to generate S (sonorous consonant), N (non-sonorous consonant), V, and SPACE. We do this by changing the spell-out to probabilistically transform CCVC, for example, into either N S V S or N S V N, both of which respect the sonority hierarchy. The result is imperfect, with the EM hijacking the extra symbols. However, if we first run our C, V, SPACE model and feed the learned model to the S, N, V, SPACE model, then it works fairly well, as shown in Figure 8(e). Learned vowels include (in order of generation probability): e, a, o, u, i, y. Learned sonorous consonants include: n, s, r, l, m. Learned non-sonorous consonants include: d, c, t, l, b, m, p, q. The model bootstrapping is good for dealing with too many parameters; we see a similar approach in Brown et al's (1993) march from Model 1 to Model 5.

There are many other constraints to explore. For example, physiological constraints make some phonetic combinations more unlikely. AE N T and AE M P work because the second sound leaves the mouth well-prepared to make the third sound, while AE N P does not. These and other constraints complement the model by also working across syllable boundaries. There are also constraints on phoneme inventory (no voiced consonant like B without its unvoiced partner like P) and syllable inventory (no CCV without CV).

7 Brute-Force Phonetic Decipherment

Another approach to universal phonetic decipherment is to build phoneme n-gram databases for all human languages, then fully decipher with respect to each in turn. At the end, we need an automatic procedure for evaluating which source language has the best fit.

There do not seem to be sizeable phoneme-sequence corpora for many languages. Therefore, we used source character models as a stand in, decoding as in Section 3. We built 80 different source models from sequences we downloaded from the UN Universal Declaration of Human Rights website.[1]

Suppose our ciphertext starts "cevzren cnegr qry..." as in Figure 9(a). We decipher it against all 80 source language models, and the results are shown in Figure 9(b-f), ordered by post-training P(c). The system believes 9(a) is enciphered Spanish, but if not, then Galician, Portuguese, or Kurdish. Spanish is actually the correct answer, as the ciphertext is again Don Quixote (put through a simple letter substitution to show the problem from the computer's point of view). Similarly, EM detects that "fpn owoktvcv hu ihgzsnwfv rqcffnw cw..." is actually English, and deciphers it as "the analysis of wocuments pritten in..."

Many writing systems do not write vowel sounds. We can also do a brute force decipherment of vowelless writing by extending our channel model: first, we deterministically remove vowel sounds (or letters, in the above case), then we probabilistically substitute letters according to P(c | p). For the ciphertext "ceze ceg qy...", EM still proposes Spanish as the best source language, with decipherment "prmr prt dl..."

8 Word-Based Decoding

Letter-based substitution/transposition schemes are technically called ciphers, while systems that make whole-word substitutions are called *codes*. As an example code, one might write "I will bring the parrot to

[1] www.un.org/Overview/right.html

```
(a) cevzren cnegr qry vatravbfb uvqnytb qba dhvwbgr qr yn znapun

    P(c)      proposed  final
 perplexity   source    edit-dist          best P(p | c) decipherment

(b) 166.28    spanish    434    primera parte del ingenioso hidalgo don quijote de la mancha
(c) 168.75    galician   741    primera palte der ingenioso cidalgo don quixote de da mancca
(d) 169.07    portug.   1487    privera porte dal ingenioso didalgo dom quivote de ho concda
(e) 169.33    kurdish   4041    xwelawe berga mas estaneini hemestu min jieziga ma se lerdhe
  ...
(f) 179.19    english   4116    wizaris asive bec uitedundl pubsctl bly whualve be ks asequs
```

Figure 9: Brute-force phonetic decipherment. (a) is ciphertext in an unknown source language, while (b-f) show the best decipherments obtained for some of the 80 candidate source languages, automatically sorted by P(c).

Canada" instead of "I will bring the money to John"—or, one might encode every word in a message. Machine translation has code-like characteristics, and indeed, the initial models of (Brown et al., 1993) took a word-substitution/transposition approach, trained on a parallel text.

Because parallel text is scarce, it would be very good to extend unsupervised letter-substitution techniques to word-substitution in MT. Success to date has been limited, however. Here we execute a small-scale example, but completely from scratch.

In this experiment, we know the Arabic cipher names of seven countries: m!lyzy!, !lmksyk, knd!, bryT!ny!, frns!, !str!ly!, and !ndwnysy!. We also know a set of English equivalents, here in no particular order: Mexico, Canada, Malaysia, Britain, Australia, France, and Indonesia. Using non-parallel corpora, can we figure out which word is a translation of which? We use neither spelling information nor exclusivity, since these are not exploitable in the general MT problem.

To create a ciphertext, we add phrases X Y and Y X to the ciphertext whenever X and Y co-occur in the same sentence in the Arabic corpus. Sorting by frequency, this ciphertext looks like:

3385	frns!	bryT!ny!
3385	bryT!ny!	frns!
450	knd!	bryT!ny!
450	bryT!ny!	knd!
410	knd!	frns!
410	frns!	knd!
386	knd!	!str!ly!
386	!str!ly!	knd!
331	frns!	!str!ly!
331	!str!ly!	frns!
	etc.	

We create an English training corpus using the same method on English text, from which we build a bigram P(p) model:

511	France/French	Britain/British
511	Britain/British	France/French
362	Canada/Canadian	Britain/British
362	Britain/British	Canada/Canadian
182	France/French	Canada/Canadian
182	Canada/Canadian	France/French
140	Britain/British	Australia/Australian
140	Australia/Australian	Britain/British
133	Canada/Canadian	Australia/Australian
133	Australia/Australian	Canada/Canadian
	etc.	

Each corpus induces a kind of world map, with high frequency indicating closeness. The task is to figure out how elements of the two world maps correspond.

We train a source English bigram model P(p) on the plaintext, then set up a uniform P(c | p) channel with 7x7=49 parameters. Our initial result is not good: EM locks up after two iterations, and every English word learns the same distribution. When we choose a random initialization for P(c | p), we get a better result, as 4 out of 7 English words correctly map to their Arabic equivalents. With 5 random restarts, we achieve 5 correct, and with 40 or more random restarts, all 7 assignments are always correct. (From among the restarts, we select the one with the best post-EM P(c), not the best accuracy on the task.) The learned P(c | p) dictionary is shown here (correct mappings are marked with *).

```
P(!str!ly!   | Australia/Australian) = 0.93 *
P(!ndwnysy! | Australia/Australian) = 0.03
P(m!lyzy!   | Australia/Australian) = 0.02
P(!mksyk    | Australia/Australian) = 0.01

P(bryT!ny!  | Britain/British) =      0.98 *
P(!ndwnysy! | Britain/British) =      0.01
P(!str!ly!  | Britain/British) =      0.01

P(knd!      | Canada/Canadian) =      0.57 *
P(frns!     | Canada/Canadian) =      0.33
P(m!lyzy!   | Canada/Canadian) =      0.06
P(!ndwnysy! | Canada/Canadian) =      0.04

P(frns!     | France/French) =        1.00 *

P(!ndwnysy! | Indonesia/Indonesian) = 1.00 *

P(m!lyzy!   | Malaysia/Malaysian) =   0.93 *
P(!lmksyk   | Malaysia/Malaysian) =   0.07

P(!lmksyk   | Mexico/Mexican) =       0.91 *
P(m!lyzy!   | Mexico/Mexican) =       0.07
```

9 Conclusion

We have discussed several decipherment problems and shown that they can all be attacked by the same basic

method. Our primary contribution is a collection of first empirical results on a number of new problems. We also studied the following techniques in action:

- executing random restarts
- cubing learned channel probabilities before decoding
- using uniform probabilities for parameters of less interest
- checking learned P(c) against the P(c) of a "correct" model
- using a well-smoothed source model P(p)
- bootstrapping larger-parameter models with smaller ones
- appealing to linguistic universals to constrain models

Results on all of our applications were substantially improved using these techniques, and a secondary contribution is to show that they lead to robust improvements across a range of decipherment problems.

All of the experiments in this paper were carried out with the Carmel finite-state toolkit, (Graehl, 1997), which supports forward-backward EM with epsilon transitions and loops, parameter tying, and random restarts. It also composes two or more transducers while keeping their transitions separate (and separately trainable) in the composed model. Work described in this paper strongly influenced the toolkit's design.

Acknowledgements

We would like to thank Kie Zuraw and Cynthia Hagstrom for conversations about phonetic universals, and Jonathan Graehl for work on Carmel. This work was funded in part by NSF Grant 759635.

References

Blevins, J. 1995. The syllable in phonological theory. In J. Goldsmith, editor, *Handbook of Phonological Theory*. Basil Blackwell, London.

Brown, P., S. Della Pietra, V. Della Pietra, and R. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2).

Chadwick, J. 1958. *The Decipherment of Linear B*. Cambridge University Press, Cambridge.

Coe, M. 1993. *Breaking the Maya Code*. Thames and Hudson, New York.

Dempster, A. P., N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(B).

Finch, S. and N. Chater. 1991. A hybrid approach to the automatic learning of linguistic categories. *Artificial Intelligence and Simulated Behaviour Quarterly*, 78.

Graehl, Jonathan. 1997. Carmel finite-state toolkit. http://www.isi.edu/licensed-sw/carmel/.

Knight, K. 1999. Decoding complexity in word-replacement translation models. *Computational Linguistics*, 25(4).

Knight, K. and K. Yamada. 1999. A computational approach to deciphering unknown scripts. In *ACL Workshop on Unsupervised Learning in Natural Language Processing*.

Peleg, S. and A. Rosenfeld. 1979. Breaking substitution ciphers using a relaxation algorithm. *Communications of the ACM*, 22(11).

Smith, L. 1943. *Cryptography*. Dover Publications, NY.

Steriade, D. 1998. Alternatives to syllable-based accounts of consonantal phonotactics. In *Proc. of Conf. on Linguistic and Phonetics (LP'98)*.

Yarowsky, D. 1994. Decision lists for lexical ambiguity resolution: Application to accent restoration in Spanish and French. In *Proc. ACL*.

Yarowsky, D. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proc. ACL*.

Mildly Non-Projective Dependency Structures

Marco Kuhlmann
Programming Systems Lab
Saarland University
Germany
kuhlmann@ps.uni-sb.de

Joakim Nivre
Växjö University and
Uppsala University
Sweden
nivre@msi.vxu.se

Abstract

Syntactic parsing requires a fine balance between expressivity and complexity, so that naturally occurring structures can be accurately parsed without compromising efficiency. In dependency-based parsing, several constraints have been proposed that restrict the class of permissible structures, such as projectivity, planarity, multi-planarity, well-nestedness, gap degree, and edge degree. While projectivity is generally taken to be too restrictive for natural language syntax, it is not clear which of the other proposals strikes the best balance between expressivity and complexity. In this paper, we review and compare the different constraints theoretically, and provide an experimental evaluation using data from two treebanks, investigating how large a proportion of the structures found in the treebanks are permitted under different constraints. The results indicate that a combination of the well-nestedness constraint and a parametric constraint on discontinuity gives a very good fit with the linguistic data.

1 Introduction

Dependency-based representations have become increasingly popular in syntactic parsing, especially for languages that exhibit free or flexible word order, such as Czech (Collins et al., 1999), Bulgarian (Marinov and Nivre, 2005), and Turkish (Eryiğit and Oflazer, 2006). Many practical implementations of dependency parsing are restricted to *projective* structures, where the projection of a head word has to form a continuous substring of the sentence. While this constraint guarantees good parsing complexity, it is well-known that certain syntactic constructions can only be adequately represented by *non-projective* dependency structures, where the projection of a head can be discontinuous. This is especially relevant for languages with free or flexible word order.

However, recent results in non-projective dependency parsing, especially using data-driven methods, indicate that most non-projective structures required for the analysis of natural language are very nearly projective, differing only minimally from the best projective approximation (Nivre and Nilsson, 2005; Hall and Novák, 2005; McDonald and Pereira, 2006). This raises the question of whether it is possible to characterize a class of *mildly* non-projective dependency structures that is rich enough to account for naturally occurring syntactic constructions, yet restricted enough to enable efficient parsing.

In this paper, we review a number of proposals for classes of dependency structures that lie between strictly projective and completely unrestricted non-projective structures. These classes have in common that they can be characterized in terms of properties of the dependency structures themselves, rather than in terms of grammar formalisms that generate the structures. We compare the proposals from a theoretical point of view, and evaluate a subset of them empirically by testing their representational adequacy with respect to two dependency treebanks: the Prague Dependency Treebank (PDT) (Hajič et al., 2001), and the Danish Dependency Treebank (DDT) (Kromann, 2003).

The rest of the paper is structured as follows. In section 2, we provide a formal definition of dependency structures as a special kind of directed graphs, and characterize the notion of projectivity. In section 3, we define and compare five different constraints on mildly non-projective dependency structures that can be found in the literature: planarity, multiplanarity, well-nestedness, gap degree, and edge degree. In section 4, we provide an experimental evaluation of the notions of planarity, well-nestedness, gap degree, and edge degree, by

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 507–514,
Sydney, July 2006.

investigating how large a proportion of the dependency structures found in PDT and DDT are allowed under the different constraints. In section 5, we present our conclusions and suggestions for further research.

2 Dependency graphs

For the purposes of this paper, a *dependency graph* is a directed graph on the set of indices corresponding to the tokens of a sentence. We write $[n]$ to refer to the set of positive integers up to and including n.

Definition 1 A *dependency graph* for a sentence $x = w_1, \ldots, w_n$ is a directed graph[1]

$$G = (V\,;\,E), \quad \text{where } V = [n] \text{ and } E \subseteq V \times V.$$

Throughout this paper, we use standard terminology and notation from graph theory to talk about dependency graphs. In particular, we refer to the elements of the set V as *nodes*, and to the elements of the set E as *edges*. We write $i \to j$ to mean that there is an edge from the node i to the node j (i.e., $(i, j) \in E$), and $i \to^* j$ to mean that the node i *dominates* the node j, i.e., that there is a (possibly empty) path from i to j. For a given node i, the set of nodes dominated by i is the *yield* of i. We use the notation $\pi(i)$ to refer to the *projection* of i: the yield of i, arranged in ascending order.

2.1 Dependency forests

Most of the literature on dependency grammar and dependency parsing does not allow arbitrary dependency graphs, but imposes certain structural constraints on them. In this paper, we restrict ourselves to dependency graphs that form *forests*.

Definition 2 A *dependency forest* is a dependency graph with two additional properties:

1. it is acyclic (i.e., if $i \to j$, then not $j \to^* i$);
2. each of its nodes has at most one incoming edge (i.e., if $i \to j$, then there is no node k such that $k \neq i$ and $k \to j$).

Nodes in a forest without an incoming edge are called *roots*. A dependency forest with exactly one root is a *dependency tree*.

Figure 1 shows a dependency forest taken from PDT. It has two roots: node 2 (corresponding to the complementizer *proto*) and node 8 (corresponding to the final punctuation mark).

[1] We only consider unlabelled dependency graphs.

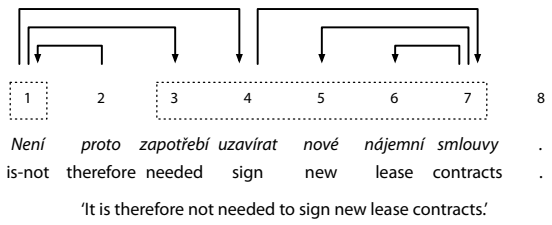


Figure 1: Dependency forest for a Czech sentence from the Prague Dependency Treebank

Some authors extend dependency forests by a special root node with position 0, and add an edge $(0, i)$ for every root node i of the remaining graph (McDonald et al., 2005). This ensures that the extended graph always is a tree. Although such a definition can be useful, we do not follow it here, since it obscures the distinction between projectivity and planarity to be discussed in section 3.

2.2 Projectivity

In contrast to acyclicity and the indegree constraint, both of which impose restrictions on the dependency relation as such, the *projectivity constraint* concerns the interaction between the dependency relation and the positions of the nodes in the sentence: it says that the nodes in a subtree of a dependency graph must form an *interval*, where an interval (with endpoints i and j) is the set

$$[i, j] \;:=\; \{\, k \in V \mid i \leq k \text{ and } k \leq j \,\}.$$

Definition 3 A dependency graph is *projective*, if the yields of its nodes are intervals.

Since projectivity requires each node to dominate a continuous substring of the sentence, it corresponds to a ban on discontinuous constituents in phrase structure representations.

Projectivity is an interesting constraint on dependency structures both from a theoretical and a practical perspective. Dependency grammars that only allow projective structures are closely related to context-free grammars (Gaifman, 1965; Obrębski and Graliński, 2004); among other things, they have the same (weak) expressivity. The projectivity constraint also leads to favourable parsing complexities: chart-based parsing of projective dependency grammars can be done in cubic time (Eisner, 1996); hard-wiring projectivity into a deterministic dependency parser leads to linear-time parsing in the worst case (Nivre, 2003).

3 Relaxations of projectivity

While the restriction to projective analyses has a number of advantages, there is clear evidence that it cannot be maintained for real-world data (Zeman, 2004; Nivre, 2006). For example, the graph in Figure 1 is non-projective: the yield of the node 1 (marked by the dashed rectangles) does not form an interval—the node 2 is 'missing'. In this section, we present several proposals for structural constraints that relax projectivity, and relate them to each other.

3.1 Planarity and multiplanarity

The notion of *planarity* appears in work on Link Grammar (Sleator and Temperley, 1993), where it is traced back to Mel'čuk (1988). Informally, a dependency graph is *planar*, if its edges can be drawn above the sentence without crossing. We emphasize the word *above*, because planarity as it is understood here does not coincide with the standard graph-theoretic concept of the same name, where one would be allowed to also use the area below the sentence to disentangle the edges.

Figure 2a shows a dependency graph that is planar but not projective: while there are no crossing edges, the yield of the node 1 (the set $\{1, 3\}$) does not form an interval.

Using the notation $linked(i, j)$ as an abbreviation for the statement 'there is an edge from i to j, or vice versa', we formalize planarity as follows:

Definition 4 A dependency graph is *planar*, if it does not contain nodes a, b, c, d such that

$$linked(a, c) \wedge linked(b, d) \wedge a < b < c < d\,.$$

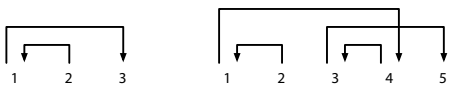
Yli-Jyrä (2003) proposes *multiplanarity* as a generalization of planarity suitable for modelling dependency analyses, and evaluates it experimentally using data from DDT.

Definition 5 A dependency graph $G = (V\,;\,E)$ is *m-planar*, if it can be split into m planar graphs

$$G_1 = (V\,;\,E_1), \ldots, G_m = (V\,;\,E_m)$$

such that $E = E_1 \uplus \cdots \uplus E_m$. The planar graphs G_i are called *planes*.

As an example of a dependency forest that is 2-planar but not planar, consider the graph depicted in Figure 2b. In this graph, the edges $(1, 4)$ and $(3, 5)$ are crossing. Moving either edge to a separate graph partitions the original graph into two planes.



(a) 1-planar (b) 2-planar

Figure 2: Planarity and multi-planarity

3.2 Gap degree and well-nestedness

Bodirsky et al. (2005) present two structural constraints on dependency graphs that characterize analyses corresponding to derivations in Tree Adjoining Grammar: the *gap degree restriction* and the *well-nestedness constraint*.

A *gap* is a discontinuity in the projection of a node in a dependency graph (Plátek et al., 2001). More precisely, let π_i be the projection of the node i. Then a gap is a pair (j_k, j_{k+1}) of nodes adjacent in π_i such that $j_{k+1} - j_k > 1$.

Definition 6 The *gap degree* of a node i in a dependency graph, $gd(i)$, is the number of gaps in π_i.

As an example, consider the node labelled i in the dependency graphs in Figure 3. In Graph 3a, the projection of i is an interval $((2, 3, 4))$, so i has gap degree 0. In Graph 3b, $\pi_i = (2, 3, 6)$ contains a single gap $((3, 6))$, so the gap degree of i is 1. In the rightmost graph, the gap degree of i is 2, since $\pi_i = (2, 4, 6)$ contains two gaps $((2, 4)$ and $(4, 6))$.

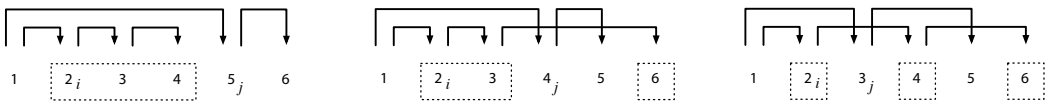
Definition 7 The *gap degree* of a dependency graph G, $gd(G)$, is the maximum among the gap degrees of its nodes.

Thus, the gap degree of the graphs in Figure 3 is 0, 1 and 2, respectively, since the node i has the maximum gap degree in all three cases.

The *well-nestedness constraint* restricts the positioning of disjoint subtrees in a dependency forest. Two subtrees are called disjoint, if neither of their roots dominates the other.

Definition 8 Two subtrees T_1, T_2 *interleave*, if there are nodes $l_1, r_1 \in T_1$ and $l_2, r_2 \in T_2$ such that $l_1 < l_2 < r_1 < r_2$. A dependency graph is *well-nested*, if no two of its disjoint subtrees interleave.

Both Graph 3a and Graph 3b are well-nested. Graph 3c is not well-nested. To see this, let T_1 be the subtree rooted at the node labelled i, and let T_2 be the subtree rooted at j. These subtrees interleave, as T_1 contains the nodes 2 and 4, and T_2 contains the nodes 3 and 5.



(a) $gd = 0$, $ed = 0$, $wn+$ (b) $gd = 1$, $ed = 1$, $wn+$ (c) $gd = 2$, $ed = 1$, $wn-$

Figure 3: Gap degree, edge degree, and well-nestedness

3.3 Edge degree

The notion of *edge degree* was introduced by Nivre (2006) in order to allow mildly non-projective structures while maintaining good parsing efficiency in data-driven dependency parsing.[2]

Define the *span* of an edge (i, j) as the interval

$$S((i, j)) \;:=\; [\min(i, j), \max(i, j)]\,.$$

Definition 9 Let $G = (V ; E)$ be a dependency forest, let $e = (i, j)$ be an edge in E, and let G_e be the subgraph of G that is induced by the nodes contained in the span of e.

- The *degree* of an edge $e \in E$, $ed(e)$, is the number of connected components c in G_e such that the root of c is not dominated by the head of e.
- The *edge degree* of G, $ed(G)$, is the maximum among the degrees of the edges in G.

To illustrate the notion of edge degree, we return to Figure 3. Graph 3a has edge degree 0: the only edge that spans more nodes than its head and its dependent is $(1, 5)$, but the root of the connected component $\{2, 3, 4\}$ is dominated by 1. Both Graph 3b and 3c have edge degree 1: the edge $(3, 6)$ in Graph 3b and the edges $(2, 4)$, $(3, 5)$ and $(4, 6)$ in Graph 3c each span a single connected component that is not dominated by the respective head.

3.4 Related work

Apart from proposals for structural constraints relaxing projectivity, there are dependency frameworks that in principle allow unrestricted graphs, but provide mechanisms to control the actually permitted forms of non-projectivity in the grammar.

The non-projective dependency grammar of Kahane et al. (1998) is based on an operation on dependency trees called *lifting*: a 'lift' of a tree T is the new tree that is obtained when one replaces one or more edges (i, k) in T by edges (j, k), where $j \rightarrow^* i$. The exact conditions under which a certain lifting may take place are specified in the rules of the grammar. A dependency tree is acceptable, if it can be lifted to form a projective graph.[3]

A similar design is pursued in Topological Dependency Grammar (Duchier and Debusmann, 2001), where a dependency analysis consists of two, mutually constraining graphs: the *ID graph* represents information about immediate dominance, the *LP graph* models the topological structure of a sentence. As a principle of the grammar, the LP graph is required to be a lift of the ID graph; this lifting can be constrained in the lexicon.

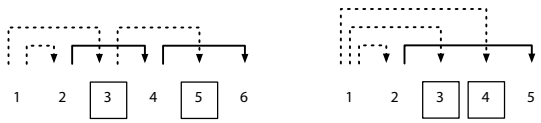
3.5 Discussion

The structural conditions we have presented here naturally fall into two groups: multiplanarity, gap degree and edge degree are *parametric constraints* with an infinite scale of possible values; planarity and well-nestedness come as *binary constraints*. We discuss these two groups in turn.

Parametric constraints With respect to the graded constraints, we find that multiplanarity is different from both gap degree and edge degree in that it involves a notion of optimization: since every dependency graph is m-planar for some sufficiently large m (put each edge onto a separate plane), the interesting question in the context of multiplanarity is about the *minimal* values for m that occur in real-world data. But then, one not only needs to show that a dependency graph *can* be decomposed into m planar graphs, but also that this decomposition is the one with the smallest number of planes among all possible decompositions. Up to now, no tractable algorithm to find the minimal decomposition has been given, so it is not clear how to evaluate the significance of the concept as such. The evaluation presented by Yli-Jyrä (2003) makes use of additional constraints that are sufficient to make the decomposition unique.

[2] We use the term *edge degree* instead of the original simple term *degree* from Nivre (2006) to mark the distinction from the notion of gap degree.

[3] We remark that, without restrictions on the lifting, *every* non-projective tree has a projective lift.



(a) $gd = 2, ed = 1$ (b) $gd = 1, ed = 2$

Figure 4: Comparing gap degree and edge degree

The fundamental difference between gap degree and edge degree is that the gap degree measures the number of discontinuities within a subtree, while the edge degree measures the number of intervening constituents spanned by a single edge. This difference is illustrated by the graphs displayed in Figure 4. Graph 4a has gap degree 2 but edge degree 1: the subtree rooted at node 2 (marked by the solid edges) has two gaps, but each of its edges only spans one connected component not dominated by 2 (marked by the squares). In contrast, Graph 4b has gap degree 1 but edge degree 2: the subtree rooted at node 2 has one gap, but this gap contains two components not dominated by 2.

Nivre (2006) shows experimentally that limiting the permissible edge degree to 1 or 2 can reduce the average parsing time for a deterministic algorithm from quadratic to linear, while omitting less than 1% of the structures found in DDT and PDT. It can be expected that constraints on the gap degree would have very similar effects.

Binary constraints For the two binary constraints, we find that well-nestedness subsumes planarity: a graph that contains interleaving subtrees cannot be drawn without crossing edges, so every planar graph must also be well-nested. To see that the converse does not hold, consider Graph 3b, which is well-nested, but not planar.

Since both planarity and well-nestedness are proper extensions of projectivity, we get the following hierarchy for sets of dependency graphs:

$$\text{projective} \subset \text{planar} \subset \text{well-nested} \subset \text{unrestricted}$$

The planarity constraint appears like a very natural one at first sight, as it expresses the intuition that 'crossing edges are bad', but still allows a limited form of non-projectivity. However, many authors use planarity in conjunction with a special representation of the root node: either as an artificial node at the sentence boundary, as we mentioned in section 2, or as the target of an infinitely long perpendicular edge coming 'from the outside', as in earlier versions of Word Grammar (Hudson, 2003). In these situations, planarity reduces to projectivity, so nothing is gained.

Even in cases where planarity is used without a special representation of the root node, it remains a peculiar concept. When we compare it with the notion of gaps, for example, we find that, in a planar dependency tree, every gap (i, j) must contain the root node r, in the sense that $i < r < j$: if the gap would only contain non-root nodes k, then the two paths from r to k and from i to j would cross. This particular property does not seem to be mirrored in any linguistic prediction.

In contrast to planarity, well-nestedness is independent from both gap degree and edge degree in the sense that for every $d > 0$, there are both well-nested and non-well-nested dependency graphs with gap degree or edge degree d. All projective dependency graphs ($d = 0$) are trivially well-nested.

Well-nestedness also brings computational benefits. In particular, chart-based parsers for grammar formalisms in which derivations obey the well-nestedness constraint (such as Tree Adjoining Grammar) are not hampered by the 'crossing configurations' to which Satta (1992) attributes the fact that the universal recognition problem of Linear Context-Free Rewriting Systems is $\mathcal{NP}$-complete.

4 Experimental evaluation

In this section, we present an experimental evaluation of planarity, well-nestedness, gap degree, and edge degree, by examining how large a proportion of the structures found in two dependency treebanks are allowed under different constraints. Assuming that the treebank structures are sampled from naturally occurring structures in natural language, this provides an indirect evaluation of the linguistic adequacy of the different proposals.

4.1 Experimental setup

The experiments are based on data from the Prague Dependency Treebank (PDT) (Hajič et al., 2001) and the Danish Dependency Treebank (DDT) (Kromann, 2003). PDT contains 1.5M words of newspaper text, annotated in three layers according to the theoretical framework of Functional Generative Description (Böhmová et al., 2003). Our experiments concern only the analytical layer, and are based on the dedicated training section of the treebank. DDT comprises 100k words of text selected from the Danish PAROLE corpus, with annotation

Table 1: Experimental results for DDT and PDT

property	DDT		PDT	
all structures	$n = 4393$		$n = 73088$	
gap degree 0	3732	84.95%	56168	76.85%
gap degree 1	654	14.89%	16608	22.72%
gap degree 2	7	0.16%	307	0.42%
gap degree 3	–	–	4	0.01%
gap degree 4	–	–	1	< 0.01%
edge degree 0	3732	84.95%	56168	76.85%
edge degree 1	584	13.29%	16585	22.69%
edge degree 2	58	1.32%	259	0.35%
edge degree 3	17	0.39%	63	0.09%
edge degree 4	2	0.05%	10	0.01%
edge degree 5	–	–	2	< 0.01%
edge degree 6	–	–	1	< 0.01%
projective	3732	84.95%	56168	76.85%
planar	3796	86.41%	60048	82.16%
well-nested	4388	99.89%	73010	99.89%
non-projective structures only	$n = 661$		$n = 16920$	
planar	64	9.68%	3880	22.93%
well-nested	656	99.24%	16842	99.54%

of primary and secondary dependencies based on Discontinuous Grammar (Kromann, 2003). Only primary dependencies are considered in the experiments, which are based on the entire treebank.[4]

4.2 Results

The results of our experiments are given in Table 1. For the binary constraints (planarity, well-nestedness), we simply report the number and percentage of structures in each data set that satisfy the constraint. For the parametric constraints (gap degree, edge degree), we report the number and percentage of structures having degree d ($d \geq 0$), where degree 0 is equivalent (for both gap degree and edge degree) to projectivity.

For DDT, we see that about 15% of all analyses are non-projective. The minimal degree of non-projectivity required to cover all of the data is 2 in the case of gap degree and 4 in the case of edge degree. For both measures, the number of structures drops quickly as the degree increases. (As an example, only 7 or 0.17% of the analyses in DDT have gap degree 2.) Regarding the binary constraints, we find that planarity accounts for slightly more than the projective structures (86.41% of the data is planar), while almost all structures in DDT (99.89%) meet the well-nestedness constraint. The difference between the two constraints becomes clearer when we base the figures on the set of non-projective structures only: out of these, less than 10% are planar, while more than 99% are well-nested.

For PDT, both the number of non-projective structures (around 23%) and the minimal degrees of non-projectivity required to cover the full data (gap degree 4 and edge degree 6) are higher than in DDT. The proportion of planar analyses is smaller than in DDT if we base it on the set of all structures (82.16%), but significantly larger when based on the set of non-projective structures only (22.93%). However, this is still very far from the well-nestedness constraint, which has almost perfect coverage on both data sets.

4.3 Discussion

As a general result, our experiments confirm previous studies on non-projective dependency parsing (Nivre and Nilsson, 2005; Hall and Novák, 2005;

[4] A total number of 17 analyses in DDT were excluded because they either had more than one root node, or violated the indegree constraint. (Both cases are annotation errors.)

McDonald and Pereira, 2006): The phenomenon of non-projectivity cannot be ignored without also ignoring a significant portion of real-world data (around 15% for DDT, and 23% for PDT). At the same time, already a small step beyond projectivity accounts for almost all of the structures occurring in these treebanks.

More specifically, we find that already an edge degree restriction of $d \leq 1$ covers 98.24% of DDT and 99.54% of PDT, while the same restriction on the gap degree scale achieves a coverage of 99.84% (DDT) and 99.57% (PDT). Together with the previous evidence that both measures also have computational advantages, this provides a strong indication for the usefulness of these constraints in the context of non-projective dependency parsing.

When we compare the two graded constraints to each other, we find that the gap degree measure partitions the data into less and larger clusters than the edge degree, which may be an advantage in the context of using the degree constraints as features in a data-driven approach towards parsing. However, our purely quantitative experiments cannot answer the question, which of the two measures yields the more informative clusters.

The planarity constraint appears to be of little use as a generalization of projectivity: enforcing it excludes more than 75% of the non-projective data in PDT, and 90% of the data in DDT. The relatively large difference in coverage between the two treebanks may at least partially be explained with their different annotation schemes for sentence-final punctuation. In DDT, sentence-final punctuation marks are annotated as dependents of the main verb of a dependency nexus. This, as we have discussed above, places severe restrictions on permitted forms of non-projectivity in the remaining sentence, as every discontinuity that includes the main verb must also include the dependent punctuation marks. On the other hand, in PDT, a sentence-final punctuation mark is annotated as a separate root node with no dependents. This scheme does not restrict the remaining discontinuities at all.

In contrast to planarity, the well-nestedness constraint appears to constitute a very attractive extension of projectivity. For one thing, the almost perfect coverage of well-nestedness on DDT and PDT (99.89%) could by no means be expected on purely combinatorial grounds—only 7% of all possible dependency structures for sentences of length 17 (the average sentence length in PDT), and only slightly more than 5% of all possible dependency structures for sentences of length 18 (the average sentence length in DDT) are well-nested.[5] Moreover, a cursory inspection of the few problematic cases in DDT indicates that violations of the well-nestedness constraint may, at least in part, be due to properties of the annotation scheme, such as the analysis of punctuation in quotations. However, a more detailed analysis of the data from both treebanks is needed before any stronger conclusions can be drawn concerning well-nestedness.

5 Conclusion

In this paper, we have reviewed a number of proposals for the characterization of mildly non-projective dependency structures, motivated by the need to find a better balance between expressivity and complexity than that offered by either strictly projective or unrestricted non-projective structures. Experimental evaluation based on data from two treebanks shows, that a combination of the well-nestedness constraint and parametric constraints on discontinuity (formalized either as gap degree or edge degree) gives a very good fit with the empirical linguistic data. Important goals for future work are to widen the empirical basis by investigating more languages, and to perform a more detailed analysis of linguistic phenomena that violate certain constraints. Another important line of research is the integration of these constraints into parsing algorithms for non-projective dependency structures, potentially leading to a better trade-off between accuracy and efficiency than that obtained with existing methods.

Acknowledgements We thank three anonymous reviewers of this paper for their comments. The work of Marco Kuhlmann is funded by the Collaborative Research Centre 378 'Resource-Adaptive Cognitive Processes' of the Deutsche Forschungsgemeinschaft. The work of Joakim Nivre is partially supported by the Swedish Research Council.

[5] The number of unrestricted dependency trees on n nodes is given by Sequence A000169, the number of well-nested dependency trees is given by Sequence A113882 in the On-Line Encyclopedia of Integer Sequences (Sloane, 2006).

References

Manuel Bodirsky, Marco Kuhlmann, and Mathias Möhl. 2005. Well-nested drawings as models of syntactic structure. In *Tenth Conference on Formal Grammar and Ninth Meeting on Mathematics of Language*.

Alena Böhmová, Jan Hajič, Eva Hajičová, and Barbora Hladká. 2003. The Prague Dependency Treebank: A three-level annotation scenario. In Anne Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*, pages 103–127. Kluwer Academic Publishers.

Michael Collins, Jan Hajič, Eric Brill, Lance Ramshaw, and Christoph Tillmann. 1999. A statistical parser for Czech. In *37th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 505–512.

Denys Duchier and Ralph Debusmann. 2001. Topological dependency trees: A constraint-based account of linear precedence. In *39th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 180–187.

Jason Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *16th International Conference on Computational Linguistics (COLING)*, pages 340–345.

Gülsen Eryiğit and Kemal Oflazer. 2006. Statistical dependency parsing of turkish. In *Eleventh Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Haim Gaifman. 1965. Dependency systems and phrase-structure systems. *Information and Control*, 8:304–337.

Jan Hajič, Barbora Vidova Hladka, Jarmila Panevová, Eva Hajičová, Petr Sgall, and Petr Pajas. 2001. Prague Dependency Treebank 1.0. LDC, 2001T10.

Keith Hall and Vaclav Novák. 2005. Corrective modeling for non-projective dependency parsing. In *Ninth International Workshop on Parsing Technologies (IWPT)*.

Richard Hudson. 2003. An encyclopedia of English grammar and Word Grammar. http://www.phon.ucl.ac.uk/home/dick/enc/intro.htm, January.

Sylvain Kahane, Alexis Nasr, and Owen Rambow. 1998. Pseudo-projectivity: A polynomially parsable non-projective dependency grammar. In *36th Annual Meeting of the Association for Computational Linguistics and 18th International Conference on Computational Linguistics (COLING-ACL)*, pages 646–652.

Matthias Trautner Kromann. 2003. The Danish Dependency Treebank and the DTAG treebank tool. In *Second Workshop on Treebanks and Linguistic Theories (TLT)*, pages 217–220.

Svetoslav Marinov and Joakim Nivre. 2005. A data-driven parser for Bulgarian. In *Fourth Workshop on Treebanks and Linguistic Theories (TLT)*, pages 89–100.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Eleventh Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *43rd Annual Meeting of the Association for Computational Linguistics (ACL)*.

Igor Mel'čuk. 1988. *Dependency Syntax: Theory and Practice*. State University of New York Press, Albany, New York, USA.

Joakim Nivre and Jens Nilsson. 2005. Pseudo-projective dependency parsing. In *43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 99–106.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Eigth International Workshop on Parsing Technologies (IWPT)*, pages 149–160.

Joakim Nivre. 2006. Constraints on non-projective dependency parsing. In *Eleventh Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

T. Obrębski and F. Graliński. 2004. Some notes on generative capacity of dependency grammar. In *COLING 2004 Workshop on Recent Advances in Dependency Grammar Workshop on Recent Advances in Dependency Grammar*.

Martin Plátek, Tomáš Holan, and Vladislav Kuboň. 2001. On relax-ability of word order by d-grammars. In *Third International Conference on Discrete Mathematics and Theoretical Computer Science*.

Giorgio Satta. 1992. Recognition of linear context-free rewriting systems. In *30th Meeting of the Association for Computational Linguistics (ACL)*, pages 89–95, Newark, Delaware, USA.

Daniel Sleator and Davy Temperley. 1993. Parsing English with a link grammar. In *Third International Workshop on Parsing Technologies*.

Neil J. A. Sloane. 2006. The on-line encyclopedia of integer sequences. Published electronically at http://www.research.att.com/ njas/sequences/.

Anssi Yli-Jyrä. 2003. Multiplanarity – a model for dependency structures in treebanks. In *Second Workshop on Treebanks and Linguistic Theories (TLT)*, pages 189–200.

Daniel Zeman. 2004. *Parsing With a Statistical Dependency Model*. Ph.D. thesis, Charles University, Prague, Czech Republic.

Parsing and Subcategorization Data

Jianguo Li and Chris Brew
Department of Linguistics
The Ohio State University
Columbus, OH, USA
{jianguo|cbrew}@ling.ohio-state.edu

Abstract

In this paper, we compare the performance of a state-of-the-art statistical parser (Bikel, 2004) in parsing written and spoken language and in generating subcategorization cues from written and spoken language. Although Bikel's parser achieves a higher accuracy for parsing written language, it achieves a higher accuracy when extracting subcategorization cues from spoken language. Our experiments also show that current technology for extracting subcategorization frames initially designed for written texts works equally well for spoken language. Additionally, we explore the utility of punctuation in helping parsing and extraction of subcategorization cues. Our experiments show that punctuation is of little help in parsing spoken language and extracting subcategorization cues from spoken language. This indicates that there is no need to add punctuation in transcribing spoken corpora simply in order to help parsers.

1 Introduction

Robust statistical syntactic parsers, made possible by new statistical techniques (Collins, 1999; Charniak, 2000; Bikel, 2004) and by the availability of large, hand-annotated training corpora such as WSJ (Marcus et al., 1993) and Switchboard (Godefrey et al., 1992), have had a major impact on the field of natural language processing. There are many ways to make use of parsers' output. One particular form of data that can be extracted from parses is information about subcategorization. Subcategorization data comes in two forms: subcategorization frame (SCF) and subcategorization cue (SCC). SCFs differ from SCCs in that SCFs contain only arguments while SCCs contain both arguments and adjuncts. Both SCFs and SCCs have been crucial to NLP tasks. For example, SCFs have been used for verb disambiguation and classification (Schulte im Walde, 2000; Merlo and Stevenson, 2001; Lapata and Brew, 2004; Merlo et al., 2005) and SCCs for semantic role labeling (Xue and Palmer, 2004; Punyakanok et al., 2005).

Current technology for automatically acquiring subcategorization data from corpora usually relies on statistical parsers to generate SCCs. While great efforts have been made in parsing written texts and extracting subcategorization data from written texts, spoken corpora have received little attention. This is understandable given that spoken language poses several challenges that are absent in written texts, including disfluency, uncertainty about utterance segmentation and lack of punctuation. Roland and Jurafsky (1998) have suggested that there are substantial subcategorization differences between written corpora and spoken corpora. For example, while written corpora show a much higher percentage of passive structures, spoken corpora usually have a higher percentage of zero-anaphora constructions. We believe that subcategorization data derived from spoken language, if of acceptable quality, would be of more value to NLP tasks involving a syntactic analysis of spoken language. We do not show this here.

The goals of this study are as follows:

1. Test the performance of Bikel's parser in parsing written and spoken language.

2. Compare the accuracy level of SCCs generated from parsed written and spoken lan-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 515–522,
Sydney, July 2006.

guage. We hope that such a comparison will shed some light on the feasibility of acquiring subcategorization data from spoken language using the current SCF acquisition technology initially designed for written language.

3. Apply our SCF extraction system (Li and Brew, 2005) to spoken and written language separately and compare the accuracy achieved for the acquired SCFs from spoken and written language.

4. Explore the utility of punctuation[1] in parsing and extraction of SCCs. It is generally recognized that punctuation helps in parsing written texts. For example, Roark (2001) finds that removing punctuation from both training and test data (WSJ) decreases his parser's accuracy from 86.4%/86.8% (LR/LP) to 83.4%/84.1%. However, spoken language does not come with punctuation. Even when punctuation is added in the process of transcription, its utility in helping parsing is slight. Both Roark (2001) and Engel et al. (2002) report that removing punctuation from both training and test data (Switchboard) results in only 1% decrease in their parser's accuracy.

2 Experiment Design

Three models will be investigated for parsing and extracting SCCs from the parser's output:

1. **punc**: leaving punctuation in both training and test data.

2. **no-punc**: removing punctuation from both training and test data.

3. **punc-no-punc**: removing punctuation from only the test data.

Following the convention in the parsing community, for written language, we selected sections 02-21 of WSJ as training data and section 23 as test data (Collins, 1999). For spoken language, we designated section 2 and 3 of Switchboard as training data and files of sw4004 to sw4135 of section 4 as test data (Roark, 2001). Since we are also interested in extracting SCCs from the parser's output, we eliminated from the two test corpora all sentences that do not contain verbs. Our experiments proceed in the following three steps:

[1]We use punctuation to refer to sentence-internal punctuation unless otherwise specified.

label	clause type	desired SCCs
S	gerundive	(NP)-GERUND
	small clause	NP-NP, (NP)-ADJP
	control	(NP)-INF-*to*
SBAR	control	(NP)-INF-*wh*-*to*
	with a complementizer	(NP)-S-*wh*, (NP)-S-*that*
	without a complementizer	(NP)-S-*that*

Table 1: SCCs for different clauses

1. Tag test data using the POS-tagger described in Ratnaparkhi (1996).

2. Parse the POS-tagged data using Bikel's parser.

3. Extract SCCs from the parser's output. The extractor we built first locates each verb in the parser's output and then identifies the syntactic categories of all its sisters and combines them into an SCC. However, there are cases where the extractor has more work to do.

   - Finite and Infinite Clauses: In the Penn Treebank, **S** and **SBAR** are used to label different types of clauses, obscuring too much detail about the internal structure of each clause. Our extractor is designed to identify the internal structure of different types of clause, as shown in Table 1.
   - Passive Structures: As noted above, Roland and Jurafsky (Roland and Jurafsky, 1998) have noticed that written language tends to have a much higher percentage of passive structures than spoken language. Our extractor is also designed to identify passive structures from the parser's output.

3 Experiment Results

3.1 Parsing and SCCs

We used EVALB measures Labeled Recall (LR) and Labeled Precision (LP) to compare the parsing performance of different models. To compare the accuracy of SCCs proposed from the parser's output, we calculated SCC Recall (SR) and SCC Precision (SP). SR and SP are defined as follows:

$$SR = \frac{\text{number of correct cues from the parser's output}}{\text{number of cues from treebank parse}} \quad (1)$$

WSJ		
model	LR/LP	SR/SP
punc	87.92%/88.29%	76.93%/77.70%
no-punc	86.25%/86.91%	76.96%/76.47%
punc-no-punc	82.31%/83.70%	74.62%/74.88%
Switchboard		
model	LR/LP	SR/SP
punc	83.14%/83.80%	79.04%/78.62%
no-punc	82.42%/83.74%	78.81%/78.37%
punc-no-punc	78.62%/80.68%	75.51%/75.02%

Table 2: Results of parsing and extraction of SCCs

$$SP = \frac{\text{number of correct cues from the parser's output}}{\text{number of cues from the parser's output}} \quad (2)$$

$$\text{SCC Balanced F-measure} = \frac{2 * SR * SP}{SR + SP} \quad (3)$$

The results for parsing WSJ and Switchboard and extracting SCCs are summarized in Table 2.

The LR/LP figures show the following trends:

1. Roark (2001) showed LR/LP of 86.4%/86.8% for punctuated written language, 83.4%/84.1% for unpunctuated written language. We achieve a higher accuracy in both punctuated and unpunctuated written language, and the decrease if punctuation is removed is less

2. For spoken language, Roark (2001) showed LR/LP of 85.2%/85.6% for punctuated spoken language, 84.0%/84.6% for unpunctuated spoken language. We achieve a lower accuracy in both punctuated and unpunctuated spoken language, and the decrease if punctuation is removed is less. The trends in (1) and (2) may be due to parser differences, or to the removal of sentences lacking verbs.

3. Unsurprisingly, if the test data is unpunctuated, but the models have been trained on punctuated language, performance decreases sharply.

In terms of the accuracy of extraction of SCCs, the results follow a similar pattern. However, the utility of punctuation turns out to be even smaller. Removing punctuation from both the training and test data results in a 0.8% drop in the accuracy of SCC extraction for written language and a 0.3% drop for spoken language.

Figure 1 exhibits the relation between the accuracy of parsing and that of extracting SCCs. If we consider WSJ and Switchboard individually, there seems to exist a positive correlation between the accuracy of parsing and that of extracting SCCs. In other words, higher LR/LP indicates

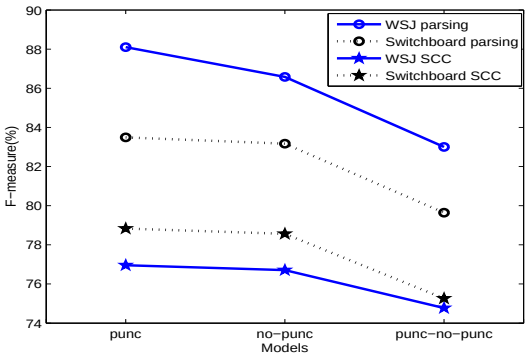


Figure 1: F-measure for parsing and extraction of SCCs

higher SR/SP. However, Figure 1 also shows that although the parser achieves a higher F-measure value for paring WSJ, it achieves a higher F-measure value for generating SCCs from Switchboard.

The fact that the parser achieves a higher accuracy of extracting SCCs from Switchboard than WSJ merits further discussion. Intuitively, it seems to be true that the shorter an SCC is, the more likely that the parser is to get it right. This intuition is confirmed by the data shown in Figure 2. Figure 2 plots the accuracy level of extracting SCCs by SCC's length. It is clear from Figure 2 that as SCCs get longer, the F-measure value drops progressively for both WSJ and Switchboard. Again, Roland and Jurafsky (1998) have suggested that one major subcategorization difference between written and spoken corpora is that spoken corpora have a much higher percentage of the zero-anaphora construction. We then examined the distribution of SCCs of different length in WSJ and Switchboard. Figure 3 shows that SCCs of length 0[2] account for a much higher percentage in Switchboard than WSJ, but it is always the other way around for SCCs of non-zero length. This observation led us to believe that the better performance that Bikel's parser achieves in extracting SCCs from Switchboard may be attributed to the following two factors:

1. Switchboard has a much higher percentage of SCCs of length 0.

2. The parser is very accurate in extracting shorter SCCs.

[2]Verbs have a length-0 SCC if they are intransitive and have no modifiers.

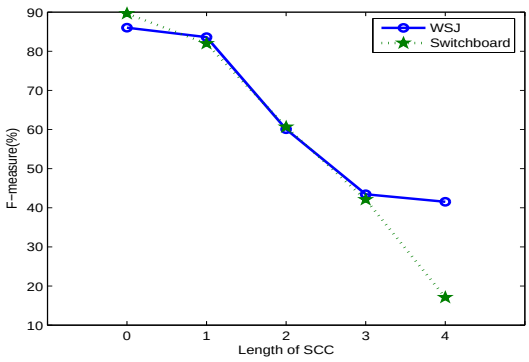


Figure 2: F-measure for SCCs of different length

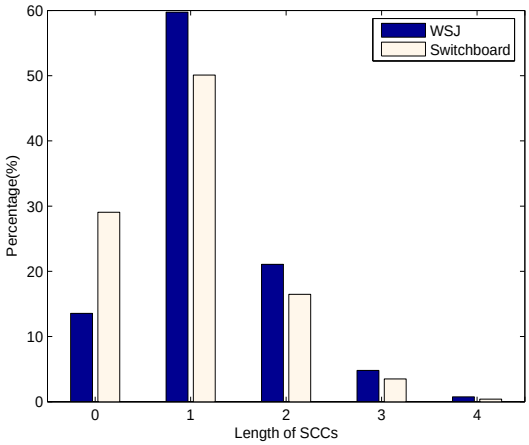


Figure 3: Distribution of SCCs by length

3.2 Extraction of Dependents

In order to estimate the effects of SCCs of length 0, we examined the parser's performance in retrieving dependents of verbs. Every constituent (whether an argument or adjunct) in an SCC generated by the parser is considered a dependent of that verb. SCCs of length 0 will be discounted because verbs that do not take any arguments or adjuncts have no dependents[3]. In addition, this way of evaluating the extraction of SCCs also matches the practice in some NLP tasks such as semantic role labeling (Xue and Palmer, 2004). For the task of semantic role labeling, the total number of dependents correctly retrieved from the parser's output affects the accuracy level of the task.

To do this, we calculated the number of dependents shared by between each SCC proposed from the parser's output and its corresponding SCC proposed from Penn Treebank. We based our calculation on a modified version of Minimum Edit Distance Algorithm. Our algorithm works by creating a shared-dependents matrix with one column for each constituent in the target sequence (SCCs proposed from Penn Treebank) and one row for each constituent in the source sequence (SCCs proposed from the parser's output). Each cell shared-dependent[*i*,*j*] contains the number of constituents shared between the first *i* constituents of the target sequence and the first *j* constituents of the source sequence. Each cell can then be computed as a simple function of the three possible paths through the matrix that arrive there. The algorithm is illustrated in Table 3.

[3]We are aware that subjects are typically also considered dependents, but we did not include subjects in our experiments

shared-dependents[*i*,*j*] = MAX(
shared-dependents[*i-1*,*j*],
shared-dependents[*i-1*,*j-1*]+1 if target[*i*] = source[*j*],
shared-dependents[*i-1*,*j-1*] if target[*i*] != source[*j*],
shared-dependents[*i*,*j-1*])

Table 3: The algorithm for computing shared dependents

INF	#5	1	1	2	**3**
ADVP	#4	1	1	2	2
PP-*in*	#3	1	1	2	2
NP	#2	1	1	1	1
NP	#1	1	1	1	1
	#0	#1	#2	#3	#4
		NP	S-*that*	PP-*in*	INF

Table 4: An example of computing the number of shared dependents

Table 4 shows an example of how the algorithm works with NP-S-*that*-PP-*in*-INF as the target sequence and NP-NP-PP-*in*-ADVP-INF as the source sequence. The algorithm returns 3 as the number of dependents shared by two SCCs.

We compared the performance of Bikel's parser in retrieving dependents from written and spoken language over all three models using Dependency Recall (DR) and Dependency Precision (DP). These metrics are defined as follows:

$$DR = \frac{\text{number of correct dependents from parser's output}}{\text{number of dependents from treebank parse}} \quad (4)$$

$$DP = \frac{\text{number of correct dependents from parser's output}}{\text{number of dependents from parser's output}} \quad (5)$$

$$\text{Dependency F-measure} = \frac{2 * DR * DP}{DR + DP} \quad (6)$$

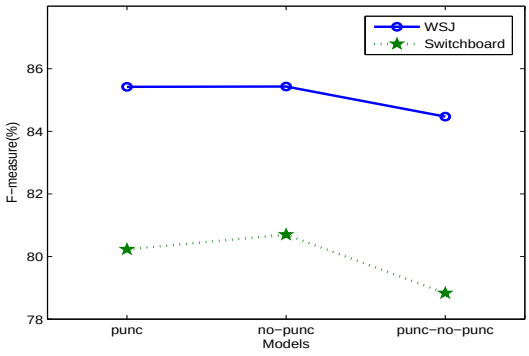


Figure 4: F-measure for extracting dependents

The results of Bikel's parser in retrieving dependents are summarized in Figure 4. Overall, the parser achieves a better performance for WSJ over all three models, just the opposite of what have been observed for SCC extraction. Interestingly, removing punctuation from both the training and test data actually slightly improves the F-measure. This holds true for both WSJ and Switchboard. This Dependency F-measure differs in detail from similar measures in Xue and Palmer (2004). For present purposes all that matters is the relative value for WSJ and Switchboard.

4 Extraction of SCFs from Spoken Language

Our experiments indicate that the SCCs generated by the parser from spoken language are as accurate as those generated from written texts. Hence, we would expect that the current technology for extracting SCFs, initially designed for written texts, should work equally well for spoken language. We previously built a system for automatically extracting SCFs from spoken BNC, and reported accuracy comparable to previous systems that work with only written texts (Li and Brew, 2005). However, Korhonen (2002) has shown that a direct comparison of different systems is very difficult to interpret because of the variations in the number of targeted SCFs, test verbs, gold standards and in the size of the test data. For this reason, we apply our SCF acquisition system separately to a written and spoken corpus of similar size from BNC and compare the accuracy of acquired SCF sets.

4.1 Overview

As noted above, previous studies on automatic extraction of SCFs from corpora usually proceed in two steps and we adopt this approach.

1. Hypothesis Generation: Identify all SCCs from the corpus data.

2. Hypothesis Selection: Determine which SCC is a valid SCF for a particular verb.

4.2 SCF Extraction System

We briefly outline our SCF extraction system for automatically extracting SCFs from corpora, which was based on the design proposed in Briscoe and Carroll (1997).

1. **A Statistical Parser**: Bikel's parser is used to parse input sentences.

2. **An SCF Extractor**: An extractor is use to extract SCCs from the parser's output.

3. **An English Lemmatizer**: MORPHA (Minnen et al., 2000) is used to lemmatize each verb.

4. **An SCF Evaluator**: An evaluator is used to filter out false SCCs based on their likelihood.

An SCC generated by the parser and extractor may be a correct SCC, or it may contain an adjunct, or it may simply be wrong due to tagging or parsing errors. We therefore need an SCF evaluator capable of filtering out false cues. Our evaluator has two parts: the Binomial Hypothesis Test (Brent, 1993) and a back-off algorithm (Sarkar and Zeman, 2000).

1. **The Binomial Hypothesis Test** (BHT): Let *p* be the probability that an scf_i occurs with $verb_j$ that is not supposed to take scf_i. If a verb occurs *n* times and *m* of those times it co-occurs with scf_i, then the scf_i cues are false cues is estimated by the summation of the binomial distribution for $m \leq k \leq n$:

$$P(m^+, n, p) = \sum_{k=m}^{n} \frac{n!}{k!(n-k)!} p^k (1-p)^{(n-k)} \quad (7)$$

If the value of $P(m^+, n, p)$ is less than or equal to a small threshold value, then the null hypothesis that $verb_j$ does not take scf_i is extremely unlikely to be true. Hence, scf_i is very likely to be a valid SCF for $verb_j$. The

SCCs	SCFs
NP-PP-*before*	
NP-S-*when*	NP
NP-PP-*at*-S-*before*	
NP-PP-*to*-S-*when*	
NP-PP-*to*-PP-*at*	NP-PP-*to*
NP-PP-*to*-S-*because*-ADVP	

Table 5: SCCs and correct SCFs for *introduce*

corpus	WC	SC
number of verb tokens	115,524	109,678
number of verb types	5,234	4,789
verb types seen more than 10 times	1,102	998
number of acquired SCFs	2,688	1,984
average number of SCFs per verb	2.43	1.99

Table 6: Training data for WC and SC

value of m and n can be directly computed from the extractor's output, but the value of p is not easy to obtain. Following Manning (1993), we empirically determined the value of p. It was between 0.005 to 0.4 depending on the likelihood of an SCC being a valid SCF.

2. **Back-off Algorithm**: Many SCCs generated by the parser and extractor tend to contain some adjuncts. However, for many SCCs, one of its subsets is likely to be the correct SCF. Table 5 shows some SCCs generated by the extractor and the corresponding SCFs.

   The Back-off Algorithm always starts with the longest SCC for each verb. Assume that this SCC fails the BHT. The evaluator then eliminates the last constituent from the rejected cue, transfers its frequency to its successor and submits the successor to the BHT again. In this way, frequency can accumulate and more valid frames survive the BHT.

4.3 Results and Discussion

We evaluated our SCF extraction system on written and spoken BNC. We chose one million word written corpus (WC) and a comparable spoken corpus (SC) from BNC. Table 6 provides relevant information on the two corpora. We only keep the verbs that occur at least 10 times in our training data.

To compare the performance of our system on WC and SC, we calculated the type precision, type recall and F-measure. Type precision is the percentage of SCF types that our system proposes which are correct according some gold standard and type recall is the percentage of correct SCF types proposed by our system that are listed in the gold standard. We used the 14 verbs [4] selected by Briscoe and Carroll (1997) and evaluated our results of these verbs against the SCF entries in two gold standards: COMLEX (Grishman et al., 1994) and a manually constructed SCF set from the training data. It makes sense to use a manually constructed SCF set while calculating type precision and recall because some of the SCFs in a syntax dictionary such as COMLEX might not occur in the training data at all. We constructed separate SCF sets for the written and spoken BNC.

gold standard	COMLEX		Manually Constructed	
corpus	WC	SC	WC	SC
type precision	93.1%	92.9%	93.1%	92.9%
type recall	49.2%	47.7%	56.5%	57.6%
F-measure	64.4%	63.1%	70.3%	71.1%

Table 7: Type precision and recall and F-measure

The results are summarized in Table 7. As shown in Table 7, the accuracy achieved for WC and SC are very comparable: Our system achieves a slightly better result for WC when using COMLEX as the gold standard and for SC when using manually constructed SCF set as gold standard, suggesting that it is feasible to apply the current technology for automatically extracting SCFs to spoken language.

5 Conclusions and Future Work

5.1 Use of Parser's Output

In this paper, we have shown that it is not necessarily true that statistical parsers always perform worse when dealing with spoken language. The conventional accuracy metrics for parsing (LR/LP) should not be taken as the only metrics in determining the feasibility of applying statistical parsers to spoken language. It is necessary to consider what information we want to extract out of parsers' output and make use of.

1. Extraction of SCFs from Corpora: This task takes SCCs generated by the parser and extractor as input. Our experiments show that

[4] The 14 verbs used in Briscoe and Carroll (1997) are *ask, begin, believe, cause, expect, find, give, help, like, move, produce, provide, seem* and *sway*. We replaced *sway* with *show* because *sway* occurs less than 10 times in our training data.

the SCCs generated for spoken language are as accurate as those generated for written language. We have also shown that it is feasible to apply the current SCF extraction technology to spoken language.

2. Semantic Role Labeling: This task usually operates on parsers' output and the number of dependents of each verb that are correctly retrieved by the parser clearly affects the accuracy of the task. Our experiments show that the parser achieves a much lower accuracy in retrieving dependents from the spoken language than written language. This seems to suggest that a lower accuracy is likely to be achieved for a semantic role labeling task performed on spoken language. We are not aware that this has yet been tried.

5.2 Punctuation and Speech Transcription Practice

Both our experiments and Roark's experiments show that parsing accuracy measured by LR/LP experiences a sharper decrease for WSJ than Switchboard after we removed punctuation from training and test data. In spoken language, commas are largely used to delimit disfluency elements. As noted in Engel et al. (2002), statistical parsers usually condition the probability of a constituent on the types of its neighboring constituents. The way that commas are used in speech transcription seems to have the effect of increasing the range of neighboring constituents, thus fragmenting the data and making it less reliable. On the other hand, in written texts, commas serve as more reliable cues for parsers to identify phrasal and clausal boundaries.

In addition, our experiment demonstrates that punctuation does not help much with extraction of SCCs from spoken language. Removing punctuation from both the training and test data results in rougly a 0.3% decrease in SR/SP. Furthermore, removing punctuation from both training and test data actually slightly improves the performance of Bikel's parser in retrieving dependents from spoken language. All these results seem to suggest that adding punctuation in speech transcription is of little help to statistical parsers including at least three state-of-the-art statistical parsers (Collins, 1999; Charniak, 2000; Bikel, 2004). As a result, there may be other good reasons why someone who wants to build a Switchboard-like corpus should choose to provide punctuation, but there is no need to do so simply in order to help parsers.

However, segmenting utterances into individual units is necessary because statistical parsers require sentence boundaries to be clearly delimited. Current statistical parsers are unable to handle an input string consisting of two sentences. For example, when presented with an input string as in (1) and (2), if the two sentences are separated by a period (1), Bikel's parser wrongly treats the second sentence as a sentential complement of the main verb *like* in the first sentence. As a result, the extractor generates an SCC NP-S for *like*, which is incorrect. The parser returns the same parse after we removed the period (2) and let the parser parse it again.

(1) I like the long hair. It was back in high school.

(2) I like the long hair It was back in high school.

Hence, while adding punctuation in transcribing a Switchboard-like corpus is not of much help to statistical parsers, segmenting utterances into individual units is crucial for statistical parsers. In future work, we plan to develop a system capable of automatically segmenting speech utterances into individual units.

6 Acknowledgments

This study was supported by NSF grant 0347799. Our thanks go to Eric Fosler-Lussier, Mike White and three anonymous reviewers for their valuable comments.

References

D. Bikel. 2004. Intricacies of Collin's parsing models. *Computational Linguistics*, 30(2):479–511.

M. Brent. 1993. From grammar to lexicon: Unsupervised learning of lexical syntax. *Computational Linguistics*, 19(3):243–262.

T. Briscoe and J. Carroll. 1997. Automatic extraction of subcategorization from corpora. In *Proceedings of the 5th ACL Conference on Applied Natural Language Processing*, pages 356–363.

E. Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 2000 Conference of the North American Chapter of the Association for Computation Linguistics*, pages 132–139.

M. Collins. 1999. *Head-driven statistical models for natural language parsing*. Ph.D. thesis, University of Pennsylvania.

D. Engel, E. Charniak, and M. Johnson. 2002. Parsing and disfluency placement. In *Proceedings of 2002 Conference on Empirical Methods of Natural Language Processing*, pages 49–54.

J. Godefrey, E. Holliman, and J. McDaniel. 1992. SWITCHBOARD: Telephone speech corpus for research and development. In *Proceedings of ICASSP-92*, pages 517–520.

R. Grishman, C. Macleod, and A. Meryers. 1994. Comlex syntax: Building a computational lexicon. In *Proceedings of the 1994 International Conference of Computational Linguistics*, pages 268–272.

A. Korhonen. 2002. *Subcategorization Acquisition*. Ph.D. thesis, Cambridge University.

M. Lapata and C. Brew. 2004. Verb class disambiguation using informative priors. *Computational Linguistics*, 30(1):45–73.

J. Li and C. Brew. 2005. Automatic extraction of subcategorization frames from spoken corpora. In *Proceedings of the Interdisciplinary Workshop on the Identification and Representation of Verb Features and Verb Classes*, Saarbracken, Germany.

C. Manning. 1993. Automatic extraction of a large subcategorization dictionary from corpora. In *Proceedings of 31st Annual Meeting of the Association for Computational Linguistics*, pages 235–242.

M. Marcus, G. Kim, and M. Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

P. Merlo and S. Stevenson. 2001. Automatic verb classification based on statistical distribution of argument structure. *Computational Linguistics*, 27(3):373–408.

P. Merlo, E. Joanis, and J. Henderson. 2005. Unsupervised verb class disambiguation based on diathesis alternations. manuscripts.

G. Minnen, J. Carroll, and D. Pearce. 2000. Applied morphological processing of English. *Natural Language Engineering*, 7(3):207–223.

V. Punyakanok, D. Roth, and W. Yih. 2005. The necessity of syntactic parsing for semantic role labeling. In *Proceedings of the 2nd Midwest Computational Linguistics Colloquium*, pages 15–22.

A. Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the Conference on Empirical Methods of Natural Language Processing*, pages 133–142.

B. Roark. 2001. *Robust Probabilistic Predictive Processing: Motivation, Models, and Applications*. Ph.D. thesis, Brown University.

D. Roland and D. Jurafsky. 1998. How verb subcategorization frequency is affected by the corpus choice. In *Proceedings of the 17th International Conference on Computational Linguistics*, pages 1122–1128.

A. Sarkar and D. Zeman. 2000. Automatic extraction of subcategorization frames for Czech. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 691–697.

S. Schulte im Walde. 2000. Clustering verbs semantically according to alternation behavior. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 747–753.

N. Xue and M. Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of 2004 Conference on Empirical Methods in Natural Language Processing*, pages 88–94.

The Role of Information Retrieval in Answering Complex Questions

Jimmy Lin
College of Information Studies
Department of Computer Science
Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742, USA
jimmylin@umd.edu

Abstract

This paper explores the role of information retrieval in answering "relationship" questions, a new class complex information needs formally introduced in TREC 2005. Since information retrieval is often an integral component of many question answering strategies, it is important to understand the impact of different term-based techniques. Within a framework of sentence retrieval, we examine three factors that contribute to question answering performance: the use of different retrieval engines, relevance (both at the document and sentence level), and redundancy. Results point out the limitations of purely term-based methods to this challenging task. Nevertheless, IR-based techniques provide a strong baseline on top of which more sophisticated language processing techniques can be deployed.

1 Introduction

The field of question answering arose from the recognition that the document does not occupy a privileged position in the space of information objects as the most ideal unit of retrieval. Indeed, for certain types of information needs, sub-document segments are preferred—an example is answers to factoid questions such as "Who won the Nobel Prize for literature in 1972?" By leveraging sophisticated language processing capabilities, factoid question answering systems are able to pinpoint the exact span of text that directly satisfies an information need.

Nevertheless, IR engines remain integral components of question answering systems, primarily as a source of candidate documents that are subsequently analyzed in greater detail. Although this two-stage architecture was initially conceived as an expedient to overcome the computational processing bottleneck associated with more sophisticated but slower language processing technology, it has worked quite well in practice. The architecture has since evolved into a widely-accepted paradigm for building working systems (Hirschman and Gaizauskas, 2001).

Due to the reliance of QA systems on IR technology, the relationship between them is an important area of study. For example, how sensitive is answer extraction performance to the initial quality of the result set? Does better document retrieval necessarily translate into more accurate answer extraction? These answers cannot be solely determined from first principles, but must be addressed through empirical experiments. Indeed, a number of works have specifically examined the effects of information retrieval on question answering (Monz, 2003; Tellex et al., 2003), including a dedicated workshop at SIGIR 2004 (Gaizauskas et al., 2004). More recently, the importance of document retrieval has prompted NIST to introduce a document ranking subtask inside the TREC 2005 QA track.

However, the connection between QA and IR has mostly been explored in the context of factoid questions such as "Who shot Abraham Lincoln?", which represent only a small fraction of all information needs. In contrast to factoid questions, which can be answered by short phrases found within an individual document, there is a large class of questions whose answers require synthesis of information from multiple sources. The so-called definition/other questions at recent TREC evaluations (Voorhees, 2005) serve as good examples: "good answers" to these questions include in-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 523–530,
Sydney, July 2006.

Qid 25: The analyst is interested in the status of Fidel Castro's brother. Specifically, the analyst would like information on his current plans and what role he may play after Fidel Castro's death.

vital	Raul Castro was formally designated his brother's successor
vital	Raul is the head of the Armed Forces
okay	Raul is five years younger than Castro
okay	Raul has enjoyed a more public role in running Cuba's Government.
okay	Raul is the number two man in the government's ruling Council of State

Figure 1: An example relationship question from TREC 2005 with its answer nuggets.

teresting "nuggets" about a particular person, organization, entity, or event. No single document can provide a complete answer, and hence systems must integrate information from multiple sources; cf. (Amigó et al., 2004; Dang, 2005).

This work focuses on so-called relationship questions, which represent a new and underexplored area in question answering. Although they require systems to extract information nuggets from multiple documents (just like definition/other questions), relationship questions demand a different approach (see Section 2). This paper explores the role of information retrieval in answering such questions, focusing primarily on three aspects: document retrieval performance, term-based measures of relevance, and term-based approaches to reducing redundancy. The overall goal is to push the limits of information retrieval technology and provide strong baselines against which linguistic processing capabilities can be compared.

The rest of this paper is organized as follows: Section 2 provides an overview of relationship questions. Section 3 describes experiments focused on document retrieval performance. An approach to answering relationship questions based on sentence retrieval is discussed in Section 4. A simple utility model that incorporates both relevance and redundancy is explored in Section 5. Before concluding, we discuss the implications of our experimental results in Section 6.

2 Relationship Questions

Relationship questions represent a new class of information needs formally introduced as a subtask in the NIST-sponsored TREC QA evaluations in 2005 (Voorhees, 2005). Previously, they were the focus of a small pilot study within the AQUAINT program, which resulted in an understanding of a "relationship" as the ability for one object to influence another. Objects in these questions can denote either entities (people, organization, countries, etc.) or events. Consider the following examples:

- Has pressure from China affected America's willingness to sell weaponry to Taiwan?
- Do the military personnel exchanges between Israel and India show an increase in cooperation? If so, what are the driving factors behind this increase?

Evidence for a relationship includes both the means to influence some entity and the motivation for doing so. Eight types of relationships ("spheres of influence") were noted: financial, movement of goods, family ties, co-location, common interest, and temporal connection.

Relationship questions are significantly different from definition questions, which can be paraphrased as "Tell me interesting things about *x*." Definition questions have received significant amounts of attention recently, e.g., (Hildebrandt et al., 2004; Prager et al., 2004; Xu et al., 2004; Cui et al., 2005). Research has shown that certain cue phrases serve as strong indicators for nuggets, and thus an approach based on matching surface patterns (e.g., appositives, parenthetical expressions) works quite well. Unfortunately, such techniques do not generalize to relationship questions because their answers are not usually captured by patterns or marked by surface cues.

Unlike answers to factoid questions, answers to relationship questions consist of an unsorted set of passages. For assessing system output, NIST employs the nugget-based evaluation methodology originally developed for definition questions; see (Voorhees, 2005) for a detailed description. Answers consist of units of information called "nuggets", which assessors manually create from system submissions and their own research (see example in Figure 1). Nuggets are divided into

two types ("vital" and "okay"), and this distinction plays an important role in scoring. The official metric is an F_3-score, where nugget recall is computed on vital nuggets, and precision is based on a length allowance derived from the number of both vital and okay nuggets retrieved.

In the original NIST setup, human assessors were required to manually determine whether a particular system's response contained a nugget. This posed a problem for researchers who wished to conduct formative evaluations outside the annual TREC cycle—the necessity of human involvement meant that system responses could not be rapidly, consistently, and automatically assessed. However, the recent introduction of POURPRE, an automatic evaluation metric for the nugget-based evaluation methodology (Lin and Demner-Fushman, 2005), fills this evaluation gap and makes possible the work reported here; cf. Nuggeteer (Marton and Radul, 2006).

This paper describes experiments with the 25 relationship questions used in the secondary task of the TREC 2005 QA track (Voorhees, 2005), which attracted a total of eleven submissions. Systems used the AQUAINT corpus, a three gigabyte collection of approximately one million news articles from the Associated Press, the New York Times, and the Xinhua News Agency.

3 Document Retrieval

Since information retrieval systems supply the initial set of documents on which a question answering system operates, it makes sense to optimize document retrieval performance in isolation. The issue of end–to–end system performance will be taken up in Section 4.

Retrieval performance can be evaluated based on the assumption that documents which contain one or more relevant nuggets (either vital or okay) are themselves relevant. From system submissions to TREC 2005, we created a set of relevance judgments, which averaged 8.96 relevant documents per question (median 7, min 1, max 21).

Our first goal was to examine the effect of different retrieval systems on performance. Two freely-available IR engines were compared: Lucene and Indri. The former is an open-source implementation of what amounts to be a modified *tf.idf* weighting scheme, while the latter employs a language modeling approach. In addition, we experimented with blind relevance feedback, a retrieval technique commonly employed to improve performance (Salton and Buckley, 1990). Following settings in typical IR experiments, the top twenty terms (by *tf.idf* value) from the top twenty documents were added to the original query in the feedback iteration.

	MAP	**R50**
Lucene	0.206	0.469
Lucene+brf	0.190 (-7.6%)$^{\circ}$	0.442 (-5.6%)$^{\circ}$
Indri	0.195 (-5.2%)$^{\circ}$	0.442 (-5.6%)$^{\circ}$
Indri+brf	0.158 (-23.3%)$^{\triangledown}$	0.377 (-19.5%)$^{\triangledown}$

Table 1: Document retrieval performance, with and without blind relevance feedback.

For each question, fifty documents from the AQUAINT collection were retrieved, representing the number of documents that a typical QA system might consider. The question itself was used verbatim as the IR query (see Section 6 for discussion). Performance is shown in Table 1. We measured Mean Average Precision (MAP), the most informative single-point metric for ranked retrieval, and recall, since it places an upper bound on the number of relevant documents available for subsequent downstream processing.

For all experiments reported in this paper, we applied the Wilcoxon signed-rank test to determine the statistical significance of the results. This test is commonly used in information retrieval research because it makes minimal assumptions about the underlying distribution of differences. Significance at the 0.90 level is denoted with a $^{\wedge}$ or $^{\vee}$, depending on the direction of change; at the 0.95 level, $^{\triangle}$ or $^{\triangledown}$; at the 0.99 level, $^{\blacktriangle}$ or $^{\blacktriangledown}$. Differences not statistically significant are marked with $^{\circ}$. Although the differences between Lucene and Indri are not significant, blind relevance feedback was found to hurt performance, significantly so in the case of Indri. These results are consistent with the findings of Monz (2003), who made the same observation in the factoid QA task.

There are a few caveats to consider when interpreting these results. First, the test set of 25 questions is rather small. Second, the number of relevant documents per question is also relatively small, and hence likely to be incomplete. Buckley and Voorhees (2004) have shown that evaluation metrics are not stable with respect to incomplete relevance judgments. Third, the distribution of relevant documents may be biased due to the small number of submissions, many of which used

Lucene. Due to these factors, one should interpret the results reported here as suggestive, not definitive. Follow-up experiments with larger data sets are required to produce more conclusive results.

4 Selecting Relevant Sentences

We adopted an extractive approach to answering relationship questions that views the task as sentence retrieval, a conception in line with the thinking of many researchers today (but see discussion in Section 6). Although oversimplified, there are several reasons why this formulation is productive: since answers consist of unordered text segments, the task is similar to passage retrieval, a well-studied problem (Callan, 1994; Tellex et al., 2003) where sentences form a natural unit of retrieval. In addition, the TREC novelty tracks have specifically tackled the questions of relevance and redundancy at the sentence level (Harman, 2002).

Empirically, a sentence retrieval approach performs quite well: when definition questions were first introduced in TREC 2003, a simple sentence-ranking algorithm outperformed all but the highest-scoring system (Voorhees, 2003). In addition, viewing the task of answering relationship questions as sentence retrieval allows one to leverage work in multi-document summarization, where extractive approaches have been extensively studied. This section examines the task of independently selecting the best sentences for inclusion in an answer; attempts to reduce redundancy will be discussed in the next section.

There are a number of term-based features associated with a candidate sentence that may contribute to its relevance. In general, such features can be divided into two types: properties of the document containing the sentence and properties of the sentence itself. Regarding the former type, two features come into play: the relevance score of the document (from the IR engine) and its rank in the result set. For sentence-based features, we experimented with the following:

- Passage match score, which sums the *idf* values of unique terms that appear in both the candidate sentence (S) and the question (Q):

$$\sum_{t \in S \cap Q} idf(t)$$

- Term *idf* precision and recall scores; cf. (Katz et al., 2005):

$$\mathcal{P} = \frac{\sum_{t \in S \cap Q} idf(t)}{\sum_{t \in A} idf(t)}, \mathcal{R} = \frac{\sum_{t \in S \cap Q} idf(t)}{\sum_{t \in Q} idf(t)}$$

- Length of the sentence (in non-whitespace characters).

Note that precision and recall values are bounded between zero and one, while the passage match score and the length of the sentence are both unbounded features.

Our baseline sentence retriever employed the passage match score to rank all sentences in the top *n* retrieved documents. By default, we used documents retrieved by Lucene, using the question verbatim as the query. To generate answers, the system selected sentences based on their scores until a hard length quota has been filled (trimming the final sentence if necessary). After experimenting with different values, we discovered that a document cutoff of ten yielded the highest performance in terms of POURPRE scores, i.e., all but the ten top-ranking documents were discarded.

In addition, we built a linear regression model that employed the above features to predict the nugget score of a sentence (the dependent variable). For the training samples, the nugget matching component within POURPRE was employed to compute the nugget score—this value quantified the "goodness" of a particular sentence in terms of nugget content.[1] Due to known issues with the vital/okay distinction (Hildebrandt et al., 2004), it was ignored for this computation; however, see (Lin and Demner-Fushman, 2006b) for recent attempts to address this issue.

When presented with a test question, the system ranked all sentences from the top ten retrieved documents using the regression model. Answers were generated by filling a quota of characters, just as in the baseline. Once again, no attempt was made to reduce redundancy.

We conducted a five-fold cross validation experiment using all sentences from the top 100 Lucene documents as training samples. After experimenting with different features, we discovered that a regression model with the following performed best: passage match score, document score, and sentence length. Surprisingly, adding

[1] Since the count variant of POURPRE achieved the highest correlation with official rankings, the nugget score is simply the highest fraction in terms of word overlap between the sentence and any of the reference nuggets.

Length	1000	2000	3000	4000	5000
F-Score					
baseline	0.275	0.268	0.255	0.234	0.225
regression	0.294 (+7.0%)°	0.268 (+0.0%)°	0.257 (+1.0%)°	0.240 (+2.5%)°	0.228 (+1.6%)°
Recall					
baseline	0.282	0.308	0.333	0.336	0.352
regression	0.302 (+7.2%)°	0.308 (+0.0%)°	0.336 (+0.8%)°	0.343 (+2.3%)°	0.358 (+1.7%)°
F-Score (all-vital)					
baseline	0.699	0.672	0.632	0.592	0.558
regression	0.722 (+3.3%)°	0.672 (+0.0%)°	0.632 (+0.0%)°	0.593 (+0.2%)°	0.554 (−0.7%)°
Recall (all-vital)					
baseline	0.723	0.774	0.816	0.834	0.856
regression	0.747 (+3.3%)°	0.774 (+0.0%)°	0.814 (−0.2%)°	0.834 (+0.0%)°	0.848 (−0.8%)°

Table 2: Question answering performance at different answer length cutoffs, as measured by POURPRE.

Length	1000	2000	3000	4000	5000
F-Score					
Lucene	0.275	0.268	0.255	0.234	0.225
Lucene+brf	0.278 (+1.3%)°	0.268 (+0.0%)°	0.251 (−1.6%)°	0.231 (−1.2%)°	0.215 (−4.3%)°
Indri	0.264 (−4.1%)°	0.260 (−2.7%)°	0.241 (−5.4%)°	0.222 (−5.0%)°	0.212 (−5.8%)°
Indri+brf	0.270 (−1.8%)°	0.257 (−3.8%)°	0.235 (−7.8%)°	0.221 (−5.7%)°	0.206 (−8.2%)°
Recall					
Lucene	0.282	0.308	0.333	0.336	0.352
Lucene+brf	0.285 (+1.3%)°	0.308 (+0.0%)°	0.319 (−4.2%)°	0.322 (−4.2%)°	0.324 (−7.9%)°
Indri	0.270 (−4.1%)°	0.300 (−2.5%)°	0.306 (−8.2%)°	0.308 (−8.1%)°	0.320 (−9.2%)°
Indri+brf	0.276 (−2.0%)°	0.296 (−3.6%)°	0.299 (−10.4%)°	0.307 (−8.5%)°	0.312 (−11.3%)°

Table 3: The effect of using different document retrieval systems on answer quality.

the term match precision and recall features to the regression model decreased overall performance slightly. We believe that precision and recall encodes information already captured by the other features.

Results of our experiments are shown in Table 2 for different answer lengths. Following the TREC QA track convention, all lengths are measured in non-whitespace characters. Both the baseline and regression conditions employed the top ten documents supplied by Lucene. In addition to the F_3-score, we report the recall component only (on vital nuggets). For this and all subsequent experiments, we used the (count, macro) variant of POURPRE, which was validated as producing the highest correlation with official rankings. The regression model yields higher scores at shorter lengths, although none of these differences were significant. In general, performance decreases with longer answers because both variants tend to rank relevant sentences before non-relevant ones.

Our results compare favorably to runs submitted to the TREC 2005 relationship task. In that evaluation, the best performing automatic run obtained a POURPRE score of 0.243, with an average answer length of 4051 character per question.

Since the vital/okay nugget distinction was ignored when training our regression model, we also evaluated system output under the assumption that all nuggets were vital. These scores are also shown in Table 2. Once again, results show higher POURPRE scores for shorter answers, but these differences are not statistically significant. Why might this be so? It appears that features based on term statistics alone are insufficient to capture nugget relevance. We verified this hypothesis by building a regression model for all 25 questions: the model exhibited an R^2 value of only 0.207.

How does IR performance affect the final system output? To find out, we applied the baseline sentence retrieval algorithm (which uses the passage match score only) on the output of different document retrieval variants. These results are shown in Table 3 for the four conditions discussed in the previous section: Lucene and Indri, with and without blind relevance feedback.

Just as with the document retrieval results, Lucene alone (without blind relevance feedback) yielded the highest POURPRE scores. However, none of the differences observed were statistically significant. These numbers point to an interesting interaction between document retrieval and question answering. The decreases in performance at-

Length	1000	2000	3000	4000	5000
F-Score					
baseline	0.275	0.268	0.255	0.234	0.225
baseline+max	0.311 (+13.2%)$^{\wedge}$	0.302 (+12.8%)$^{\blacktriangle}$	0.281 (+10.5%)$^{\blacktriangle}$	0.256 (+9.5%)$^{\triangle}$	0.235 (+4.6%)$^{\circ}$
baseline+avg	0.301 (+9.6%)$^{\circ}$	0.294 (+9.8%)$^{\wedge}$	0.271 (+6.5%)$^{\wedge}$	0.256 (+9.5%)$^{\triangle}$	0.237 (+5.6%)$^{\circ}$
regression+max	0.275 (+0.3%)$^{\circ}$	0.303 (+13.3%)$^{\wedge}$	0.275 (+8.1%)$^{\circ}$	0.258 (+10.4%)$^{\circ}$	0.244 (+8.4%)$^{\circ}$
Recall					
baseline	0.282	0.308	0.333	0.336	0.352
baseline+max	0.324 (+15.1%)$^{\wedge}$	0.355 (+15.4%)$^{\triangle}$	0.369 (+10.6%)$^{\triangle}$	0.369 (+9.8%)$^{\triangle}$	0.369 (+4.7%)$^{\circ}$
baseline+avg	0.314 (+11.4%)$^{\circ}$	0.346 (+12.3%)$^{\wedge}$	0.354 (+6.2%)$^{\wedge}$	0.369 (+9.8%)$^{\triangle}$	0.371 (+5.5%)$^{\circ}$
regression+max	0.287 (+2.0%)$^{\circ}$	0.357 (+16.1%)$^{\wedge}$	0.360 (+8.0%)$^{\circ}$	0.371 (+10.4%)$^{\wedge}$	0.379 (+7.6%)$^{\circ}$

Table 4: Evaluation of different utility settings.

tributed to blind relevance feedback in end–to–end QA were in general *less* than the drops observed in the document retrieval runs. It appears possible that the sentence retrieval algorithm was able to recover from a lower-quality result set, i.e., one with relevant documents ranked lower. Nevertheless, just as with factoid QA, the coupling between IR and answer extraction merits further study.

5 Reducing Redundancy

The methods described in the previous section for choosing relevant sentences do not take into account information that may be conveyed more than once. Drawing inspiration from research in sentence-level redundancy within the context of the TREC novelty track (Allan et al., 2003) and work in multi-document summarization, we experimented with term-based approaches to reducing redundancy.

Instead of selecting sentences for inclusion in the answer based on relevance alone, we implemented a simple utility model, which takes into account sentences that have already been added to the answer A. For each candidate c, utility is defined as follows:

$$\text{Utility}(c) = \text{Relevance}(c) - \lambda \max_{s \in A} sim(s, c)$$

This model is the baseline variant of the Maximal Marginal Relevance method for summarization (Goldstein et al., 2000). Each candidate is compared to all sentences that have already been selected for inclusion in the answer. The maximum of these pairwise similarity comparisons is deducted from the relevance score of the sentence, subjected to λ, a parameter that we tune. For our experiments, we used cosine distance as the similarity function. All relevance scores were normalized to a range between zero and one.

At each step in the answer generation process, utility values are computed for all candidate sentences. The one with the highest score is selected for inclusion in the final answer. Utility values are then recomputed, and the process iterates until the length quota has been filled.

We experimented with two different sources for the relevance scores: the baseline sentence retriever (passage match score only) and the regression model. In addition to taking the max of all pairwise similarity values, as in the above formula, we also experimented with the average.

Results of our runs are shown in Table 4. We report values for the baseline relevance score with the max and avg aggregation functions, as well as the regression relevance scores with max. These experimental conditions were compared against the baseline run that used the relevance score only (no redundancy penalty). To compute the optimal λ, we swept across the parameter space from zero to one in increments of a tenth. We determined the optimal value of λ by averaging POURPRE scores across all length intervals. For all three conditions, we discovered 0.4 to be the optimal value.

These experiments suggest that a simple term-based approach to reducing redundancy yields statistically significant gains in performance. This result is not surprising since similar techniques have proven effective in multi-document summarization. Empirically, we found that the max operator outperforms the avg operator in quantifying the degree of redundancy. The observation that performance improvements are more noticeable at shorter answer lengths confirms our intuitions. Redundancy is better tolerated in longer answers because a redundant nugget is less likely to "squeeze out" a relevant, novel nugget.

While it is productive to model the relationship task as sentence retrieval where independent decisions are made about sentence-level relevance,

this simplification fails to capture overlap in information content, and leads to redundant answers. We found that a simple term-based approach was effective in tackling this issue.

6 Discussion

Although this work represents the first formal study of relationship questions that we are aware of, by no means are we claiming a solution—we see this as merely the first step in addressing a complex problem. Nevertheless, information retrieval techniques lay the groundwork for systems aimed at answering complex questions. The methods described here will hopefully serve as a starting point for future work.

Relationship questions represent an important problem because they exemplify complex information needs, generally acknowledged as the future of QA research. Other types of complex needs include analytical questions such as "How close is Iran to acquiring nuclear weapons?", which are the focus of the AQUAINT program in the U.S., and opinion questions such as "How does the Chilean government view attempts at having Pinochet tried in Spanish Court?", which were explored in a 2005 pilot study also funded by AQUAINT. In 2006, there will be a dedicated task within the TREC QA track exploring complex questions within an interactive setting. Furthermore, we note the convergence of the QA and summarization communities, as demonstrated by the shift from generic to query-focused summaries starting with DUC 2005 (Dang, 2005). This development is also compatible with the conception of "distillation" in the current DARPA GALE program. All these trends point to same problem: how do we build advanced information systems to address complex information needs?

The value of this work lies in the generality of IR-based approaches. Sophisticated linguistic processing algorithms are typically unable to cope with the enormous quantities of text available. To render analysis more computationally tractable, researchers commonly employ IR techniques to reduce the amount of text under consideration. We believe that the techniques introduced in this paper are applicable to the different types of information needs discussed above.

While information retrieval techniques form a strong baseline for answering relationship questions, there are clear limitations of term-based approaches. Although we certainly did not experiment with every possible method, this work examined several common IR techniques (e.g., relevance feedback, different term-based features, etc.). In our regression experiments, we discovered that our feature set was unable to adequately capture sentence relevance. On the other hand, simple IR-based techniques appeared to work well at reducing redundancy, suggesting that determining content overlap is a simpler problem.

To answer relationship questions well, NLP technology must take over where IR techniques leave off. Yet, there are a number of challenges, the biggest of which is that question classification and named-entity recognition, which have worked well for factoid questions, are not applicable to relationship questions, since answer types are difficult to anticipate. For factoids, there exists a significant amount of work on question analysis—the results of which include important query terms and the expected answer type (e.g., person, organization, etc.). Relationship questions are more difficult to process: for one, they are often not phrased as direct *wh*-questions, but rather as indirect requests for information, statements of doubt, etc. Furthermore, since these complex questions cannot be answered by short noun phrases, existing answer type ontologies are not very useful. For our experiments, we decided to simply use the question verbatim as the query to the IR systems, but undoubtedly performance can be gained by better query formulation strategies. These are difficult challenges, but recent work on applying semantic models to QA (Narayanan and Harabagiu, 2004; Lin and Demner-Fushman, 2006a) provide a promising direction.

While our formulation of answering relationship questions as sentence retrieval is productive, it clearly has limitations. The assumption that information nuggets do not span sentence boundaries is false and neglects important work in anaphora resolution and discourse modeling. The current setup of the task, where answers consist of unordered strings, does not place any value on coherence and readability of the responses, which will be important if the answers are intended for human consumption. Clearly, there are ample opportunities here for NLP techniques to shine.

The other value of this work lies in its use of an automatic evaluation metric (POURPRE) for system development—the first instance in complex

QA that we are aware of. Prior to the introduction of this automatic scoring technique, studies such as this were difficult to conduct due to the necessity of involving humans in the evaluation process. POURPRE was developed to enable rapid exploration of the solution space, and experiments reported here demonstrate its usefulness in doing just that. Although automatic evaluation metrics are no stranger to other fields such as machine translation (e.g., BLEU) and document summarization (e.g., ROUGE, BE, etc.), this represents a new development in question answering research.

7 Conclusion

Although many findings in this paper are negative, the conclusions are positive for NLP researchers. An exploration of a variety of term-based approaches for answering relationship questions has demonstrated the impact of different techniques, but more importantly, this work highlights limitations of purely IR-based methods. With a strong baseline as a foundation, the door is wide open for the integration of natural language understanding techniques.

8 Acknowledgments

This work has been supported in part by DARPA contract HR0011-06-2-0001 (GALE). I would like to thank Esther and Kiri for their loving support.

References

J. Allan, C. Wade, and A. Bolivar. 2003. Retrieval and novelty detection at the sentence level. In *SIGIR 2003*.

E. Amigó, J. Gonzalo, V. Peinado, A. Peñas, and F. Verdejo. 2004. An empirical study of information synthesis task. In *ACL 2004*.

C. Buckley and E. Voorhees. 2004. Retrieval evaluation with incomplete information. In *SIGIR 2004*.

J. Callan. 1994. Passage-level evidence in document retrieval. In *SIGIR 1994*.

H. Cui, M.-Y. Kan, and T.-S. Chua. 2005. Generic soft pattern models for definitional question answering. In *SIGIR 2005*.

H. Dang. 2005. Overview of DUC 2005. In *DUC 2005*.

R. Gaizauskas, M. Hepple, and M. Greenwood. 2004. *Proceedings of the SIGIR 2004 Workshop on Information Retrieval for Question Answering (IR4QA)*.

J. Goldstein, V. Mittal, J. Carbonell, and J. Callan. 2000. Creating and evaluating multi-document sentence extract summaries. In *CIKM 2000*.

D. Harman. 2002. Overview of the TREC 2002 novelty track. In *TREC 2002*.

W. Hildebrandt, B. Katz, and J. Lin. 2004. Answering definition questions with multiple knowledge sources. In *HLT/NAACL 2004*.

L. Hirschman and R. Gaizauskas. 2001. Natural language question answering: The view from here. *Natural Language Engineering*, 7(4):275–300.

B. Katz, G. Marton, G. Borchardt, A. Brownell, S. Felshin, D. Loreto, J. Louis-Rosenberg, B. Lu, F. Mora, S. Stiller, O. Uzuner, and A. Wilcox. 2005. External knowledge sources for question answering. In *TREC 2005*.

J. Lin and D. Demner-Fushman. 2005. Automatically evaluating answers to definition questions. In *HLT/EMNLP 2005*.

J. Lin and D. Demner-Fushman. 2006a. The role of knowledge in conceptual retrieval: A study in the domain of clinical medicine. In *SIGIR 2006*.

J. Lin and D. Demner-Fushman. 2006b. Will pyramids built of nuggets topple over? In *HLT/NAACL 2006*.

G. Marton and A. Radul. 2006. Nuggeteer: Automatic nugget-based evaluation using descriptions and judgements. In *HLT/NAACL 2006*.

C. Monz. 2003. *From Document Retrieval to Question Answering*. Ph.D. thesis, Institute for Logic, Language, and Computation, University of Amsterdam.

S. Narayanan and S. Harabagiu. 2004. Question answering based on semantic structures. In *COLING 2004*.

J. Prager, J. Chu-Carroll, and K. Czuba. 2004. Question answering using constraint satisfaction: QA–by–Dossier–with–Constraints. In *ACL 2004*.

G. Salton and C. Buckley. 1990. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4):288–297.

S. Tellex, B. Katz, J. Lin, G. Marton, and A. Fernandes. 2003. Quantitative evaluation of passage retrieval algorithms for question answering. In *SIGIR 2003*.

E. Voorhees. 2003. Overview of the TREC 2003 question answering track. In *TREC 2003*.

E. Voorhees. 2005. Overview of the TREC 2005 question answering track. In *TREC 2005*.

J. Xu, R. Weischedel, and A. Licuanan. 2004. Evaluation of an extraction-based approach to answering definition questions. In *SIGIR 2004*.

Examining the Content Load of Part of Speech Blocks for Information Retrieval

Christina Lioma
Department of Computing Science
University of Glasgow
17 Lilybank Gardens
Scotland, U.K.
xristina@dcs.gla.ac.uk

Iadh Ounis
Department of Computing Science
University of Glasgow
17 Lilybank Gardens
Scotland, U.K.
ounis@dcs.gla.ac.uk

Abstract

We investigate the connection between part of speech (POS) distribution and content in language. We define POS blocks to be groups of parts of speech. We hypothesise that there exists a directly proportional relation between the frequency of POS blocks and their content salience. We also hypothesise that the class membership of the parts of speech within such blocks reflects the content load of the blocks, on the basis that open class parts of speech are more content-bearing than closed class parts of speech. We test these hypotheses in the context of Information Retrieval, by syntactically representing queries, and removing from them content-poor blocks, in line with the aforementioned hypotheses. For our first hypothesis, we induce POS distribution information from a corpus, and approximate the probability of occurrence of POS blocks as per two statistical estimators separately. For our second hypothesis, we use simple heuristics to estimate the content load within POS blocks. We use the Text REtrieval Conference (TREC) queries of 1999 and 2000 to retrieve documents from the WT2G and WT10G test collections, with five different retrieval strategies. Experimental outcomes confirm that our hypotheses hold in the context of Information Retrieval.

1 Introduction

The task of an Information Retrieval (IR) system is to retrieve documents from a collection, in response to a user need, which is expressed in the form of a query. Very often, this task is realised by indexing the documents in the collection with keyword descriptors. Retrieval consists in matching the query against the descriptors of the documents, and returning the ones that appear closest, in ranked lists of relevance (van Rijsbergen, 1979). Usually, the keywords that constitute the document descriptors are associated with individual weights, which capture the importance of the keywords to the content of the document. Such weights, commonly referred to as term weights, can be computed using various term weighting schemes. Not all words can be used as keyword descriptors. In fact, a relatively small number of words accounts for most of a document's content (van Rijsbergen, 1979). Function words make 'noisy' index terms, and are usually ignored during the retrieval process. This is practically realised with the use of stopword lists, which are lists of words to be exempted when indexing the collection and the queries.

The use of stopword lists in IR is a manifestation of a well-known bifurcation in linguistics between open and closed classes of words (Lyons, 1977). In brief, open class words are more content-bearing than closed class words. Generally, the open class contains parts of speech that are morphologically and semantically flexible, while the closed class contains words that primarily perform linguistic well-formedness functions. The membership of the closed class is mostly fixed and largely restricted to function words, which are not prone to semantic or morphological alterations.

We define a block of parts of speech (*POS block*) as a block of fixed length n, where n is set empirically. We define *POS block* tokens as individual instances of *POS blocks*, and *POS block*

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 531–538,
Sydney, July 2006.

types as distinct *POS blocks* in a corpus. The purpose of this paper is to test two hypotheses.

The intuition behind both of these hypotheses is that, just as individual words can be content-rich or content-poor, the same can hold for blocks of parts of speech. According to our first hypothesis, *POS blocks* can be categorized as content-rich or content-poor, on the basis of their distribution within a corpus. Specifically, we hypothesise that the more frequently a *POS block* occurs in language, the more content it is likely to bear. According to our second hypothesis, *POS blocks* can be categorized as content-rich or content-poor, on the basis of the part of speech class membership of their individual components. Specifically, we hypothesise that the more closed class components found in a *POS block*, the less content the block is likely to bear.

Both aforementioned hypotheses are evaluated in the context of IR as follows. We observe the distribution of *POS blocks* in a corpus. We create a list of *POS block* types with their respective probabilities of occurrence. As a first step, to test our first hypothesis, we remove the *POS blocks* with a low probability of occurrence from each query, on the assumption that these blocks are content-poor. The decision regarding the threshold θ of low probability of occurrence is realised empirically. As a second step, we further remove from each query *POS blocks* that contain less open class than closed class components, in order to test the validity of our second hypothesis, as an extension of the first hypothesis. We retrieve documents from two standard IR English test collections, namely WT2G and WT10G. Both of these collections are commonly used for retrieval effectiveness evaluations in the Text REtrieval Conference (TREC), and come with sets of queries and query relevance assessments[1]. Query relevance assessments are lists of relevant documents, given a query. We retrieve relevant documents using firstly the original queries, secondly the queries produced after step 1, and thirdly the queries produced after step 2. We use five statistically different term weighting schemes to match the query terms to the document keywords, in order to assess our hypotheses across a range of retrieval techniques. We associate improvement of retrieval performance with successful noise reduction in the queries. We assume noise reduction to reflect the correct identification of content-poor blocks, in line with our hypotheses.

Section 2 presents related studies in this field. Section 3 introduces our methodology. Section 4 presents the experimental settings used to test our hypotheses, and their evaluation outcomes. Section 5 provides our conclusions and remarks.

2 Related Studies

We examine the distribution of *POS blocks* in language. This is but one type of language distribution analysis that can be realised. One can also examine the distribution of character or word n-grams, e.g. Language Modeling (Croft and Lafferty, 2003), phrases (Church and Hanks, 1990; Lewis, 1992), and so on. In class-based n-gram modeling (Brown et al., 1992) for example, class-based n-grams are used to determine the probability of occurrence of a POS class, given its preceding classes, and the probability of a particular word, given its own POS class. Unlike the class-based n-gram model, we do not use *POS blocks* to make predictions. We estimate their probability of occurrence as blocks, not the individual probabilities of their components, motivated by the intuition that the more frequently a *POS block* occurs, the more content it bears. In the context of IR, efforts have been made to use syntactic information to enhance retrieval (Smeaton, 1999; Strzalkowski, 1996; Zukerman and Raskutti, 2002), but not by using POS block-based distribution representations.

3 Methodology

We present the steps realised in order to assess our hypotheses in the context of IR. Firstly, *POS blocks* with their respective frequencies are extracted from a corpus. The probability of occurrence of each *POS block* is statistically estimated. In order to test our first hypothesis, we remove from the query all but *POS blocks* of high probability of occurrence, on the assumption that the latter are content-rich. In order to test our second hypothesis, *POS blocks* that contain more closed class than open class tags are removed from the queries, on the assumption that these blocks are content-poor.

3.1 Inducing *POS blocks* from a corpus

We extract *POS blocks* from a corpus and estimate their probability of occurrence, as follows.

[1] http://trec.nist.gov/

The corpus is POS tagged. All lexical word forms are eliminated. Thus, sentences are constituted solely by sequences of POS tags. The following example illustrates this point.

> [Original sentence] Many of the proposals for directives and action programmes planned by the Commission have for some obscure reason never seen the light of day.
>
> [Tagged sentence] Many/JJ of/IN the/DT proposals/NNS for/IN directives/NNS and/CC action/NN programmes/NNS planned/VVN by/IN the/DT Commission/NP have/VHP for/IN some/DT obscure/JJ reason/NN never/RB seen/VVN the/DT light/NN of/IN day/NN
>
> [Tags-only sentence] JJ IN DT NNS IN NNS CC NN NNS VVN IN DT NP VHP IN DT JJ NN RB VVN DT NN IN NN

For each sentence in the corpus, all possible *POS blocks* are extracted. Thus, for a given sentence ABCDEFGH, where POS tags are denoted by single letters, and where POS block length n = 4, the *POS blocks* extracted are ABCD, BCDE, CDEF, and so on. The extracted *POS blocks* overlap. The order in which the *POS blocks* occur in the sentence is disregarded.

We statistically infer the probability of occurrence of each *POS block*, on the basis of the individual *POS block* frequencies counted in the corpus. Maximum Likelihood inference is eschewed, as it assigns the maximum possible likelihood to the *POS blocks* observed in the corpus, and no probability to unseen *POS blocks*. Instead, we employ statistical estimation that accounts for unseen *POS blocks*, namely Laplace and Good-Turing (Manning and Schutze, 1999).

3.2 Removing *POS blocks* from the queries

In order to test our first hypothesis, *POS blocks* of low probability of occurrence are removed from the queries. Specifically, we POS tag the queries, and remove the *POS blocks* that have a probability of occurrence below an empirical threshold θ. The following example illustrates this point.

> [Original query] A relevant document will focus on the causes of the lack of integration in a significant way; that is, the mere mention of immigration difficulties is not relevant. Documents that discuss immigration problems unrelated to Germany are also not relevant.
>
> [Tags-only query] DT JJ NN MD VV IN DT NNS IN DT NN IN NN IN DT JJ NN; WDT VBZ DT JJ NN IN NN NNS VBZ RB JJ. NNS WDT VVP NN NNS JJ TO NP VBP RB RB JJ
>
> [Query with high-probability *POS blocks*] DT NNS IN DT NN IN NN IN NN IN NN NNS
>
> [Resulting query] the causes of the lack of integration in mention of immigration difficulties

Some of the low-probability *POS blocks*, which are removed from the query in the above example, are DT JJ NN MD, JJ NN MD VV, NN MD VV IN, and so on. The resulting query contains fragments of the original query, assumed to be content-rich. In the context of the bag-of-words approach to IR investigated here, the grammatical well-formedness of the query is thus not an issue to be considered.

In order to test the second hypothesis, we remove from the queries *POS blocks* that contain less open class than closed class components. We propose a simple heuristic *Content Load* algorithm, to 'count' the presence of content within a *POS block*, on the premise that open class tags bear more content than closed class tags. The order of tags within a *POS block* is ignored. Figure 1 displays our *Content Load* algorithm.

After the n^{th} *POS block* component has been 'counted', if the *Content Load* is zero or more, we consider the *POS block* content-rich. If the

Figure 1: The *Content Load* algorithm

```
function CONTENT-LOAD(POSblock)
returns ContentLoad
INITIALISE-FOR-EACH-POSBLOCK(query)
for pos ← from 1 to POSblock-size do
if(current-tag = = OpenClass)
(ContentLoad)+ +
elseif(current-tag = = ClosedClass)
(ContentLoad)- -
end
return(ContentLoad)
```

Content Load is strictly less than zero, we consider the *POS block* content-poor. We assume an underlying equivalence of content in all open class parts of speech, which albeit being linguistically counter-intuitive, is shown to be effective when applied to IR (Section 4). The following example illustrates this point. In this example, *POS block* length $n = 4$.

> [Original query] A relevant document will focus on the causes of the lack of integration in a significant way; that is, the mere mention of immigration difficulties is not relevant. Documents that discuss immigration problems unrelated to Germany are also not relevant.
>
> [Tags-only query] DT JJ NN MD VV IN DT NNS IN DT NN IN NN IN DT JJ NN; WDT VBZ DT JJ NN IN NN NNS VBZ RB JJ. NNS WDT VVP NN NNS JJ TO NP VBP RB RB JJ
>
> [Query with high-probability POS blocks] DT NNS IN DT NN IN NN IN NN IN NN NNS
>
> [Content Load of *POS blocks*]
> DT NNS IN DT (-2), NN IN NN IN (0), NN IN NN NNS (+2)
>
> [Query with high-probability *POS blocks* of zero or positive Content Load] NN IN NN IN NN IN NN NNS
>
> [Resulting query] lack of integration in mention of immigration difficulties

4 Evaluation

We present the experiments realised to test the two hypotheses formulated in Section 1. Section 4.1 presents our experimental settings, and Section 4.2 our evaluation results.

4.1 Experimental Settings

We induce *POS blocks* from the English language component of the second release of the parallel Europarl corpus(75MB)[2]. We POS tag the corpus using the TreeTagger[3], which is a probabilistic POS tagger that uses the Penn TreeBank tagset (Marcus et al., 1993). Since we are solely interested in a POS analysis, we introduce a stage of tagset simplification, during which, any information on top of surface POS classification is lost (Table 1). Practically, this leads to 48 original TreeBank (TB) tag classes being narrowed down to 15 Reduced TreeBank (RTB) tag classes. Additionally, tag names are shortened into two-letter names, for reasons of computational efficiency. We consider the TBR tags JJ, FW, NN, and VB as open-class, and the remaining tags as closed class (Lyons, 1977). We extract 214,398,227 *POS block* tokens and 19,343 *POS block* types from the corpus.

Table 1: Correspondence between the TreeBank (TB) and Reduced TreeBank (RTB) tags.

TB	TBR
JJ, JJR, JJS	JJ
RB,RBR,RBS	RB
CD, LS	CD
CC	CC
DT, WDT, PDT	DT
FW	FW
MD, VB, VBD, VBG, VBN, VBP, VBZ, VH, VHD, VHG, VHN, VHP, VHZ	MD
NN, NNS, NP, NPS	NN
PP, WP, PP$, WP$, EX, WRB	PP
IN, TO	IN
POS	PO
RP	RP
SYM	SY
UH	UH
VV, VVD, VVG, VVN, VVP, VVZ	VB

We retrieve relevant documents from two standard TREC test collections, namely WT2G (2GB) and WT10G (10GB), from the 1999 and 2000 TREC Web tracks, respectively. We use the queries 401-450 from the ad-hoc task of the 1999 Web track, for the WT2G test collection, and the queries 451-500 from the ad-hoc task of the 2000 Web track, for the WT10G test collection, with their respective relevance assessments. Each query contains three fields, namely *title*, *description*, and *narrative*. The *title* contains keywords describing the information need. The *description* expands briefly on the information need. The *narrative* part consists of sentences denoting key concepts to be considered or ignored. We use all three

[2] http://people.csail.mit.edu/koehn/publications/europarl/

[3] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/

query fields to match query terms to document keyword descriptors, but extract *POS blocks* only from the narrative field of the queries. This choice is motivated by the two following reasons. Firstly, the *narrative* includes the longest sentences in the whole query. For our experiments, longer sentences provide better grounds upon which we can test our hypotheses, since the longer a sentence, the more *POS blocks* we can match within it. Secondly, the *narrative* field contains the most noise in the whole query. Especially when using bag-of-words term weighting, such as in our evaluation, information on what is not relevant to the query only introduces noise. Thus, we select the most noisy field of the query to test whether the application of our hypotheses indeed results in the reduction of noise.

During indexing, we remove stopwords, and stem the collections and the queries, using Porter's[4] stemming algorithm. We use the Terrier[5] IR platform, and apply five different weighting schemes to match query terms to document descriptors. In IR, term weighting schemes estimate the relevance $R(d, Q)$ of a document d for a query Q, as: $R(d, Q) = \sum_{t \in Q} qtw \cdot w(t, d)$, where t is a term in Q, qtw is the query term weight, and $w(t, d)$ is the weight of document d for term t. For example, we use the classical TF_IDF weighting scheme (Sparck-Jones, 1972; Robertson et al., 1995): $w(t, d) = tfn \cdot \log_2 \frac{N}{df+1}$, where tfn is the normalised term frequency in a document: $tfn = \frac{k_1 \cdot tf}{tf + k_1(1-b+b\frac{l}{avg\_l})}$; tf is the frequency of a term in a document; $k1$, and b are parameters; l and avg_l are the document length and the average document length in the collection, respectively; N is the number of documents in the collection; and df is the number of documents containing the term t. For all weighting schemes we use, $qtw = \frac{qtf}{qtf_{max}}$, where qtf is the query term frequency, and qtf_{max} is the maximum qtf among all query terms. We also use the well-established probabilistic BM25 weighting scheme (Robertson et al., 1995), and three distinct weighting schemes from the more recent Divergence From Randomness (DFR) framework (Amati, 2003), namely BB2, PL2, and DLH. Note that, even though we use three weighting schemes from the DFR framework, the said schemes are statistically different to one another. Also, DLH is the only parameter-free weighting scheme we use, as it computes all of the $w(t, d)$ variables automatically from the collection statistics.

We use the default values of all parameters, namely, for the TF_IDF and BM25 weighting schemes (Robertson et al., 1995), $k1 = 1.2$, $k3 = 1000$, and $b = 0.75$ for both test collections; while for the PL2 and BB2 term weighting schemes (Amati, 2003), $c = 4.80$ for the WT2G test collection, and $c = 5.58$ for the WT10G test collection. We use default values, instead of tuning the term weighting parameters, because our focus lies in testing our hypotheses, and not in optimising retrieval performance. If the said parameters are optimised, retrieval performance may be further improved. We measure the retrieval performance using the Mean Average Precision (MAP) measure (van Rijsbergen, 1979).

Throughout all experiments, we set *POS block* length at n = 4. We employ Good-Turing and Laplace smoothing, and set the threshold of high probability of occurrence empirically at θ = 0.01. We present all evaluation results in tables, the format of which is as follows: GT and LA indicate Good-Turing and Laplace respectively, and $\Delta\%$ denotes the % difference in MAP from the baseline. Statistically significant scores, as per the Wilcoxon test ($p < 0.05$), appear in boldface, while highest Δ percentages appear in italics.

4.2 Evaluation Results

Our retrieval baseline consists in testing the performance of each term weighting scheme, with each of the two test collections, using the original queries. We introduce two retrieval combinations on top of the baseline, which we call POS and POSC. The POS retrieval experiments, which relate to our first hypothesis, and the POSC retrieval experiments, which relate to our second hypothesis, are described in Section 4.2.1. Section 4.2.2 presents the assessment of our hypotheses using a performance-boosting retrieval technique, namely query expansion.

4.2.1 POS and POSC Retrieval Experiments

The aim of the POS and POSC experiments is to test our first and second hypotheses, respectively. Firstly, to test the first hypothesis, namely that there is a direct connection between the removal of low-frequency *POS blocks* from the queries and noise reduction in the queries, we remove all low-frequency *POS blocks* from the *narrative* field of

[4] http://snowball.tartarus.org/

[5] http://ir.dcs.gla.ac.uk/terrier/

the queries. Secondly, to test our second hypothesis as an extension of our first hypothesis, we refilter the queries used in the POS experiments by removing from them *POS blocks* that contain more closed class than open class tags. The processes involved in both hypotheses take place prior to the removal of stop words and stemming of the queries. Table 2 displays the relevant evaluation results.

Overall, the removal of low-probability *POS blocks* from the queries (*Hypothesis 1* section in Table 2) is associated with an improvement in retrieval performance over the baseline in most cases, which sometimes is statistically significant. This improvement is quite similar across the two statistical estimators. Moreover, two interesting patterns emerge. Firstly, the DFR weighting schemes seem to be divided, performance-wise, between the parametric BB2 and PL2, which are associated with the highest improvement in retrieval performance, and the non-parametric DLH, which is associated with the lowest improvement, or even deterioration in retrieval performance. This may indicate that the parameter used in BB2 and PL2 is not optimal, which would explain a low baseline, and thus a very high improvement over it. Secondly, when comparing the improvement in performance related to the WT2G and the WT10G test collections, we observe a more marked improvement in retrieval performance with WT2G than with WT10G.

The combination of our two hypotheses (*Hypotheses 1+2* section in Table 2) is associated with an improvement in retrieval performance over the baseline in most cases, which sometimes is statistically significant. This improvement is very similar across the two statistical estimators, namely Good-Turing and Laplace. When combining hypotheses 1+2, retrieval performance improves more than it did for hypothesis 1 only, for the WT2G test collection, which indicates that our second hypothesis might further reduce the amount of noise in the queries successfully. For the WT10G collection, we object similar results, with the exception of DLH. Generally, the improvement in performance associated to the WT2G test collection is more marked than the improvement associated to WT10G.

To recapitulate on the evaluation outcomes of our two hypotheses, we report an improvement in retrieval performance over the baseline for most, but not all cases, which is sometimes statistically significant. This may be indicative of successful noise reduction in the queries, as per our hypotheses. Also, the difference in the improvement in retrieval performance across the two test collections may suggest that data sparseness affects retrieval performance.

4.2.2 POS and POSC Retrieval Experiments with Query Expansion

Query expansion (QE) is a performance-boosting technique often used in IR, which consists in extracting the most relevant terms from the top retrieved documents, and in using these terms to expand the initial query. The expanded query is then used to retrieve documents anew. Query expansion has the distinct property of improving retrieval performance when queries do not contain noise, but harming retrieval performance when queries contain noise, furnishing us with a strong baseline, against which we can measure our hypotheses. We repeat the experiments described in Section 4.2.1 with query expansion.

We use the Bo1 query expansion scheme from the DFR framework (Amati, 2003). We optimise the query expansion settings, so as to maximise its performance. This provides us with an even stronger baseline, against which we can compare our proposed technique, which we tune empirically too through the tuning of the threshold θ. We optimise query expansion on the basis of the corresponding relevance assessments available for the queries and collections employed, by selecting the most relevant terms from the top retrieved documents. For the WT2G test collection, the relevant terms / top retrieved documents ratio we use is (i) 20/5 with TF_IDF, BM25, and DLH; (ii) 30/5 with PL2; and (iii) 10/5 with BB2. For the WT10G collection, the said ratio is (i) 10/5 for TF_IDF; (ii) 20/5 for BM25 and DLH; and (iii) 5/5 for PL2 and BB2.

We repeat our POS and POSC retrieval experiments with query expansion. Table 3 displays the relevant evaluation results.

Query expansion has overall improved retrieval performance (compare Tables 2 and 3), for both test collections, with two exceptions, where query expansion has made no difference at all, namely for BB2 and PL2, with the WT10G collection. The removal of low-probability *POS blocks* from the queries, as per our first hypothesis, combined with query expansion, is associated with an im-

Table 2: Mean Average Precision (MAP) scores of the POS and POSC experiments.

WT2G collection									
		Hypothesis 1				Hypotheses 1+2			
w(t,d)	base	POSGT	Δ%	POSLA	Δ%	POSCGT	Δ%	POSCLA	Δ%
TFIDF	0.276	0.295	+6.8	0.293	+6.1	0.298	+8.0	0.294	+6.4
BM25	0.280	0.294	+4.8	0.292	+4.1	0.297	+5.9	0.293	+4.5
BB2	0.237	0.291	***+22.8***	0.287	*+21.0*	0.295	***+24.2***	0.288	*+21.5*
PL2	0.268	0.298	+11.2	0.297	+10.9	0.306	+14.1	0.302	+12.8
DLH	0.237	0.239	+0.7	0.238	+0.4	0.243	+2.3	0.241	+1.6
WT10G collection									
		Hypothesis 1				Hypotheses 1+2			
w(t,d)	base	POSGT	Δ%	POSLA	Δ%	POSCGT	Δ%	POSCLA	Δ%
TFIDF	0.231	0.234	+1.2	0.238	+2.8	0.233	+0.7	0.237	+2.6
BM25	0.234	0.234	none	0.238	+1.5	0.233	-0.4	0.237	+1.2
BB2	0.206	0.213	+3.5	0.214	+4.0	0.216	+5.0	0.220	+6.7
PL2	0.237	0.253	*+6.8*	0.253	***+7.0***	0.251	*+6.1*	0.256	***+8.2***
DLH	0.232	0.231	-0.7	0.233	+0.5	0.230	-1.0	0.234	+0.9

Table 3: Mean Average Precision (MAP) scores of the POS and POSC experiments with Query Expansion.

WT2G collection									
		Hypothesis 1				Hypotheses 1+2			
w(t,d)	base	POSGT	Δ%	POSLA	Δ%	POSCGT	Δ%	POSCLA	Δ%
TFIDF	0.299	0.323	+8.0	0.329	+10.0	0.322	+7.7	0.325	+8.7
BM25	0.302	0.320	+5.7	0.326	+7.9	0.319	+5.6	0.322	+6.6
BB2	0.239	0.291	***+21.7***	0.288	*+20.5*	0.291	***+21.7***	0.287	*+20.1*
PL2	0.285	0.312	+9.5	0.315	+10.5	0.315	+10.5	0.316	+10.9
DLH	0.267	0.283	+6.0	0.283	+6.0	0.284	+6.4	0.283	+6.0
WT10G collection									
		Hypothesis 1				Hypotheses 1+2			
w(t,d)	base	POSGTQE	Δ%	POSLAQE	Δ%	POSCGT	Δ%	POSCLA	Δ%
TFIDF	0.233	0.241	+3.4	0.249	***+6.9***	0.240	+3.0	0.250	+7.3
BM25	0.240	0.248	+3.3	0.250	+4.2	0.244	+1.7	0.249	+3.7
BB2	0.206	0.213	+3.4	0.214	+3.9	0.216	+4.8	0.220	+6.8
PL2	0.237	0.253	*+6.7*	0.253	+6.7	0.251	*+5.9*	0.256	***+8.0***
DLH	0.236	0.250	+5.9	0.246	+4.2	0.250	*+5.9*	0.253	+7.2

provement in retrieval performance over the new baseline at all times, which is sometimes statistically significant. This may indicate that noise has been further reduced in the queries. Also, the two statistical estimators lead to similar improvements in retrieval performance. When we compare these results to the ones reported with identical settings but without query expansion (Table 2), we observe the following. Firstly, the previously reported division in the DFR weighting schemes, where BB2 and PL2 improved the most from our hypothesised noise reduction in the queries, while DLH improved the least, is no longer valid. The improvement in retrieval performance now associated to DLH is similar to the improvement associated with the other weighting schemes. Secondly, the difference in the retrieval improvement previously observed between the two test collections is now smaller.

To recapitulate on the evaluation outcomes of our two hypotheses combined with query expansion, we report an improvement in retrieval performance over the baseline at all times, which is sometimes statistically significant. It appears that the combination of our hypotheses with query expansion tones down previously reported sharp differences in retrieval improvements over the baseline (Table 2), which may be indicative of further noise reduction.

5 Conclusion

We described a block-based part of speech (POS) modeling of language distribution, induced from a corpus, and statistically smoothened using two different estimators. We hypothesised that high-frequency *POS blocks* bear more content than low-frequency *POS blocks*. Also, we hypothesised that the more closed class components a *POS block* contains, the less content it bears. We evaluated both hypotheses in the context of Information Retrieval, across two standard test collections, and five statistically different term weighting schemes. Our hypotheses led to a general improvement in retrieval performance. This improvement was overall higher for the smaller of the two collections, indicating that data sparseness may have an effect on retrieval. The use of query expansion worked well with our hypotheses, by helping weaker weighting schemes to benefit more from the reduction of noise in the queries.

In the future, we wish to investigate varying the size n of *POS blocks*, as well as testing our hypotheses on shorter queries.

References

Alan F. Smeaton. 1999. *Using NLP or NLP resources for information retrieval tasks. Natural language information retrieval*. Kluwer Academic Publishers Dordrecht, NL.

Bruce Croft and John Lafferty. 2003. *Language Modeling for Information Retrieval*. Springer.

Christopher D. Manning and Hinrich Schutze. 1999. *Foundations of Statistical Language Processing*. The MIT Press, London.

David D. Lewis. 1992. An Evaluation of Phrasal and Clustered Representations on a Text Categorization Task. *ACM SIGIR 1992*, 37–50.

Gianni Amati. 2003. *Probabilistic Models for Information Retrieval based on Divergence from Randomness*. Ph.D. Thesis, University of Glasgow.

Ingrid Zukerman and Bhavani Raskutti. 2002. Lexical Query Paraphrasing for Document Retrieval. *COLING 2002*, 1177–1183.

John Lyons. 1977. *Semantics: Volume 2*. CUP, Cambridge.

Karen Sparck-Jones. 1972. A statistical interpretation of term specifi city and its application in retrieval. *Journal of Documentation*, 28:11–21.

'Keith' (C. J.) van Rijsbergen. 1979. *Information Retrieval*. Butterworths, London.

Kenneth W. Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313–330.

Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jennifer C. Lai, and Robert L. Mercer. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

Stephen Robertson, Steve Walker, Micheline Beaulieu, Mike Gatford, and A. Payne. 1995. Okapi at TREC-4. *NIST Special Publication 500-236: TREC-4*, 73–96.

Tomek Strzalkowski. 1996. Robust Natural Language Processing and user-guided concept discovery for Information retrieval, extraction and summarization. *Tipster Text Phase III Kickoff Workshop*.

Stochastic Iterative Alignment for Machine Translation Evaluation

Ding Liu and **Daniel Gildea**
Department of Computer Science
University of Rochester
Rochester, NY 14627

Abstract

A number of metrics for automatic evaluation of machine translation have been proposed in recent years, with some metrics focusing on measuring the adequacy of MT output, and other metrics focusing on fluency. Adequacy-oriented metrics such as BLEU measure n-gram overlap of MT outputs and their references, but do not represent sentence-level information. In contrast, fluency-oriented metrics such as ROUGE-W compute longest common subsequences, but ignore words not aligned by the LCS. We propose a metric based on stochastic iterative string alignment (SIA), which aims to combine the strengths of both approaches. We compare SIA with existing metrics, and find that it outperforms them in overall evaluation, and works specially well in fluency evaluation.

1 Introduction

Evaluation has long been a stumbling block in the development of machine translation systems, due to the simple fact that there are many correct translations for a given sentence. Human evaluation of system output is costly in both time and money, leading to the rise of automatic evaluation metrics in recent years. In the 2003 Johns Hopkins Workshop on Speech and Language Engineering, experiments on MT evaluation showed that BLEU and NIST do not correlate well with human judgments at the sentence level, even when they correlate well over large test sets (Blatz et al., 2003). Liu and Gildea (2005) also pointed out that due to the limited references for every MT output, using the overlapping ratio of n-grams longer than 2 did not improve sentence level evaluation performance of BLEU. The problem leads to an even worse result in BLEU'S fluency evaluation, which is supposed to rely on the long n-grams. In order to improve sentence-level evaluation performance, several metrics have been proposed, including ROUGE-W, ROUGE-S (Lin and Och, 2004) and METEOR (Banerjee and Lavie, 2005). ROUGE-W differs from BLEU and NIST in that it doesn't require the common sequence between MT output and the references to be consecutive, and thus longer common sequences can be found. There is a problem with loose-sequence-based metrics: the words outside the longest common sequence are not considered in the metric, even if they appear both in MT output and the reference. ROUGE-S is meant to alleviate this problem by computing the common skipped bigrams instead of the LCS. But the price ROUGE-S pays is falling back to the shorter sequences and losing the advantage of long common sequences. METEOR is essentially a unigram based metric, which prefers the monotonic word alignment between MT output and the references by penalizing crossing word alignments. There are two problems with METEOR. First, it doesn't consider gaps in the aligned words, which is an important feature for evaluating the sentence fluency; second, it cannot use multiple references simultaneously.[1] ROUGE and METEOR both use WordNet and Porter Stemmer to increase the chance of the MT output words matching the reference words. Such morphological processing and synonym extraction tools are available for English, but are not always available for other languages. In order to take advantage of loose-sequence-based metrics and avoid the problems in ROUGE and METEOR, we propose a new metric SIA, which is based on loose sequence alignment but enhanced with the following features:

[1] METEOR and ROUGE both compute the score based on the best reference

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 539–546,
Sydney, July 2006.

- Computing the string alignment score based on the gaps in the common sequence. Though ROUGE-W also takes into consider the gaps in the common sequence between the MT output and the reference by giving more credits to the n-grams in the common sequence, our method is more flexible in that not only do the strict n-grams get more credits, but also the tighter sequences.

- Stochastic word matching. For the purpose of increasing hitting chance of MT outputs in references, we use a stochastic word matching in the string alignment instead of WORD-STEM and WORD-NET used in METEOR and ROUGE. Instead of using exact matching, we use a soft matching based on the similarity between two words, which is trained in a bilingual corpus. The corpus is aligned in the word level using IBM Model4 (Brown et al., 1993). Stochastic word matching is a uniform replacement for both morphological processing and synonym matching. More importantly, it can be easily adapted for different kinds of languages, as long as there are bilingual parallel corpora available (which is always true for statistical machine translation).

- Iterative alignment scheme. In this scheme, the string alignment will be continued until there are no more co-occuring words to be found between the MT output and any one of the references. In this way, every co-occuring word between the MT output and the references can be considered and contribute to the final score, and multiple references can be used simultaneously.

The remainder of the paper is organized as follows: section 2 gives a recap of BLEU, ROUGE-W and METEOR; section 3 describes the three components of SIA; section 4 compares the performance of different metrics based on experimental results; section 5 presents our conclusion.

2 Recap of BLEU, ROUGE-W and METEOR

The most commonly used automatic evaluation metrics, BLEU (Papineni et al., 2002) and NIST (Doddington, 2002), are based on the assumption that "The closer a machine translation is to a professional human translation, the better it is" (Papineni et al., 2002). For every hypothesis, BLEU computes the fraction of n-grams which also appear in the reference sentences, as well as a brevity penalty. NIST uses a similar strategy to BLEU but further considers that n-grams with different frequency should be treated differently in the evaluation (Doddington, 2002). BLEU and NIST have been shown to correlate closely with human judgments in ranking MT systems with different qualities (Papineni et al., 2002; Doddington, 2002).

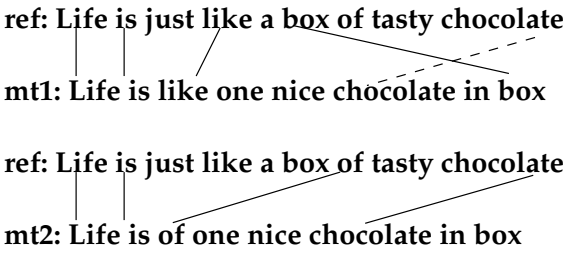


Figure 1: Alignment Example for ROUGE-W

ROUGE-W is based on the weighted longest common subsequence (LCS) between the MT output and the reference. The common subsequences in ROUGE-W are not necessarily strict n-grams, and gaps are allowed in both the MT output and the reference. Because of the flexibility, long common subsequences are feasible in ROUGE-W and can help to reflect the sentence-wide similarity of MT output and references. ROUGE-W uses a weighting strategy where the LCS containing strict n-grams is favored. Figure 1 gives two examples that show how ROUGE-W searches for the LCS. For *mt1*, ROUGE-W will choose either *life is like chocolate* or *life is like box* as the LCS, since neither of the sequences 'like box' and 'like chocolate' are strict n-grams and thus make no difference in ROUGE-W (the only strict n-grams in the two candidate LCS is *life is*). For *mt2*, there is only one choice of the LCS: *life is of chocolate*. The LCS of *mt1* and *mt2* have the same length and the same number of strict n-grams, thus they get the same score in ROUGE-W. But it is clear to us that *mt1* is better than *mt2*. It is easy to verify that *mt1* and *mt2* have the same number of common 1-grams, 2-grams, and skipped 2-grams with the reference (they don't have common n-grams longer than 2 words), thus BLEU and ROUGE-S are also not able to differentiate them.

METEOR is a metric sitting in the middle of the n-gram based metrics and the loose se-

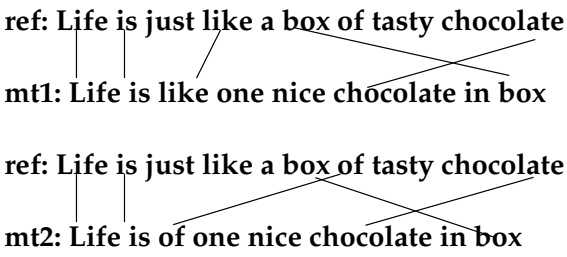


Figure 2: Alignment Example for METEOR

quence based metrics. It has several phases and in each phase different matching techniques (EXACT, PORTER-STEM, WORD-NET) are used to make an alignment for the MT output and the reference. METEOR doesn't require the alignment to be monotonic, which means crossing word mappings (e.g. a b is mapped to b a) are allowed, though doing so will get a penalty. Figure 2 shows the alignments of METEOR based on the same example as ROUGE. Though the two alignments have the same number of word mappings, *mt2* gets more crossed word mappings than *mt1*, thus it will get less credits in METEOR. Both ROUGE and METEOR normalize their evaluation result based on the MT output length (precision) and the reference length (recall), and the final score is computed as the F-mean of them.

3 Stochastic Iterative Alignment (SIA) for Machine Translation Evaluation

We introduce three techniques to allow more sensitive scores to be computed.

3.1 Modified String Alignment

This section introduces how to compute the string alignment based on the word gaps. Given a pair of strings, the task of string alignment is to obtain the longest monotonic common sequence (where gaps are allowed). SIA uses a different weighting strategy from ROUGE-W, which is more flexible. In SIA, the alignments are evaluated based on the geometric mean of the gaps in the reference side and the MT output side. Thus in the dynamic programming, the state not only includes the current covering length of the MT output and the reference, but also includes the last aligned positions in them. The algorithm for computing the alignment score in SIA is described in Figure 3. The subroutine COMPUTE_SCORE, which computes the score gained from the current aligned positions, is shown in Figure 4. From the algorithm, we can

function GET_ALIGN_SCORE(*mt*, *M*, *ref*, *N*)
▷ Compute the alignment score of the MT output *mt* with length *M* and the reference *ref* with length *N*
  for $i = 1; i \leq M; i = i + 1$ **do**
    for $j = 1; j \leq N; j = j + 1$ **do**
      for $k = 1; k \leq i; k = k + 1$ **do**
        for $m = 1; m \leq j; m = m + 1$ **do**
          $score_{i,j,k,m} = \max\{score_{i-1,j,k,m}, score_{i,j-1,k,m}\}$;
        end for
      end for
      $score_{i,j,i,j} = \max_{n=1,M;p=1,N}\{score_{i,j,i,j}, score_{i-1,j-1,n,p} + \text{COMPUTE\_SCORE}(mt, ref, i, j, n, p)\}$;
    end for
  end for
  return $\frac{score_{M,N,M,N}}{M}$;
end function

Figure 3: Alignment Algorithm Based on Gaps

function COMPUTE_SCORE(*mt*, *ref*, *i*, *j*, *n*, *p*)
  if *mt*[*i*] == *ref*[*j*] **then**
    return $1/\sqrt{(i-n)\times(j-p)}$;
  else
    return 0;
  end if
end function

Figure 4: Compute Word Matching Score Based on Gaps

see that not only will strict n-grams get higher scores than non-consecutive sequences, but also the non-consecutive sequences with smaller gaps will get higher scores than those with larger gaps. This weighting method can help SIA capture more subtle difference of MT outputs than ROUGE-W does. For example, if SIA is used to align *mt1* and *ref* in Figure 1, it will choose *life is like box* instead of *life is like chocolate*, because the average distance of 'box-box' to its previous mapping 'like-like' is less than 'chocolate-chocolate'. Then the score SIA assigns to *mt1* is:

$$\left(\frac{1}{1\times 1}+\frac{1}{1\times 1}+\frac{1}{\sqrt{1\times 2}}+\frac{1}{\sqrt{2\times 5}}\right)\times\frac{1}{8}=0.399 \quad (1)$$

For *mt2*, there is only one possible alignment, its score in SIA is computed as:

$$\left(\frac{1}{1\times 1}+\frac{1}{1\times 1}+\frac{1}{\sqrt{1\times 5}}+\frac{1}{\sqrt{2\times 3}}\right)\times\frac{1}{8}=0.357 \quad (2)$$

Thus, *mt1* will be considered better than *mt2* in SIA, which is reasonable. As mentioned in section 1, though loose-sequence-based metrics give a better reflection of the sentence-wide similarity of the MT output and the reference, they cannot

make full use of word-level information. This defect could potentially lead to a poor performance in adequacy evaluation, considering the case that the ignored words are crucial to the evaluation. In the later part of this section, we will describe an iterative alignment scheme which is meant to compensate for this defect.

3.2 Stochastic Word Mapping

In ROUGE and METEOR, PORTER-STEM and WORD-NET are used to increase the chance of the MT output words matching the references. We use a different stochastic approach in SIA to achieve the same purpose. The string alignment has a good dynamic framework which allows the stochastic word matching to be easily incorporated into it. The stochastic string alignment can be implemented by simply replacing the function COMPUTE_SCORE with the function of Figure 5. The function *similarity(word1, word2)* returns a ratio which reflects how similar the two words are. Now we consider how to compute the similarity ratio of two words. Our method is motivated by the phrase extraction method of Bannard and Callison-Burch (2005), which computes the similarity ratio of two words by looking at their relationship with words in another language. Given a bilingual parallel corpus with aligned sentences, say English and French, the probability of an English word given a French word can be computed by training word alignment models such as IBM Model4. Then for every English word *e*, we have a set of conditional probabilities given each French word: $p(e|f_1)$, $p(e|f_2)$, ... , $p(e|f_N)$. If we consider these probabilities as a vector, the similarities of two English words can be obtained by computing the dot product of their corresponding vectors.[2] The formula is described below:

$$similarity(e_i, e_j) = \sum_{k=1}^{N} p(e_i|f_k)p(e_j|f_k) \quad (3)$$

Paraphrasing methods based on monolingual parallel corpora such as (Pang et al., 2003; Barzilay and Lee, 2003) can also be used to compute the similarity ratio of two words, but they don't have as rich training resources as the bilingual methods do.

[2]Although the marginalized probability (over all French words) of an English word given the other English word ($\sum_{k=1}^{N} p(e_i|f_k)p(f_k|e_j)$) is a more intuitive way of measuring the similarity, the dot product of the vectors $p(e|f)$ described above performed slightly better in our experiments.

```
function STO_COMPUTE_SCORE(mt, ref, i, j, n, p)
    if mt[i] == ref[j] then
        return 1/sqrt((i − n) × (j − p));
    else
        return similarity(mt[i],ref[i]) / sqrt((i−n)×(j−p));
    end if
end function
```

Figure 5: Compute Stochastic Word Matching Score

3.3 Iterative Alignment Scheme

ROUGE-W, METEOR, and WER all score MT output by first computing a score based on each available reference, and then taking the highest score as the final score for the MT output. This scheme has the problem of not being able to use multiple references simultaneously. The iterative alignment scheme proposed here is meant to alleviate this problem, by doing alignment between the MT output and one of the available references until no more words in the MT output can be found in the references. In each alignment round, the score based on each reference is computed and the highest one is taken as the score for the round. Then the words which have been aligned in best alignment will not be considered in the next round. With the same number of aligned words, the MT output with fewer alignment rounds should be considered better than those requiring more rounds. For this reason, a decay factor α is multiplied with the scores of each round. The final score of the MT output is then computed by summing the weighted scores of each alignment round. The scheme is described in Figure 6.

The function GET_ALIGN_SCORE_1 used in GET_ALIGN_SCORE_IN_MULTIPLE_REFS is slightly different from GET_ALIGN_SCORE described in the prior subsection. The dynamic programming algorithm for getting the best alignment is the same, except that it has two more tables as input, which record the unavailable positions in the MT output and the reference. These positions have already been used in the prior best alignments and should not be considered in the ongoing alignment. It also returns the aligned positions of the best alignment. The pseudocode for GET_ALIGN_SCORE_1 is shown in Figure 7. The computation of the length penalty is similar to BLEU: it is set to 1 if length of the MT output is longer than the arithmetic mean of length of the

function GET_ALIGN_SCORE_IN_MULTIPLE_REFS(*mt*, ref_1, ..., ref_N, α)
▷ Iteratively Compute the Alignment Score Based on Multiple References and the Decay Factor α
final_score = 0;
while *max_score* != 0 **do**
for *i* = 1, ..., *N* **do**
(*score*, *align*) = GET_ALIGN_SCORE_1(*mt*, ref_i, *mt_table*, ref_table_i);
if *score* > *max_score* **then**
max_score = *score*;
max_align = *align*;
max_ref = *i*;
end if
end for
final_score += *max_score* $\times\alpha$;
$\alpha \times = \alpha$;
Add the words in *align* to *mt_table* and $ref\_table_{max\_ref}$;
end while
return *final_score*$\times$ *length_penalty*;
end function

Figure 6: Iterative Alignment Scheme

function GET_ALIGN_SCORE1(*mt*, *ref*, *mttable*, *reftable*)
▷ Compute the alignment score of the MT output *mt* with length *M* and the reference *ref* with length *N*, without considering the positions in *mttable* and *reftable*
$M = |mt|$; $N = |ref|$;
for $i = 1; i \leq M; i = i+1$ **do**
for $j = 1; j \leq N; j = j+1$ **do**
for $k = 1; k \leq i; k = k+1$ **do**
for $m = 1; m \leq j; m = m+1$ **do**
$score_{i,j,k,m} = \max\{score_{i-1,j,k,m}, score_{i,j-1,k,m}\}$;
end for
end for
if *i* is not in *mttable* and *j* is not in *reftable* **then**
$score_{i,j,i,j} = \max_{n=1,M;p=1,N}\{score_{i,j,i,j}, score_{i-1,j-1,n,p} + \text{COMPUTE\_SCORE}(mt, ref, i, j, n, p)\}$;
end if
end for
end for
return $\frac{score_{M,N,M,N}}{M}$ and the corresponding alignment;
end function

Figure 7: Alignment Algorithm Based on Gaps Without Considering Aligned Positions

references, and otherwise is set to the ratio of the two. Figure 8 shows how the iterative alignment scheme works with an evaluation set containing one MT output and two references. The selected alignment in each round is shown, as well as the unavailable positions in MT output and references. With the iterative scheme, every common word between the MT output and the reference set can make a contribution to the metric, and by such means SIA is able to make full use of the word-level information. Furthermore, the order (alignment round) in which the words are aligned provides a way to weight them. In BLEU, multiple references can be used simultaneously, but the common n-grams are treated equally.

4 Experiments

Evaluation experiments were conducted to compare the performance of different metrics including BLEU, ROUGE, METEOR and SIA.[3] The test data for the experiments are from the MT evaluation workshop at ACL05. There are seven sets of MT outputs (E09 E11 E12 E14 E15 E17 E22), all of which contain 919 English sentences. These sentences are the translation of the same Chinese input generated by seven different MT systems. The fluency and adequacy of each sentence are manually ranked from 1 to 5. For each MT output, there are two sets of human scores available, and

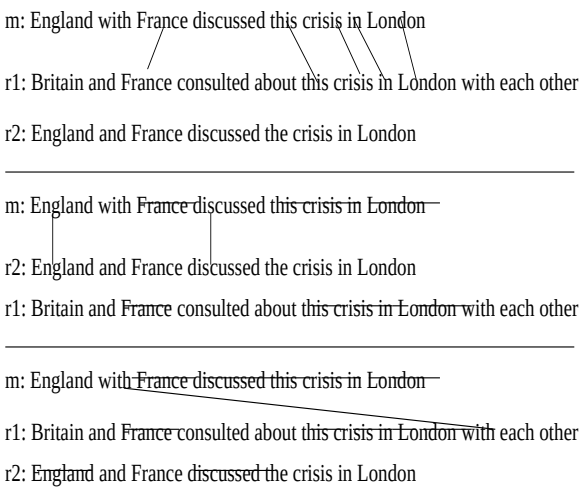


Figure 8: Alignment Example for SIA

[3]METEOR and ROUGE can be downloaded at http://www.cs.cmu.edu/~alavie/METEOR and http://www.isi.edu/licensed-sw/see/rouge

we randomly choose one as the score used in the experiments. The human overall scores are calculated as the arithmetic means of the human fluency scores and adequacy scores. There are four sets of human translations (E01, E02, E03, E04) serving as references for those MT outputs. The MT outputs and reference sentences are transformed to lower case. Our experiments are carried out as follows: automatic metrics are used to evaluate the MT outputs based on the four sets of references, and the Pearson's correlation coefficient of the automatic scores and the human scores is computed to see how well they agree.

4.1 N-gram vs. Loose Sequence

One of the problems addressed in this paper is the different performance of n-gram based metrics and loose-sequence-based metrics in sentence-level evaluation. To see how they really differ in experiments, we choose BLEU and ROUGE-W as the representative metrics for the two types, and used them to evaluate the 6433 sentences in the 7 MT outputs. The Pearson correlation coefficients are then computed based on the 6433 samples. The experimental results are shown in Table 1. BLEU-n denotes the BLEU metric with the longest n-gram of length n. F denotes fluency, A denotes adequacy, and O denotes overall. We see that with the increase of n-gram length, BLEU's performance does not increase monotonically. The best result in adequacy evaluation is achieved at 2-gram and the best result in fluency is achieved at 4-gram. Using n-grams longer than 2 doesn't buy much improvement for BLEU in fluency evaluation, and does not compensate for the loss in adequacy evaluation. This confirms Liu and Gildea (2005)'s finding that in sentence level evaluation, long n-grams in BLEU are not beneficial. The loose-sequence-based ROUGE-W does much better than BLEU in fluency evaluation, but it does poorly in adequacy evaluation and doesn't achieve a significant improvement in overall evaluation. We speculate that the reason is that ROUGE-W doesn't make full use of the available word-level information.

4.2 METEOR vs. SIA

SIA is designed to take the advantage of loose-sequence-based metrics without losing word-level information. To see how well it works, we choose E09 as the development set and the sentences in the other 6 sets as the test data. The decay factor in SIA is determined by optimizing the overall evaluation for E09, and then used with SIA to evaluate the other 5514 sentences based on the four sets of references. The similarity of English words is computed by training IBM Model 4 in an English-French parallel corpus which contains seven hundred thousand sentence pairs. For every English word, only the entries of the top 100 most similar English words are kept and the similarity ratios of them are then re-normalized. The words outside the training corpus will be considered as only having itself as its similar word. To compare the performance of SIA with BLEU, ROUGE and METEOR, the evaluation results based on the same testing data is given in Table 2. B-3 denotes BLEU-3; R_1 denotes the skipped bigram based ROUGE metric which considers all skip distances and uses PORTER-STEM; R_2 denotes ROUGE-W with PORTER-STEM; M denotes the METEOR metric using PORTER-STEM and WORD-NET synonym; S denotes SIA.

	B-3	R_1	R_2	M	S
F	0.167	0.152	0.192	0.167	0.202
A	0.306	0.304	0.287	0.332	0.322
O	0.265	0.256	0.266	0.280	0.292

Table 2: Sentence level evaluation results of BLEU, ROUGE, METEOR and SIA

We see that METEOR, as the other metric sitting in the middle of n-gram based metrics and loose sequence metrics, achieves improvement over BLEU in both adequacy and fluency evaluation. Though METEOR gets the best results in adequacy evaluation, in fluency evaluation, it is worse than the loose-sequence-based metric ROUGE-W-STEM. SIA is the only one among the 5 metrics which does well in both fluency and adequacy evaluation. It achieves the best results in fluency evaluation and comparable results to METEOR in adequacy evaluation, and the balanced performance leads to the best overall evaluation results in the experiment. To estimate the significance of the correlations, bootstrap resampling (Koehn, 2004) is used to randomly select 5514 sentences with replacement out of the whole test set of 5514 sentences, and then the correlation coefficients are computed based on the selected sentence set. The resampling is repeated 5000 times, and the 95% confidence intervals are shown in Tables 3, 4, and 5. We can see that it is very diffi-

	BLEU-1	BLEU-2	BLEU-3	BLEU-4	BLEU-5	BLEU-6	ROUGE-W
F	0.147	0.162	0.166	0.168	0.165	0.164	0.191
A	0.288	0.296	0.291	0.285	0.279	0.274	0.268
O	0.243	0.256	0.255	0.251	0.247	0.244	0.254

Table 1: Sentence level evaluation results of BLEU and ROUGE-W

	low	mean	high
B-3	(-16.6%) 0.138	0.165	0.192 (+16.4%)
R_1	(-17.8%) 0.124	0.151	0.177 (+17.3%)
R_2	(-14.3%) 0.164	0.191	0.218 (+14.2%)
M	(-15.8%) 0.139	0.166	0.191 (+15.5%)
S	(-13.3%) 0.174	0.201	0.227 (+13.3%)

Table 3: 95% significance intervals for sentence-level fluency evaluation

	low	mean	high
B-3	(-08.2%) 0.280	0.306	0.330 (+08.1%)
R_1	(-08.5%) 0.278	0.304	0.329 (+08.4%)
R_2	(-09.2%) 0.259	0.285	0.312 (+09.5%)
M	(-07.3%) 0.307	0.332	0.355 (+07.0%)
S	(-07.9%) 0.295	0.321	0.346 (+07.8%)

Table 4: 95% significance intervals for sentence-level adequacy evaluation

cult for one metric to significantly outperform another metric in sentence-level evaluation. The results show that the mean of the correlation factors converges right to the value we computed based on the whole testing set, and the confidence intervals correlate with the means.

While sentence-level evaluation is useful if we are interested in a confidence measure on MT outputs, syste-x level evaluation is more useful for comparing MT systems and guiding their development. Thus we also present the evaluation results based on the 7 MT output sets in Table 6. SIA uses the same decay factor as in the sentence-level evaluation. Its system-level score is computed as the arithmetic mean of the sentence level scores, and

	low	mean	high
B-3	(-09.8%) 0.238	0.264	0.290 (+09.9%)
R_1	(-10.2%) 0.229	0.255	0.281 (+10.0%)
R_2	(-10.0%) 0.238	0.265	0.293 (+10.4%)
M	(-09.0%) 0.254	0.279	0.304 (+08.8%)
S	(-08.7%) 0.265	0.291	0.316 (+08.8%)

Table 5: 95% significance intervals for sentence-level overall evaluation

	WLS	WLS PROB	WLS INCS	WLS PROB INCS
F	0.189	0.202	0.188	0.202
A	0.295	0.310	0.311	0.322
O	0.270	0.285	0.278	0.292

Table 7: Results of different components in SIA

	WLS INCS	WLS INCS STEM	WLS INCS WN	WLS INCS STEM WN
F	0.188	0.188	0.187	0.191
A	0.311	0.313	0.310	0.317
O	0.278	0.280	0.277	0.284

Table 8: Results of SIA working with Porter-Stem and WordNet

so are ROUGE, METEOR and the human judgments. We can see that SIA achieves the best performance in both fluency and adequacy evaluation of the 7 systems. Though the 7-sample based results are not reliable, we can get a sense of how well SIA works in the system-level evaluation.

4.3 Components in SIA

To see how the three components in SIA contribute to the final performance, we conduct experiments where one or two components are removed in SIA, shown in Table 7. The three components are denoted as WLS (weighted loose sequence alignment), PROB (stochastic word matching), and INCS (iterative alignment scheme) respectively. WLS without INCS does only one round of alignment and chooses the best alignment score as the final score. This scheme is similar to ROUGE-W and METEOR. We can see that INCS, as expected, improves the adequacy evaluation without hurting the fluency evaluation. PROB improves both adequacy and fluency evaluation performance. The result that SIA works with PORTER-STEM and WordNet is also shown in Table 8. When PORTER-STEM and WordNet are

	B-6	R_1	R_2	M	S
F	0.514	0.466	0.458	0.378	0.532
A	0.876	0.900	0.906	0.875	0.928
O	0.794	0.790	0.792	0.741	0.835

Table 6: Results of BLEU, ROUGE, METEOR and SIA in system level evaluation

both used, PORTER-STEM is used first. We can see that they are not as good as using the stochastic word matching. Since INCS and PROB are independent of WLS, we believe they can also be used to improve other metrics such as ROUGE-W and METEOR.

5 Conclusion

This paper describes a new metric SIA for MT evaluation, which achieves good performance by combining the advantages of n-gram-based metrics and loose-sequence-based metrics. SIA uses stochastic word mapping to allow soft or partial matches between the MT hypotheses and the references. This stochastic component is shown to be better than PORTER-STEM and WordNet in our experiments. We also analyzed the effect of other components in SIA and speculate that they can also be used in other metrics to improve their performance.

Acknowledgments This work was supported by NSF ITR IIS-09325646 and NSF ITR IIS-0428020.

References

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judegments. In *Proceedings of the ACL-04 workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, Ann Arbor, Michigan.

Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Conference of the Association for Computational Linguistics (ACL-05)*.

Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Meeting of the North American chapter of the Association for Computational Linguistics (NAACL-03)*, pages 16–23.

John Blatz, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto Sanchis, and Nicola Ueffing. 2003. Confidence estimation for machine translation. Technical report, Center for Language and Speech Processing, Johns Hopkins University, Baltimore. Summer Workshop Final Report.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

G. Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *In HLT 2002, Human Language Technology Conference*, San Diego, CA.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 388–395, Barcelona, Spain, July.

Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42th Annual Conference of the Association for Computational Linguistics (ACL-04)*, Barcelona, Spain.

Ding Liu and Daniel Gildea. 2005. Syntactic features for evaluation of machine translation. In *ACL 2005 Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*.

Bo Pang, Kevin Knight, and Daniel Marcu. 2003. Syntax-based alignment of multiple translations: Extracting paraphrases and generating new sentences. In *Proceedings of the 2003 Meeting of the North American chapter of the Association for Computational Linguistics (NAACL-03)*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Conference of the Association for Computational Linguistics (ACL-02)*, Philadelphia, PA.

Discriminating image senses by clustering with multimodal features

Nicolas Loeff
Dept. of Computer Science
University of Illinois, UC
loeff@uiuc.edu

Cecilia Ovesdotter Alm
Dept. of Linguistics
University of Illinois, UC
ebbaalm@uiuc.edu

David A. Forsyth
Dept. of Computer Science
University of Illinois, UC
daf@uiuc.edu

Abstract

We discuss *Image Sense Discrimination* (ISD), and apply a method based on spectral clustering, using multimodal features from the image and text of the embedding web page. We evaluate our method on a new data set of annotated web images, retrieved with ambiguous query terms. Experiments investigate different levels of sense granularity, as well as the impact of text and image features, and global versus local text features.

1 Introduction and problem clarification

Semantics extends beyond words. We focus on *image sense discrimination* (ISD)[1] for web images retrieved from ambiguous keywords, given a multimodal feature set, including text from the document which the image was embedded in. For instance, a search for CRANE retrieves images of crane machines, crane birds, associated other machinery or animals etc., people, as well as images of irrelevant meanings. Current displays for image queries (e.g. Google or Yahoo!) simply list retrieved images in any order. An application is a user display where images are presented in semantically sensible clusters for improved image browsing. Another usage of the presented model is automatic creation of sense discriminated image data sets, and determining available image senses automatically.

ISD differs from word sense discrimination and disambiguation (WSD) by increased complexity in several respects. As an initial complication, both **word and iconographic sense distinctions** matter. Whereas a search term like CRANE can refer to, e.g. a MACHINE or a BIRD; iconographic distinctions could additionally include birds standing, vs. in a marsh land, or flying, i.e. sense-distinctions encoded by further descriptive modification in text. Therefore, as the number of text senses grow with corpus size, the iconographic senses grow even faster, and enumerating iconographic senses is extremely challenging; especially since dictionary senses do not capture iconographic distinctions. Thus, we focus on image-driven word senses for ISD, but we acknowledge the importance of iconography for visual meaning.

Also, an image often **depicts a related meaning**. E.g. a picture retrieved for SQUASH may depict a squash bug (i.e. an insect on a leaf of a squash plant) instead of a squash vegetable, whereas this does not really apply in WSD, where each instance concerns the ambiguous term itself. Therefore, it makes sense to consider the division between **core sense, related sense, and unrelated sense** in ISD, and, as an additional complication, their boundaries are often blurred. Most importantly, whereas the one-sense-per-discourse assumption (Yarowsky, 1995) also applies to discriminating images, there is **no guarantee of a local collocational or co-occurrence context** around the target image. Design or aesthetics may instead determine image placement. Thus, considering local text around the image may not be as helpful as local context is for standard WSD. In fact, the **query term may even not occur** in the text body. On the other hand, one can assume that an image spotlights the web page topic and that it highlights important document information. Also, images mostly depict concrete senses. Lastly, ISD from web data is complicated by web pages being more domain-independent than news wire, the fa-

[1] Cf. (Schütze, 1998) for a definition of sense discrimination in NLP.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 547–554,
Sydney, July 2006.

(a) *squash flower* (b) *tennis?* (c) *hook* (d) *food* (e) *bow* (f) *speaker*

Figure 1: **Example RELATED images** for (a) *vegetable* and (b) *sports* senses for SQUASH, and for (c-d) *fish* and (e-f) *musical instrument* for BASS. Related senses are associated with the semantic field of a core sense, but the core sense is visually absent or undeterminable.

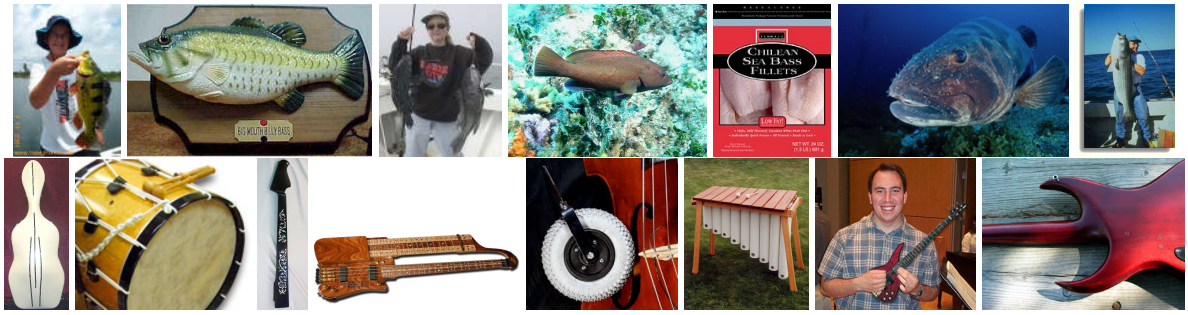


Figure 2: ***Which fish or instruments are* BASS?** Image sense annotation is more vague and subjective than in text.

vored corpus for WSD. As noted by (Yanai and Barnard, 2005), whereas current image retrieval engines include many irrelevant images, a data set of web images gives a more real-world point of departure for image recognition.

Outline Section 2 discusses the corpus data and image annotation. Section 3 presents the feature set and the clustering model. Subsequently, section 4 introduces the evaluation used, and discusses experimental work and results. In section 5, this work is positioned with respect to previous work. We conclude with an outline of plans for future work in section 6.

2 Data and annotation

Yahoo!'s image query API was used to obtain a corpus of pairs of semantically ambiguous images, in thumbnail and true size, and their corresponding web sites for three ambiguous keywords inspired by (Yarowsky, 1995): BASS, CRANE, and SQUASH. We apply query augmentation (cf. Table 1), and exact duplicates were filtered out by identical image URLs, but cases occurred where both thumbnail and true-size image were included. Also, some images shared the same webpage or came from the same site. Generally, the latter gives important information about shared discourse topic, however the images do not necessarily depict the same sense (e.g. a CRANE bird vs. a meadow), and image features can separate them into different clusters.

Annotation overview The images were annotated with one of several labels by one of the authors out of context (*without* considering the web site and its text), after applying text-based filtering (cf. section 3.1). For annotation purposes, images were numbered and displayed on a web page in thumbnail size. In case the thumbnail was not sufficient for disambiguation, the image linked at its true size to the thumbnail was inspected.[2] The true-size view depended on the size of the original picture and showed the image and its name. However, the annotator tried to resist name influence, and make judgements based just on the image. For each query, 2 to 4 core word senses (e.g. *squash vegetable* and *squash sport* for SQUASH) were distinguished from inspecting the data. However, because "context" was restricted to the image content, and there was no guarantee that the image actually depicts the query term, additional annotator senses were introduced. Thus, for most core senses, a RELATED label was included, accounting for meanings that seemed related to core meaning but lacked a core sense object in the image. Some examples for RELATED senses are in Fig. 1. In addition, for each query term, a PEOPLE label was included because such images are common due to the nature of how people take pictures (e.g. portraits of persons or group pictures of crowds, when core or related senses did not apply), as was an

[2] We noticed a few cases where Yahoo! retrieved a thumbnail image different from the true size image.

Word (#Annot. images)	QueryTerms	Senses	Coverage	Examples of visual annotation cues
BASS (2881)	5: bass, bass guitar, bass instrument, bass fishing, sea bass	1. **fish**	35%	any fish, people holding catch
		2. **musical instrument**	28%	any bass-looking instrument, playing
		3. related: fish	10%	fishing (gear, boats, farms), rel. food, rel. charts/maps
		4. related: musical instrument	8%	speakers, accessories, works, chords, rel. music
		5. unrelated	12%	miscellaneous (above senses not applicable)
		6. people	7%	faces, crowd (above senses not applicable)
CRANE (2650)	5: crane, construction cranes, whooping crane, sandhill crane, origami cranes	1. **machine**	21%	machine crane, incl. panoramas
		2. **bird**	26%	crane bird or chick
		3. **origami**	4%	origami bird
		4. related: machine	11%	other machinery, construction, motor, steering, seat
		5. related: bird	11%	egg, other birds, wildlife, insects, hunting, rel. maps/charts
		6. related: origami	1%	origami shapes (stars, pigs), paper folding
		7. people	7%	faces, crowd (above senses not applicable)
		8. unrelated	18%	miscellaneous (above senses not applicable)
		9. **karate**	1%	martial arts
SQUASH (1948)	10: squash+: rules, butternut, vegetable, grow, game of, spaghetti, winter, types of, summer	1. **vegetable**	24%	squash vegetable
		2. **sport**	13%	people playing, court, equipment
		3. related:vegetable	31%	agriculture, food, plant, flower, insect, vegetables
		4. related:sport	6%	other sports, sports complex
		5. people	10%	faces, crowd (above senses not applicable)
		6. unrelated	16%	miscellaneous (above senses not applicable)

Table 1: **Web images for three ambiguous query terms** were annotated manually out of context (*without* considering the web page document). For each term, the number of annotated images, the query retrieval terms, the senses, their distribution, and rough sample annotation guidelines are provided, with core senses marked in bold face. Because image retrieval engines restrict hits to 1000 images, query expansion was conducted by adding narrowing query terms from `askjeeves.com` to increase corpus size. We selected terms relevant to core senses, i.e. the main discrimination phenomenon.

UNRELATED label for irrelevant images which did not fit other labels or were undeterminable.

For a human annotator, even when using more natural word senses, assigning sense labels to images based on image alone is more challenging and subjective than labeling word senses in textual context. First of all, the annotation is heavily dependent on **domain-knowledge** and it is not feasible for a layperson to recognize fine-grained semantics. For example, it is straightforward for the layperson to distinguish between a robin and a crane, but determining whether a given fish should have the common name *bass* applied to it, or whether an instrument is indeed a bass instrument or not, is extremely difficult (see Fig. 2; e.g. deciding if a picture of a fish fillet is a picture of a fish is tricky). Furthermore, most images **display objects only partially**; for example just the neck of a classical double bass instead of the whole instrument. In addition, **scaling, proportions, and components** are key cues for object discrimination in real-life, e.g. for singling out an electric bass from an electric guitar, but an image may not provide these detail. Thus, **senses are even fuzzier** for ISD than WSD labeling. Given that laypeople are in the majority, it is fair to assume their perspective and naiveness. This latter fact also led to annotations' level of specificity differing according to search term. Annotation criteria depended on the keyword term and its senses and their coverage, as shown in Table 1. Nevertheless, several border-line cases for label assignment occurred. Considering that the annotation task is quite subjective, this is to be expected. In fact, one person's labeling often appears as justifiable as a contradicting label provided by another person. We explore the vagueness and subjective nature of image annotation further in a companion paper (Alm, Loeff, Forsyth, 2006).

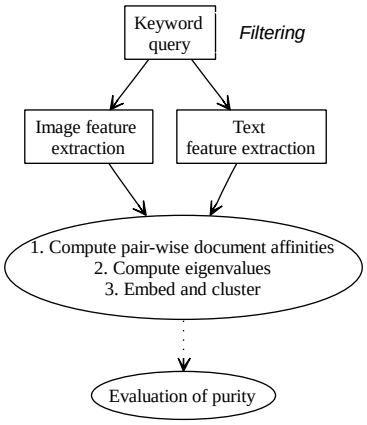


Figure 3: Overview of algorithm

3 Model

Our goal is to provide a mapping between images and a set of iconographically coherent clusters for a given query word, in an unsupervised framework. Our approach involves extracting and weighting unordered bags-of-words (BOWs; henceforth) features from the webpage text, simple local and global features from the image, and running spectral clustering on top. Fig. 3 shows an overview of the implementation.

3.1 Feature extraction

Document and text filtering A pruning process was used to filter out image-document pairs based on e.g. language specification, exclusion of *"Index of"* pages, pages lacking an extractable target image, or a cutoff threshold of number of tokens in the body. For remaining documents, text was preprocessed (e.g. lower-casing, removing punctuation, tokens being very short, having numbers or no vowels, etc.). We used a stop word list, but avoided stemming to make the algorithm language independent in other respects. When using image features, grayscale images (no color histograms) and images without salient regions (no keypoints detected) were also removed.

Text features We used the following BOWs: (a) tokens in the page *body*; (b) tokens in a ± 10 window around the target image (if multiple, the first was considered); (c) tokens in a ± 10 window around any instances of the query keyword (e.g. *squash*); (d) tokens of the target image's *alt* attribute; (e) tokens of the *title* tag; (f) some *meta* tokens.[3] Tf-idf was applied to a weighted average of the BOWs. Webpage design is flexible, and some inconsistencies and a certain degree of noise remained in the text features.

Image features Given the large variability in the retrieved image set for a given query, it is difficult to model images in an unsupervised fashion. Simple features have been shown to provide performance rivaling that of more elaborate models in object recognition (Csurka et al, 2004) and (Chapelle, Haffner, and Vapnik, 1999), and the following image bags of features were considered:

Bags of keypoints: In order to obtain a compact representation of the textures of an image, patches are extracted automatically around interesting regions or *keypoints* in each image. The keypoint detection algorithm (Kadir and Brady, 2001) uses a saliency measure based on entropy to select regions. After extraction, keypoints were represented by a histogram of gradient magnitude of the pixel values in the region (SIFT) (Lowe, 2004). These descriptors were clustered using a Gaussian Mixture with ≈ 300 components, and the resulting global patch *codebook* (i.e. histogram of codebook entries) was used as lookup table to assign each keypoint to a codebook entry.

[3] Adding to META *content*, *keywords* was an attribute, but is irregular. Embedded BODY pairs are rare; thus not used.

Color histograms: Due to its similarity to how humans perceive color, HSV (hue, saturation, brightness) color space was used to bin pixel color values for each image. Eight bins were used per channel, obtaining an 8^3 dimensional vector.

3.2 Measuring similarity between images

For the BOWs text representation, we use the common measure of *cosine similarity* (cs) of two *tf-idf* vectors (Jurafsky and Martin, 2000). The cosine similarity measure is also appropriate for keypoint representation as it is also an unordered bag. There are several measures for histogram comparison (i.e. $L1$, χ^2). As in (Fowlkes et al, 2004) we use the χ^2 distance measure between histograms h_i and h_j.

$$\chi^2{}_{i,j} = \frac{1}{2}\sum_{k=1}^{512}\frac{(h_i(k)-h_j(k))^2}{h_i(k)+h_j(k)} \quad (1)$$

3.3 Spectral Clustering

Spectral clustering is a powerful way to separate non-convex groups of data. Spectral methods for clustering are a family of algorithms that work by first constructing a pairwise-affinity matrix from the data, computing an eigendecomposition of the data, embedding the data into this low-dimensional manifold, and finally applying traditional clustering techniques (i.e. k-means) to it.

Consider a graph with a set of n vertices each one representing an image document, and the edges of the graph represent the pairwise affinities between the vertices. Let W be an $n \times n$ symmetric matrix of pairwise affinities. We define these as the Gaussian-weighted distance

$$W_{ij} = \exp\left(-\alpha^t(1-cs^t_{i,j}) - \alpha^k(1-cs^k_{i,j}) - \alpha^c\chi^2_{i,j}\right), \quad (2)$$

where $\{\alpha^t, \alpha^k, \alpha^c\}$ are scaling parameters for text, keypoints, and color features.

It has been shown that the use of multiple eigenvectors of W is a valid space onto which the data can be embedded (Ng, Jordan, Weiss, 2002). In this space *noise* is reduced while the most significant affinities are preserved. After this, any traditional clustering algorithm can be applied in this new space to get the final clusters. Note that this is a nonlinear mapping of the *original* space. In particular, we employ a variant of k-means, which includes a *selective* step that is quasi-optimal in a Vector Quantization sense (Ueda and Nakano, 1994). It has the added advantage of being more

robust to initialization than traditional k-means. The algorithm follows,

1. For given documents, compute the affinity matrix W as defined in equation 2.
2. Let D be a diagonal matrix whose (i, i)-th element is the sum of W's i-th row, and define $\mathcal{L} = D^{-1/2}WD^{-1/2}$.
3. Find the k largest eigenvectors V of $\mathcal{L}$.
4. Define E as V, with normalized rows.
5. Perform clustering on the columns of E, which represent the embedding of each image into the new space, using a *selective* step as in (Ueda and Nakano, 1994).

Why Spectral Clustering? Why apply a variant of k-means in the embedded space as opposed to the *original* feature space? The k-means algorithm cannot separate non-convex clusters. Furthermore, it is unable to cope with noisy dimensions (this is especially true in the case of the text data) and highly non-ellipsoid clusters. (Ng, Jordan, Weiss, 2002) stated that spectral clustering outperforms k-means not only on these high dimensional problems, but also in low-dimensional, multi-class data sets. Moreover, there are problems where Euclidean measures of distance required by k-means are not appropriate (for instance histograms), or others where there is not even a natural vector space representation. Also, spectral clustering provides a simple way of combining dissimilar vector spaces, like in this case text, keypoint and color features.

4 Experiments and results

In the first set of experiments, we used all features for clustering. We considered three levels of sense granularity: (1) all senses (*All*), (2) merging related senses with their corresponding core sense (*Meta*), (3) just the core senses (*Core*). For experiments (1) and (2), we used 40 clusters and all labeled images. For (3), we considered only images labeled with core senses, and thus reduced the number of clusters to 20 for a more fair comparison. Results were evaluated according to global cluster purity, cf. Equation 3.[4]

$$\text{Global purity} = \sum_{\text{clusters}} \frac{\text{\# of most common sense in cluster}}{\text{total \# images}} \quad (3)$$

[4]Purity did not include the small set of outlier images, defined as images whose ratio of distances to the second closest and closest clusters was below a threshold.

Word	All senses	Meta senses	Core senses
BASS	6 senses	4 senses	2 senses
Median	0.60	0.73	0.94
Range	0.03	0.02	0.02
Baseline	0.35	0.45	0.55
CRANE	9 senses	6 senses	4 senses
Median	0.49	0.65	0.86
Range	0.05	0.07	0.07
Baseline	0.27	0.37	0.50
SQUASH	6 senses	4 senses	2 senses
Median	0.52	0.71	0.94
Range	0.03	0.04	0.03
Baseline	0.32	0.56	0.64

Table 2: **Median and range of global clustering purity** for 5 runs with different initializations. For each keyword, the table lists the number of senses, median, and range of global cluster purity, followed by the baseline. **All** senses used the full set of sense labels and 40 clusters. **Meta** senses merged core senses with their respective related senses, considering all images and using 40 clusters. **Core** senses were clustered into 20 clusters, using only images labeled with core sense labels. Purity was stable across runs, and peaked for **Core**. The baseline reflected the frequency of the most common sense.

Word	Img	TxtWin	BodyTxt	Baseline
BASS				
Median	0.71	0.83	0.93	0.55
Range	0.05	0.03	0.05	
CRANE				
Median	0.61	0.84	0.85	0.50
Range	0.07	0.04	0.05	
SQUASH				
Median	0.71	0.91	0.96	0.64
Range	0.05	0.04	0.03	

Table 3: **Global and local features' performance**. **Core sense** images were grouped into 20 clusters, on the basis of individual feature types, and global cluster purity was measured. The table lists the median and range from 5 runs with different initializations. **Img** included just image features; **TxtWin** local tokens in a ± 10 window around the target image anchor; **BodyTxt** global tokens in the page BODY; and **Baseline** uses the most common sense. Text performed better than image features, and global text appeared better than local. All features performed above the baseline.

Median and range results are reported for five runs, given each condition, comparing against the baseline (i.e. choosing the most common sense). Table 2 shows that purity was surprisingly good, stable across query terms, and that it was highest when only core sense data was considered. In addition, purity tended to be slightly higher for BASS, which may be related to the annotator being less confident about its fine-grained sense distinctions, and thus less strict for assigning core sense labels for this query term.[5] In addition, we looked at the relative performance of individual global and local features using 20 clusters and only core

[5]A slightly modified HTML extractor yielded similar results ($\pm$0-2% median, $\pm$0-5% range cf. to Tables 2 - 4).

Figure 4: **First 30 images from a CRANE BIRD cluster** consisting of 81 images in the median run. Individual cluster purity for all senses was 0.67, and for meta senses 0.83. Not all clusters were as pure as this one; global purity for all 40 cluster was 0.49. This cluster appeared to show some iconography; mostly standing cranes. Interestingly, another cluster contained several images of flying cranes. Most weighted tokens: *cranes whooping birds wildlife species*. Table 1 has sense labels.

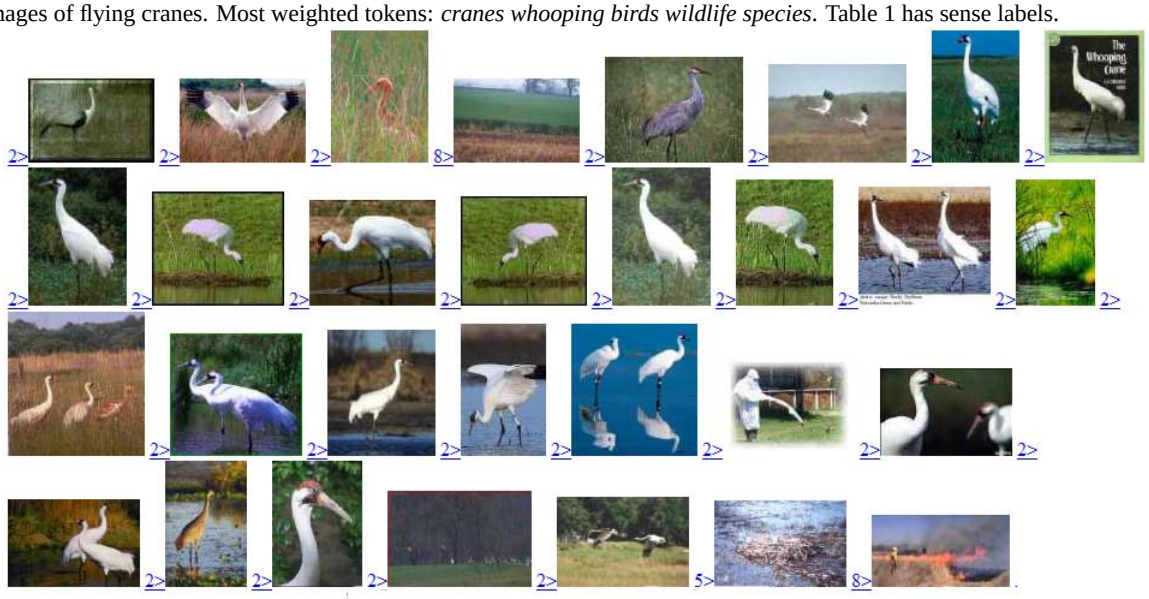
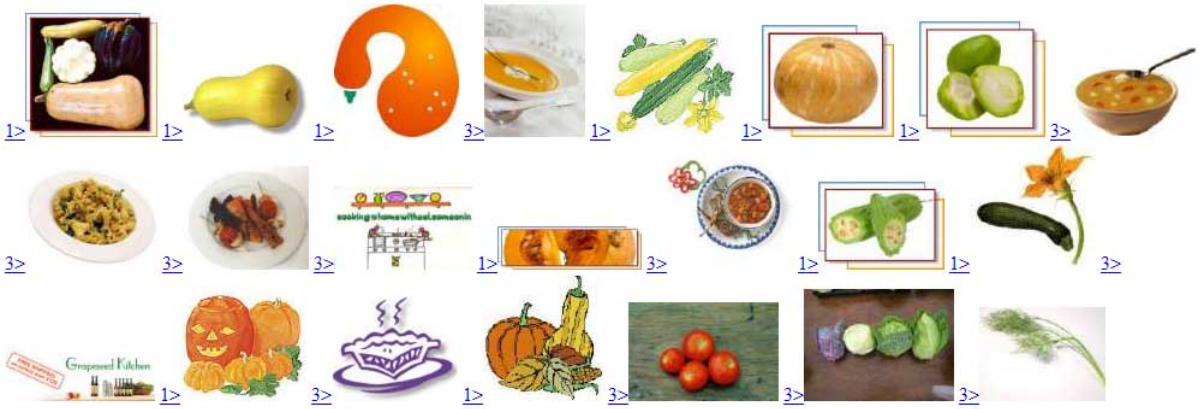


Figure 5: **Global purity does not tell the whole story** SQUASH VEGETABLE cluster of 22 images in the median run. Individual cluster purity for all senses was 0.5, and for meta senses 1.0. Global purity for all 40 cluster was 0.52. This cluster both shows visually coherent images, and a sensible meta semantic field. Most weighted tokens: *chayote calabaza add bitter cup*. Presumably, some tokens reflect the vegetable's use within the cooking domain.



sense data based on a particular feature. Table 3 shows that global text features were most informative (although not homogenously), but also that each feature type performed better than the baseline in isolation. This indicates that an optimal feature combination may improve over current performance, using manually selected parameters. In addition, purity is not the whole story. Figs. 4 and 5 show examples of two selected interesting clusters obtained for CRANE and SQUASH, respectively, using combined image and text features and all individual senses.[6] Inspection of image clusters indicated that image features, both in isolation and when used in combination, appeared to contribute to more visually balanced clusters, especially in terms of colors and shading. This shows that further exploring image features may be vital for attaining more subtle iconographic senses. Moreover, as discussed in the introduction, images are not necessarily anchored in the immediate text which they refer to. This could explain why local text features do not perform as well as global ones. Lastly, in addition, Fig. 6 shows an example of a partial cluster where the algorithm inferred a specific related sense.

We also experimented with different number of clusters for BASS. The results are in Table 4, lacking a clear trend, with comparable variation to different initializations. This is surprising, since we would expect purity to increase with number of

[6]The *UIUC-ISD data set* and results are currently at `http://www.visionpc.cs.uiuc.edu/isd/`.

Figure 6: **RELATED: SQUASH VEGETABLE cluster, consisting of 27 images.** The algorithm discovered a specific SQUASH BUG-PLANT sense, which appears iconographic. Individual cluster purity for all senses was 0.85, and individual meta purity: 1.0. Global purity for all 40 clusters: 0.52. Most weighted tokens: *bugs bug beetle leaf-footed kentucky*.

# Clusters	6	10	20	40	80
All					
Median	0.61	0.55	0.58	0.60	0.61
Range	0.03	0.05	0.03	0.03	0.04
Meta					
Median	0.75	0.70	0.70	0.73	0.72
Range	0.04	0.07	0.04	0.02	0.04

Table 4: **Impact of cluster size?** We ran BASS for different number of clusters (5 runs each with distinct initializations), and recorded median and range of global purity for all six senses of the query term, and for the four meta senses, without a clear trend.

clusters (Schütze, 1998), but may be due to the spectral clustering. Inspection showed that 6 clusters were dominated by core senses, whereas with 40 clusters a few were also dominated by RELATED senses or PEOPLE. No cluster was dominated by an UNRELATED label, which makes sense since semantic linkage should be absent between unrelated items.

5 Comparison to previous work

Space does not allow a complete review of the WSD literature. (Yarowsky, 1995) demonstrated that semi-supervised WSD could be successful. (Schütze, 1998) and (Lin and Pantel, 2002a, b) show that clustering methods are helpful in this area.

While ISD has received less attention, image categorization has been approached previously by adding text features. For example, (Frankel, Swain, and Athitsos, 1996)'s WebSeer system attempted to mutually distinguish photos, hand-drawn, and computer-drawn images, using a combination of HTML markup, web page text, and image information. (Yanai and Barnard, 2005) found that adding text features could benefit identifying relevant web images. Using text-annotated images (i.e. images annotated with relevant keywords), (Barnard and Forsyth, 2001) clustered them exploring a semantic hierarchy; similarly (Barnard, Duygulu, and Forsyth, 2002) conducted art clustering, and (Barnard and Johnson, 2005) used text-annotated images to improve WSD. The latter paper obtained best results when combining text and image features, but contrary to our findings, image features performed better in isolation than just text. They did use a larger set of image features and segmentation, however, we suspect that differences can rather be attributed to corpus type. In fact, (Yanai, Shirahatti, and Barnard, 2005) noted that human evaluators rated images obtained via a keyword retrieval method higher compared to image-based retrieval methods, which they relate to the importance of semantics for what humans regard as matching, and because pictorial semantics is hard to detect.

(Cai et al, 2004) use similar methods to rank visual search results. While their work does not focus explicitly on sense and does not provide in-depth discussion of visual sense phenomena, these do appear in, for example, figs. 7 and 9 of their paper. An interesting aspect of their work is the use of page layout segmentation to associate text with images in web documents. Unfortunately, the au-

thors only provide an illustrative query example, and no numerical evaluation, making any comparison difficult. (Wang et al, 2004) use similar features with the goal to improve image retrieval through similarity propagation, querying specific web sites. (Fuji and Ishikawa, 2005) deal with image ambiguity for establishing an online multimedia encyclopedia, but their method does not integrate image features, and appears to depend on previous encyclopedic background knowledge, limited to a domain set.

6 Conclusion

It is remarkable how high purity is, considering that we are using relatively simple image and text representation. In most corpora used to date for research on illustrated text, word sense is an entirely secondary phenomenon, whereas our data set was collected as to emphasize possible ambiguities associated with word sense. Our results suggest that a surprisingly degree of the meaning of an illustrated object is exposed on the surface.

This work is an initial attempt at addressing the ISD problem. Future work will involve learning the algorithm's parameters without supervision, and develop a semantically meaningful image taxonomy. In particular, we intend to explore the notion of iconographic senses; surprisingly good results on image classification by (Chapelle, Haffner, and Vapnik, 1999) using image features suggest that iconography plays an important role in the semantics of images. An important aspect is to enhance our understanding of the interplay between text and image features for this purpose. Also, it remains an unsolved problem how to enumerate iconographic senses, and use them in manual annotation and classification. Experimental work with humans performing similar tasks may provide increased insight into this issue, and can also be used to validate clustering performance.

7 Acknowledgements

We are grateful to Roxana Girju and Richard Sproat for helpful feedback, and to Alexander Sorokin.

References

C. O. Alm, N. Loeff, and D. Forsyth. 2006. Challenges for annotating images for sense disambiguation. *ACL workshop on Frontiers in Linguistically Annotated Corpora*.

K. Barnard and D. Forsyth. 2001. Learning the semantics of words and pictures. *ICCV*, 408–415.

K. Barnard, P. Duygulu, and D. Forsyth. 2002. Modeling the statistics of image features and associated text. *SPIE*.

K. Barnard and M. Johnson. 2005. Word sense disambiguation with pictures. *Artificial Intelligence*, 167, 13–30.

D. Cai et al. 2004. Hierarchical clustering of WWW image search results using visual, textual and link information. *ACM Multimedia*, 952-959.

O. Chapelle and P. Haffner and V. Vapnik. 1999. Support vector machines for histogram-based image classification. *IEEE Neural Networks*, 10(5), 1055–1064.

G. Csurka et al. 2004. Visual categorization with bags of keypoints. *ECCV Int. Workshop on Stat. Learning in Computer Vision*.

C. Frankel, M. Swain, and V. Athitsos. 1996. WebSeer: an image search engine for the World Wide Web. *Univ. of Chicago, Computer Science, Technical report #96-14*.

C. Fowlkes, S. Belongie, F. Chung, and J. Malik. 2004. Spectral grouping using the Nyström method. *IEEE PAMI*, 26(2),214-225.

A. Fuji and T. Ishikawa. 2005. Toward the automatic compilation of multimedia encyclopedias: associating images with term descriptions on the web. *IEEE WI*, 536-542.

D. Jurafsky and J. Martin 2000. *Speech and Language Processing*, Prentice Hall.

T. Kadir and M. Brady. 2001. Scale, saliency and image description. *Int. Journal of Computer Vision*, 45 (2):83–105.

D. Lin and P. Pantel. 2002a. Concept discovery from text. *COLING*, 577–583.

D. Lowe. 2004. Distinctive image features from scale-invariant keypoints. *Int. Journal of Computer Vision*, 60(2), 91–110.

A. Ng, M. Jordan, and Y. Weiss. 2002. On spectral clustering: analysis and an algorithm. *NIPS 14*.

P. Pantel and D. Lin. 2002b. Discovering word senses from text. *KDD*, 613–619.

H. Schuetze. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123.

J. Shi and J. Malik. 2000. Normalized cuts and image segmentation. *IEEE PAMI*, 22(8):888–905.

N. Ueda. and R. Nakano. 1994. A new competitive learning approach based on an equidistortion principle for designing optimal vector quantizers. *Neural Networks*, 7(8):1211–1227.

X.-J. Wang et al. 2004. Multi-model similarity propagation and its application for image retrieval. *MM*,944–951.

K. Yanai and K. Barnard. 2005. Probabilistic web image gathering. *SIGMM*, 57–64.

K. Yanai, N. V. Shirahatti, and K. Barnard. 2005. Evaluation strategies for image understanding and retrieval. *SIGMM*, 217-226.

D. Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. *ACL*, 189–196.

Modeling Adjectives in Computational Relational Lexica

Palmira Marrafa
Department of Linguistics, Faculty of Arts,
University of Lisbon and
CLG – Group for the Computation of Lexical and Grammatical Knowledge,
Center of Linguistics – University of Lisbon,
Avenida Professor Gama Pinto, 2
1649-003 Lisbon Portugal
palmira.marrafa@netcabo.pt

Sara Mendes
CLG – Group for the Computation of Lexical and Grammatical Knowledge
Center of Linguistics – University of Lisbon
Avenida Professor Gama Pinto, 2
1649-003 Lisbon, Portugal
sara.mendes@clul.ul.pt

Abstract

In this paper we propose a small set of lexical conceptual relations which allow to encode adjectives in computational relational lexica in a principled and integrated way. Our main motivation comes from the fact that adjectives and certain classes of verbs, related in a way or another with adjectives, do not have a satisfactory representation in this kind of lexica. This is due to a great extent to the heterogeneity of their semantic and syntactic properties. We sustain that such properties are mostly derived from the relations holding between adjectives and other POS. Accordingly, our proposal is mainly concerned with the specification of appropriate cross-POS relations to encode adjectives in lexica of the type considered here.

1 Introduction

As well known, the experiment conducted by George Miller on the mental lexicon properties in the early 80s pointed out that lexical meaning is derived from a set of lexical and conceptual relations among concepts. Subsequently, a computational lexicon conceived as a semantic network has been built (the Princeton WordNet (Miller, 1990; Fellbaum, 1998)). Given its psychological plausibility and its crucial role for applications like machine translation, information retrieval and language learning systems, among many others, this relational lexicon model is being extensively adopted for machine lexical knowledge representations, playing a leading role in this field.

One of the most salient undertaking in this domain is EuroWordNet (Vossen, 1998), a multilingual database which stores wordnets for several European languages that follow the same main lines as the Princeton WordNet (Miller, 1990; Fellbaum, 1998) and are inter-related amongst them.

EuroWordNet wordnets follow the Princeton WordNet model, but they are richer concerning both the number and the nature of conceptual relations.

The work depicted here programmatically adopts the EuroWordNet framework.

In general terms, it deals with the specifications for an accurate modeling of lexical knowledge in a EuroWordNet wordnet-like database for Portuguese (WordNet.PT, henceforth), specifically focusing on the lexical semantics of adjectives.

Although WordNet.PT (Marrafa, 2001; Marrafa, 2002) is being developed in the general EuroWordNet framework, basic research has been carried out on Portuguese in order to guarantee the WordNet.PT accuracy. This work has already led to some changes and new directions (cf. Marrafa et al., (2006) and Amaro et al., (2006), for instance).

In this paper we propose a small set of new relations which allow a strongly empirical motivated encoding of the major POS in WordNet.PT, despite the fact that we particularly focus on adjectives. The empirical issues at stake are described in section 2. In section 3 we discuss the strategies adopted in previous work carried out both in WordNet and EuroWordNet frameworks, in order to make their shortcomings apparent. In section 4 we present our proposal

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 555–562,
Sydney, July 2006.

and argue for its relevance and soundness. Section 5 presents some results concerning the encoding of adjectives in WordNet.PT. We conclude the paper with some final remarks.

2 Empirical Issues

Adjective semantic analysis and representation is far from being a trivial issue, as adjectives show a very particular linguistic behavior, namely in what concerns sense change depending on linguistic context. Being so, there are several different typologies and classifications of adjectives in the literature: semantic based classifications, syntactic based classifications, classifications regarding the relation holding between the adjective and the modified noun, and so on.

As our work on this issue progresses, it has become clear that only a combination of syntactic and semantic criteria can offer interesting insights concerning adjective linguistic behavior and the identification of relevant common features, which may set the basis for an accurate modeling of this POS in computational relational lexica. In this section we will briefly look at some of the main adjective classifications.

Regarding the way adjectives relate to the noun they modify, we consider two classes: property ascribing adjectives (in (1)), which add a new restriction to the properties introduced by the modified noun; and reference modifying adjectives (in (2)), which behave like a semantic operator, taking the reference of the modified noun as its argument[1].

(1) o livro azul
'the blue book'
(2) o diamante falso
'the fake diamond'

Adjectives like *falso* (fake), for instance, deal with concepts instead of real or referential objects, showing how a concept applies to a particular object. These adjectives constitute a closed class with very particular properties, which makes them somewhat close to semantic operators. In this work we will therefore focus on property ascribing adjectives.

Demonte (1999) classifies property ascribing adjectives based on their intrinsic meaning, a classification combining syntactic and semantic criteria to determine which adjectives belong to which class. Two main subclasses are considered: descriptive adjectives and relational adjectives. Each of these classes displays specific semantic and syntactic properties.

In languages like Portuguese, descriptive adjectives can occur both in attributive and predicative contexts, while relational adjectives occur almost exclusively in attributive contexts[2]. Both prenominal and postnominal positions are possible for descriptive adjectives in attributive contexts. Relational adjectives, on the contrary, can only occur in postnominal position. Finally, descriptive adjectives are gradable, i.e. they can co-occur with degree adverbs, which is not the case for relational adjectives. However, these criteria are not always sufficient to make a clear-cut distinction between relational and descriptive adjectives. Demonte (1999) proposes some additional criteria in order to make a more accurate distinction between these adjectives: their occurrence in comparative structures, and the formation of polarity systems.

(3) a. O sabor desta laranja é mais doce do que o daquela.
'this orange taste is sweeter than that one's'
b. o rapaz alto / o rapaz baixo
'the tall boy / the short boy'
(4) a. *Este sabor é mais mineral do que aquele.
'this taste is more mineral than that one'
b. o sabor mineral / *o sabor amineral
'the mineral taste / the amineral taste'

But most of all, and besides all the syntactical contrasts we have mentioned above, there is a clear contrast in the way these two adjective classes relate to the noun they modify. Descriptive adjectives ascribe a single property, setting a value for an attribute, whereas relational adjectives introduce a set of properties.

(5) o prédio alto
'the high building'

1 This distinction between *property ascribing adjectives* and *reference modifying adjectives* is basically equivalent to the one used in the SIMPLE project (Lenci et al., 2000) (*extensional* vs. *intensional adjectives*, following Chierchia and McConnel-Ginet (1990)) to address the semantics of adjectives. This distinction is also included in the EAGLES recommendations for a semantic typology of adjectives.

2 Predicative contexts with relational adjectives are generally ruled out in Portuguese. Nonetheless, some specific contexts, like contrastive contexts, for instance, seem to license predicative uses of relational adjectives:

(I) As próximas eleições são autárquicas, não são presidenciais.
'next election will be autarchic, not presidential'

(6) a indústria alimentar
'the alimentary industry'

Looking at (5) and (6), we see that, while *alto* (high) sets the value of the **height** attribute of *prédio* (building) to **high**, *alimentar* (alimentary) does not ascribe a single property, but a set of properties to *indústria* (industry). Moreover, this set of properties corresponds to the main features describing another noun – *alimento* (food) in the example above. In fact, the way properties are ascribed to the modified nouns in (5) and in (6) are quite different. Ascribing a singular property usually corresponds to an incidence relation of this property in the nominal referent, while ascribing sets of properties usually entails more complex and diversified semantic relations.

However, despite the relevance of the descriptive/relational dichotomy, it cannot account for the following contrasts:

(7) a. *Ele viu a Maria alta.
'He saw Mary tall'
b. Ele viu a Maria triste.
'He saw Mary sad'.

Both *alta* and *triste* are descriptive adjectives, but they do not behave in the same way regarding secondary predication.

We can refine the classification, considering, for instance, the opposition between accidental properties and permanent or inherent properties (this distinction goes back to Milsark (1974; 1977) and Carlson (1977)). According to this distinction, the property denoted by *alta* (tall) belongs to the latter class and the property denoted by *triste* (sad) to the former one. However, as pointed out by Marrafa (2004) and previous work, the characterization of adjectives on the basis of this dichotomy is not straightforward, since certain adjectives are ambiguous with regard to those properties, as it is the case of *triste* (sad). In the example above *triste* (sad) denotes an accidental property, but in an expression like um *livro triste* (a sad book) it denotes a permanent property.

Intuitively, we can say that *triste* (sad) expresses a state of *tristeza* (sadness), but we let the discussion of the status of this relation out of the scope of this paper.

Nevertheless, this kind of adjectives is of great importance to model telic verbs. The semantics of telic verbs involves a change of state of their theme argument, i.e. the subevent that closes the whole event is an atomic event, (a state) that affects the theme and is different from its initial state. As argued in Marrafa (2005) and previous work, by default, verbs like *lavar* (to wash) are associated to the following Lexical-Conceptual Structure (LCS' in Pustejovsky (1991)):

(8) $[_T [_P$ act(x,y)and ~ Q(y)], $[_e$Q(y)]]
T:transition, P:process, e: event, Q: atomic event

When syntactically realized, the telic subevent generally corresponds to an adjectival constituent, like in the example below:

(9) Ele lavou a camisa bem lavada.
'He washed the shirt well washed'

In (9) the absence of the telic expression *bem lavada* (well washed) does not induce ungrammaticality. However, in the case of verbs like *tornar* (to make*)*, it seems impossible to assign a value to *Q* independently of the telic expression.

(10) a. Ele tornou a Maria triste.
'He made Mary sad'
b. *Ele tornou a Maria.
'He made Mary'

Along the lines of Marrafa (1993) and further work, verbs like *tornar* (to make*)* are assumed here to be LCS deficitary, the telic expression filling the gap of the LCS of the verb.

As shown below, the troponyms of these verbs incorporate the telic state:

(12) a. Ele entristeceu a Maria.
'He saddened Mary'
b. *Ele entristeceu a Maria triste.
'He saddened Mary sad'

The grammaticality contrast above is due to the fact that *entristecer* (to sadden) incorporates the telic state. This justifies that this verb can be paraphrased by *tornar triste* (to make sad).

In this section we have mainly focused on property ascribing adjectives. We have considered two main subclasses, descriptive and relational adjectives, briefly presenting their syntactic and semantic behavior with regard to gradability, formation of polarity systems and their occurrence in predicative and attributive (both pronominally and postnominally) contexts and comparative structures. We have also addressed the issue of adjective relation with the noun they modify. Different adjective behavior regarding secondary predication is also discussed and analyzed in terms of the opposition between acci-

dental and permanent properties. The properties discussed in this section should be encoded in computational relational lexica such as wordnets.

3 Adjectives in WordNet and in EuroWordNet

Hyponymy is the main structuring relation both in WordNet and in EuroWordNet. However, the semantic organization of adjectives is entirely different from that of other POS: nothing like the hierarchies of hyponymic (in the semantic organization of nouns) and troponymic relations (in the semantic organization of verbs) is available for adjectives. Even if it is possible to find some small local hierarchies, hyperonymy/hyponymy is far from being the crucial semantic relation in the organization of adjectives in relational lexical databases such as wordnets.

However, some authors working within the EuroWordNet framework have reconsidered the possibility of encoding hyponymy for adjectives. Hamp and Feldweg (1998), in the development of GermaNet, abandon the cluster organization of WordNet in favor of a hierarchical structuring of adjectives, arguing for a uniform treatment of all POS. Even though taxonomic chains of adjectives yield rather flat in comparison to those of nouns and verbs, these authors claim to derive more structural information from these small taxonomies than from clusters, as they seek to eliminate what they consider to be the 'rather fuzzy concept of indirect antonyms'. Even though the concept of indirect antonymy is not completely clear, it is not obvious to us why this fact should entail that adjectives must show a hierarchical organization instead.

In ItalWordNet, Alonge et al. (2000) also organize adjectives into classes sharing a superordinate. These classes correspond to adjectives sharing some semantic features, and are generally rather flat. These authors argue for the possibility of inferring semantic preferences and syntactic characteristics of adjectives found in the same taxonomy. The SIMPLE project addresses the semantics of adjectives in a similar way, identifying a set of common features relevant for classifying and describing adjective behavior. However, as noted by Peters and Peters (2000), even though similarities exist "adjectives belonging to the same semantic class may differ from each other in numerous ways", i.e. the classes established in this way are not homogeneous.

In WordNet, descriptive and relational adjectives are distinguished, first, by being encoded in separate files, and second, by the relations holding between synsets.

Descriptive adjectives are organized in clusters of synsets, each cluster being associated by semantic similarity to a focal adjective which is linked to a contrasting cluster through an antonymy relation. Therefore, antonymy is the basic semantic relation used in WordNet to encode descriptive adjectives. As argued for in Miller (1998), this cluster organization of adjectives seems to mirror psychological principles. In fact, this organization is clearly motivated if we recognize that these adjectives main function regards the expression of attributes, and that an important number of attributes are bipolar.

Relational adjectives, on the other hand, do not have antonyms. Therefore, they cannot be organized in opposite clusters. As pointed out by Levi (1978), the intrinsic meaning of these adjectives is something along the following lines: 'of, relating/pertaining to, associated with' some noun. The way these adjectives are encoded in WordNet mirrors this as it links relational adjectives to the nouns they relate to.

In GermaNet a distinct treatment of relational and descriptive adjectives is abandoned, as the distinction between these two classes is considered to be 'not at all clear'. Nonetheless, the WordNet strategy for distinguishing between different adjective classes is maintained: listing lexical items in different files[3].

As pointed out in the previous section, even if the distinction between these two classes is not always clear-cut, testing adjectives against the set of syntactic and semantic criteria presented in section 2 allows us to distinguish descriptive from relational adjectives. We consider that this distinction can be mirrored in the database via the semantic relations expressed in the network, adjective listing in different files not being therefore necessary. In order to do this we propose several cross-POS relations, since in the EuroWordNet model, unlike what happens in WordNet where each POS forms a separate system, it is possible to relate lexical items belonging to different POS. Such an approach has the

[3] GermaNet classifies the adjectives into 15 semantic classes, following the classes proposed by Hundsnurscher and Splett (1982), with some minor changes: perceptional, spatial, temporality-related, motion-related, material-related, weather-related, body-related, mood-related, spirit-related, behaviour-related, social-related, quantity-related, relational and general adjectives. One special class is added for pertainyms.

advantage of coping with adjective representation in lexical semantic databases without using strategies external to the lexical model, such as *a priori* semantic classes or separate files corresponding to different classes.

4 Relating adjectives, nouns and verbs

It is undeniable that important structural information can be extracted from the hierarchical organization of lexical items, namely of nouns and verbs. However, extending wordnets to all the main POS involves a revision of certain commonly used relations and the specification of several cross-POS relations.

We previously mentioned that adjectives show a very particular semantic organization. Thus, encoding adjectives in wordnets calls for the specification of a number of cross-POS semantic relations. Here we use these cross-POS semantic relations to mirror adjectives main features in wordnet-like databases, which allows us to make adjective classes emerge from the relations expressed in the network.

According to the strategies discussed in Mendes (2006), we present here the relations we argue are appropriate to encode adjectives and show how they conform to some complex phenomena.

4.1 Relating Adjectives and Nouns

To put it somewhat simplistically, descriptive adjectives ascribe a value of an attribute to a noun. We link each descriptive adjective to the attribute it modifies via the semantic relation *characterizes with regard to/can be characterized by*[4]. Thus, instead of linking adjectives amongst themselves by a similarity relation, following what is done in WordNet, all adjectives modifying the same attribute are linked to the noun that lexicalizes this attribute. This way, and in combination with the *antonymy* relation, we obtain the cluster effect argued to be the basis of the organization of adjectives (Miller, 1998; Fellbaum et al, 1993), without having to encode it directly in the database.

As shown by word association tests, *antonymy* is also a basic relation in the organization of descriptive adjectives. Nonetheless, this relation does not correspond to conceptual opposition, which is one of the semantic relations used for the definition of adjective clusters. We argue that conceptual opposition does not have to be explicitly encoded in wordnets, since it is possible to infer it from the combination of *synonymy* and *antonymy* relations (see Mendes (2006) for more details).

Concerning relational adjectives, even though they are also property ascribing adjectives, they entail more complex and diversified relations between the set of properties they introduce and the modified noun, often pointing to the denotation of another noun (cf. section 2). We use the *is related to* relation to encode this.

Therefore, the *characterizes with regard to*/*can be characterized by* and the *antonymy* relations, for descriptive adjectives, and the *is related to* relation for relational adjectives, allows us to encode the basic features of these adjectives in computational relational lexica such as wordnets, while making it possible to derive membership to these classes from the relations expressed in the network.

Another issue regarding adjectives is that they have a rather sparse net of relations. We introduce a new relation to encode salient characteristics of nouns: *is characteristic of*/*has as a characteristic to be*. These characteristics are often expressed by adjectival expressions. Although in terms of lexical knowledge we can discuss the status of this relation, it regards crucial information for many wordnet-based applications, namely those using inference systems, allowing for richer and clearer synsets.

Also, it may allow for deducing semantic domains from the database, as it makes it possible to identify the typical semantic domains of application of adjectives. Research on the classes and semantic domains emerging from the relations expressed in the database is still ongoing.

Thus, the combination of these relations allows us to encode a less sparse net of adjectives. Besides the importance of having a more dense net from the point of view of wordnet-based applications, as mentioned above, this is also crucial with regard to relational lexica such as wordnets themselves, as the meaning of each unit is determined by the set of relations it holds with other units. Thus, a denser network of relations allows for richer and clearer synsets. Fig. 1 illustrates this idea, presenting an example of the way adjectives are being encoded in WordNet.PT.

4 This semantic relation is very close to the *is a value of*/*attributes* relation used in WordNet. We have changed its label in order to make it more straightforward to the common user.

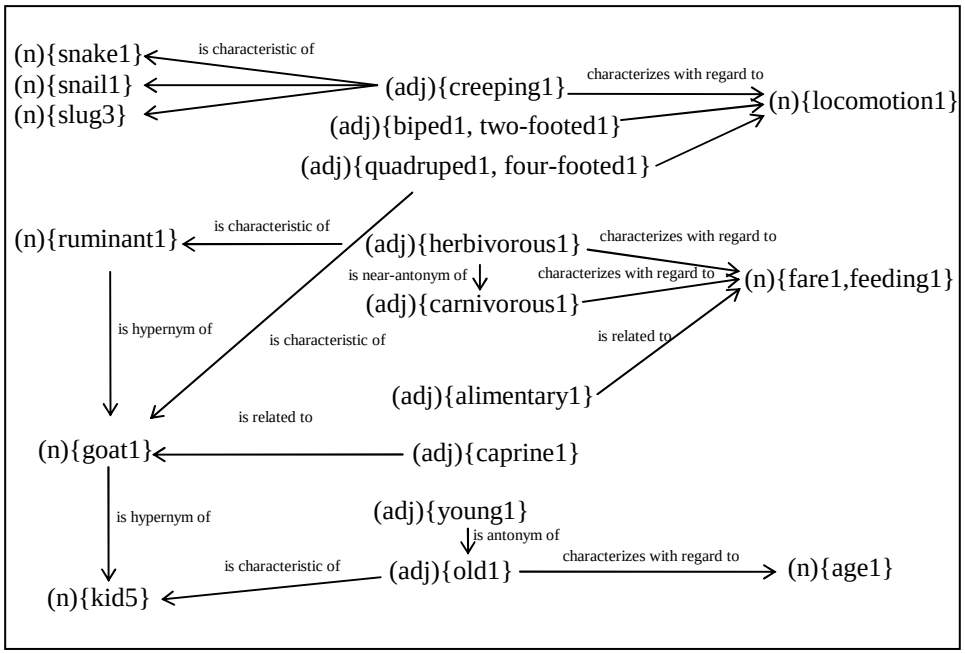


Figure 1. Fragment showing relations between adjectives and nouns[5]

4.2 Relating Adjectives and Verbs

We also introduce new semantic relations to encode telic verbs in the database (on this issue see also Marrafa, 2005; Amaro et al., 2006).

As shown in section 2, the facts render evident that the representation of LCS deficitary telic verbs has to include information regarding the telic expression. Obviously, it would not be adequate to overtly include in the synset all the expressions that can integrate the predicate, among other reasons, because they seem to constitute an open set. Rather, we claim that we can capture the telicity of these verbs by including a new relation in the set of internal relations of wordnets: the *telic sub-event* relation, as exemplified below.

(13) {*make*} has_telic_sub-event {*state*}
{*state*} is_telic_sub-event_of{*make*}
(defeasible)[6]

Relating *make* to *state* by means of this relation, we capture the telic properties of the verb and let the specific nature of the final state underspecified. This way, we also account for the weakness of the verb selection restrictions. As expected, we can also use this relation to encode telicity in the case of the troponyms of the class of verbs discussed in section 2.

In these cases, we use the *telic sub-event* relation to relate the verb to the expression corresponding to the incorporated telic information:

(14) {*sadden*} has_telic_sub-event {*sad*}
{*sad*} is_telic_sub-event of {*sadden*}
(defeasable)

The global solution is schematically presented below:

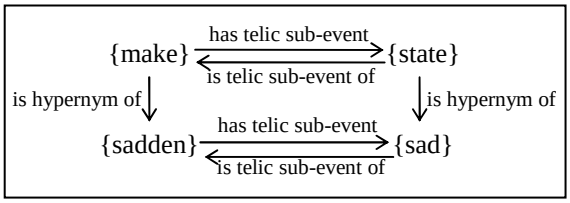


Figure 2. Relations between adjectives and verbs

As shown, the *telic sub-event* relation straightforwardly allows the encoding of lexical telicity in wordnets, in accordance with the empirical evidence.

It should be noticed that the existing *sub-event* relation in the EuroWordNet framework is different from the relation proposed here. It only stands for lexical entailment involving temporal proper inclusion. Therefore, it does not account for the geometry of the event. On the contrary, the *telic sub-event* relation regards the atomic sub-event that is the ending point of the global event.

[5] Word senses presented here correspond to Princeton WordNet synsets (2.1 version).

[6] The relation is not obligatory in this direction.

5 Encoding adjectives in WordNet.PT

As previously mentioned, the proposal presented in this paper is mainly concerned with the specification of appropriate cross-POS relations to encode adjectives in computational relational lexica.

In order to test whether the set of relations presented here is appropriate and allows the encoding of adjectives in wordnet-like lexica, we have introduced a selection of Portuguese adjectives in WordNet.PT.

In the first phase of the WordNet.PT project mostly nouns were encoded in the database. Thus, we have mainly focused on the encoding of relations between adjectives and nouns[7]. Table 1 presents the number of entries and relations specified at the present stage.

total number of adjectives	1462
synonymy relation	252
antonymy relation	134
near-antonymy relation	40
is related to relation	331
is characteristic of relation	1293
characterizes with regard to relation	261
total number of relations	2311

Table1. Statistics concerning the encoding of adjectives in WordNet.PT

Besides the discussion presented above, the implemented data, being already a representative sample, show that the cross-POS relations proposed here effectively allow for a fine-grained encoding of adjectives in relational lexica (specifically in wordnet-like lexica) through the specification of a denser network of relations.

6 Conclusion

In this paper we argue that the semantics of adjectives can be appropriately captured in wordnet-like lexica by means of the implementation of a small set of new relations, which have a strong linguistic motivation and preserve the coherence of the model.

We focus on property ascribing adjectives and we distinguish between descriptive and relational adjectives. Besides the relevance of this dichotomy, we also address the opposition between accidental and permanent properties, as adjective association to certain kind of properties determines their syntactic and semantic behavior, namely with regard to secondary predication. Here, we model these distinctions in WordNet.PT via cross-POS relations: *characterizes with regard to/can be characterized by* to model descriptive adjectives introducing permanent properties; *has_telic_subevent/is_telic_subevent* to model descriptive adjectives associated to accidental properties; and the *is related to* to model relational adjectives.

Moreover, we make apparent that increasing the expressive power of the system has an important impact in precision concerning the specifications of all POS, mainly induced by the cross-POS relations.

This way, we provide a simple and integrated solution for a complex and heterogeneous problem.

[7] Nevertheless, relations between adjectives and verbs are already being implemented at the current stage.

7 Acknowledgements

We wish to thank Fundação para a Ciência e Tecnologia who has partially funded the research presented in this paper (grant SFRH/BD/8524/2002). We also have to thank Instituto Camões for the support it has been giving to our research in computational relational lexica.

References

A. Alonge, F. Bertagna, N. Calzolari, A. Roventini and A. Zampoli. 2000. Encoding information on adjectives in a lexical-semantic net for computational applications. *Proceedings of NAACL 2000*. Seattle, pp. 42-49.

R. Amaro, R. P. Chaves, P. Marrafa and S. Mendes. 2006. Enriching wordnets with new Relations and with event and argument structures. *Proceedings of CICLing 2006 – Conferences on Computational Linguistics and Intelligent Text Processing*. Mexico City, Mexico, pp. 28-40.

G. Carlson. 1977. *Reference to Kinds in English*, PhD dissertation, University of Massachusetts-Amherst.

G. Chierchia and S. McConnel-Ginet. 1990. *Meaning and Grammar: an Introduction to Semantics*, Cambridge, MA: The MIT Press.

V. Demonte. 1999. El Adjectivo: classes y usos. La posición del adjectivo en el sintagma nominal. in I. Bosque and V. Demonte (orgs.) *Gramática Descriptiva de la Lengua Española.* volume 1. Madrid: Espasa.

EAGLES Lexicon Interest Group. 1998. *Preliminary Recommendations on Semantic Encoding Interim Report*.

C. Fellbaum, D. Gross and K. J. Millar. 1993. Adjectives in WordNet. in Miller et al., *Five papers on*

WordNet, Technical Report, Cognitive Science Laboratory, Princeton University, pp. 26–39.

C. Fellbaum. 1998 A Semantic Network of English: The Mother of all WordNets. in P. Vossen (ed.) *EuroWordNet: A Multilingual Database with Lexical Semantic Networks.* Dordrecht: Kluwer Academic Publishers, pp. 137-148.

B. Hamp and H. Feldweg. 1997. GermaNet – a Lexical Semantic Net for German. *Proceedings of ACL workshop on Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications.* Madrid.

A. Lenci, N. Bel, F. Busa, N. Calzolari, E. Gola, M. Monachini, A. Ogonoski, I. Peters. W. Peters, N. Ruimy, M. Villegas & A. Zampolli. 2000. SIMPLE - A General Framework for the Development of Multilingual Lexicons. in T. Fontenelle (ed.) *International Journal of Lexicography.* volume 13. pp. 249-263. Oxford University Press.

J. N. Levi. 1978. *The Syntax and Semantic of complex nominals*, New York: Academic Press.

P. Marrafa. 1993. *Predicação Secundária e Predicados Complexos: Modelização e Análise*, PhD. dissertation, Lisbon, University of Lisbon.

P. Marrafa. 2001. *WordNet do Português: uma base de dados de conhecimento linguístico*, Lisboa: Instituto Camões.

P. Marrafa. 2002. Portuguese WordNet: general architecture and internal semantic relations. *D.E.L.T.A.*, 18.

P. Marrafa. 2004. Modelling Constituency and Predication in Portuguese. *Revista PaLavra*. volume 12 (special issue: Linguística Computacional), pp. 106-118.

P. Marrafa. 2005. The Representation of Complex Telic Predicates in WordNets: the Case of Lexical-Conceptual Structure Deficitary Verbs. *Research on Computing Science*. volume 12, pp. 109–116.

P. Marrafa, R. Amaro, R. P. Chaves, S. Lourosa, C. Martins and S. Mendes. 2006. WordNet.PT new directions. *Proceedings of GWC'06: 3rd International Wordnet Conference.* Jeju Island, Korea.

S. Mendes. 2006. Adjectives in WordNet.PT. *Proceedings of the GWA 2006 – Global WordNet Association Conference*. Jeju Island, Korea.

G. A. Miller. 1990. WordNet: an on-line Lexical Database. *Special Issue of International Journal of Lexicography*. volume 3, nº 4.

K. J. Miller. 1998. Modifiers in WordNet. in C. Fellbaum (ed.) *WordNet: an electronic lexical database.* Cambridge, MA: The MIT Press, pp. 47-68.

G. Milsark. 1974. *Existential Sentences in English*. PhD dissertation, MIT.

G. Milsark. 1977. Toward an Explanation of Certain Pecularities of the Existencial Construction in English. *Linguistic Analysis*, 3, pp. 1-29.

I. Peters and W. Peters. 2000. The Treatment of Adjectives in SIMPLE: Theoretical Observations. *Proceedings of LREC 2000*.

J. Pustejovsky. 1991. The Syntax of Event Structure. *Cognition*, 41, pp. 47–81.

P. Vossen. 1998. (ed.) *EuroWordNet: A Multilingual Database with Lexical Semantic Networks*, Dordrecht: Kluwer Academic Publishers.

Segmented and unsegmented dialogue-act annotation with statistical dialogue models*

Carlos D. Martínez Hinarejos, Ramón Granell, José Miguel Benedí
Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Camino de Vera, s/n, 46022, Valencia
{cmartine,rgranell,jbenedi}@dsic.upv.es

Abstract

Dialogue systems are one of the most challenging applications of Natural Language Processing. In recent years, some statistical dialogue models have been proposed to cope with the dialogue problem. The evaluation of these models is usually performed by using them as annotation models. Many of the works on annotation use information such as the complete sequence of dialogue turns or the correct segmentation of the dialogue. This information is not usually available for dialogue systems. In this work, we propose a statistical model that uses only the information that is usually available and performs the segmentation and annotation at the same time. The results of this model reveal the great influence that the availability of a correct segmentation has in obtaining an accurate annotation of the dialogues.

1 Introduction

In the Natural Language Processing (NLP) field, one of the most challenging applications is dialogue systems (Kuppevelt and Smith, 2003). A dialogue system is usually defined as a computer system that can interact with a human being through dialogue in order to complete a specific task (e.g., ticket reservation, timetable consultation, bank operations,...) (Aust et al., 1995; Hardy et al., 2002). Most dialogue system have a characteristic behaviour with respect to dialogue management, which is known as dialogue strategy. It defines what the dialogue system must do at each point of the dialogue.

Most of these strategies are rule-based, i.e., the dialogue strategy is defined by rules that are usually defined by a human expert (Gorin et al., 1997; Hardy et al., 2003). This approach is usually difficult to adapt or extend to new domains where the dialogue structure could be completely different, and it requires the definition of new rules.

Similar to other NLP problems (like speech recognition and understanding, or statistical machine translation), an alternative data-based approach has been developed in the last decade (Stolcke et al., 2000; Young, 2000). This approach relies on statistical models that can be automatically estimated from annotated data, which in this case, are dialogues from the task.

Statistical modelling learns the appropriate parameters of the models from the annotated dialogues. As a simplification, it could be considered that each label is associated to a situation in the dialogue, and the models learn how to identify and react to the different situations by estimating the associations between the labels and the dialogue events (words, the speaker, previous turns, etc.). An appropriate annotation scheme should be defined to capture the elements that are really important for the dialogue, eliminating the information that is irrelevant to the dialogue process. Several annotation schemes have been proposed in the last few years (Core and Allen, 1997; Dybkjaer and Bernsen, 2000).

One of the most popular annotation schemes at the dialogue level is based on Dialogue Acts (DA). A DA is a label that defines the function of the annotated utterance with respect to the dialogue process. In other words, every turn in the dialogue

* Work partially supported by the Spanish project TIC2003-08681-C02-02 and by Spanish Ministry of Culture under FPI grants.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 563–570,
Sydney, July 2006.

is supposed to be composed of one or more utterances. In this context, from the dialogue management viewpoint an utterance is a relevant subsequence . Several DA annotation schemes have been proposed in recent years (DAMSL (Core and Allen, 1997), VerbMobil (Alexandersson et al., 1998), Dihana (Alcácer et al., 2005)).

In all these studies, it is necessary to annotate a large amount of dialogues to estimate the parameters of the statistical models. Manual annotation is the usual solution, although is very time-consuming and there is a tendency for error (the annotation instructions are not usually easy to interpret and apply, and human annotators can commit errors) (Jurafsky et al., 1997).

Therefore, the possibility of applying statistical models to the annotation problem is really interesting. Moreover, it gives the possibility of evaluating the statistical models. The evaluation of the performance of dialogue strategies models is a difficult task. Although many proposals have been made (Walker et al., 1997; Fraser, 1997; Stolcke et al., 2000), there is no real agreement in the NLP community about the evaluation technique to apply.

Our main aim is the evaluation of strategy models, which provide the reaction of the system given a user input and a dialogue history. Using these models as annotation models gives us a possible evaluation: the correct recognition of the labels implies the correct recognition of the dialogue situation; consequently this information can help the system to react appropriately. Many recent works have attempted this approach (Stolcke et al., 2000; Webb et al., 2005).

However, many of these works are based on the hypothesis of the availability of the segmentation into utterances of the turns of the dialogue. This is an important drawback in order to evaluate these models as strategy models, where segmentation is usually not available. Other works rely on a decoupled scheme of segmentation and DA classification (Ang et al., 2005).

In this paper, we present a new statistical model that computes the segmentation and the annotation of the turns at the same time, using a statistical framework that is simpler than the models that have been proposed to solve both problems at the same time (Warnke et al., 1997). The results demonstrate that segmentation accuracy is really important in obtaining an accurate annotation of the dialogue, and consequently in obtaining quality strategy models. Therefore, more accurate segmentation models are needed to perform this process efficiently.

This paper is organised as follows: Section 2, presents the annotation models (for both the unsegmented and segmented versions); Section 3, describes the dialogue corpora used in the experiments; Section 4 establishes the experimental framework and presents a summary of the results; Section 5, presents our conclusions and future research directions.

2 Annotation models

The statistical annotation model that we used initially was inspired by the one presented in (Stolcke et al., 2000). Under a maximum likelihood framework, they developed a formulation that assigns DAs depending on the conversation evidence (transcribed words, recognised words from a speech recogniser, phonetic and prosodic features,...). Stolcke's model uses simple and popular statistical models: N-grams and Hidden Markov Models. The N-grams are used to model the probability of the DA sequence, while the HMM are used to model the evidence likelihood given the DA. The results presented in (Stolcke et al., 2000) are very promising.

However, the model makes some unrealistic assumptions when they are evaluated to be used as strategy models. One of them is that there is a complete dialogue available to perform the DA assignation. In a real dialogue system, the only available information is the information that is prior to the current user input. Although this alternative is proposed in (Stolcke et al., 2000), no experimental results are given.

Another unrealistic assumption corresponds to the availability of the segmentation of the turns into utterances. An utterance is defined as a dialogue-relevant subsequence of words in the current turn (Stolcke et al., 2000). It is clear that the only information given in a turn is the usual information: transcribed words (for text systems), recognised words, and phonetic/prosodic features (for speech systems). Therefore, it is necessary to develop a model to cope with both the segmentation and the assignation problem.

Let $U_1^d = U_1 U_2 \cdots U_d$ be the sequence of DA assigned until the current turn, corresponding to the first d segments of the current dialogue. Let

$W = w_1 w_2 \dots w_l$ be the sequence of the words of the current turn, where subsequences $W_i^j = w_i w_{i+1} \dots w_j$ can be defined ($1 \leq i \leq j \leq l$).

For the sequence of words W, a segmentation is defined as $s_1^r = s_0 s_1 \dots s_r$, where $s_0 = 0$ and $W = W_{s_0+1}^{s_1} W_{s_1+1}^{s_2} \dots W_{s_{r-1}+1}^{s_r}$. Therefore, the optimal sequence of DA for the current turn will be given by:

$$\hat{U} = \underset{U}{\operatorname{argmax}} \Pr(U|W_1^l, U_1^d) =$$

$$\underset{U_{d+1}^{d+r}}{\operatorname{argmax}} \sum_{(s_1^r, r)} \Pr(U_{d+1}^{d+r}|W_1^l, U_1^d)$$

After developing this formula and making several assumptions and simplifications, the final model, called *unsegmented model*, is:

$$\hat{U} = \underset{U_{d+1}^{d+r}}{\operatorname{argmax}} \max_{(s_1^r, r)}$$

$$\prod_{k=d+1}^{d+r} \Pr(U_k|U_{k-n-1}^{k-1}) \Pr(W_{s_{k-(d+1)}+1}^{s_{k-d}}|U_k)$$

This model can be easily implemented using simple statistical models (N-grams and Hidden Markov Models). The decoding (segmentation and DA assignation) was implemented using the Viterbi algorithm. A Word Insertion Penalty (WIP) factor, similar to the one used in speech recognition, can be incorporated into the model to control the number of utterances and avoid excessive segmentation.

When the segmentation into utterances is provided, the model can be simplified into the *segmented model*, which is:

$$\hat{U} = \underset{U_{d+1}^{d+r}}{\operatorname{argmax}}$$

$$\prod_{k=d+1}^{d+r} \Pr(U_k|U_{k-n-1}^{k-1}) \Pr(W_{s_{k-(d+1)}+1}^{s_{k-d}}|U_k)$$

All the presented models only take into account word transcriptions and dialogue acts, although they could be extended to deal with other features (like prosody, sintactical and semantic information, etc.).

3 Experimental data

Two corpora with very different features were used in the experiment with the models proposed in Section 2. The SwitchBoard corpus is composed of human-human, non task-oriented dialogues with a large vocabulary. The Dihana corpus is composed of human-computer, task-oriented dialogues with a small vocabulary.

Although two corpora are not enough to let us draw general conclusions, they give us more reliable results than using only one corpus. Moreover, the very different nature of both corpora makes our conclusions more independent from the corpus type, the annotation scheme, the vocabulary size, etc.

3.1 The SwitchBoard corpus

The first corpus used in the experiments was the well-known SwitchBoard corpus (Godfrey et al., 1992). The SwitchBoard database consists of human-human conversations by telephone with no directed tasks. Both speakers discuss about general interest topics, but without a clear task to accomplish.

The corpus is formed by 1,155 conversations, which comprise 126,754 different turns of spontaneous and sometimes overlapped speech, using a vocabulary of 21,797 different words. The corpus was segmented into utterances, each of which was annotated with a DA following the simplified DAMSL annotation scheme (Jurafsky et al., 1997). The set of labels of the simplified DAMSL scheme is composed of 42 different labels, which define categories such as statement, backchannel, opinion, etc. An example of annotation is presented in Figure 1.

3.2 The Dihana corpus

The second corpus used was a task-oriented corpus called Dihana (Benedí et al., 2004). It is composed of computer-to-human dialogues, and the main aim of the task is to answer telephone queries about train timetables, fares, and services for long-distance trains in Spanish. A total of 900 dialogues were acquired by using the Wizard of Oz technique and semicontrolled scenarios. Therefore, the voluntary caller was always free to express him/herself (there were no syntactic or vocabulary restrictions); however, in some dialogues, s/he had to achieve some goals using a set of restrictions that had been given previously (e.g. departure/arrival times, origin/destination, travelling on a train with some services, etc.).

These 900 dialogues comprise 6,280 user turns and 9,133 system turns. Obviously, as a task-

Utterance	Label
YEAH, TO GET REFERENCES AND THAT, SO, BUT, UH, I DON'T FEEL COMFORTABLE ABOUT LEAVING MY KIDS IN A BIG DAY CARE CENTER, SIMPLY BECAUSE THERE'S SO MANY KIDS AND SO MANY <SNIFFING> <THROAT_CLEARING>	
Yeah, to get references and that, so, but, uh, I don't feel comfortable about leaving my kids in a big day care center, simply because there's so many kids and so many <sniffing> <throat_clearing>	aa sd % sd
I THINK SHE HAS PROBLEMS WITH THAT, TOO.	
I think she has problems with that, too.	sd

Figure 1: An example of annotated turns in the SwitchBoard corpus.

oriented and medium size corpus, the total number of different words in the vocabulary, 812, is not as large as the Switchboard database.

The turns were segmented into utterances. It was possible for more than one utterance (with their respective labels) to appear in a turn (on average, there were 1.5 utterances per user/system turn). A three-level annotation scheme of the utterances was defined (Alcácer et al., 2005). These labels represent the general purpose of the utterance (first level), as well as more specific semantic information (second and third level): the second level represents the data focus in the utterance and the third level represents the specific data present in the utterance. An example of three-level annotated user turns is given in Figure 2. The corpus was annotated by means of a semiautomatic procedure, and all the dialogues were manually corrected by human experts using a very specific set of defined rules.

After this process, there were 248 different labels (153 for user turns, 95 for system turns) using the three-level scheme. When the detail level was reduced to the first and second levels, there were 72 labels (45 for user turns, 27 for system turns). When the detail level was limited to the first level, there were only 16 labels (7 for user turns, 9 for system turns). The differences in the number of labels and in the number of examples for each label with the SwitchBoard corpus are significant.

4 Experiments and results

The SwitchBoard database was processed to remove certain particularities. The main adaptations performed were:

- The interrupted utterances (which were labelled with '+') were joined to the correct previous utterance, thereby avoiding interruptions (i.e., all the words of the interrupted utterance were annotated with the same DA).

Table 1: SwitchBoard database statistics (mean for the ten cross-validation partitions)

	Training	Test
Dialogues	1,136	19
Turns	113,370	1,885
Utterances	201,474	3,718
Running words	1,837,222	33,162
Vocabulary	21,248	2,579

- All the words were transcribed in lowercase.
- Puntuaction marks were separated from words.

The experiments were performed using a cross-validation approach to avoid the statistical bias that can be introduced by the election of fixed training and test partitions. This cross-validation approach has also been adopted in other recent works on this corpus (Webb et al., 2005). In our case, we performed 10 different experiments. In each experiment, the training partition was composed of 1,136 dialogues, and the test partition was composed of 19 dialogues. This proportion was adopted so that our results could be compared with the results in (Stolcke et al., 2000), where similar training and test sizes were used. The mean figures for the training and test partitions are shown in Table 1.

With respect to the Dihana database, the preprocessing included the following points:

- A categorisation process was performed for categories such as town names, the time, dates, train types, etc.
- All the words were transcribed in lowercase.
- Puntuaction marks were separated from words.
- All the words were preceded by the speaker identification (U for user, M for system).

Utterance	1st level	2nd level	3rd level
YES, TIMES AND FARES.			
Yes, times and fares	Acceptance Question	Dep_Hour Dep_Hour,Fare	Nil Nil
YES, I WANT TIMES AND FARES OF TRAINS THAT ARRIVE BEFORE SEVEN.			
Yes, I want times and fares of trains that arrive before seven.	Question	Dep_Hour,Fare	Arr_Hour
ON THURSDAY IN THE AFTERNOON.			
On thursday in the afternoon	Answer Answer	Day Time	Day Time

Figure 2: An example of annotated turns in the Dihana corpus. Original turns were in Spanish.

Table 2: Dihana database statistics (mean for the five cross-validation partitions)

	Training	Test
Dialogues	720	180
Turns	12,330	3,083
User turns	5,024	1,256
System turns	7,206	1,827
Utterances	18,837	4,171
User utterances	7,773	1,406
System utterances	11,064	2,765
Running words	162,613	40,765
User running words	42,806	10,815
System running words	119,807	29,950
Vocabulary	832	485
User vocabulary	762	417
System vocabulary	208	174

A cross-validation approach was adopted in Dihana as well. In this case, only 5 different partitions were used. Each of them had 720 dialogues for training and 180 for testing. The statistics on the Dihana corpus are presented in Table 2.

For both corpora, different N-gram models, with N $= 2, 3, 4$, and HMM of one state were trained from the training database. In the case of the SwitchBoard database, all the turns in the test set were used to compute the labelling accuracy. However, for the Dihana database, only the user turns were taken into account (because system turns follow a regular, template-based scheme, which presents artificially high labelling accuracies). Furthermore, in order to use a really significant set of labels in the Dihana corpus, we performed the experiments using only two-level labels instead of the complete three-level labels. This restriction allowed us to be more independent from the understanding issues, which are strongly related to the third level. It also allowed us to concentrate on the dialogue issues, which relate more to the first and second levels.

Table 3: SwitchBoard results for the segmented model

N-gram	Utt. accuracy	Turn accuracy
2-gram	68.19%	59.33%
3-gram	68.50%	59.75%
4-gram	67.90%	59.14%

The results in the case of the segmented approach described in Section 2 for SwitchBoard are presented in Table 3. Two different definitions of accuracy were used to assess the results:

- Utterance accuracy: computes the proportion of well-labelled utterances.
- Turn accuracy: computes the proportion of totally well-labelled turns (i.e.: if the labelling has the same labels in the same order as in the reference, it is taken as a well-labelled turn).

As expected, the utterance accuracy results are a bit worse than those presented in (Stolcke et al., 2000). This may be due to the use of only the past history and possibly to the cross-validation approach used in the experiments. The turn accuracy was calculated to compare the segmented and the unsegmented models. This was necessary because the utterance accuracy does not make sense for the unsegmented model.

The results for the unsegmented approach for SwitchBoard are presented in Table 4. In this case, three different definitions of accuracy were used to assess the results:

- Accuracy at DA level: the edit distance between the reference and the labelling of the turn was computed; then, the number of correct substitutions (c), wrong substitutions (s), deletions (d) and insertions (i) was com-

Table 4: SwitchBoard results for the unsegmented model (WIP=50)

N-gram	DA acc.	Turn acc.	Segm. acc.
2-gram	38.19%	39.47%	38.92%
3-gram	38.58%	39.61%	39.06%
4-gram	38.49%	39.52%	38.96%

puted, and the accuracy was calculated as $100 \cdot \frac{c}{(c+s+i+d)}$.

- Accuracy at turn level: this provides the proportion of well-labelled turns, without taking into account the segmentation (i.e., if the labelling has the same labels in the same order as in the reference, it is taken as a well-labelled turn).
- Accuracy at segmentation level: this provides the proportion of well-labelled and segmented turns (i.e., the labels are the same as in the reference and they affect the same utterances).

The WIP parameter used in Table 4 was 50, which is the one that offered the best results. The segmentation accuracy in Table 4 must be compared with the turn accuracy in Table 3. As Table 4 shows, the accuracy of the labelling decreased dramatically. This reveals the strong influence of the availability of the real segmentation of the turns.

To confirm this hypothesis, similar experiments were performed with the Dihana database. Table 5 presents the results with the segmented corpus, and Table 6 presents the results with the unsegmented corpus (with WIP=50, which gave the best results). In this case, only user turns were taken into account to compute the accuracy, although the model was applied to all the turns (both user and system turns). For the Dihana corpus, the degradation of the results of the unsegmented approach with respect to the segmented approach was not as high as in the SwitchBoard corpus, due to the smaller vocabulary and complexity of the dialogues.

These results led us to the same conclusion, even for such a different corpus (much more labels, task-oriented, etc.). In any case, these accuracy figures must be taken as a lower bound on the model performance because sometimes an incorrect recognition of segment boundaries or dialogue acts does not cause an inappropriate reaction of the dialogue strategy.

Table 5: Dihana results for the segmented model (only two-level labelling for user turns)

N-gram	Utt. accuracy	Turn accuracy
2-gram	75.70%	74.46%
3-gram	76.28%	74.93%
4-gram	76.39%	75.10%

Table 6: Dihana results for the unsegmented model (WIP=50, only two-level labelling for user turns)

N-gram	DA acc.	Turn acc.	Segm. acc.
2-gram	60.36%	62.86%	58.15%
3-gram	60.05%	62.49%	57.87%
4-gram	59.81%	62.44%	57.88%

An illustrative example of annotation errors in the SwitchBoard database, is presented in Figure 3 for the same turns as in Figure 1. An error analysis of the segmented model was performed. The results reveals that, in the case of most of the errors were produced by the confusion of the 'sv' and 'sd' classes (about 50% of the times 'sv' was badly labelled, the wrong label was 'sd') The second turn in Figure 3 is an example of this type of error. The confusions between the 'aa' and 'b' classes were also significant (about 27% of the times 'aa' was badly labelled, the wrong label was 'b'). This was reasonable due to the similar definitions of these classes (which makes the annotation difficult, even for human experts). These errors were similar for all the N-grams used. In the case of the unsegmented model, most of the errors were produced by deletions of the 'sd' and 'sv' classes, as in the first turn in Figure 3 (about 50% of the errors). This can be explained by the presence of very short and very long utterances in both classes (i.e., utterances for 'sd' and 'sv' did not present a regular length).

Some examples of errors in the Dihana corpus are shown in Figure 4 (in this case, for the same turns as those presented in Figure 2). In the segmented model, most of the errors were substitutions between labels with the same first level (especially questions and answers) where the second level was difficult to recognise. The first and third turn in Figure 4 are examples of this type of error. This was because sometimes the expressions only differed with each other by one word, or

Utt	Label	
1	%	Yeah, to get references and that, so, but, uh, I don't
2	sd	feel comfortable about leaving my kids in a big day care center, simply because there's so many kids and so many <sniffing> <throat_clearing>
Utt	Label	
1	sv	I think she has problems with that, too.

Figure 3: An example of errors produced by the model in the SwitchBoard corpus

the previous segment influence (i.e., the language model weight) was not enough to get the appropriate label. This was true for all the N-grams tested. In the case of the unsegmented model, most of the errors were caused by similar misrecognitions in the second level (which are more frequent due to the absence of utterance boundaries); however, deletion and insertion errors were also significant. The deletion errors corresponded to acceptance utterances, which were too short (most of them were "Yes"). The insertion errors corresponded to "Yes" words that were placed after a new-consult system utterance, which is the case of the second turn presented in Figure 4. These words should not have been labelled as a separate utterance. In both cases, these errors were very dependant on the WIP factor, and we had to get an adequate WIP value which did not increase the insertions and did not cause too many deletions.

5 Conclusions and future work

In this work, we proposed a method for simultaneous segmentation and annotation of dialogue utterances. In contrast to previous models for this task, our model does not assume manual utterance segmentation. Instead of treating utterance segmentation as a separate task, the proposed method selects utterance boundaries to optimize the accuracy of the generated labels. We performed experiments to determine the effect of the availability of the correct segmentation of dialogue turns in utterances in the statistical DA labelling framework. Our results reveal that, as shown in previous work (Warnke et al., 1999), having the correct segmentation is very important in obtaining accurate results in the labelling task. This conclusion is supported by the results obtained in very different dialogue corpora: different amounts of training and test data, different natures (general and task-oriented), different sets of labels, etc.

Future work on this task will be carried out in several directions. As segmentation appears to be an important step in these tasks, it would be interesting to obtain an automatic and accurate segmentation model that can be easily integrated in our statistical model. The application of our statistical models to other tasks (like VerbMobil (Alexandersson et al., 1998)) would allow us to confirm our conclusions and compare results with other works.

The error analysis we performed shows the need for incorporating new and more reliable information resources to the presented model. Therefore, the use of alternative models in both corpora, such as the N-gram-based model presented in (Webb et al., 2005) or an evolution of the presented statistical model with other information sources would be useful. The combination of these two models might be a good way to improve results.

Finally, it must be pointed out that the main task of the dialogue models is to allow the most correct reaction of a dialogue system given the user input. Therefore, the correct evaluation technique must be based on the system behaviour as well as on the accurate assignation of DA to the user input. Therefore, future evaluation results should take this fact into account.

Acknowledgements

The authors wish to thank Nick Webb, Mark Hepple and Yorick Wilks for their comments and suggestions and for providing the preprocessed SwitchBoard corpus. We also want to thank the anonymous reviewers for their criticism and suggestions.

References

N. Alcácer, J. M. Benedí, F. Blat, R. Granell, C. D. Martínez, and F. Torres. 2005. Acquisition and labelling of a spontaneous speech dialogue corpus. In *Proceedings of SPECOM*, pages 583–586, Patras, Greece.

Jan Alexandersson, Bianka Buschbeck-Wolf, Tsutomu Fujinami, Michael Kipp, Stephan Koch, Elis-

Utterance	1st level	2nd level
Yes, times	Acceptance	Dep_Hour,Fare
and fares	Question	Dep_Hour,Fare
Yes, I want	Acceptance	Dep_Hour,Fare
times and fares of trains that arrive before seven.	Question	Dep_Hour,Fare
On thursday in the afternoon	Answer	Time

Figure 4: An example of errors produced by the model in the Dihana corpus

abeth Maier, Norbert Reithinger, Birte Schmitz, and Melanie Siegel. 1998. Dialogue acts in VERBMOBIL-2 (second edition). Technical Report 226, DFKI GmbH, Saarbrücken, Germany, July.

J. Ang, Y. Liu, and E. Shriberg. 2005. Automatic dialog act segmentation and classification in multiparty meetings. In *Proceedings of the International Conference of Acoustics, Speech, and Signal Processings*, volume 1, pages 1061–1064, Philadelphia.

H. Aust, M. Oerder, F. Seide, and V. Steinbiss. 1995. The philips automatic train timetable information system. *Speech Communication*, 17:249–263.

J. M. Benedí, A. Varona, and E. Lleida. 2004. Dihana: Dialogue system for information access using spontaneous speech in several environments tic2002-04103-c03. In *Reports for Jornadas de Seguimiento - Programa Nacional de Tecnologías Informáticas*, Málaga, Spain.

Mark G. Core and James F. Allen. 1997. Coding dialogs with the damsl annotation scheme. In *Working Notes of AAAI Fall Symposium on Communicative Action in Humans and Machines*, Boston, MA, November.

Layla Dybkjaer and Niels Ole Bernsen. 2000. The mate workbench.

N. Fraser, 1997. *Assessment of interactive systems*, pages 564–614. Mouton de Gruyter.

J. Godfrey, E. Holliman, and J. McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *Proc. ICASSP-92*, pages 517–520.

A. Gorin, G. Riccardi, and J. Wright. 1997. How may i help you? *Speech Communication*, 23:113–127.

Hilda Hardy, Kirk Baker, Laurence Devillers, Lori Lamel, Sophie Rosset, Tomek Strzalkowski, Cristian Ursu, and Nick Webb. 2002. Multi-layer dialogue annotation for automated multilingual customer service. In *Proceedings of the ISLE Workshop on Dialogue Tagging for Multi-Modal Human Computer Interaction*, Edinburgh, Scotland, December.

Hilda Hardy, Tomek Strzalkowski, and Min Wu. 2003. Dialogue management for an automated multilingual call center. In *Proceedings of HLT-NAACL 2003 Workshop: Research Directions in Dialogue Processing*, pages 10–12, Edmonton, Canada, June.

D. Jurafsky, E. Shriberg, and D. Biasca. 1997. Switchboard swbd-damsl shallow- discourse-function annotation coders manual - draft 13. Technical Report 97-01, University of Colorado Institute of Cognitive Science.

J. Van Kuppevelt and R. W. Smith. 2003. *Current and New Directions in Discourse and Dialogue*, volume 22 of *Text, Speech and Language Technology*. Springer.

A. Stolcke, N. Coccaro, R. Bates, P. Taylor, C. van Ess-Dykema, K. Ries, E. Shriberg, D. Jurafsky, R. Martin, and M. Meteer. 2000. Dialogue act modelling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):1–34.

Marilyn A. Walker, Diane Litman J., Candace A. Kamm, and Alicia Abella. 1997. PARADISE: A framework for evaluating spoken dialogue agents. In Philip R. Cohen and Wolfgang Wahlster, editors, *Proceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, pages 271–280, Somerset, New Jersey. Association for Computational Linguistics.

V. Warnke, R. Kompe, H. Niemann, and E. Nöth. 1997. Integrated Dialog Act Segmentation and Classification using Prosodic Features and Language Models. In *Proc. European Conf. on Speech Communication and Technology*, volume 1, pages 207–210, Rhodes.

V. Warnke, S. Harbeck, E. Nöth, H. Niemann, and M. Levit. 1999. Discriminative Estimation of Interpolation Parameters for Language Model Classifiers. In *Proceedings of the IEEE Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 525–528, Phoenix, AZ, March.

N. Webb, M. Hepple, and Y. Wilks. 2005. Dialogue act classification using intra-utterance features. In *Proceedings of the AAAI Workshop on Spoken Language Understanding*, Pittsburgh.

S. Young. 2000. Probabilistic methods in spoken dialogue systems. *Philosophical Trans Royal Society (Series A)*, 358(1769):1389–1402.

ARE: Instance Splitting Strategies for Dependency Relation-based Information Extraction

Mstislav Maslennikov **Hai-Kiat Goh** **Tat-Seng Chua**

Department of Computer Science
School of Computing
National University of Singapore
{maslenni, gohhaiki, chuats}@ comp.nus.edu.sg

Abstract

Information Extraction (IE) is a fundamental technology for NLP. Previous methods for IE were relying on co-occurrence relations, soft patterns and properties of the target (for example, syntactic role), which result in problems of handling paraphrasing and alignment of instances. Our system ARE (Anchor and Relation) is based on the dependency relation model and tackles these problems by unifying entities according to their dependency relations, which we found to provide more invariant relations between entities in many cases. In order to exploit the complexity and characteristics of relation paths, we further classify the relation paths into the categories of 'easy', 'average' and 'hard', and utilize different extraction strategies based on the characteristics of those categories. Our extraction method leads to improvement in performance by 3% and 6% for MUC4 and MUC6 respectively as compared to the state-of-art IE systems.

1 Introduction

Information Extraction (IE) is one of the fundamental problems of natural language processing. Progress in IE is important to enhance results in such tasks as Question Answering, Information Retrieval and Text Summarization. Multiple efforts in MUC series allowed IE systems to achieve near-human performance in such domains as biological (Humphreys et al., 2000), terrorism (Kaufmann, 1992; Kaufmann, 1993) and management succession (Kaufmann, 1995).

The IE task is formulated for MUC series as filling of several predefined slots in a template. The terrorism template consists of slots Perpetrator, Victim and Target; the slots in the management succession template are Org, PersonIn, PersonOut and Post. We decided to choose both terrorism and management succession domains, from MUC4 and MUC6 respectively, in order to demonstrate that our idea is applicable to multiple domains.

Paraphrasing of instances is one of the crucial problems in IE. This problem leads to data sparseness in situations when information is expressed in different ways. As an example, consider the excerpts "Terrorists attacked victims" and "Victims *were* attacked *by unidentified* terrorists". These instances have very similar semantic meaning. However, context-based approaches such as Autoslog-TS by Riloff (1996) and Yangarber et al. (2002) may face difficulties in handling these instances effectively because the context of entity 'victims' is located on the left context in the first instance and on the right context in the second. For these cases, we found that we are able to verify the context by performing dependency relation parsing (Lin, 1997), which outputs the word 'victims' as an object in both instances, with 'attacked' as a verb and 'terrorists' as a subject. After grouping of same syntactic roles in the above examples, we are able to unify these instances.

Another problem in IE systems is word alignment. Insertion or deletion of tokens prevents instances from being generalized effectively during learning. Therefore, the instances "Victims *were* attacked *by* terrorists" and "Victims *were recently* attacked *by* terrorists" are difficult to unify. The common approach adopted in GRID by Xiao et al. (2003) is to apply more stable chunks such as noun phrases and verb phrases. Another recent approach by Cui et al. (2005) utilizes soft patterns for probabilistic matching of tokens. However, a longer insertion leads to a more complicated structure, as in the instance "Victims, living near the shop, *went out for a walk and were* attacked *by* terrorists". Since there may be many inserted words, both approaches may also be inefficient for this case. Similar to the paraphrasing problem, the word alignment problem may be handled with dependency relations in many cases. We found that the relation subject-verb-object for words 'victims', 'attacked' and 'terrorists' remains invariant for the above two instances.

Before IE can be performed, we need to identify sentences containing possible slots. This is

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 571–578,
Sydney, July 2006.

done through the identification of cue phrases which we call *anchors* or *anchor cues*. However, natural texts tend to have diverse terminologies, which require semantic features for generalization. These features include semantic classes, Named Entities (NE) and support from ontology (for example, synsets in Wordnet). If such features are predefined, then changes in terminology (for instance, addition of new terrorism organization) will lead to a loss in recall. To avoid this, we exploit automatic mining techniques for anchor cues. Examples of anchors are the words "terrorists" or "guerrilla" that signify a possible candidate for the Perpetrator slot.

From the reviewed works, we observe that the inefficient use of relations causes problems of paraphrasing and alignment and the related data sparseness problem in current IE systems. As a result, training and testing instances in the systems often lack generality. This paper aims to tackle these problems with the help of dependency relation-based model for IE. Although dependency relations provide invariant structures for many instances as illustrated above, they tend to be efficient only for short sentences and make errors on long distance relations. To tackle this problem, we classify relations into 'simple', 'average' and 'hard' categories, depending on the complexity of the dependency relation paths. We then employ different strategies to perform IE in each category.

The main contributions of our work are as follows. First, we propose a dependency relation based model for IE. Second, we perform classification of instances into several categories based on the complexity of dependency relation structures, and employ the action promotion strategy to tackle the problem of long distance relations.

The remaining parts of the paper are organized as follows. Section 2 discusses related work and Section 3 introduces our approach for constructing ARE. Section 4 introduces our method for splitting instances into categories. Section 5 describes our experimental setups and results and, finally, Section 6 concludes the paper.

2 Related work

There are several research directions in Information Extraction. We highlight a few directions in IE such as case frame based modeling in PALKA by Kim and Moldovan (1995) and CRYSTAL by Soderland et al. (1995); rule-based learning in Autoslog-TS by Riloff et al. (1996); and classification-based learning by Chieu et al. (2002). Although systems representing these directions have very different learning models, paraphrasing and alignment problems still have no reliable solution.

Case frame based IE systems incorporate domain-dependent knowledge in the processing and learning of semantic constraints. However, concept hierarchy used in case frames is typically encoded manually and requires additional human labor for porting across domains. Moreover, the systems tend to rely on heuristics in order to match case frames. PALKA by Kim and Moldovan (1995) performs keyword-based matching of concepts, while CRYSTAL by Soderland et al. (1995) relied on additional domain-specific annotation and associated lexicon for matching.

Rule-based IE models allow differentiation of rules according to their performance. Autoslog-TS by Riloff (1996) learns the context rules for extraction and ranks them according to their performance on the training corpus. Although this approach is suitable for automatic training, Xiao et al. (2004) stated that hard matching techniques tend to have low recall due to data sparseness problem. To overcome this problem, $(LP)^2$ by Ciravegna (2002) utilizes rules with high precision in order to improve the precision of rules with average recall. However, $(LP)^2$ is developed for semi-structured textual domain, where we can find consistent lexical patterns at surface text level. This is not the same for free-text, in which different order of words or an extra clause in a sentence may cause paraphrasing and alignment problems respectively, such as the example excerpts "terrorists attacked peasants" and "peasants *were* attacked *2 months ago by* terrorists".

The classification-based approaches such as by Chieu and Ng (2002) tend to outperform rule-based approaches. However, Ciravegna (2001) argued that it is difficult to examine the result obtained by classifiers. Thus, interpretability of the learned knowledge is a serious bottleneck of the classification approach. Additionally, Zhou and Su (2002) trained classifiers for Named Entity extraction and reported that performance degrades rapidly if the training corpus size is below 100KB. It implies that human experts have to spend long hours to annotate a sufficiently large amount of training corpus.

Several recent researches focused on the extraction of relationships using classifiers. Roth and Yih (2002) learned the entities and relations together. The joint learning improves the performance of NE recognition in cases such as "X killed Y". It also prevents the propagation of mistakes in NE extraction to the extraction of relations. However, long distance relations between entities are likely to cause mistakes in relation extraction. A possible approach for modeling relations of different complexity is the use of dependency-based kernel trees in support vector machines by Culotta and Sorensen (2004). The authors reported that non-relation instances are very heterogeneous, and

hence they suggested the additional step of extracting candidate relations before classification.

3 Our approach

Differing from previous systems, the language model in ARE is based on dependency relations obtained from Minipar by Lin (1997). In the first stage, ARE tries to identify possible candidates for filling slots in a sentence. For example, words such as 'terrorist' or 'guerrilla' can fill the slot for Perpetrator in the terrorism domain. We refer to these candidates as *anchors* or *anchor cues*. In the second stage, ARE defines the dependency relations that connect anchor cues. We exploit dependency relations to provide more invariant structures for similar sentences with different syntactic structures. After extracting the possible relations between anchor cues, we form several possible parsing paths and rank them. Based on the ranking, we choose the optimal filling of slots.

Ranking strategy may be unnecessary in cases when entities are represented in the SVO form. Ranking strategy may also fail in situations of long distance relations. To handle such problems, we categorize the sentences into 3 categories of: simple, average and hard, depending on the complexity of the dependency relations. We then apply different strategies to tackle sentences in each category effectively. The following subsections discuss details of our approach.

Features	**Perpetrator_Cue (A)**	**Action_Cue (D)**	**Victim_Cue (A)**	**Target_Cue (A)**
Lexical (Head noun)	terrorists, individuals, soldiers	attacked, murder, massacre	mayor, general, priests	bridge, house, ministry
Part-of-Speech	Noun	Verb	Noun	Noun
Named Entities	Soldiers (PERSON)	-	Jesuit priests (PERSON)	WTC (OBJECT)
Synonyms	Synset 130, 166	Synset 22	Synset 68	Synset 71
Concept Class	ID 2, 3	ID 9	ID 22, 43	ID 61, 48
Co-referenced entity	He -> terrorist, soldier	-	They -> peasants	-

Table 1. Linguistic features for anchor extraction

Every token in ARE may be represented at a different level of representations, including: Lexical, Part-of-Speech, Named Entities, Synonyms and Concept classes. The synonym set and concept classes are mainly obtained from Wordnet. We use NLProcessor from Infogistics Ltd for the extraction of part-of-speech, noun phrases and verb phrases (we refer to them as *phrases*). Named Entities are extracted with the program used in Yang et al. (2003). Additionally, we employed the co-reference module for the extraction of meaningful pronouns. It is used for linking entities across clauses or sentences, for example in "John *works in* XYZ Corp. He *was* appointed *as a* vice-president *a month ago*" and could achieve an accuracy of 62%. After preprocessing and feature extraction, we obtain the linguistic features in Table 1.

3.1 Mining of anchor cues

In order to extract possible anchors and relations from every sentence, we need to select features to support the generalization of words. This generalization may be different for different classes of words. For example, person names may be generalized as a Named Entity PERSON, whereas for 'murder' and 'assassinate', the optimal generalization would be the concept class 'kill' in the WordNet hypernym tree. To support several generalizations, we need to store multiple representations of every word or token.

Mining of anchor cues or anchors is crucial in order to unify meaningful entities in a sentence, for example words 'terrorists', 'individuals' and 'soldiers' from Table 1. In the terrorism domain, we consider 4 types of anchor cues: Perpetrator, Action, Victim, and Target of destruction. For management succession domain, we have 6 types: Post, Person In, Person Out, Action and Organization. Each set of anchor cues may be seen as a pre-defined semantic type where the tokens are mined automatically. The anchor cues are further classified into two categories: general type *A* and action type *D*. Action type anchor cues are those with verbs or verb phrases describing a particular action or movement. General type encompasses any predefined type that does not fall under the action type cues.

In the first stage, we need to extract anchor cues for every type. Let P be an input phrase, and A_j be the anchor of type j that we want to match. The similarity score of P for A_j in sentence S is given by:

$$\begin{aligned} Phrase\_Score_S(P,A_j) = {} & \delta_1 * S\_lexical_S(P,A_j + \delta_2 * S\_POS_S(P,A_j) \\ & + \delta_3 * S\_NE_S(P,A_j) + \delta_4 * S\_Syn_S(P,A_j) \\ & + \delta_5 * S\_Concept\text{-}Class_S(P,A_j) \end{aligned} \quad (1)$$

where $S\_XXX_S(P,A_j)$ is a score function for the type A_j and δ_i is the importance weight for A_j. In order to extract the score function, we use entities from slots in the training instances. Each $S\_XXX_S(P,A_j)$ is calculated as a ratio of occurrence in positive slots versus all the slots:

$$S\_XXX_S(P,A_j) = \frac{\#(P \text{ in positive slots of the type } A_j)}{\#(\text{all slots of the type } A_j)} \quad (2)$$

We classify the phrase P as belonging to an anchor cue A of type j if $Phrase\_Score_S(P,A_j) \geq \omega$, where ω is an empirically determined threshold. The weights $\vec{\delta} = (\delta_1, ..., \delta_5)$ are learned automatically using Expectation Maximization by Dempster et al. (1977). Using anchors from training instances as ground truth, we iteratively input different sets of weights into EM to maximize the overall score.

Consider the excerpts "Terrorists attacked victims", "Peasants were murdered by unidentified individuals" and "Soldiers participated in massacre of Jesuit priests". Let W_i denotes the position of token i in the instances. After mining of anchors, we are able to extract meaningful anchor cues in these sentences as shown in Table 2:

W_{-3}	W_{-2}	W_{-1}	W_0	W_1	W_2	W_3
	Perp_Cue	Action_Cue	Victim_Cue			
			Victim_Cue	were	Action_Cue	by
In	Action_Cue	Of	Victim_Cue			

Table 2. Instances with anchor cues

3.2 Relationship extraction and ranking

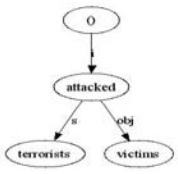


Figure 1. Dependency tree

In the next stage, we need to find meaningful relations to unify instances using the anchor cues. This unification is done using dependency trees of sentences. The dependency relations for the first sentence are given in Figure 1.

From the dependency tree, we need to identify the SVO relations between anchor cues. In cases when there are multiple relations linking many potential subjects, verbs or objects, we need to select the best relations under the circumstances. Our scheme for relation ranking is as follows.

First, we rank each single relation individually based on the probability that it appears in the respective context template slot in the training data. We use the following formula to capture the quality of a relation Rel which gives higher weight to more frequently occurring relations:

$$Quality\ (Rel\ , A_1, A_2) = \frac{\sum_{\bar{S}} \| \{R_i \mid R_i \in R, R_i = Rel\ \} \|}{\sum_{\bar{S}} \| \{R_i \mid R_i \in S_i\} \|} \quad (3)$$

where $\bar{S}$ is a set of sentences containing relation *Rel*, anchors A_1 and A_2; R denotes relation path connecting A_1 and A_2 in a sentence S_i; $||X||$ denotes size of the set X.

Second, we need to take into account the entity height in the dependency tree. We calculate height as a distance to the root node. Our intuition is that the nodes on the higher level of dependency tree are more important, because they may be linked to more nodes or entities. The following example in Figure 2 illustrates it.

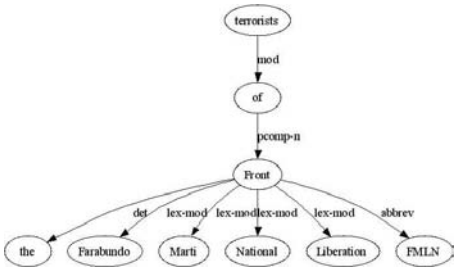


Figure 2. Example of entity in a dependency tree

Here, the node 'terrorists' is the most representative in the whole tree, and thus relations nearer to 'terrorists' should have higher weight. Therefore, we give a slightly higher weight to the links that are closer to the root node as follows:

$$Height_s(Rel) = log_2(Const - Distance(Root, Rel)) \quad (4)$$

where *Const* is set to be larger than the depth of nodes in the tree.

Third, we need to calculate the score of relation path $R_{i->j}$ between each pair of anchors A_i and A_j, where A_i and A_j belong to different anchor cue types. The path score of $R_{i->j}$ depends on both quality and height of participating relations:

$$Score_s(A_i, A_j) = \Sigma_{Ri \in R} \{Height_s(R_i) * Quality(R_i)\} / Length_{ij} \quad (5)$$

where $Length_{ij}$ is the length of path $R_{i->j}$. Division on $Length_{ij}$ allows normalizing *Score* against the length of $R_{i->j}$. The formula (5) tends to give higher scores to shorter paths. Therefore, the path ending with 'terrorist' will be preferred in the previous example to the equivalent path ending with 'MRTA'.

Finally, we need to find optimal filling of a template T. Let $C = \{C_1, .. , C_K\}$ be the set of slot types in T and $A = \{A_1, .., A_L\}$ be the set of extracted anchors. First, we regroup anchors A according to their respective types. Let $A^{(k)} = \{A_1^{(k)}, ..., A_{L_k}^{(k)}\}$ be the projection of A onto the type C_k, $\forall k \in N$, $k \le K$. Let $F = A^{(1)} \times A^{(2)} \times .. \times A^{(K)}$ be the set of possible template fillings. The elements of F are denoted as $F_1, .., F_M$, where every $F_i \in F$ is represented as $F_i = \{A_i^{(1)}, .., A_i^{(K)}\}$. Our aim is to evaluate F and find the optimal filling $F_0 \in F$. For this purpose, we use the previously calculated scores of relation paths between every two anchors A_i and A_j.

Based on the previously defined $Score_S(A_i, A_j)$, it is possible to rank all the fillings in F. For each filling $F_i \in F$ we calculate the aggregate score for all the involved anchor pairs:

$$Relation\_Score_s(F_i) = \frac{\sum_{1 \le i, j \le K} Score_s(A_i, A_j)}{M} \quad (7)$$

where K is number of slot types and M denotes the number of relation paths between anchors in F_i.

After calculating $Relation\_Score_S(F_i)$, it is used for ranking all possible template fillings. The next step is to join entity and relation scores. We defined the entity score of F_i as an average of the scores of participating anchors:

$$Entity\_Score_s(F_i) = \sum_{1 \le k \le K} Phrase\_Score_s(A_i^{(k)}) / K \quad (8)$$

We combine entity and relation scores of F_i into the overall formula for ranking.

$$Rank_S(F_i) = \lambda * Entity\_Score_S(F_i) + (1-\lambda) * Relation\_Score_S(F_i) \quad (9)$$

The application of Subject-Verb-Object (SVO) relations facilitates the grouping of subjects,

verbs and objects together. For the 3 instances in Table 2 containing the anchor cues, the unified SVO relations are given in Table 3.

W_{-2}	W_{-1}	W_0	Instance is
Perp_Cue	attacked	Victim_Cue	+
Perp_Cue	murdered	Victim_Cue	+
Perp_Cue	participated	?	-

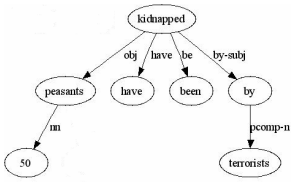
Table 3. Unification based on SVO relations

The first 2 instances are unified correctly. The only exception is the slot in the third case, which is missing because the target is not an object of 'participated'.

4 Category Splitting

Through our experiments, we found that the combination of relations and anchors are essential for improving IE performance. However, relations alone are not applicable across all situations because of long distance relations and possible dependency relation parsing errors, especially for long sentences. Since the relations in long sentences are often complicated, parsing errors are very difficult to avoid. Furthermore, application of dependency relations on long sentences may lead to incorrect extractions and decrease the performance.

Through the analysis of instances, we noticed that dependency trees have different complexity for different sentences. Therefore, we decided to classify sentences into 3 categories based on the complexity of dependency relations between the action cues (V) and the likely subject (S) and object cues (O). Category 1 is when the potential SVO's are connected directly to each other (simple category); Category 2 is when S or O is one link away from V in terms of nouns or verbs (average category); and Category 3 is when the path distances between potential S, V, and Os are more than 2 links away (hard category).



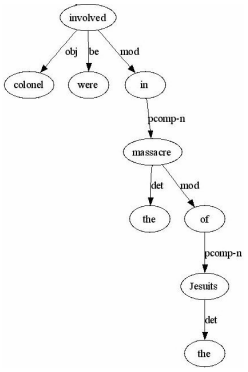


Figure 3. Simple category Figure 4. Average category

Figure 3 and Figure 4 illustrate the dependency parse trees for the simple and average categories respectively derived from the sentences: "50 peasants of have been kidnapped by terrorists" and "a colonel was involved in the massacre of the Jesuits". These trees represent 2 common structures in the MUC4 domain. By taking advantage of this commonality, we can further improve the performance of extraction. We notice that in the simple category, the perpetrator cue ('terrorists') is always a subject, action cue ('kidnapped') a verb, and victim cue ('peasants') an object. For the average category, perpetrator and victim commonly appear under 3 relations: subject, object and pcomp-n. The most difficult category is the hard category, since in this category relations can be distant. We thus primarily rely on anchors for extraction and have to give less importance to dependency parsing.

In order to process the different categories, we utilize the specific strategies for each category. As an example, the instance "X murdered Y" requires only the analysis of the context verb 'murdered' in the simple category. It is different from the instances "X *investigated* murder *of* Y" and "X *conducted* murder *of* Y" in the average category, in which transition of word 'investigated' into 'conducted' makes X a perpetrator. We refer to the anchor 'murder' in the first and second instances as *promotable* and *non-promotable* respectively. Additionally, we denote that the token 'conducted' is the optimal node for promotion of 'murder', whereas the anchor 'investigate' is not. This example illustrates the importance of support verb analysis specifically for the average category.

Algorithm

1) Analyze category
   If(simple)
   - Perform token reordering based on SVO relations
   Else if (average) *ProcessAverage*
   Else *ProcessHard*
2) Fill template slots

Function ProcessAverage

1) Find the nearest missing anchor in the previous sentences
2) Find the optimal linking node for action anchor in every F_i
3) Find the filling $F_i^{(0)} = \text{argmax}_i \text{Rank}(F_i)$
4) Use F_i for filling the template if $\text{Rank}_0 > \theta_2$, where θ_2 is an empirical threshold

Function ProcessHard

1) Perform token reordering based on anchors
2) Use linguistic+ syntactic + semantic feature of the head noun. Eg. Caps, 'subj', etc
3) Find the optimal linking node for action anchor in every F_i
4) Find the filling $F_i^{(0)} = \text{argmax}_i \text{Rank}(F_i)$
5) Use F_i for filling the template if $\text{Rank}_0 > \theta_3$, where θ_3 is an empirical threshold

Figure 5. Category processing

The main steps of our algorithm for performing IE in different categories are given in Figure 5. Although some steps are common for every category, the processing strategies are different.

Simple category

For simple category, we reorder tokens according to their slot types. Based on this reordering, we fill the template.

Average category

For average category, our strategy consists of 4 steps. First, in the case of missing anchor type we try to find it in the nearest previous sentence. Consider an example from MUC-6: "*Look at what happened to* John Sculley, *Apple Computer's* former chairman. *Earlier this month* he *abruptly* resigned *as* chairman *of troubled* Spectrum Information Technologies." In this example, a noisy cue 'he' needs to be substituted with "John Sculley", which is a strong anchor cue. Second, we need to find an optimal promotion of a support verb. For example, in "X *conducted* murder of Y", the verb 'murder' should be linked with X and in the excerpt "X *investigated* murder of Y", it should not be promoted. Thus, we need to make 2 steps for promotion: (a) calculate importance of every word connecting the action cue such as 'murder' and 'distributed' and (b) find the optimal promotion for the word 'murder'. Third, using the predefined threshold λ we cutoff the instances with irrelevant support verbs (e.g., 'investigated'). Fourth, we reorder the tokens in order to group them according to the anchor types.

The following algorithm in Figure 6 estimates the importance of a token *W* for type *D* in the support verb structure. The input of the algorithm consists of sentences $S_1 \ldots S_N$ and two sets of tokens V_{neg}, V_{pos} co-occurring with anchor cue of type D. V_{neg} and V_{pos} are automatically tagged as irrelevant and relevant respectively based on preliminary marked keys in the training instances. The algorithm output represents the importance value between 0 to 1.

CalculateImportance (W, D)
1) Select sentences that contain anchor cue D
2) Extract linguistic features of V_{pos}, V_{neg} and D
3) Train using SVM on instances (V_{pos},D) and instances (V_{neg},D)
4) Return Importance(W) using SVM

Figure 6. Evaluation of word importance

We use the linguistic features for *W* and *D* as given in Table 1 to form the instances.

Hard category

In the hard category, we have to deal with long-distance relations: at least 2 anchors are more than 2 links away in the dependency tree. Consequently, dependency tree alone is not reliable for connecting nodes. To find an optimal connection, we primarily rely on comparison between several possible fillings of slots based on previously extracted anchor cues. Depending on the results of such comparison, we chose the filling that has the highest score. As an example, consider the hard category in the excerpt "MRTA *today distributed leaflets claiming responsibility for the* murder *of former defense* minister Enrique Lopez Albujar". The dependency tree for this instance is given in Figure 7.

Although words 'MRTA', 'murder' and 'minister' might be correctly extracted as anchors, the challenging problem is to decide whether 'MRTA' is a perpetrator. Anchors 'MRTA' and 'minister' are connected via the verb 'distributed'. However, the word 'murder' belongs to another branch of this verb.

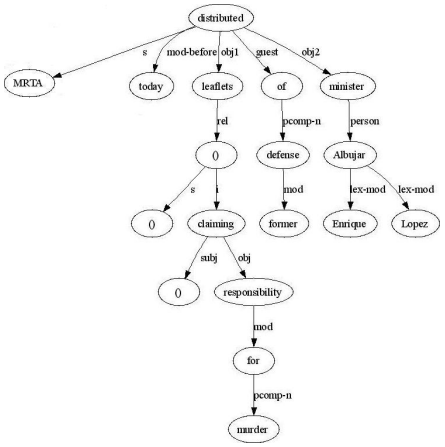


Figure 7. Hard case

Processing of such categories is challenging. Since relations are not reliable, we first need to rely on the anchor extraction stage. Nevertheless, the promotion strategy for the anchor cue 'murder' is still possible, although the corresponding branch in the dependency tree is long. Henceforth, we try to replace the verb 'distributed' by promoting the anchor 'murder'. To do so, we need to evaluate whether the nodes in between may be eliminated. For example, such elimination is possible in the pairs 'conducted' -> 'murder' and not possible in the pair 'investigated' -> 'murder', since in the excerpt "X *investigated* murder" X is not a perpetrator. If the elimination is possible, we apply the promotion algorithm given on Figure 8:

FindOptimalPromotion (F_i)
1) $Z = \varnothing$
2) For each $A_i^{(j1)}, A_i^{(j2)} \in F_i$
   $Z = Z \cup P_{j1->j2}$
   End_for
3) Output Top(Z)

Figure 8. Token promotion algorithm

The algorithm checks path $P_{j1->j2}$ that connect anchors $A_i^{(j1)}$ and $A_i^{(j2)}$ in the filling F_i; the nodes from $P_{j1->j2}$ are added to the set *Z*. Finally, the top node of the set *Z* is chosen as an optimal node for the promotion. The example optimal node for promotion of the word 'murder' on Figure 7 is the node 'distributed'.

Another important difference between the hard and average cases is in the calculation of $Rank_S(F_i)$ in Equation (9). We set $\lambda_{hard} > \lambda_{average}$ because long distance relations are less reliable in the hard case than in the average case.

5 Evaluation

In order to evaluate the efficiency of our method, we conduct our experiments in 2 domains: MUC4 (Kaufmann, 1992) and MUC6 (Kaufmann, 1995). The official corpus of MUC4 is released with MUC3; it covers terrorism in the Latin America region and consists of 1,700 texts. Among them, 1,300 documents belong to the training corpus. Testing was done on 25 relevant and 25 irrelevant texts from TST3, plus 25 relevant and 25 irrelevant texts from TST4, as is done in Xiao et al. (2004). MUC6 covers news articles in Management Succession domain. Its training corpus consists of 1201 instances, whereas the testing corpus consists of 76 person-ins, 82 person-outs, 123 positions, and 79 organizations. These slots we extracted in order to fill templates on a sentence-by-sentence basis, as is done by Chieu et al. (2002) and Soderland (1999).

Our experiments were designed to test the effectiveness of both case splitting and action verb promotion. The performance of ARE is compared to both the state-of-art systems and our baseline approach. We use 2 state-of-art systems for MUC4 and 1 system for MUC6. Our baseline system, *Anc+rel*, utilizes only anchors and relations without category splitting as described in Section 3. For our ARE system with case splitting, we present the results on *Overall* corpus, as well as separate results on *Simple*, *Average* and *Hard* categories. The *Overall* performance of ARE represents the result for all the categories combined together. Additionally, we test the impact of the action promotion (in the right column) for the average and hard categories.

Case (%)	Without promotion			With promotion		
	P	**R**	**F_1**	**P**	**R**	**F_1**
GRID	58%	56%	57%	-	-	-
Riloff'05	46%	52%	48%	-	-	-
Anc+rel (100%)	58%	59%	58%	58%	59%	58%
Overall (100%)	57%	60%	59%	58%	61%	60%
Simple (13%)	79%	86%	82%	79%	86%	82%
Average (22%)	64%	70%	67%	67%	71%	69%
Hard (65%)	50%	52%	51%	51%	53%	52%

Table 4. Results on MUC4 with case splitting

The comparative results are presented in Table 4 and Table 5 for MUC4 and MUC6, respectively. First, we review our experimental results on MUC4 corpus without promotion (left column) before proceeding to the right column.

a) From the results on Table 4 we observe that our baseline approach *Anc+rel* outperforms all the state-of-art systems. It demonstrates that both anchors and relations are useful. Anchors allow us to group entities according to their semantic meanings and thus to select of the most prominent candidates. Relations allow us to capture more invariant representation of instances. However, a sentence may contain very few high-quality relations. It implies that the relations ranking step is fuzzy in nature. In addition, we noticed that some anchor cues may be missing, whereas the other anchor types may be represented by several anchor cues. All these factors lead only to moderate improvement in performance, especially in comparison with GRID system.

b) *Overall*, the splitting of instances into categories turned out to be useful. Due to the application of specific strategies the performance increased by 1% over the baseline. However, the large dominance of the hard cases (65%) made this improvement modest.

c) We notice that the amount of variations for connecting anchor cues in the *Simple* category is relatively small. Therefore, the overall performance for this case reaches F_1=82%. The main errors here come from missing anchors resulting partly from mistakes in such component as NE detection.

d) The performance in the *Average* category is F_1=67%. It is lower than that for the simple category because of higher variability in relations and negative influence of support verbs. For example, for excerpt such as "X *investigated* murder *of* Y", the processing tends to make mistake without the analysis of semantic value of support verb 'investigated'.

e) *Hard* category achieves the lowest performance of F_1=51% among all the categories. Since for this category we have to rely mostly on anchors, the problem arises if these anchors provide the wrong clues. It happens if some of them are missing or are wrongly extracted. The other cause of mistakes is when ARE finds several anchor cues which belong to the same type.

Additional usage of promotion strategies allowed us to improve the performance further.

f) Overall, the addition of promotion strategy enables the system to further boost the performance to F_1=60%. It means that the promotion strategy is useful, especially for the average case. The improvement in comparison to the state-of-art system GRID is about 3%.

g) It achieved an F_1=69%, which is an improvement of 2%, for the *Average* category. It implies that the analysis of support verbs helps in revealing the differences between the instances such as "X *was involved in* kidnapping *of* Y" and "X *reported* kidnapping *of* Y".

h) The results in the *Hard* category improved moderately to F_1=52%. The reason for the improvement is that more anchor cues are captured after the promotion. Still, there are 2 types of common mis-

takes: 1) multiple or missing anchor cues of the same type and 2) anchors can be spread across several sentences or several clauses in the same sentence.

Case (%)	Without promotion			With promotion		
	P	R	F_1	P	R	F_1
Chieu et al.'02	74%	49%	59%	-	-	-
Anc+rel (100%)	78%	52%	62%	78%	52%	62%
Overall (100%)	72%	58%	64%	73%	58%	65%
Simple (45%)	85%	67%	75%	87%	68%	76%
Average (27%)	61%	55%	58%	64%	56%	60%
Hard (28%)	59%	44%	50%	59%	44%	50%

Table 5. Results on MUC6 with case splitting

For the MUC6 results given in Table 5, we observe that the overall improvement in performance of ARE system over Chieu et al.'02 is 6%. The trends of results for MUC6 are similar to that in MUC4. However, there are few important differences. First, 45% of instances in MUC6 fall into the S*imple* category, therefore this category dominates. The reason for this is that the terminologies used in Management Succession domain are more stable in comparison to the Terrorism domain. Second, there are more anchor types for this case and therefore the promotion strategy is applicable also to the simple case. Third, there is no improvement in performance for the *Hard* category. We believe the primary reason for it is that more stable language patterns are used in MUC6. Therefore, dependency relations are also more stable in MUC6 and the promotion strategy is not very important. Similar to MUC4, there are problems of missing anchors and mistakes in dependency parsing.

6 Conclusion

The current state-of-art IE methods tend to use co-occurrence relations for extraction of entities. Although context may provide a meaningful clue, the use of co-occurrence relations alone has serious limitations because of alignment and paraphrasing problems. In our work, we proposed to utilize dependency relations to tackle these problems. Based on the extracted anchor cues and relations between them, we split instances into 'simple', 'average' and 'hard' categories. For each category, we applied specific strategy. This approach allowed us to outperform the existing state-of-art approaches by 3% on Terrorism domain and 6% on Management Succession domain. In our future work we plan to investigate the role of semantic relations and integrate ontology in the rule generation process. Another direction is to explore the use of bootstrapping and transduction approaches that may require less training instances.

References

H.L. Chieu and H.T. Ng. 2002. A Maximum Entropy Approach to Information Extraction from Semi-Structured and Free Text. *In Proc of AAAI-2002*, 786-791.

H. Cui, M.Y. Kan, and Chua T.S. 2005. *Generic Soft Pattern Models for Definitional Question Answering.* In Proc of ACM SIGIR-2005.

A. Culotta and J. Sorensen J. 2004. Dependency tree kernels for relation extraction. *In Proc of ACL-2004*.

F. Ciravegna. 2001. Adaptive Information Extraction from Text by Rule Induction and Generalization. *In Proc of IJCAI-2001*.

A. Dempster, N. Laird, and D. Rubin. 1977. *Maximum likelihood from incomplete data via the EM algorithm.* Journal of the Royal Statistical Society B, 39(1):1–38

K. Humphreys, G. Demetriou and R. Gaizuskas. 2000. Two applications of Information Extraction to Biological Science: Enzyme interactions and Protein structures. *In Proc of the Pacific Symposium on Biocomputing*, 502-513

M. Kaufmann. 1992. MUC-4. *In Proc of MUC-4.*

M. Kaufmann. 1995. MUC-6. *In Proc of MUC-6.*

J. Kim and D. Moldovan. 1995. Acquisition of linguistic patterns for knowledge-based information extraction. *IEEE Transactions on KDE,* 7(5): 713-724

D. Lin. 1997. Using Syntactic Dependency as Local Context to Resolve Word Sense Ambiguity. *In Proc of ACL-97.*

E. Riloff. 1996. Automatically Generating Extraction Patterns from Untagged Text. *In Proc of AAAI-96*, 1044-1049.

D. Roth and W. Yih. 2002. Probabilistic Reasoning for Entity & Relation Recognition. In Proc of *COLING-2002.*

S. Soderland, D. Fisher, J. Aseltine and W. Lehnert. 1995. Crystal: Inducing a Conceptual Dictionary. *In Proc of IJCAI-95*, 1314-1319.

S. Soderland. 1999. Learning Information Extraction Rules for Semi-Structured and Free Text. *Machine Learning* 34:233-272.

J. Xiao, T.S. Chua and H. Cui. 2004. Cascading Use of Soft and Hard Matching Pattern Rules for Weakly Supervised Information Extraction. *In Proc of COLING-2004.*

H. Yang, H. Cui, M.-Y. Kan, M. Maslennikov, L. Qiu and T.-S. Chua. 2003. QUALIFIER in TREC 12 QA Main Task. *In Proc of TREC-12,* 54-65.

R. Yangarber, W. Lin, R. Grishman. 2002. Unsupervised Learning of Generalized Names. *In Proc of COLING-2002.*

G.D. Zhou and J. Su. 2002. Named entity recognition using an HMM-based chunk tagger. *In Proc of ACL-2002,* 473-480

Integrating Pattern-based and Distributional Similarity Methods for Lexical Entailment Acquisition

Shachar Mirkin
School of Computer Science and Engineering
The Hebrew University, Jerusalem, Israel, 91904
mirkin@cs.huji.ac.il

Ido Dagan **Maayan Geffet**
Department of Computer Science
Bar-Ilan University, Ramat Gan, Israel, 52900
{dagan,zitima}@cs.biu.ac.il

Abstract

This paper addresses the problem of acquiring lexical semantic relationships, applied to the lexical entailment relation. Our main contribution is a novel conceptual integration between the two distinct acquisition paradigms for lexical relations – the pattern-based and the distributional similarity approaches. The integrated method exploits mutual complementary information of the two approaches to obtain candidate relations and informative characterizing features. Then, a small size training set is used to construct a more accurate supervised classifier, showing significant increase in both recall and precision over the original approaches.

1 Introduction

Learning lexical semantic relationships is a fundamental task needed for most text understanding applications. Several types of lexical semantic relations were proposed as a goal for automatic acquisition. These include lexical ontological relations such as synonymy, hyponymy and meronymy, aiming to automate the construction of WordNet-style relations. Another common target is learning general distributional similarity between words, following Harris' Distributional Hypothesis (Harris, 1968). Recently, an applied notion of entailment between lexical items was proposed as capturing major inference needs which cut across multiple semantic relationship types (see Section 2 for further background).

The literature suggests two major approaches for learning lexical semantic relations: distributional similarity and pattern-based. The first approach recognizes that two words (or two multiword terms) are semantically similar based on distributional similarity of the different contexts in which the two words occur. The distributional method identifies a somewhat loose notion of semantic similarity, such as between *company* and *government*, which does not ensure that the meaning of one word can be substituted by the other. The second approach is based on identifying joint occurrences of the two words within particular patterns, which typically indicate directly concrete semantic relationships. The pattern-based approach tends to yield more accurate hyponymy and (some) meronymy relations, but is less suited to acquire synonyms which only rarely co-occur within short patterns in texts. It should be noted that the pattern-based approach is commonly applied also for information and knowledge extraction to acquire factual instances of concrete meaning relationships (e.g. *born in*, *located at*) rather than generic lexical semantic relationships in the language.

While the two acquisition approaches are largely complementary, there have been just few attempts to combine them, usually by pipeline architecture. In this paper we propose a methodology for integrating distributional similarity with the pattern-based approach. In particular, we focus on learning the lexical entailment relationship between common nouns and noun phrases (to be distinguished from learning relationships for proper nouns, which usually falls within the knowledge acquisition paradigm).

The underlying idea is to first identify candidate relationships by both the distributional approach, which is applied exhaustively to a local corpus, and the pattern-based approach, applied to the web. Next, each candidate is represented by a unified set of distributional and pattern-based features. Finally, using a small training set we devise a supervised (SVM) model that classifies new candidate relations as correct or incorrect.

To implement the integrated approach we developed state of the art pattern-based acquisition

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 579–586,
Sydney, July 2006.

methods and utilized a distributional similarity method that was previously shown to provide superior performance for lexical entailment acquisition. Our empirical results show that the integrated method significantly outperforms each approach in isolation, as well as the naïve combination of their outputs. Overall, our method reveals complementary types of information that can be obtained from the two approaches.

2 Background

2.1 Distributional Similarity and Lexical Entailment

The general idea behind distributional similarity is that words which occur within similar contexts are semantically similar (Harris, 1968). In a computational framework, words are represented by feature vectors, where features are context words weighted by a function of their statistical association with the target word. The degree of similarity between two target words is then determined by a vector comparison function. Amongst the many proposals for distributional similarity measures, (Lin, 1998) is maybe the most widely used one, while (Weeds et al., 2004) provides a typical example for recent research. Distributional similarity measures are typically computed through exhaustive processing of a corpus, and are therefore applicable to corpora of bounded size.

It was noted recently by Geffet and Dagan (2004, 2005) that distributional similarity captures a quite loose notion of semantic similarity, as exemplified by the pair *country – party* (identified by Lin's similarity measure). Consequently, they proposed a definition for the *lexical entailment* relation, which conforms to the general framework of applied textual entailment (Dagan et al., 2005). Generally speaking, a word *w* lexically entails another word *v* if *w* can substitute *v* in some contexts while implying *v*'s original meaning. It was suggested that lexical entailment captures major application needs in modeling lexical variability, generalized over several types of known ontological relationships. For example, in Question Answering (QA), the word *company* in a question can be substituted in the text by *firm* (synonym), *automaker* (hyponym) or *subsidiary* (meronym), all of which entail *company*. Typically, hyponyms entail their hypernyms and synonyms entail each other, while entailment holds for meronymy only in certain cases.

In this paper we investigate automatic acquisition of the lexical entailment relation. For the distributional similarity component we employ the similarity scheme of (Geffet and Dagan, 2004), which was shown to yield improved predictions of (non-directional) lexical entailment pairs. This scheme utilizes the symmetric similarity measure of (Lin, 1998) to induce improved feature weights via bootstrapping. These weights identify the most characteristic features of each word, yielding cleaner feature vector representations and better similarity assessments.

2.2 Pattern-based Approaches

Hearst (1992) pioneered the use of lexical-syntactic patterns for automatic extraction of lexical semantic relationships. She acquired hyponymy relations based on a small predefined set of highly indicative patterns, such as "*X, . . . , Y and/or other Z*", and "*Z such as X, . . . and/or Y*", where *X* and *Y* are extracted as hyponyms of *Z*. Similar techniques were further applied to predict hyponymy and meronymy relationships using lexical or lexico-syntactic patterns (Berland and Charniak, 1999; Sundblad, 2002), and web page structure was exploited to extract hyponymy relationships by Shinzato and Torisawa (2004). Chklovski and Pantel (2004) used patterns to extract a set of relations between verbs, such as similarity, strength and antonymy. Synonyms, on the other hand, are rarely found in such patterns. In addition to their use for learning lexical semantic relations, patterns were commonly used to learn instances of concrete semantic relations for Information Extraction (IE) and QA, as in (Riloff and Shepherd, 1997; Ravichandran and Hovy, 2002; Yangarber et al., 2000).

Patterns identify rather specific and informative structures within particular co-occurrences of the related words. Consequently, they are relatively reliable and tend to be more accurate than distributional evidence. On the other hand, they are susceptive to data sparseness in a limited size corpus. To obtain sufficient coverage, recent works such as (Chklovski and Pantel, 2004) applied pattern-based approaches to the web. These methods form search engine queries that match likely pattern instances, which may be verified by post-processing the retrieved texts.

Another extension of the approach was automatic enrichment of the pattern set through bootstrapping. Initially, some instances of the sought

1	NP_1 such as NP_2
2	Such NP_1 as NP_2
3	NP_1 or other NP_2
4	NP_1 and other NP_2
5	NP_1 ADV known as NP_2
6	NP_1 especially NP_2
7	NP_1 like NP_2
8	NP_1 including NP_2
9	NP_1-sg is (a OR an) NP_2-sg
10	NP_1-sg (a OR an) NP_2-sg
11	NP_1-pl are NP_2-pl

Table 1: The patterns we used for entailment acquisition based on (Hearst, 1992) and (Pantel et al., 2004). Capitalized terms indicate variables. *pl* and *sg* stand for plural and singular forms.

relation are found based on a set of manually defined patterns. Then, additional co-occurrences of the related terms are retrieved, from which new patterns are extracted (Riloff and Jones, 1999; Pantel et al., 2004). Eventually, the list of effective patterns found for ontological relations has pretty much converged in the literature. Amongst these, Table 1 lists the patterns that were utilized in our work.

Finally, the selection of candidate pairs for a target relation was usually based on some function over the statistics of matched patterns. To perform more systematic selection Etzioni et al. (2004) applied a supervised Machine Learning algorithm (Naïve Bayes), using pattern statistics as features. Their work was done within the IE framework, aiming to extract semantic relation instances for proper nouns, which occur quite frequently in indicative patterns. In our work we incorporate and extend the supervised learning step for the more difficult task of acquiring general language relationships between common nouns.

2.3 Combined Approaches

It can be noticed that the pattern-based and distributional approaches have certain complementary properties. The pattern-based method tends to be more precise, and also indicates the direction of the relationship between the candidate terms. The distributional similarity approach is more exhaustive and suitable to detect symmetric synonymy relations. Few recent attempts on related (though different) tasks were made to classify (Lin et al., 2003) and label (Pantel and Ravichandran, 2004) distributional similarity output using lexical-syntactic patterns, in a pipeline architecture. We aim to achieve tighter integration of the two approaches, as described next.

3 An Integrated Approach for Lexical Entailment Acquisition

This section describes our integrated approach for acquiring lexical entailment relationships, applied to common nouns. The algorithm receives as input a *target term* and aims to acquire a set of terms that either entail or are entailed by it. We denote a pair consisting of the input target term and an acquired entailing/entailed term as *entailment pair*. Entailment pairs are directional, as in *bank* → *company*.

Our approach applies a supervised learning scheme, using SVM, to classify candidate entailment pairs as correct or incorrect. The SVM training phase is applied to a small constant number of training pairs, yielding a classification model that is then used to classify new test entailment pairs. The designated training set is also used to tune some additional parameters of the method. Overall, the method consists of the following main components:

1: Acquiring candidate entailment pairs for the input term by pattern-based and distributional similarity methods (Section 3.2);

2: Constructing a feature set for all candidates based on pattern-based and distributional information (Section 3.3);

3: Applying SVM training and classification to the candidate pairs (Section 3.4).

The first two components, of acquiring candidate pairs and collecting features for them, utilize a generic module for pattern-based extraction from the web, which is described first in Section 3.1.

3.1 Pattern-based Extraction Module

The general pattern-based extraction module receives as input a set of lexical-syntactic patterns (as in Table 1) and either a target term or a candidate pair of terms. It then searches the web for occurrences of the patterns with the input term(s). A small set of effective queries is created for each pattern-terms combination, aiming to retrieve as much relevant data with as few queries as possible.

Each pattern has two variable slots to be instantiated by candidate terms for the sought relation. Accordingly, the extraction module can be

used in two modes: (a) receiving a single target term as input and searching for instantiations of the other variable to identify candidate related terms (as in Section 3.2); (b) receiving a candidate pair of terms for the relation and searching pattern instances with both terms, in order to validate and collect information about the relationship between the terms (as in Section 3.3). Google proximity search[1] provides a useful tool for these purposes, as it allows using a wildcard which might match either an un-instantiated term or optional words such as modifiers. For example, the query *"such ** as *** (war OR wars)"* is one of the queries created for the input pattern *such NP_1 as NP_2* and the input target term *war*, allowing new terms to match the first pattern variable. For the candidate entailment pair *war* → *struggle*, the first variable is instantiated as well. The corresponding query would be: *"such * (struggle OR struggles) as *** (war OR wars)"*. This technique allows matching terms that are sub-parts of more complex noun phrases as well as multi-word terms.

The automatically constructed queries, covering the possible combinations of multiple wildcards, are submitted to Google[2] and a specified number of snippets is downloaded, while avoiding duplicates. The snippets are passed through a word splitter and a sentence segmenter[3], while filtering individual sentences that do not contain all search terms. Next, the sentences are processed with the OpenNLP[4] POS tagger and NP chunker. Finally, pattern-specific regular expressions over the chunked sentences are applied to verify that the instantiated pattern indeed occurs in the sentence, and to identify variable instantiations.

On average, this method extracted more than 3300 relationship instances for every 1MB of downloaded text, almost third of them contained multi-word terms.

3.2 Candidate Acquisition

Given an input target term we first employ pattern-based extraction to acquire entailment pair candidates and then augment the candidate set with pairs obtained through distributional similarity.

[1] Previously used by (Chklovski and Pantel, 2004).
[2] http://www.google.com/apis/
[3] Available from the University of Illinois at Urbana-Champaign, http://l2r.cs.uiuc.edu/~cogcomp/tools.php
[4] www.opennlp.sourceforge.net/

3.2.1 Pattern-based Candidates

At the candidate acquisition phase pattern instances are searched with one input target term, looking for instantiations of the other pattern variable to become the candidate related term (the first querying mode described in Section 3.1). We construct two types of queries, in which the target term is either the first or second variable in the pattern, which corresponds to finding either entailing or entailed terms that instantiate the other variable.

In the candidate acquisition phase we utilized patterns 1-8 in Table 1, which we empirically found as most suitable for identifying directional lexical entailment pairs. Patterns 9-11 are not used at this stage as they produce too much noise when searched with only one instantiated variable. About 35 queries are created for each target term in each entailment direction for each of the 8 patterns. For every query, the first *n* snippets are downloaded (we used *n=50*). The downloaded snippets are processed as described in Section 3.1, and candidate related terms are extracted, yielding candidate entailment pairs with the input target term.

Quite often the entailment relation holds between multi-word noun-phrases rather than merely between their heads. For example, *trade center* lexically entails *shopping complex*, while *center* does not necessarily entail *complex*. On the other hand, many complex multi-word noun phrases are too rare to make a statistically based decision about their relation with other terms. Hence, we apply the following two criteria to balance these constraints:

1. For the entailing term we extract only the complete noun-chunk which instantiate the pattern. For example: we extract *housing project* → *complex*, but do not extract *project* as entailing *complex* since the head noun alone is often too general to entail the other term.
2. For the entailed term we extract both the complete noun-phrase and its head in order to create two separate candidate entailment pairs with the entailing term, which will be judged eventually according to their overall statistics.

As it turns out, a large portion of the extracted pairs constitute trivial hyponymy relations, where one term is a modified version of the other, like *low interest loan* → *loan*. These pairs were removed, along with numerous pairs including proper nouns, following the goal of learning en-

tailment relationships for distinct common nouns.

Finally, we filter out the candidate pairs whose frequency in the extracted patterns is less than a threshold, which was set empirically to 3. Using a lower threshold yielded poor precision, while a threshold of 4 decreased recall substantially with just little effect on precision.

3.2.2 Distributional Similarity Candidates

As mentioned in Section 2, we employ the distributional similarity measure of (Geffet and Dagan, 2004) (denoted here *GD04* for brevity), which was found effective for extracting non-directional lexical entailment pairs. Using local corpus statistics, this algorithm produces for each target noun a scored list of up to a few hundred words with positive distributional similarity scores.

Next we need to determine an optimal threshold for the similarity score, considering words above it as likely entailment candidates. To tune such a threshold we followed the original methodology used to evaluate *GD04*. First, the top-*k* (*k=40*) similarities of each training term are manually annotated by the lexical entailment criterion (see Section 4.1). Then, the similarity value which yields the maximal micro-averaged F1 score is selected as threshold, suggesting an optimal recall-precision tradeoff. The selected threshold is then used to filter the candidate similarity lists of the test words.

Finally, we remove all entailment pairs that already appear in the candidate set of the pattern-based approach, in either direction (recall that the distributional candidates are non-directional). Each of the remaining candidates generates two directional pairs which are added to the unified candidate set of the two approaches.

3.3 Feature Construction

Next, each candidate is represented by a set of features, suitable for supervised classification. To this end we developed a novel feature set based on both pattern-based and distributional data.

To obtain pattern statistics for each pair, the second mode of the pattern-based extraction module is applied (see Section 3.1). As in this case, both variables in the pattern are instantiated by the terms of the pair, we could use all eleven patterns in Table 1, creating a total of about 55 queries per pair and downloading *m*=20 snippets for each query. The downloaded snippets are processed as described in Section 3.1 to identify pattern matches and obtain relevant statistics for feature scores.

Following is the list of feature types computed for each candidate pair. The feature set was designed specifically for the task of extracting the complementary information of the two methods.

***Conditional Pattern Probability*:** This type of feature is created for each of the 11 individual patterns. The feature value is the estimated conditional probability of having the pattern matched in a sentence given that the pair of terms does appear in the sentence (calculated as the fraction of pattern matches for the pair amongst all unique sentences that contain the pair). This feature yields normalized scores for pattern matches regardless of the number of snippets retrieved for the given pair. This normalization is important in order to bring to equal grounds candidate pairs identified through either the pattern-based or distributional approaches, since the latter tend to occur less frequently in patterns.

Aggregated Conditional Pattern Probability: This single feature is the conditional probability that *any* of the patterns match in a retrieved sentence, given that the two terms appear in it. It is calculated like the previous feature, with counts aggregated over all patterns, and aims to capture overall appearance of the pair in patterns, regardless of the specific pattern.

***Conditional List-Pattern Probability*:** This feature was designed to eliminate the typical non-entailing cases of co-hyponyms (words sharing the same hypernym), which nevertheless tend to co-occur in entailment patterns. We therefore also check for pairs' occurrences in lists, using appropriate list patterns, expecting that correct entailment pairs would not co-occur in lists. The probability estimate, calculated like the previous one, is expected to be a negative feature for the learning model.

Relation Direction Ratio: The value of this feature is the ratio between the overall number of pattern matches for the pair and the number of pattern matches for the reversed pair (a pair created with the same terms in the opposite entailment direction). We found that this feature strongly correlates with entailment likelihood. Interestingly, it does not deteriorate performance for synonymous pairs.

Distributional Similarity Score: The *GD04* similarity score of the pair was used as a feature. We

PATTERN-BASED	DISTRIBU-TIONAL	TOTAL
1186	1420	2350

Table 2: The numbers of distinct entailment pair candidates obtained for the test words by each of the methods, and when combined.

also attempted adding Lin's (1998) similarity scores but they appeared to be redundant.

Intersection Feature: A binary feature indicating candidate pairs acquired by both methods, which was found to indicate higher entailment likelihood.

In summary, the above feature types utilize mutually complementary pattern-based and distributional information. Using cross validation over the training set we verified that each feature makes marginal contribution to performance when added on top of the remaining features.

3.4 Training and Classification

In order to systematically integrate different feature types we used the state-of-the-art supervised classifier SVM^{light} (Joachims, 1999) for entailment pair classification. Using 10-fold cross-validation over the training set we obtained the SVM configuration that yields an optimal micro-averaged F1 score. Through this optimization we chose the RBF kernel function and obtained optimal values for the *J*, *C* and the RBF's *Gamma* parameters. The candidate test pairs classified as correct entailments constitute the output of our integrated method.

4 Empirical Results

4.1 Data Set and Annotation

We utilized the experimental data set from Geffet and Dagan (2004). The dataset includes the similarity lists calculated by *GD04* for a sample of 30 target (common) nouns, computed from an 18 million word subset of the Reuters corpus[5]. We randomly picked a small set of 10 terms for training, leaving the remaining 20 terms for testing. Then, the set of entailment pair candidates for all nouns was created by applying the filtering method of Section 3.2.2 to the distributional similarity lists, and by extracting pattern-based candidates from the web as described in Section 3.2.1.

[5] Reuters Corpus, Volume 1, English Language, 1996-08-20 to 1997-08-19.

METHOD	P	R	F
Pattern-based	0.44	0.61	0.51
Distributional Similarity	0.33	0.53	0.40
Naïve Combination	0.36	1.00	0.53
Integrated	0.57	0.69	0.62

Table 3: Precision, Recall and F1 figures for the test words under each method.

Gold standard annotations for entailment pairs were created by three judges. The judges were guided to annotate as "Correct" the pairs conforming to the lexical entailment definition, which was reflected in two operational tests: i) *Word meaning entailment*: whether the meaning of the first (entailing) term implies the meaning of the second (entailed) term under some common sense of the two terms; and ii) *Substitutability*: whether the first term can substitute the second term in some natural contexts, such that the meaning of the modified context entails the meaning of the original one. The obtained Kappa values (varying between 0.7 and 0.8) correspond to *substantial agreement* on the task.

4.2 Results

The numbers of candidate entailment pairs collected for the test terms are shown in Table 2. These figures highlight the markedly complementary yield of the two acquisition approaches, where only about 10% of all candidates were identified by both methods. On average, 120 candidate entailment pairs were acquired for each target term.

The SVM classifier was trained on a quite small annotated sample of 700 candidate entailment pairs of the 10 training terms. Table 3 presents comparative results for the classifier, for each of the two sets of candidates produced by each method alone, and for the union of these two sets (referred as *Naïve Combination*). The results were computed for an annotated random sample of about 400 candidate entailment pairs of the test terms. Following common pooling evaluations in Information Retrieval, recall is calculated relatively to the total number of correct entailment pairs acquired by both methods together.

Pattern-based	Distributional
abduction → abuse	assault ↔ abuse
government → organization	government ↔ administration
drug therapy → treatment	budget deficit →gap
gap → hazard*	broker → analyst*
management → issue*	government → parliament*

Table 4: Typical entailment pairs acquired by the integrated method, illustrating Section 4.3. The columns specify the method that produced the candidate pair. Asterisk indicates a non-entailing pair.

The first two rows of the table show quite moderate precision and recall for the candidates of each separate method. The next row shows the great impact of method combination on recall, relative to the amount of correct entailment pairs found by each method alone, validating the complementary yield of the approaches. The integrated classifier, applied to the combined set of candidates, succeeds to increase precision substantially by 21 points (a relative increase of almost 60%), which is especially important for many precision-oriented applications like Information Retrieval and Question Answering. The precision increase comes with the expense of some recall, yet having F1 improved by 9 points. The integrated method yielded on average about 30 correct entailments per target term. Its classification *accuracy* (percent of correct classifications) reached 70%, which nearly doubles the naïve combination's accuracy.

It is impossible to directly compare our results with those of other works on lexical semantic relationships acquisition, since the particular task definition and dataset are different. As a rough reference point, our result figures do match those of related papers reviewed in Section 2, while we notice that our setting is relatively more difficult since we excluded the easier cases of proper nouns. (Geffet and Dagan, 2005), who exploited the distributional similarity approach over the web to address the same task as ours, obtained higher precision but substantially lower recall, considering only distributional candidates. Further research is suggested to investigate integrating their approach with ours.

4.3 Analysis and Discussion

Analysis of the data confirmed that the two methods tend to discover different types of relations. As expected, the distributional similarity method contributed most (75%) of the synonyms that were correctly classified as mutually entailing pairs (e.g. *assault* ↔ *abuse* in Table 4). On the other hand, about 80% of all correctly identified hyponymy relations were produced by the pattern-based method (e.g. *abduction* → *abuse*).

The integrated method provides a means to determine the entailment direction for distributional similarity candidates which by construction are non-directional. Thus, amongst the (non-synonymous) distributional similarity pairs classified as entailing, the direction of 73% was correctly identified. In addition, the integrated method successfully filters 65% of the non-entailing co-hyponym candidates (hyponyms of the same hypernym), most of them originated in the distributional candidates, which is a large portion (23%) of all correctly discarded pairs. Consequently, the precision of distributional similarity candidates approved by the integrated system was nearly doubled, indicating the additional information that patterns provide about distributionally similar pairs.

Yet, several error cases were detected and categorized. First, many non-entailing pairs are context-dependent, such as a *gap* which might constitute a *hazard* in some particular contexts, even though these words do not entail each other in their general meanings. Such cases are more typical for the pattern-based approach, which is sometimes permissive with respect to the relationship captured and may also extract candidates from a relatively small number of pattern occurrences. Second, synonyms tend to appear less frequently in patterns. Consequently, some synonymous pairs discovered by distributional similarity were rejected due to insufficient pattern matches. Anecdotally, some typos and spelling alternatives, like *privatization* ↔ *privatisation*, are also included in this category as they never co-occur in patterns.

In addition, a large portion of errors is caused by pattern ambiguity. For example, the pattern *"NP_1, a|an NP_2"*, ranked among the top IS-A patterns by (Pantel et al., 2004), can represent both apposition (entailing) and a list of co-hyponyms (non-entailing). Finally, some misclassifications can be attributed to technical web-based processing errors and to corpus data sparseness.

5 Conclusion

The main contribution of this paper is a novel integration of the pattern-based and distributional approaches for lexical semantic acquisition, applied to lexical entailment. Our investigation highlights the complementary nature of the two approaches and the information they provide. Notably, it is possible to extract pattern-based information that complements the weaker evidence of distributional similarity. Supervised learning was found effective for integrating the different information types, yielding noticeably improved performance. Indeed, our analysis reveals that the integrated approach helps eliminating many error cases typical to each method alone. We suggest that this line of research may be investigated further to enrich and optimize the learning processes and to address additional lexical relationships.

Acknowledgement

We wish to thank Google for providing us with an extended quota for search queries, which made this research feasible.

References

Berland, Matthew and Charniak, Eugene. 1999. Finding parts in very large corpora. In Proc. of ACL-99. Maryland, USA.

Chklovski, Timothy and Patrick Pantel. 2004. VerbOcean: Mining the Web for Fine-Grained Semantic Verb Relations. In Proc. of EMNLP-04. Barcelona, Spain.

Dagan, Ido, Oren Glickman and Bernardo Magnini. 2005. The PASCAL Recognizing Textual Entailment Challenge. In Proc. of the PASCAL Challenges Workshop for Recognizing Textual Entailment. Southampton, U.K.

Etzioni, Oren, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D.S. Weld, and A. Yates. 2004. Web-scale information extraction in KnowItAll. In Proc. of WWW-04. NY, USA.

Geffet, Maayan and Ido Dagan. 2004. Feature Vector Quality and Distributional Similarity. In Proc. of COLING-04. Geneva, Switzerland.

Geffet, Maayan and Ido Dagan. 2005. The Distributional Inclusion Hypothesis and Lexical Entailment. In Proc of ACL-05. Michigan, USA.

Harris, Zelig S. 1968. Mathematical Structures of Language. Wiley.

Hearst, Marti. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. In Proc. of COLING-92. Nantes, France.

Joachims, Thorsten. 1999. Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning, B. Schölkopf and C. Burges and A. Smola (ed.), MIT-Press.

Lin, Dekang. 1998. Automatic Retrieval and Clustering of Similar Words. In Proc. of COLING–ACL98, Montreal, Canada.

Lin, Dekang, Shaojun Zhao, Lijuan Qin, and Ming Zhou. 2003. Identifying synonyms among distributionally similar words. In Proc. of IJCAI-03. Acapulco, Mexico.

Pantel, Patrick, Deepak Ravichandran, and Eduard Hovy. 2004. Towards Terascale Semantic Acquisition. In Proc. of COLING-04. Geneva, Switzerland.

Pantel, Patrick and Deepak Ravichandran. 2004. Automatically Labeling Semantic Classes. In Proc. of HLT/NAACL-04. Boston, MA.

Ravichandran, Deepak and Eduard Hovy. 2002. Learning Surface Text Patterns for a Question Answering System. In Proc. of ACL-02. Philadelphia, PA.

Riloff, Ellen and Jessica Shepherd. 1997. A corpus-based approach for building semantic lexicons. In Proc. of EMNLP-97. RI, USA.

Riloff, Ellen and Rosie Jones. 1999. Learning Dictionaries for Information Extraction by Multi-Level Bootstrapping. In Proc. of AAAI-99. Florida, USA.

Shinzato, Kenji and Kentaro Torisawa. 2004. Acquiring Hyponymy Relations from Web Documents. In Proc. of HLT/NAACL-04. Boston, MA.

Sundblad, H. Automatic Acquisition of Hyponyms and Meronyms from Question Corpora. 2002. In Proc. of the ECAI-02 Workshop on Natural Language Processing and Machine Learning for Ontology Engineering. Lyon, France.

Weeds, Julie, David Weir, and Diana McCarthy. 2004. Characterizing Measures of Lexical Distributional Similarity. In Proc. of COLING-04. Geneva, Switzerland.

Yangarber, Roman, Ralph Grishman, Pasi Tapanainen and Silja Huttunen. 2000. Automatic Acquisition of Domain Knowledge for Information Extraction. In Proc. of COLING-00. Saarbrücken, Germany.

Machine-Learning-Based Transformation of Passive Japanese Sentences into Active by Separating Training Data into Each Input Particle

Masaki Murata
National Institute of Information and Communications Technology
3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0289, Japan
murata@nict.go.jp

Toshiyuki Kanamaru
National Institute of Information and Communications Technology
3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0289, Japan
kanamaru@nict.go.jp

Tamotsu Shirado
National Institute of Information and Communications Technology
3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0289, Japan
shirado@nict.go.jp

Hitoshi Isahara
National Institute of Information and Communications Technology
3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0289, Japan
isahara@nict.go.jp

Abstract

We developed a new method of transforming Japanese case particles when transforming Japanese passive sentences into active sentences. It separates training data into each input particle and uses machine learning for each particle. We also used numerous rich features for learning. Our method obtained a high rate of accuracy (94.30%). In contrast, a method that did not separate training data for any input particles obtained a lower rate of accuracy (92.00%). In addition, a method that did not have many rich features for learning used in a previous study (Murata and Isahara, 2003) obtained a much lower accuracy rate (89.77%). We confirmed that these improvements were significant through a statistical test. We also conducted experiments utilizing traditional methods using verb dictionaries and manually prepared heuristic rules and confirmed that our method obtained much higher accuracy rates than traditional methods.

1 Introduction

This paper describes how passive Japanese sentences can be automatically transformed into active. There is an example of a passive Japanese sentence in Figure 1. The Japanese suffix *reta* functions as an auxiliary verb indicating the passive voice. There is a corresponding active-voice sentence in Figure 2. When the sentence in Figure 1 is transformed into an active sentence, (i) *ni* (by), which is a case postpositional particle with the meaning of "by", is changed into *ga*, which is a case postpositional particle indicating the subjective case, and (ii) *ga* (subject), which is a case postpositional particle indicating the subjective case, is changed into *wo* (object), which is a case postpositional particle indicating the objective case. In this paper, we discuss the transformation of Japanese case particles (i.e., *ni* → *ga*) through machine learning.[1]

The transformation of passive sentences into active is useful in many research areas including generation, knowledge extraction from databases written in natural languages, information extraction, and answering questions. For example, when the answer is in the passive voice and the question is in the active voice, a question-answering system cannot match the answer with the question because the sentence structures are different and it is thus difficult to find the answer to the question. Methods of transforming passive sentences into active are important in natural language processing.

The transformation of case particles in transforming passive sentences into active is not easy because particles depend on verbs and their use.

We developed a new method of transforming Japanese case particles when transforming passive Japanese sentences into active in this study. Our method separates training data into each input particle and uses machine learning for each input particle. We also used numerous rich features for learning. Our experiments confirmed that our method was effective.

[1] In this study, we did not handle the transformation of auxiliary verbs and the inflection change of verbs because these can be transformed based on Japanese grammar.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 587–594,
Sydney, July 2006.

inu *ni* *watashi* *ga* *kama-* *reta.*
(dog) (by) (I) *subjective-case postpositional particle* (bite) passive voice
(I was bitten by a dog.)

Figure 1: Passive sentence

inu *ni* *watashi* *ga* *kama-* *reta.*
ga wo
(dog) (by) (I) *subjective-case postpositional particle* (bite) passive voice
(I was bitten by a dog.)

Figure 3: Example in corpus

inu *ga* *watashi* *wo* *kanda.*
(dog) subject (I) object (bite)
(Dog bit me.)

Figure 2: Active sentence

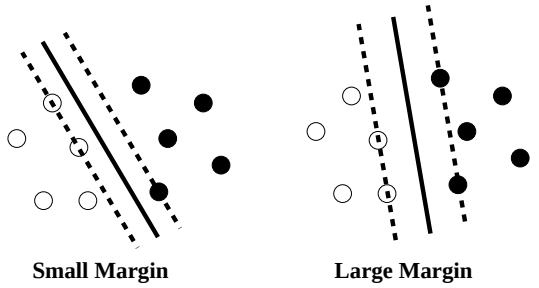


Figure 4: Maximizing margin

2 Tagged corpus as supervised data

We used the Kyoto University corpus (Kurohashi and Nagao, 1997) to construct a corpus tagged for the transformation of case particles. It has approximately 20,000 sentences (16 editions of the Mainichi Newspaper, from January 1st to 17th, 1995). We extracted case particles in passive-voice sentences from the Kyoto University corpus. There were 3,576 particles. We assigned a corresponding case particle for the active voice to each case particle. There is an example in Figure 3. The two underlined particles, "ga" and "wo" that are given for "*ni*" and "*ga*" are tags for case particles in the active voice. We called the given case particles for the active voice *target case particles*, and the original case particles in passive-voice sentences *source case particles*. We created tags for target case particles in the corpus. If we can determine the target case particles in a given sentence, we can transform the case particles in passive-voice sentences into case particles for the active voice. Therefore, our goal was to determine the target case particles.

3 Machine learning method (support vector machine)

We used a support vector machine as the basis of our machine-learning method. This is because support vector machines are comparatively better than other methods in many research areas (Kudoh and Matsumoto, 2000; Taira and Haruno, 2001; Murata et al., 2002).

Data consisting of two categories were classified by using a hyperplane to divide a space with the support vector machine. When these two categories were, positive and negative, for example, enlarging the margin between them in the training data (see Figure 4[2]), reduced the possibility of incorrectly choosing categories in blind data (test data). A hyperplane that maximized the margin was thus determined, and classification was done using that hyperplane. Although the basics of this method are as described above, the region between the margins through the training data can include a small number of examples in extended versions, and the linearity of the hyperplane can be changed to non-linear by using kernel functions. Classification in these extended versions is equivalent to classification using the following discernment function, and the two categories can be classified on the basis of whether the value output by the function is positive or negative (Cristianini and Shawe-Taylor, 2000; Kudoh, 2000):

[2]The open circles in the figure indicate positive examples and the black circles indicate negative. The solid line indicates the hyperplane dividing the space, and the broken lines indicate the planes depicting margins.

$$f(\mathbf{x}) = sgn\left(\sum_{i=1}^{l} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right) \quad (1)$$

$$b = \frac{max_{i,y_i=-1} b_i + min_{i,y_i=1} b_i}{2}$$

$$b_i = -\sum_{j=1}^{l} \alpha_j y_j K(\mathbf{x}_j, \mathbf{x}_i),$$

where $\mathbf{x}$ is the context (a set of features) of an input example, $\mathbf{x}_i$ indicates the context of a training datum, and y_i $(i = 1, ..., l, y_i \in \{1, -1\})$ indicates its category. Function sgn is:

$$sgn(x) = \begin{array}{ll} 1 & (x \geq 0), \\ -1 & (otherwise). \end{array} \quad (2)$$

Each α_i $(i = 1, 2...)$ is fixed as a value of α_i that maximizes the value of $L(\alpha)$ in Eq. (3) under the conditions set by Eqs. (4) and (5).

$$L(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j K(\mathbf{x_i}, \mathbf{x_j}) \quad (3)$$

$$0 \leq \alpha_i \leq C \ (i = 1, ..., l) \quad (4)$$

$$\sum_{i=1}^{l} \alpha_i y_i = 0 \quad (5)$$

Although function K is called a kernel function and various functions are used as kernel functions, we have exclusively used the following polynomial function:

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^d \quad (6)$$

C and d are constants set by experimentation. For all experiments reported in this paper, C was fixed as 1 and d was fixed as 2.

A set of $\mathbf{x}_i$ that satisfies $\alpha_i > 0$ is called a support vector, (SV_s)[3], and the summation portion of Eq. (1) is only calculated using examples that are support vectors. Equation 1 is expressed as follows by using support vectors.

$$f(\mathbf{x}) = sgn\left(\sum_{i:x_i \in SV_s} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right) \quad (7)$$

$$b = \frac{b_{i:y_i=-1, x_i \in SV_s} + b_{i:y_i=1, x_i \in SV_s}}{2}$$

$$b_i = -\sum_{i:x_i \in SV_s} \alpha_j y_j K(\mathbf{x}_j, \mathbf{x}_i),$$

[3]The circles on the broken lines in Figure 4 indicate support vectors.

Table 1: Features

F1	part of speech (POS) of P
F2	main word of P
F3	word of P
F4	first 1, 2, 3, 4, 5, and 7 digits of category number of P[5]
F5	auxiliary verb attached to P
F6	word of N
F7	first 1, 2, 3, 4, 5, and 7 digits of category number of N
F8	case particles and words of nominals that have dependency relationship with P and are other than N
F9	first 1, 2, 3, 4, 5, and 7 digits of category number of nominals that have dependency relationship with P and are other than N
F10	case particles of nominals that have dependency relationship with P and are other than N
F11	the words appearing in the same sentence
F12	first 3 and 5 digits of category number of words appearing in same sentence
F13	case particle taken by N (source case particle)
F14	target case particle output by KNP (Kurohashi, 1998)
F15	target case particle output with Kondo's method (Kondo et al., 2001)
F16	case patterns defined in IPAL dictionary (IPAL) (IPA, 1987)
F17	combination of predicate semantic primitives defined in IPAL
F18	predicate semantic primitives defined in IPAL
F19	combination of semantic primitives of N defined in IPAL
F20	semantic primitives of N defined in IPAL
F21	whether P is defined in IPAL or not
F22	whether P can be in passive form defined in VDIC[6]
F23	case particles of P defined in VDIC
F24	type of P defined in VDIC
F25	transformation rule used for P and N in Kondo's method
F26	whether P is defined in VDIC or not
F27	pattern of case particles of nominals that have dependency relationship with P
F28	pair of case particles of nominals that have dependency relationship with P
F29	case particles of nominals that have dependency relationship with P and appear before N
F30	case particles of nominals that have dependency relationship with P and appear after N
F31	case particles of nominals that have dependency relationship with P and appear just before N
F32	case particles of nominals that have dependency relationship with P and appear just after N

Table 2: Frequently occurring target case particles in source case particles

Source case particle	Occurrence rate	Frequent target case particles in source case particles	Occurrence rate in source case particles
ni (indirect object)	27.57% (493/1788)	*ni* (indirect object)	70.79% (349/493)
		ga (subject)	27.38% (135/493)
ga (subject)	26.96% (482/1788)	*wo* (direct object)	96.47% (465/482)
de (with)	17.17% (307/1788)	*ga* (subject)	79.15% (243/307)
		de (with)	13.36% (41/307)
to (with)	16.11% (288/1788)	*to* (with)	99.31% (286/288)
wo (direct object)	6.77% (121/1788)	*wo* (direct object)	99.17% (120/121)
kara (from)	4.53% (81/1788)	*ga* (subject)	49.38% (40/ 81)
		kara (from)	44.44% (36/ 81)
made (to)	0.78% (14/1788)	*made* (to)	100.00% (14/ 14)
he (to)	0.06% (1/1788)	*ga* (subject)	100.00% (1/ 1)
no (subject)	0.06% (1/1788)	*wo* (direct object)	100.00% (1/ 1)

Support vector machines are capable of handling data consisting of two categories. Data consisting of more than two categories is generally handled using the pair-wise method (Kudoh and Matsumoto, 2000).

Pairs of two different categories (N(N-1)/2 pairs) are constructed for data consisting of N categories with this method. The best category is determined by using a two-category classifier (in this paper, a support vector machine[4] is used as the two-category classifier), and the correct category is finally determined on the basis of "voting" on the N(N-1)/2 pairs that result from analysis with the two-category classifier.

The method discussed in this paper is in fact a combination of the support vector machine and the pair-wise method described above.

4 Features (information used in classification)

The features we used in our study are listed in Table 1, where N is a noun phrase connected to the case particle being analyzed, and P is the phrase's predicate. We used the Japanese syntactic parser, KNP (Kurohashi, 1998), for identifying N, P, parts of speech and syntactic relations.

In the experiments conducted in this study, we selected features. We used the following procedure to select them.

- Feature selection

  We first used all the features for learning. We next deleted only one feature from all the features for learning. We did this for every feature. We decided to delete features that would make the most improvement. We repeated this until we could not improve the rate of accuracy.

5 Method of separating training data into each input particle

We developed a new method of separating training data into each input (source) particle that uses machine learning for each particle. For example, when we identify a target particle where the source particle is *ni*, we use only the training data where the source particle is *ni*. When we identify a target particle where the source particle is *ga*, we use only the training data where the source particle is *ga*.

Frequently occurring target case particles are very different in source case particles. Frequently occurring target case particles in all source case particles are listed in Table 2. For example, when *ni* is a source case particle, frequently occurring

[4]We used Kudoh's TinySVM software (Kudoh, 2000) as the support vector machine.

[5]The category number indicates a semantic class of words. A Japanese thesaurus, the *Bunrui Goi Hyou* (NLRI, 1964), was used to determine the category number of each word. This thesaurus is 'is-a' hierarchical, in which each word has *a category number*. This is a 10-digit number that indicates seven levels of 'is-a' hierarchy. The top five levels are expressed by the first five digits, the sixth level is expressed by the next two digits, and the seventh level is expressed by the last three digits.

[6]Kondo et al. constructed a rich dictionary for Japanese verbs (Kondo et al., 2001). It defined types and characteristics of verbs. We will refer to it as VDIC.

Table 3: Occurrence rates for target case particles

Target case particle	Occurrence rate	
	Closed	Open
wo (direct object)	33.05%	29.92%
ni (indirect object)	19.69%	17.79%
to (with)	16.00%	18.90%
de (with)	13.65%	15.27%
ga (subject)	11.07%	10.01%
ga or *de*	2.40%	2.46%
kara (from)	2.13%	3.47%
Other	2.01%	1.79%

target case particles are *ni* or *ga*. In contrast, when *ga* is a source case particle, a frequently occurring target case particle is *wo*.

In this case, it is better to separate training data into each source particle and use machine learning for each particle. We therefore developed this method and confirmed that it was effective through experiments (Section 6).

6 Experiments

6.1 Basic experiments

We used the corpus we constructed described in Section 2 as supervised data. We divided the supervised data into closed and open data (Both the closed data and open data had 1788 items each.). The distribution of target case particles in the data are listed in Table 3. We used the closed data to determine features that were deleted in feature selection and used the open data as test data (data for evaluation). We used 10-fold cross validation for the experiments on closed data and we used closed data as the training data for the experiments on open data. The target case particles were determined by using the machine-learning method explained in Section 3. When multiple target particles could have been answers in the training data, we used pairs of them as answers for machine learning.

The experimental results are listed in Tables 4 and 5. Baseline 1 outputs a source case particle as the target case particle. Baseline 2 outputs the most frequent target case particle (*wo* (direct object)) in the closed data as the target case particle in every case. Baseline 3 outputs the most frequent target case particle for each source target case particle in the closed data as the target case particle. For example, *ni* (indirect object) is the most frequent target case particle when the source case particle is *ni*, as listed in Table 2. Baseline 3 outputs *ni* when the source case particle is *ni*. KNP indicates the results that the Japanese syntactic parser, KNP (Kurohashi, 1998), output. Kondo indicates the results that Kondo's method, (Kondo et al., 2001), output. KNP and Kondo can only work when a target predicate is defined in the IPAL dictionary or the VDIC dictionary. Otherwise, KNP and Kondo output nothing. "KNP/Kondo + Baseline X" indicates the use of outputs by Baseline X when KNP/Kondo have output nothing. KNP and Kondo are traditional methods using verb dictionaries and manually prepared heuristic rules. These traditional methods were used in this study to compare them with ours. "Murata 2003" indicates results using a method they developed in a previous study (Murata and Isahara, 2003). This method uses F1, F2, F5, F6, F7, F10, and F13 as features and does not have training data for any source case particles. "Division" indicates separating training data into each source particle. "No-division" indicates not separating training data for any source particles. "All features" indicates the use of all features with no features being selected. "Feature selection" indicates features are selected. We did two kinds of evaluations: "Eval. A" and "Eval. B". There are some cases where multiple target case particles can be answers. For example, *ga* and *de* can be answers. We judged the result to be correct in "Eval. A" when *ga* and *de* could be answers and the system output the pair of *ga* and *de* as answers. We judged the result to be correct in "Eval. B" when *ga* and *de* could be answers and the system output *ga*, *de*, or the pair of *ga* and *de* as answers.

Table 4 lists the results using all data. Table 5 lists the results where a target predicate is defined in the IPAL and VDIC dictionaries. There were 551 items in the closed data and 539 in the open.

We found the following from the results.

Although selection of features obtained higher rates of accuracy than use of all features in the closed data, it did not obtain higher rates of accuracy in the open data. This indicates that feature selection was not effective and we should have used all features in this study.

Our method using all features in the open data and separating training data into each source particle obtained the highest rate of accuracy (94.30% in Eval. B). This indicates that our method is ef-

Table 4: Experimental results

Method	Closed data		Open data	
	Eval. A	Eval. B	Eval. A	Eval. B
Baseline 1	58.67%	61.41%	62.02%	64.60%
Baseline 2	33.05%	33.56%	29.92%	30.37%
Baseline 3	84.17%	88.20%	84.17%	88.20%
KNP	27.35%	28.69%	27.91%	29.14%
KNP + Baseline 1	64.32%	67.06%	67.79%	70.36%
KNP + Baseline 2	48.10%	48.99%	45.97%	46.48%
KNP + Baseline 3	81.21%	84.84%	80.82%	84.45%
Kondo	39.21%	40.88%	39.32%	41.00%
Kondo + Baseline 1	65.27%	68.57%	67.34%	70.41%
Kondo + Baseline 2	54.87%	56.54%	53.52%	55.26%
Kondo + Baseline 3	78.08%	81.71%	78.30%	81.88%
Murata 2003	86.86%	89.09%	87.86%	89.77%
Our method, no-division + all features	89.99%	92.39%	90.04%	92.00%
Our method, no-division + feature selection	91.28%	93.40%	90.10%	92.00%
Our method, division + all features	91.22%	93.79%	92.28%	94.30%
Our method, division + feature selection	92.06%	94.41%	91.89%	93.85%

Table 5: Experimental results on data that can use IPAL and VDIC dictionaries

Method	Closed data		Open data	
	Eval. A	Eval. B	Eval. A	Eval. B
Baseline 1	57.71%	58.98%	58.63%	58.81%
Baseline 2	37.39%	37.39%	37.29%	37.29%
Baseline 3	84.03%	86.57%	86.83%	88.31%
KNP	74.59%	75.86%	75.88%	76.07%
Kondo	76.04%	77.50%	78.66%	78.85%
Our method, no-division + all features	94.19%	95.46%	94.81%	94.81%
Our method, division + all features	95.83%	96.91%	97.03%	97.03%

fective.

Our method that used all the features and did not separate training data for any source particles obtained an accuracy rate of 92.00% in Eval. B. The technique of separating training data into each source particles made an improvement of 2.30%. We confirmed that this improvement has a significance level of 0.01 by using a two-sided binomial test (two-sided sign test). This indicates that the technique of separating training data for all source particles is effective.

Murata 2003 who used only seven features and did not separate training data for any source particles obtained an accuracy rate of 89.77% with Eval. B. The method (92.00%) of using all features (32) made an improvement of 2.23% against theirs. We confirmed that this improvement had a significance level of 0.01 by using a two-sided binomial test (two-sided sign test). This indicates that our increased features are effective.

KNP and Kondo obtained low accuracy rates (29.14% and 41.00% in Eval. B for the open data). We did the evaluation using data and proved that these methods could work well. A target predicate in the data is defined in the IPAL and VDIC dictionaries. The results are listed in Table 5. KNP and Kondo obtained relatively higher accuracy rates (76.07% and 78.85% in Eval. B for the open data). However, they were lower than that for Baseline 3.

Baseline 3 obtained a relatively high accuracy rate (84.17% and 88.20% in Eval. B for the open data). Baseline 3 is similar to our method in terms of separating the training data into source particles. Baseline 3 separates the training data into

Table 6: Deletion of features

Deleted features	Closed data				Open data			
	Eval. A		Eval. B		Eval. A		Eval. B	
	Acc.	Diff.	Acc.	Diff.	Acc.	Diff.	Acc.	Diff.
Not deleted	91.22%	—	93.79%	—	92.28%	—	94.30%	—
F1	91.16%	-0.06%	93.74%	-0.05%	92.23%	-0.05%	94.24%	-0.06%
F2	91.11%	-0.11%	93.68%	-0.11%	92.23%	-0.05%	94.18%	-0.12%
F3	91.11%	-0.11%	93.68%	-0.11%	92.23%	-0.05%	94.18%	-0.12%
F4	91.50%	0.28%	94.13%	0.34%	91.72%	-0.56%	93.68%	-0.62%
F5	91.22%	0.00%	93.62%	-0.17%	91.95%	-0.33%	93.96%	-0.34%
F6	91.00%	-0.22%	93.51%	-0.28%	92.23%	-0.05%	94.24%	-0.06%
F7	90.66%	-0.56%	93.18%	-0.61%	91.78%	-0.50%	93.90%	-0.40%
F8	91.22%	0.00%	93.79%	0.00%	92.39%	0.11%	94.24%	-0.06%
F9	91.28%	0.06%	93.62%	-0.17%	92.45%	0.17%	94.07%	-0.23%
F10	91.33%	0.11%	93.85%	0.06%	92.00%	-0.28%	94.07%	-0.23%
F11	91.50%	0.28%	93.74%	-0.05%	92.06%	-0.22%	93.79%	-0.51%
F12	91.28%	0.06%	93.62%	-0.17%	92.56%	0.28%	94.35%	0.05%
F13	91.22%	0.00%	93.79%	0.00%	92.28%	0.00%	94.30%	0.00%
F14	91.16%	-0.06%	93.74%	-0.05%	92.39%	0.11%	94.41%	0.11%
F15	91.22%	0.00%	93.79%	0.00%	92.23%	-0.05%	94.24%	-0.06%
F16	91.39%	0.17%	93.90%	0.11%	92.34%	0.06%	94.30%	0.00%
F17	91.22%	0.00%	93.79%	0.00%	92.23%	-0.05%	94.24%	-0.06%
F18	91.16%	-0.06%	93.74%	-0.05%	92.39%	0.11%	94.46%	0.16%
F19	91.33%	0.11%	93.90%	0.11%	92.28%	0.00%	94.30%	0.00%
F20	91.11%	-0.11%	93.68%	-0.11%	92.34%	0.06%	94.35%	0.05%
F21	91.22%	0.00%	93.79%	0.00%	92.28%	0.00%	94.30%	0.00%
F22	91.16%	-0.06%	93.74%	-0.05%	92.23%	-0.05%	94.24%	-0.06%
F23	91.28%	0.06%	93.79%	0.00%	92.28%	0.00%	94.24%	-0.06%
F24	91.22%	0.00%	93.74%	-0.05%	92.23%	-0.05%	94.24%	-0.06%
F25	89.54%	-1.68%	92.11%	-1.68%	90.04%	-2.24%	92.39%	-1.91%
F26	91.16%	-0.06%	93.74%	-0.05%	92.28%	0.00%	94.30%	0.00%
F27	91.22%	0.00%	93.68%	-0.11%	92.23%	-0.05%	94.18%	-0.12%
F28	90.94%	-0.28%	93.51%	-0.28%	92.11%	-0.17%	94.13%	-0.17%
F29	91.28%	0.06%	93.85%	0.06%	92.28%	0.00%	94.30%	0.00%
F30	91.16%	-0.06%	93.74%	-0.05%	92.23%	-0.05%	94.24%	-0.06%
F31	91.28%	0.06%	93.85%	0.06%	92.28%	0.00%	94.24%	-0.06%
F32	91.22%	0.00%	93.79%	0.00%	92.28%	0.00%	94.30%	0.00%

source particles and uses the most frequent target case particle. Our method involves separating the training data into source particles and using machine learning for each particle. The fact that Baseline 3 obtained a relatively high accuracy rate supports the effectiveness of our method separating the training data into source particles.

6.2 Experiments confirming importance of features

We next conducted experiments where we confirmed which features were effective. The results are listed in Table 6. We can see the accuracy rate for deleting features and the accuracy rate for using all features. We can see that not using F25 greatly decreased the accuracy rate (about 2%). This indicates that F25 is particularly effective. F25 is the transformation rule Kondo used for P and N in his method. The transformation rules in Kondo's method were made precisely for *ni* (indirect object), which is particularly difficult to handle. F25 is thus effective. We could also see not using F7 decreased the accuracy rate (about 0.5%). F7 has the semantic features for N. We found that the semantic features for N were also effective.

6.3 Experiments changing number of training data

We finally did experiments changing the number of training data. The results are plotted in Figure 5. We used our two methods of all features "Division" and "Non-division". We only plotted the

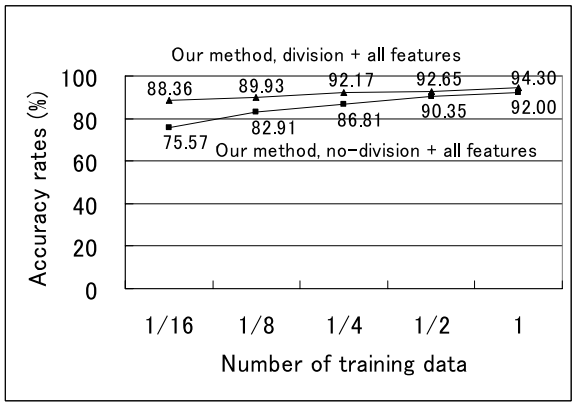


Figure 5: Changing number of training data

accuracy rates for Eval. B in the open data in the figure. We plotted accuracy rates when 1, 1/2, 1/4, 1/8, and 1/16 of the training data were used. "Division", which separates training data for all source particles, obtained a high accuracy rate (88.36%) even when the number of training data was small. In contrast, "Non-division", which does not separate training data for any source particles, obtained a low accuracy rate (75.57%), when the number of training data was small. This indicates that our method of separating training data for all source particles is effective.

7 Conclusion

We developed a new method of transforming Japanese case particles when transforming Japanese passive sentences into active sentences. Our method separates training data for all input (source) particles and uses machine learning for each particle. We also used numerous rich features for learning. Our method obtained a high rate of accuracy (94.30%). In contrast, a method that did not separate training data for all source particles obtained a lower rate of accuracy (92.00%). In addition, a method that did not have many rich features for learning used in a previous study obtained a much lower accuracy rate (89.77%). We confirmed that these improvements were significant through a statistical test. We also undertook experiments utilizing traditional methods using verb dictionaries and manually prepared heuristic rules and confirmed that our method obtained much higher accuracy rates than traditional methods.

We also conducted experiments on which features were the most effective. We found that Kondo's transformation rule used as a feature in our system was particularly effective. We also found that semantic features for nominal targets were effective.

We finally did experiments on changing the number of training data. We found that our method of separating training data for all source particles could obtain high accuracy rates even when there were few training data. This indicates that our method of separating training data for all source particles is effective.

The transformation of passive sentences into active sentences is useful in many research areas including generation, knowledge extraction from databases written in natural languages, information extraction, and answering questions. In the future, we intend to use the results of our study for these kinds of research projects.

References

Nello Cristianini and John Shawe-Taylor. 2000. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.

IPA. 1987. (Information–Technology Promotion Agency, Japan). *IPA Lexicon of the Japanese Language for Computers IPAL (Basic Verbs)*. (in Japanese).

Keiko Kondo, Satoshi Sato, and Manabu Okumura. 2001. Paraphrasing by case alternation. *Transactions of Information Processing Society of Japan*, 42(3):465–477. (in Japanese).

Taku Kudoh and Yuji Matsumoto. 2000. Use of support vector learning for chunk identification. *CoNLL-2000*, pages 142–144.

Taku Kudoh. 2000. TinySVM: Support Vector Machines. http://cl.aist-nara.ac.jp/taku-ku//software/TinySVM/index.html.

Sadao Kurohashi and Makoto Nagao. 1997. Kyoto University text corpus project. *3rd Annual Meeting of the Association for Natural Language Processing*, pages 115–118. (in Japanese).

Sadao Kurohashi, 1998. *Japanese Dependency/Case Structure Analyzer KNP version 2.0b6*. Department of Informatics, Kyoto University. (in Japanese).

Masaki Murata and Hitoshi Isahara, 2003. *Conversion of Japanese Passive/Causative Sentences into Active Sentences Using Machine Learning*, pages 115–125. Springer Publisher.

Masaki Murata, Qing Ma, and Hitoshi Isahara. 2002. Comparison of three machine-learning methods for Thai part-of-speech tagging. *ACM Transactions on Asian Language Information Processing*, 1(2):145–158.

NLRI. 1964. *Bunrui Goi Hyou*. Shuuei Publishing.

Hirotoshi Taira and Masahiko Haruno. 2001. Feature selection in svm text categorization. In *Proceedings of AAAI2001*, pages 480–486.

Reinforcing English Countability Prediction with One Countability per Discourse Property

Ryo Nagata
Hyogo University of Teacher Education
6731494, Japan
rnagata@hyogo-u.ac.jp

Atsuo Kawai
Mie University
5148507, Japan
kawai@ai.info.mie-u.ac.jp

Koichiro Morihiro
Hyogo University of Teacher Education
6731494, Japan
mori@hyogo-u.ac.jp

Naoki Isu
Mie University
5148507, Japan
isu@ai.info.mie-u.ac.jp

Abstract

Countability of English nouns is important in various natural language processing tasks. It especially plays an important role in machine translation since it determines the range of possible determiners. This paper proposes a method for reinforcing countability prediction by introducing a novel concept called *one countability per discourse*. It claims that when a noun appears more than once in a discourse, they will all share the same countability in the discourse. The basic idea of the proposed method is that mispredictions can be correctly overridden using efficiently the one countability per discourse property. Experiments show that the proposed method successfully reinforces countability prediction and outperforms other methods used for comparison.

1 Introduction

Countability of English nouns is important in various natural language processing tasks. It is particularly important in machine translation from a source language that does not have an article system similar to that of English, such as Chinese and Japanese, into English since it determines the range of possible determiners including articles. It also plays an important role in determining whether a noun can take singular and plural forms. Another useful application is to detect errors in article usage and singular/plural usage in the writing of second language learners. Given countability, these errors can be detected in many cases. For example, an error can be detected from "We have a *furniture*." given that the noun *furniture* is uncountable since uncountable nouns do not tolerate the indefinite article.

Because of the wide range of applications, researchers have done a lot of work related to countability. Baldwin and Bond (2003a; 2003b) have proposed a method for automatically learning countability from corpus data. Lapata and Keller (2005) and Peng and Araki (2005) have proposed web-based models for learning countability. Others including Bond and Vatikiotis-Bateson (2002) and O'Hara et al. (2003) use ontology to determine countability.

In the application to error detection, researchers have explored alternative approaches since sources of evidence for determining countability are limited compared to other applications. Articles and the singular/plural distinction, which are informative for countability, cannot be used in countability prediction aiming at detecting errors in article usage and singular/plural usage. Returning to the previous example, the countability of the noun *furniture* cannot be determined as uncountable by the indefinite article; first, its countability has to be predicted without the indefinite article, and only then whether or not it tolerates the indefinite article is examined using the predicted countability. Also, unlike in machine translation, the source language is not given in the writing of second language learners such as essays, which means that information available is limited.

To overcome these limitations, Nagata et al. (2005a) have proposed a method for predicting countability that relies solely on words (except articles and other determiners) surrounding the target noun. Nagata et al. (2005b) have shown that the method is effective to detecting errors in article usage and singular/plural usage in the writing of Japanese learners of English. They

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 595–602,
Sydney, July 2006.

also have shown that it is likely that performance of the error detection will improve as accuracy of the countability prediction increases since most of false positives are due to mispredictions.

In this paper, we propose a method for reinforcing countability prediction by introducing a novel concept called *one countability per discourse* that is an extension of *one sense per discourse* proposed by Gale et al. (1992). It claims that when a noun appears more than once in a discourse, they will all share the same countability in the discourse. The basic idea of the proposed method is that initially mispredicted countability can be corrected using efficiently the one countability per discourse property.

The next section introduces the one countability per discourse concept and shows that it can be a good source of evidence for predicting countability. Section 3 discusses how it can be efficiently exploited to predict countability. Section 4 describes the proposed method. Section 5 describes experiments conducted to evaluate the proposed method and discusses the results.

2 One Countability per Discourse

One countability per discourse is an extension of *one sense per discourse* proposed by Gale et al. (1992). One sense per discourse claims that when a polysemous word appears more than once in a discourse it is likely that they will all share the same sense. Yarowsky (1995) tested the claim on about 37,000 examples and found that when a polysemous word appeared more than once in a discourse, they took on the majority sense for the discourse 99.8% of the time on average.

Based on one sense per discourse, we hypothesize that when a noun appears more than once in a discourse, they will all share the same countability in the discourse, that is, one countability per discourse. The motivation for this hypothesis is that if one sense per discourse is satisfied, so is one countability per discourse because countability is often determined by word sense. For example, if the noun *paper* appears in a discourse and it has the sense of newspaper, which is countable, the rest of *paper*s in the discourse also have the same sense according to one sense per discourse, and thus they are also countable.

We tested this hypothesis on a set of nouns[1] as Yarowsky (1995) did. We calculated how accurately the majority countability for each discourse predicted countability of the nouns in the discourse when they appeared more than once. If the one countability per discourse property is always satisfied, the majority countability for each discourse should predict countability with the accuracy of 100%. In other others, the obtained accuracy represents how often the one countability per discourse property is satisfied.

Table 1 shows the results. "MCD" in Table 1 stands for Majority Countability for Discourse and its corresponding column denotes accuracy where countability of individual nouns was predicted by the majority countability for the discourse in which they appeared. Also, "Baseline" denotes accuracy where it was predicted by the majority countability for the whole corpus used in this test.

Table 1: Accuracy obtained by Majority Countability for Discourse

Target noun	MCD	Baseline
advantage	0.772	0.618
aid	0.943	0.671
authority	0.864	0.771
building	0.850	0.811
cover	0.926	0.537
detail	0.829	0.763
discipline	0.877	0.652
duty	0.839	0.714
football	0.938	0.930
gold	0.929	0.929
hair	0.914	0.902
improvement	0.735	0.685
necessity	0.769	0.590
paper	0.807	0.647
reason	0.858	0.822
sausage	0.821	0.750
sleep	0.901	0.765
stomach	0.778	0.778
study	0.824	0.781
truth	0.783	0.724
use	0.877	0.871
work	0.861	0.777
worry	0.871	0.843
Average	0.851	0.754

Table 1 reveals that the one countability per dis-

[1]The conditions of this test are shown in Section 5. Note that although the source of the data is the same as in Section 5, discourses in which the target noun appears only once are excluded from this test unlike in Section 5.

course property is a good source of evidence for predicting countability compared to the baseline while it is not as strong as the one sense per discourse property is. It also reveals that the tendency of one countability per discourse varies from noun to noun. For instance, nouns such as *aid* and *cover* show a strong tendency while others such as *advantage* and *improvement* do not. On average, "MCD" achieves an improvement of approximately 10% in accuracy over the baseline.

Having observed the results, it is reasonable to exploit the one countability per discourse property for predicting countability. In order to do it, however, the following two questions should be addressed. First, how can the majority countability be obtained from a novel discourse? Since our intention is to predict values of countability of instances in a novel discourse, none of them are known. Second, even if the majority countability is known, how can it be efficiently exploited for predicting countability? Although we could simply predict countability of individual instances of a target noun in a discourse by the majority countability for the discourse, it is highly possible that this simple method will cause side effects considering the results in Table 1. These two questions are addressed in the next section.

3 Basic Idea

3.1 How Can the Majority Countability be Obtained from a Novel Discourse?

Although we do not know the true value of the majority countability for a novel discourse, we can at least estimate it because we have a method for predicting countability to be reinforced by the proposed method. That is, we can predict countability of the target noun in a novel discourse using the method. Simply counting the results would give the majority countability for it.

Here, we should note that countability of each instance is not the true value but a predicted one. Considering this fact, it is sensible to set a certain criterion in order to filter out spurious predictions. Fortunately, most methods based on machine learning algorithms give predictions with their confidences. We use the confidences as the criterion. Namely, we only take account of predictions whose confidences are greater than a certain threshold when we estimate the majority countability for a novel discourse.

3.2 How Can the Majority Countability be Efficiently Exploited?

In order to efficiently exploit the one countability per discourse property, we treat the majority countability for each discourse as a feature in addition to other features extracted from instances of the target noun. Doing so, we let a machine learning algorithm decide which features are relevant to the prediction. If the majority countability feature is relevant, the machine learning algorithm should give a high weight to it compared to others.

To see this, let us suppose that we have a set of discourses in which instances of the target noun are tagged with their countability (either *countable* or *uncountable*[2]) for the moment; we will describe how to obtain it in Subsection 4.1. For each discourse, we can know its majority countability by counting the numbers of *countable*s and *uncountable*s. We can also generate a model for predicting countability from the set of discourses using a machine learning algorithm. All we have to do is to extract a set of training data from the tagged instances and to apply a machine learning algorithm to it. This is where the majority countability feature comes in. The majority countability for each instance is added to its corresponding training data as a feature to create a new set of training data before applying a machine learning algorithm; then a machine learning algorithm is applied to the new set. The resulting model takes the majority countability feature into account as well as the other features when making predictions.

It is important to exercise some care in counting the majority countability for each discourse. Note that one countability per discourse is always satisfied in discourses where the target noun appears only once. This suggests that it is highly possible that the resulting model too strongly favors the majority countability feature. To avoid this, we could split the discourses into two sets, one for where the target noun appears only once and one for where it appears more than once, and train a model on each set. However, we do not take this strategy because we want to use as much data as possible for training. As a compromise, we approximate the majority countability for discourses where the target noun appears only once to the value *unknown*.

[2]This paper concentrates solely on countable and uncountable nouns, since they account for the vast majority of nouns (Lapata and Keller, 2005).

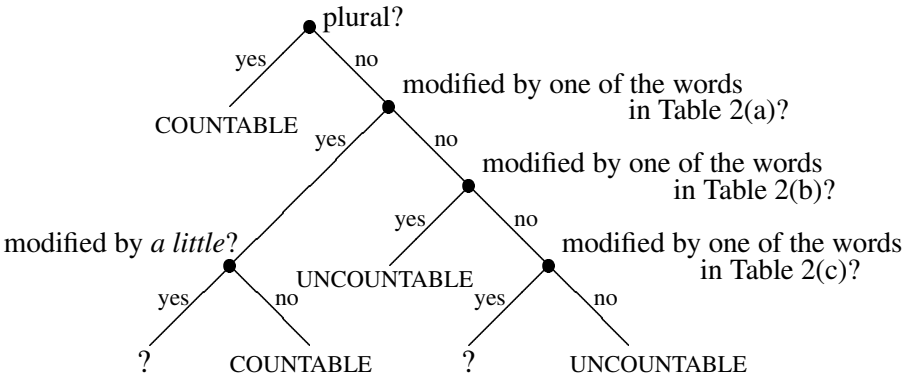


Figure 1: Framework of the tagging rules

Table 2: Words used in the tagging rules

(a)	(b)	(c)
the indefinite article	much	*the definite article*
another	less	*demonstrative adjectives*
one	enough	*possessive adjectives*
each	sufficient	*interrogative adjectives*
—	—	*quantifiers*
—	—	*'s genitives*

4 Proposed Method

4.1 Generating Training Data

As discussed in Subsection 3.2, training data are needed to exploit the one countability per discourse property. In other words, the proposed method requires a set of discourses in which instances of the target noun are tagged with their countability. Fortunately, Nagata et al. (2005b) have proposed a method for tagging nouns with their countability. This paper follows it to generate training data.

To generate training data, first, instances of the target noun used as a head noun are collected from a corpus with their surrounding words. This can be simply done by an existing chunker or parser.

Second, the collected instances are tagged with either *countable* or *uncountable* by tagging rules. For example, the underlined *paper*:

... read a *paper* in the morning ...

is tagged as

... read a *paper/countable* in the morning ...

because it is modified by the indefinite article.

Figure 1 and Table 2 represent the tagging rules based on Nagata et al. (2005b)'s method. Figure 1 shows the framework of the tagging rules. Each node in Figure 1 represents a question applied to the instance in question. For instance, the root node reads "Is the instance in question plural?". Each leaf represents a result of the classification. For instance, if the answer is "yes" at the root node, the instance in question is tagged with *countable*. Otherwise, the question at the lower node is applied and so on. The tagging rules do not classify instances in some cases. These unclassified instances are tagged with the symbol "?". Unfortunately, they cannot readily be included in training data. For simplicity of implementation, they are excluded from training data (we will discuss the use of these excluded data in Section 6).

Note that the tagging rules cannot be used for countability prediction aiming at detecting errors in article usage and singular/plural usage. The reason is that they are useless in error detection where whether determiners and the singular/plural distinction are correct or not is unknown. Obviously, the tagging rules assume that the target text contains no error.

Third, features are extracted from each instance. As the features, the following three types of contextual cues are used: (i) words in the noun phrase that the instance heads, (ii) three words to the left of the noun phrase, and (iii) three words to its right. Here, the words in Table 2 are excluded. Also, function words (except prepositions) such as pronouns, cardinal and quasi-cardinal numer-

als, and the target noun are excluded. All words are reduced to their morphological stem and converted entirely to lower case when collected. In addition to the features, the majority countability is used as a feature. For each discourse, the numbers of *countable*s and *uncountable*s are counted to obtain its majority countability. In case of ties, it is set to *unknown*. Also, it is set to *unknown* when only one instance appears in the discourse as explained in Subsection 3.2.

To illustrate feature extraction, let us consider the following discourse (target noun: *paper*):

... writing a new *paper/countable* in his room ...
... read *papers/countable* with ...

The discourse would give a set of features:

-3=write, NP=new, +3=in, +3=room, MC=c
-3=read, +3=with, MC=c

where "MC=c" denotes that the majority countability for the discourse is *countable*. In this example (and in the following examples), the features are represented in a somewhat simplified manner for the purpose of illustration. In practice, features are represented as a vector.

Finally, the features are stored in a file with their corresponding countability as training data. Each piece of training data would be as follows:

-3=read, +3=with, MC=c, LABEL=c

where "LABEL=c" denotes that the countability for the instance is *countable*.

4.2 Model Generation

The model used in the proposed method can be regarded as a function. It takes as its input a feature vector extracted from the instance in question and predicts countability (either *countable* or *uncountable*). Formally, $f : \boldsymbol{v} \rightarrow c$ where f, $\boldsymbol{v}$, and c denote the model, the feature vector, and $c = 0, 1$, respectively; here, 0 and 1 correspond to *countable* and *uncountable*, respectively.

Given the specification, almost any kind of machine learning algorithm cab be used to generate the model used in the proposed method. In this paper, the Maximum Entropy (ME) algorithm is used which has been shown to be effective in a wide variety of natural language processing tasks.

Model generation is done by applying the ME algorithm to the training data. The resulting model takes account of the features including the majority countability feature and is used for reinforcing countability prediction.

4.3 Reinforcing Countability Prediction

Before explaining the reinforcement procedure, let us introduce the following discourse for illustration (target noun: *paper*):

... writing *paper* in room ... wrote *paper* in ...
... submitted *paper* to ...

Note that articles and the singular/plural distinction are deliberately removed from the discourse. This kind of situation can happen in machine translation from a source language that does not have articles and the singular/plural distinction[3]. The situation is similar in the writing of second language learners of English since they often omit articles and the singular/plural distinction or use improper ones. Here, suppose that the true values of the countability for all instances are *countable*.

A method to be reinforced by the proposed method would predict countability as follows:

... writing *paper/countable* (0.97) in room ...
... wrote *paper/countable* (0.98) in ...
... submitted *paper/uncountable* (0.57) to ...

where the numbers in brackets denote the confidences given by the method. The third instance is mistakenly predicted as *uncountable*[4].

Now let us move on to the reinforcement procedure. It is divided into three steps. First, the majority countability for the discourse in question is estimated by counting the numbers of the predicted *countable*s and *uncountable*s whose confidences are greater than a certain threshold. In case of ties, the values of the majority countability is set to *unknown*. In the above example, the majority countability for the discourse is estimated to be *countable* when the threshold is set to 0.95 (two *countable*s). Second, features explained in Subsection 4.1 are extracted from each instance. As for the majority countability feature, the estimated one is used. Returning to the above example, the three instances would give a set of features:

-3=write, +3=in, +3=room, MC=c,
-3=write, +3=in, MC=c,
-3=submit, +3=to, MC=c.

Finally, the model generated in Subsection 4.2 is applied to the features to predict countability. Because of the majority countability feature, it

[3]For instance, the Japanese language does not have an article system similar to that of English, neither does it mark the singular/plural distinction.

[4]The reason would be that the contextual cues did not appear in the training data used in the method.

is likely that previous mispredictions are overridden by correct ones. In the above example, the third one would be correctly overridden by *countable* because of the majority countability feature (MC=c) that is informative for the instance being *countable*.

5 Experiments

5.1 Experimental Conditions

In the experiments, we chose Nagata et al. (2005a)'s method as the one to be reinforced by the proposed method. In this method, the decision list (DL) learning algorithm (Yarowsky, 1995) is used. However, we used the ME algorithm because we found that the method with the ME algorithm instead of the DL learning algorithm performed better when trained on the same training data.

As the target noun, we selected 23 nouns that were also used in Nagata et al. (2005a)'s experiments. They are exemplified as nouns that are used as both countable and uncountable by Huddleston and Pullum (2002).

Training data were generated from the written part of the British National Corpus (Burnard, 1995). A text tagged with the text tags was used as a discourse unit. From the corpus, 314 texts, which amounted to about 10% of all texts, were randomly taken to obtain test data. The rest of texts were used to generate training data.

We evaluated performance of prediction by accuracy. We defined accuracy by the ratio of the number of correct predictions to that of instances of the target noun in the test data.

5.2 Experimental Procedures

First, we generated training data for each target noun from the texts using the tagging rules explained in Subsection 4.1. We used the OAK system[5] to extract noun phrases and their heads. Of the extracted instances, we excluded those that had no contextual cues from the training data (and also the test data). We also generated another set of training data by removing the majority countability features from them. This set of training data was used for comparison.

Second, we obtained test data by applying the tagging rules described in Subsection 4.1 to each instance of the target noun in the 314 texts. Nagata et al. (2005b) showed that the tagging rules achieved an accuracy of 0.997 in the texts that contained no errors. Considering these results, we used the tagging rules to obtain test data. Instances tagged with "?" were excluded in the experiments.

Third, we applied the ME algorithm[6] to the training data without the majority countability feature. Using the resulting model, countability of the target nouns in the test data was predicted. Then, the predictions were reinforced by the proposed method. The threshold to filter out spurious predictions was set to 0.95. For comparison, the predictions obtained by the ME model were simply replaced with the estimated majority countability for each discourse. In this method, the original predictions were used when the estimated majority countability was *unknown*. Also, Nagata et al. (2005a)'s method that was based on the DL learning algorithm was implemented for comparison.

Finally, we calculated accuracy of each method. In addition to the results, we evaluated the baseline on the same test data where all predictions were done by the majority countability for the whole corpus (training data).

5.3 Experimental Results and Discussion

Table 3 shows the accuracies[7]. "ME" and "Proposed" in Table 3 refer to accuracies of the ME model and the ME model reinforced by the proposed method, respectively. "ME+MCD" refers to accuracy obtained by replacing predictions of the ME model with the estimated majority countability for each discourse. Also, "DL" refers to accuracy of the DL-based method.

Table 3 shows that the three ME-based methods ("Proposed", "ME", and "ME+MCD") perform better than "DL" and the baseline. Especially, "Proposed" outperforms the other methods in most of the target nouns.

Figure 2 summarizes the comparison between the three ME-based methods. Each plot in Figure 2 represents each target noun. The horizontal and vertical axises correspond to accuracy of "ME" and that of "Proposed" (or "ME+MCD"), respectively. The diagonal line corresponds to the line $y = x$. So if "Proposed" (or "ME+MCD") achieved no improvement at all over "ME", all the

[5] http://www.cs.nyu.edu/ sekine/PROJECT/OAK/

[6] All ME models were generated using the opennlp.maxent package (http://maxent.sourceforge.net/).

[7] The baseline in Table 3 is different from that in Table 1 because discourses where the target noun appears only once are not taken into account in Table 1.

Table 3: Experimental results

Target noun	Freq.	Baseline	Proposed	ME	ME+MCD	DL
advantage	570	0.604	0.933	0.921	0.811	0.882
aid	385	0.665	0.909	0.873	0.896	0.722
authority	1162	0.760	0.857	0.851	0.840	0.804
building	1114	0.803	0.848	0.842	0.829	0.807
cover	210	0.567	0.790	0.757	0.800	0.714
detail	1157	0.760	0.906	0.904	0.821	0.869
discipline	204	0.593	0.804	0.745	0.750	0.696
duty	570	0.700	0.879	0.877	0.828	0.847
football	281	0.907	0.925	0.907	0.925	0.911
gold	140	0.929	0.929	0.929	0.921	0.929
hair	448	0.902	0.908	0.902	0.904	0.904
improvement	362	0.696	0.735	0.715	0.685	0.738
necessity	83	0.566	0.831	0.843	0.831	0.783
paper	1266	0.642	0.859	0.836	0.808	0.839
reason	1163	0.824	0.885	0.893	0.834	0.843
sausage	45	0.778	0.778	0.733	0.756	0.778
sleep	107	0.776	0.925	0.897	0.897	0.813
stomach	30	0.633	0.800	0.800	0.800	0.733
study	1162	0.779	0.832	0.819	0.782	0.808
truth	264	0.720	0.761	0.777	0.765	0.731
use	1390	0.869	0.879	0.863	0.871	0.873
work	3002	0.778	0.858	0.842	0.837	0.806
worry	119	0.798	0.874	0.840	0.849	0.849
Average	662	0.741	0.857	0.842	0.828	0.812

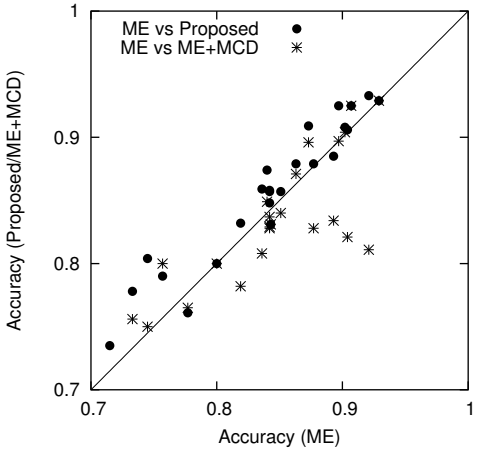


Figure 2: Comparison between "ME" and "Proposed/ME+MCD" in each target noun

plots would be on the line. Plots above the line mean improvement over "ME" and the distance from the line expresses the amount of improvement. Plots below the line mean the opposite.

Figure 2 clearly shows that most of the plots (•) corresponding to the comparison between "ME" and "Proposed" are above the line. This means that the proposed method successfully reinforced "ME" in most of the target nouns. Indeed, the average accuracy of "Proposed" is significantly superior to that of "ME" at the 99% confidence level (paired *t-test*). This improvement is close to that of one sense per discourse (Yarowsky, 1995) (improvement ranging from 1.3% to 1.7%), which seems to be a sensible upper bound of the proposed method. By contrast, about half of the plots (∗) corresponding to the comparison between "ME" and "ME+MCD" are below the line.

From these results, it follows that the one countability per discourse property is a good source of evidence for predicting countability, but it is crucial to devise a way of exploiting the property as we did in this paper. Namely, simply replacing original predictions with the majority countability for the discourse causes side effects, which has been already suggested in Table 1. This is

also exemplified as follows. Suppose that several instances of the target noun *advantage* appear in a discourse and that its majority countably is *countable*. Further suppose that an idiomatic phrase "take *advantage* of" of which countability is *uncountable* happens to appear in it. On one hand, simply replacing all the predictions with its majority countability (*countable*) would lead to a misprediction for the idiomatic phrase even if the original prediction is correct. On the other hand, the proposed method would correctly predict the countability because the contextual cues strongly indicate that it is *uncountable*.

6 Conclusions

This paper has proposed a method for reinforcing English countability prediction by introducing one countability per discourse. The experiments have shown that the proposed method successfully overrode original mispredictions using efficiently the one countability per discourse property. They also have shown that it outperformed other methods used for comparison. From these results, we conclude that the proposed method is effective in reinforcing English countability prediction.

In addition, the proposed method has two advantages. The first is its applicability. It can reinforce almost any earlier method. Even to hand-coded rules, it can be applied as long as they give predictions with their confidences. This further gives an additional advantage. Recall that the instances tagged with "?" by the tagging rules are discarded when training data are generated as described in Subsection 4.1. These instances can be retagged with their countability by using the proposed method and some kind of bootstrapping (Yarowsky, 1995). This means increase in training data, which might eventually result in further improvement. The second is that the proposed method is unsupervised. It requires no human intervention to reinforce countability prediction.

For future work, we will investigate what models are most appropriate for exploiting the one countability per discourse property. We will also explore a method for including instances tagged with "?" in training data by using the proposed method and bootstrapping.

Acknowledgments

The authors would like to thank Satoshi Sekine who has developed the OAK System. The authors also would like to thank three anonymous reviewers for their useful comments on this paper.

References

T. Baldwin and F. Bond. 2003a. Learning the countability of English nouns from corpus data. In *Proc. of 41st Annual Meeting of ACL*, pages 463–470.

T. Baldwin and F. Bond. 2003b. A plethora of methods for learning English countability. In *Proc. of 2003 Conference on Empirical Methods in Natural Language Processing*, pages 73–80.

F. Bond and C. Vatikiotis-Bateson. 2002. Using an ontology to determine English countability. In *Proc. of 19th International Conference on Computational Linguistics*, pages 99–105.

L. Burnard. 1995. *Users Reference Guide for the British National Corpus. version 1.0.* Oxford University Computing Services, Oxford.

W.A. Gale, K.W. Church, and D. Yarowsky. 1992. One sense per discourse. In *Proc. of 4th DARPA Speech and Natural Language Workshop*, pages 233–237.

R. Huddleston and G.K. Pullum. 2002. *The Cambridge Grammar of the English Language*. Cambridge University Press, Cambridge.

M. Lapata and F. Keller. 2005. Web-based models for natural language processing. *ACM Transactions on Speech and Language Processing*, 2(1):1–31.

R. Nagata, F. Masui, A. Kawai, and N. Isu. 2005a. An unsupervised method for distinguishing mass and count noun in context. In *Proc. of 6th International Workshop on Computational Semantics*, pages 213–224.

R. Nagata, T. Wakana, F. Masui, A. Kawai, and N. Isu. 2005b. Detecting article errors based on the mass count distinction. In *Proc. of 2nd International Joint Conference on Natural Language Processing*, pages 815–826.

T. O'Hara, N. Salay, M. Witbrock, D. Schneider, B. Aldag, S. Bertolo, K. Panton, F. Lehmann, J. Curtis, M. Smith, D. Baxter, and P. Wagner. 2003. Inducing criteria for mass noun lexical mappings using the Cyc KB, and its extension to WordNet. In *Proc. of 5th International Workshop on Computational Semantics*, pages 425–441.

J. Peng and K. Araki. 2005. Detecting the countability of English compound nouns using web-based models. In *Companion Volume to Proc. of 2nd International Joint Conference on Natural Language Processing*, pages 105–109.

D. Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proc. of 33rd Annual Meeting of ACL*, pages 189–196.

An Automatic Method for Summary Evaluation Using Multiple Evaluation Results by a Manual Method

Hidetsugu Nanba
Faculty of Information Sciences,
Hiroshima City University
3-4-1 Ozuka, Hiroshima, 731-3194 Japan
nanba@its.hiroshima-cu.ac.jp

Manabu Okumura
Precision and Intelligence Laboratory,
Tokyo Institute of Technology
4259 Nagatsuta, Yokohama, 226-8503 Japan
oku@pi.titech.ac.jp

Abstract

To solve a problem of how to evaluate computer-produced summaries, a number of automatic and manual methods have been proposed. Manual methods evaluate summaries correctly, because humans evaluate them, but are costly. On the other hand, automatic methods, which use evaluation tools or programs, are low cost, although these methods cannot evaluate summaries as accurately as manual methods. In this paper, we investigate an automatic evaluation method that can reduce the errors of traditional automatic methods by using several evaluation results obtained manually. We conducted some experiments using the data of the Text Summarization Challenge 2 (TSC-2). A comparison with conventional automatic methods shows that our method outperforms other methods usually used.

1 Introduction

Recently, the evaluation of computer-produced summaries has become recognized as one of the problem areas that must be addressed in the field of automatic summarization. To solve this problem, a number of automatic (Donaway et al., 2000, Hirao et al., 2005, Lin et al., 2003, Lin, 2004, Hori et al., 2003) and manual methods (Nenkova et al., 2004, Teufel et al., 2004) have been proposed. Manual methods evaluate summaries correctly, because humans evaluate them, but are costly. On the other hand, automatic methods, which use evaluation tools or programs, are low cost, although these methods cannot evaluate summaries as accurately as manual methods. In this paper, we investigate an automatic method that can reduce the errors of traditional automatic methods by using several evaluation results obtained manually. Unlike other automatic methods, our method estimates manual evaluation scores. Therefore, our method makes it possible to compare a new system with other systems that have been evaluated manually.

There are two research studies related to our work (Kazawa et al., 2003, Yasuda et al., 2003). Kazawa et al. (2003) proposed an automatic evaluation method using multiple evaluation results from a manual method. In the field of machine translation, Yasuda et al. (2003) proposed an automatic method that gives an evaluation result of a translation system as a score for the Test of English for International Communication (TOEIC). Although the effectiveness of both methods was confirmed experimentally, further discussion of four points, which we describe in Section 3, is necessary for a more accurate summary evaluation. In this paper, we address three of these points based on Kazawa's and Yasuda's methods. We also investigate whether these methods can outperform other automatic methods.

The remainder of this paper is organized as follows. Section 2 describes related work. Section 3 describes our method. To investigate the effectiveness of our method, we conducted some examinations and Section 4 reports on these. We present some conclusions in Section 5.

2 Related Work

Generally, similar summaries are considered to obtain similar evaluation results. If there is a set of summaries (pooled summaries) produced from a document (or multiple documents) and if these are evaluated manually, then we can estimate a manual evaluation score for any summary to be evaluated with the evaluation results for those pooled summaries. Based on this idea, Kazawa et

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 603–610,
Sydney, July 2006.

al. (2003) proposed an automatic method using multiple evaluation results from a manual method. First, n summaries for each document, m, were prepared. A summarization system generated summaries from m documents. Here, we represent the i^{th} summary for the j^{th} document and its evaluation score as x_{ij} and y_{ij}, respectively. The system was evaluated using Equation 1.

$$scr(x) = \sum_{i=1}^{m} \sum_{j=1}^{n} w_j y_{ij} Sim(x, x_{ij}) + b \quad (1)$$

The evaluation score of summary x was obtained by summing parameter b for all the subscores calculated for each pooled summary, x_{ij}. A subscore was obtained by multiplying a parameter w_j, by the evaluation score y_{ij}, and the similarity between x and x_{ij}.

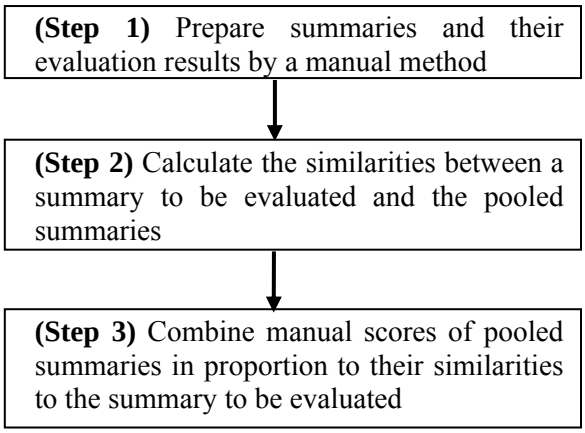
In the field of machine translation, there is another related study. Yasuda et al. (2003) proposed an automatic method that gives an evaluation result of a translation system as a score for TOEIC. They prepared 29 human subjects, whose TOEIC scores were from 300s to 800s, and asked them to translate 23 Japanese conversations into English. They also generated translations using a system for each conversation. Then, they evaluated both translations using an automatic method, and obtained W_H, which indicated the ratio of system translations that were superior to human translations. Yasuda et al. calculated W_H for each subject and plotted the values along with their corresponding TOEIC scores to produce a regression line. Finally, they defined a point where the regression line crossed W_H = 0.5 to provide the TOEIC score for the system.

Though, the effectiveness of Kazawa's and Yasuda's methods were confirmed experimentally, further discussions of four points, which we describe in the next section, are necessary for a more accurate summary evaluation.

3 Investigation of an Automatic Method using Multiple Manual Evaluation Results

3.1 Overview of Our Evaluation Method and Essential Points to be Discussed

We investigate an automatic method using multiple evaluation results by a manual method based on Kazawa's and Yasuda's method. The procedure of our evaluation method is shown as follows;



For each step, we need to discuss the following points.

(Step 1)

1. How many summaries, and what type (variety) of summaries should be prepared? Kazawa et al. prepared 6 summaries for each document, and Yasuda et al. prepared 29 translations for each conversation. However, they did not examine about the number and the type of pooled summaries required to the evaluation.

(Step 2)

2. Which measure is better for calculating the similarities between a summary to be evaluated and the pooled summaries? Kazawa et al. used Equation 2 to calculate similarities.

$$Sim(x, x_{ij}) = \frac{|\, x_{ij} \cap x \,|}{\min(|\, x_{ij} \,|, |\, x \,|)} \quad (2)$$

where $x_{ij} \cap x$ indicates the number of discourse units[1] that appear in both x_{ij} and x, and $|\, x \,|$ represents the number of words in x. However, there are many other measures that can be used to calculate the topical similarities between two documents (or passages).

As well as Yasuda's method does, using W_H is another way to calculate similarities between a summary to be evaluated and pooled summaries indirectly. Yasuda et al. (2003) tested DP matching (Su et al., 1992), BLEU (Papineni et al., 2002), and NIST[2], for the calculation of W_H. However there are many other measures for summary evaluation.

[1] Rhetorical Structure Theory Discourse Treebank. www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2002T07 Linguistic Data Consortium.

[2] http://www.nist.gov/speech/tests/mt/mt2001/resource/

3. How many summaries should be used to calculate the score of a summary to be evaluated? Kazawa et al. used all the pooled summaries but this does not ensure the best performance of their evaluation method.

(Step 3)

4. How to combine the manual scores of the pooled summaries? Kazawa et al. calculated the score of a summary as a weighted linear sum of the manual scores. Applying regression analysis (Yasuda et al., 2003) is another method of combining several manual scores.

3.2 Three Points Addressed in Our Study

We address the second, third and fourth points in Section 3.1.

(Point 2) A measure for calculating similarities between a summary to be evaluated and pooled summaries:
There are many measures that can calculate the topical similarities between two documents (or passages). We tested several measures, such as ROUGE (Lin, 2004) and the cosine distance. We describe these measures in detail in Section 4.2.

(Point 3) The number of summaries used to calculate the score of a summary to be evaluated:
We use summaries whose similarities to a summary to be evaluated are higher than a threshold value.

(Point 4) Combination of manual scores:
We used both Kazawa's and Yasuda's methods.

4 Experiments

4.1 Experimental Methods

To investigate the three points described in Section 3.2, we conducted the following four experiments.

- **Exp-1:** We examined Points 2 and 3 based on Kazawa's method. We tested threshold values from 0 to 1 at 0.005 intervals. We also tested several similarity measures, such as cosine distance and 11 kinds of ROUGE.
- **Exp-2:** In order to investigate whether the evaluation based on Kazawa's method can outperform other automatic methods, we compared the evaluation with other automatic methods. In this experiment, we used the similarity measure, which obtain the best performance in Exp-1.
- **Exp-3:** We also examined Point 2 based on Yasuda's method. As a similarity measure, we tested cosine distance and 11 kinds of ROUGE. Then, we examined Point 4 by comparing the result of Yasuda's method with that of Kazawa's.
- **Exp-4:** In the same way as Exp-2, we compared the evaluation with other automatic methods, which we describe in the next section, to investigate whether the evaluation based on Yasuda's method can outperform other automatic methods.

4.2 Automatic Evaluation Methods Used in the Experiments

In the following, we show the automatic evaluation methods used in our experiments.

Content-based evaluation (Donaway et al., 2000)
This measure evaluates summaries by comparing their content words with those of the human-produced extracts. The score of the content-based measure is obtained by computing the similarity between the term vector using tf*idf weighting of a computer-produced summary and the term vector of a human-produced summary by cosine distance.

ROUGE-N (Lin, 2004)
This measure compares n-grams of two summaries, and counts the number of matches. The measure is defined by Equation 3.

$$ROUGE-N = \frac{\sum_{S \in R} \sum_{gram_N \in S} Count_{match}(gram_N)}{\sum_{S \in R} \sum_{gram_N \in S} Count(gram_N)} \quad (3)$$

where $\text{Count}(\text{gram}_N)$ is the number of an N-gram and $\text{Count}_{\text{match}}(\text{gram}_N)$ denotes the number of n-gram co-occurrences in two summaries.

ROUGE-L (Lin, 2004)
This measure evaluates summaries by longest common subsequence (LCS) defined by Equation 4.

$$ROUGE-L = \frac{\sum_{i=i}^{u} LCS_{\cup}(r_i, C)}{m} \quad (4)$$

where $LCS_U(r_i,C)$ is the LCS score of the union's longest common subsequence between reference sentences r_i and the summary to be evaluated, and m is the number of words contained in a reference summary.

ROUGE-S (Lin, 2004)
Skip-bigram is any pair of words in their sentence order, allowing for arbitrary gaps. ROUGE-S measures the overlap of skip-bigrams in a candidate summary and a reference summary. Several variations of ROUGE-S are possible by limiting the maximum skip distance between the two in-order words that are allowed to form a skip-bigram. In the following, ROUGE-SN denotes ROUGE-S with maximum skip distance N.

ROUGE-SU (Lin, 2004)
This measure is an extension of ROUGE-S; it adds a unigram as a counting unit. In the following, ROUGE-SUN denotes ROUGE-SU with maximum skip distance N.

4.3 Evaluation Methods

In the following, we elaborate on the evaluation methods for each experiment.

Exp-1: An experiment for Points 2 and 3 based on Kazawa's method
We evaluated Kazawa's method from the viewpoint of "Gap". Differing from other automatic methods, the method uses multiple manual evaluation results and estimates the manual scores of the summaries to be evaluated or the summarization systems. We therefore evaluated the automatic methods using Gap, which manually indicates the difference between the scores from a manual method and each automatic method that estimates the scores. First, an arbitrary summary is selected from the 10 summaries in a dataset, which we describe in Section 4.4, and an evaluation score is calculated by Kazawa's method using the other nine summaries. The score is compared with a manual score of the summary by Gap, which is defined by Equation 5.

$$Gap = \frac{\sum_{k=1}^{m}\sum_{l=1}^{n} | scr'(x_{kl}) - y_{kl} |}{m \times n} \quad (5)$$

where x_{kl} is the k^{th} system's l^{th} summary, and y_{kl} is the score from a manual evaluation method for the k^{th} system's l^{th} summary. To distinguish our evaluation function from Kazawa's, we denote it as $scr'(x)$. As a similarity measure in $scr'(x)$, we tested ROUGE and the cosine distance.

We also tested the coverage of the automatic method. The method cannot calculate scores if there are no similar summaries above a given threshold value. Therefore, we checked the coverage of the method, which is defined by Equation 6.

$$Coverage = \frac{\textit{The number of summaries evaluated by the method}}{\textit{The number of given summaries}} \quad (6)$$

Exp-2: Comparison of Kazawa's method with other automatic methods
Traditionally, automatic methods have been evaluated by "Ranking". This means that summarization systems are ranked based on the results of the automatic and manual methods. Then, the effectiveness of the automatic method is evaluated by the number of matches between both rankings using Spearman's rank correlation coefficient and Pearson's rank correlation coefficient (Lin et al., 2003, Lin, 2004, Hirao et al., 2005). However, we did not use both correlation coefficients, because evaluation scores are not always calculated by a Kazawa-based method, which we described in Exp-1. Therefore, we ranked the summaries instead of the summarization systems. Two arbitrary summaries from the 10 summaries in a dataset were selected and ranked by Kazawa's method. Then, Kazawa's method was evaluated using "Precision," which calculates the percentage of cases where the order of the manual method of the two summaries matches the order of their ranks calculated by Kazawa's method. The two summaries were also ranked by ROUGE and by cosine distance, and both Precision values were calculated. Finally, the Precision value of Kazawa's method was compared with those of ROUGE and cosine distance.

Exp-3: An experiment for Point 2 based on Yasuda's method
An arbitrary system was selected from the 10 systems, and Yasuda's method estimated its manual score from the other nine systems. Yasuda's method was evaluated by Gap, which is defined by Equation 7.

$$Gap = \frac{\sum_{k=1}^{m} | s(x_k) - y_k |}{m} \quad (7)$$

where x_k is the k^{th} system, $s(x_k)$ is a score of x_k by Yasuda's method, and y_k is the manual score for the k^{th} system. Yasuda et al. (2003) tested DP matching (Su et al., 1992), BLEU (Papineni et al., 2002), and NIST[3], as automatic methods used in their evaluation. Instead of those methods, we

[3] http://www.nist.gov/speech/tests/mt/mt2001/resource/

tested ROUGE and cosine distance, both of which have been used for summary evaluation.

If a score by Yasuda's method exceeds the range of the manual score, the score is modified to be within the range. In our experiments, we used evaluation by revision (Fukushima et al., 2002) as the manual evaluation method. The range of the score of this method is between zero and 0.5. If the score is less than zero, it is changed to zero and if greater than 0.5 it is changed to 0.5.

Exp-4: Comparison of Yasuda's method and other automatic methods

In the same way as for the evaluation of Kazawa's method in Exp-2, we evaluated Yasuda's method by Precision. Two arbitrary summaries from the 10 summaries in a dataset were selected, and ranked by Yasuda's method. Then, Yasuda's method was evaluated using Precision. Two summaries were also ranked by ROUGE and by cosine distance and both Precision values were calculated. Finally, the Precision value of Yasuda's method was compared with those of ROUGE and cosine distance.

4.4 The Data Used in Our Experiments

We used the TSC-2 data (Fukushima et al., 2002) in our examinations. The data consisted of human-produced extracts (denoted as "PART"), human-produced abstracts (denoted as "FREE"), computer-produced summaries (eight systems and a baseline system using the lead method (denoted as "LEAD"))[4], and their evaluation results by two manual methods. All the summaries were derived from 30 newspaper articles, written in Japanese, and were extracted from the Mainichi newspaper database for the years 1998 and 1999. Two tasks were conducted in TSC-2, and we used the data from a single document summarization task. In this task, participants were asked to produce summaries in plain text in the ratios of 20% and 40%.

Summaries were evaluated using a ranking evaluation method and the revision method evaluation. In our experiments, we used the results of evaluation from the revision method. This method evaluates summaries by measuring the degree to which computer-produced summaries are revised. The judges read the original texts and revised the computer-produced summaries in terms of their content and readability. The human revisions were made with only three editing operations (insertion, deletion, replacement). The degree of the human revision, called the "edit distance," is computed from the number of revised characters divided by the number of characters in the original summary. If the summary's quality was so low that a revision of more than half of the original summary was required, the judges stopped the revision and a score of 0.5 was given.

The effectiveness of evaluation by the revision method was confirmed in our previous work (Nanba et al., 2004). We compared evaluation by revision with ranking evaluation. We also tested other automatic methods: content-based evaluation, BLEU (Papineni et al., 2001) and ROUGE-1 (Lin, 2004), and compared their results with that of evaluation by revision as reference. As a result, we found that evaluation by revision is effective for recognizing slight differences between computer-produced summaries.

4.5 Experimental Results and Discussion

Exp-1: An experiment for Points 2 and 3 based on Kazawa's method

To address Points 2 and 3, we evaluated summaries by the method based on Kazawa's method using 12 measures, described in Section 4.4, as measures to calculate topical similarities between summaries, and compared these measures by Gap. The experimental results for summarization ratios of 40% and 20% are shown in Tables 1 and 2, respectively. Tables show the Gap values of 12 measures for each Coverage value from 0.2 to 1.0 at 0.1 intervals. Average values of Gap for each measure are also shown in these tables. As can be seen from Tables 1 and 2, the larger the threshold value, the smaller the value of Gap. From the result, we can conclude for Point 3 that more accurate evaluation is possible when we use similar pooled summaries (Point 2). However, the number of summaries that can be evaluated by this method was limited when the threshold value was large.

Of the 12 measures, unigram-based methods, such as cosine distance and ROUGE-1, produced good results. However, there were no significant differences between measures except for when ROUGE-L was used.

[4] In Exp-2 and 4, we evaluated "PART", "LEAD", and eight systems (candidate summaries) by automatic methods using "FREE" as the reference summaries.

Table 1 Comparison of Gap values for several measures (ratio: 40%)

Measure \ Coverage	1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	Average
R-1	0.080	0.070	0.067	0.057	0.064	0.062	0.058	0.045	0.041	0.062
R-2	0.082	0.074	0.070	0.070	0.069	0.063	0.059	0.051	0.042	0.065
R-3	0.083	0.074	0.075	0.071	0.069	0.063	0.059	0.051	0.045	0.066
R-4	0.085	0.078	0.076	0.073	0.069	0.064	0.060	0.051	0.043	0.067
R-L	0.102	0.100	0.097	0.094	0.091	0.090	0.089	0.082	0.078	0.091
R-S	0.083	0.077	0.073	0.073	0.069	0.067	0.064	0.060	0.045	0.068
R-S4	0.083	0.072	0.071	0.069	0.066	0.066	0.060	0.054	0.044	0.065
R-S9	0.083	0.075	0.069	0.070	0.067	0.066	0.066	0.057	0.046	0.067
R-SU	0.083	0.077	0.070	0.071	0.069	0.068	0.064	0.057	0.043	0.067
R-SU4	0.082	0.073	0.069	0.069	0.065	0.068	0.063	0.051	0.043	0.065
R-SU9	0.083	0.074	0.070	0.068	0.066	0.067	0.066	0.054	0.046	0.066
Cosine	0.081	0.074	0.065	0.062	0.059	0.056	0.057	0.039	0.043	**0.059**
Threshold	Small ◄								► Large	

Table 2 Comparison of Gap values for several measures (ratio: 20%)

Measure \ Coverage	1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	Average
R-1	0.129	0.104	0.102	0.976	0.090	0.089	0.089	0.083	0.082	0.096
R-2	0.132	0.115	0.107	0.109	0.096	0.093	0.079	0.081	0.082	0.099
R-3	0.132	0.115	0.116	0.111	0.102	0.092	0.080	0.078	0.079	0.101
R-4	0.134	0.121	0.121	0.112	0.103	0.090	0.080	0.080	0.078	0.102
R-L	0.140	0.135	0.134	0.125	0.117	0.110	0.105	0.769	0.060	0.111
R-S	0.130	0.119	0.113	0.106	0.098	0.099	0.089	0.089	0.087	0.103
R-S4	0.130	0.114	0.109	0.105	0.102	0.092	0.085	0.088	0.085	0.101
R-S9	0.130	0.119	0.113	0.105	0.095	0.097	0.095	0.085	0.084	0.103
R-SU	0.130	0.118	0.109	0.109	0.097	0.098	0.088	0.089	0.079	0.102
R-SU4	0.130	0.111	0.107	0.106	0.100	0.090	0.086	0.084	0.087	0.100
R-SU9	0.130	0.116	0.108	0.105	0.096	0.090	0.085	0.085	0.082	0.099
Cosine	0.128	0.106	0.102	0.094	0.091	0.090	0.079	0.080	0.057	**0.092**
Threshold	Small ◄								► Large	

Exp-2: Comparison of Kazawa's method with other automatic methods (Point 2)

In Exp-1, cosine distance outperformed the other 11 measures. We therefore used cosine distance in Kazawa's method in Exp-2. We ranked summaries by Kazawa's method, ROUGE and cosine distance, calculated using Precision.

The results of the evaluation by Precision for summarization ratios of 40% and 20% are shown in Figures 1 and 2, respectively. We plotted the Precision value of Kazawa's method by changing the threshold value from 0 to 1 at 0.05 intervals. We also plotted the Precision values of ROUGE-2 as dotted lines. ROUGE-2 was superior to the other 11 measures in terms of Ranking. The X and Y axes in Figures 1 and 2 show the threshold value of Kazawa's method and the Precision values, respectively. From the result shown in Figure 1, we found that Kazawa's method outperformed ROUGE-2, when the threshold value was greater than 0.968. The Coverage value of this point was 0.203. In Figure 2, the Precision curve of Kazawa's method crossed the dotted line at a threshold value of 0.890. The Coverage value of this point was 0.405.

To improve these Coverage values, we need to prepare more summaries and their manual evaluation results, because the Coverage is critically dependent on the number and variety of pooled summaries. This is exactly the first point in Section 3.1, which we do not address in this paper. We will investigate this point as the next step in our future work.

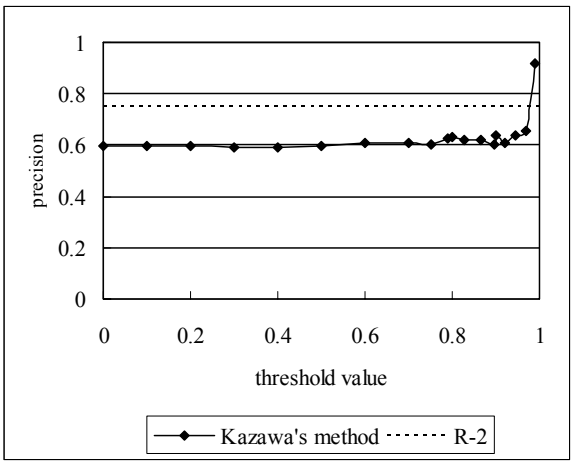


Figure 1 Comparison of Kazawa's method and ROUGE-2 (ratio: 40%)

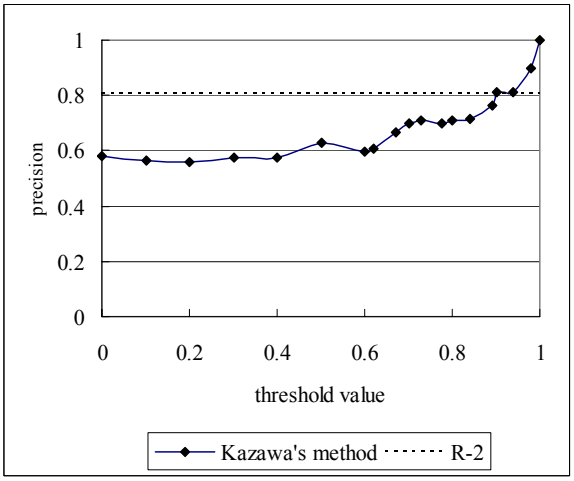


Figure 2 Comparison of Kazawa's method and ROUGE-2 (ratio: 20%)

Exp-3: An experiment for Point 3 based on Yasuda's method

For Point 2 in Section 3.2, we also examined Yasuda's method. The experimental result by Gap is shown in Table 3. When the ratio is 20%, ROUGE-SU4 is the best. The N-gram and the skip-bigram are both useful when the summarization ratio is low.

For Point 4, we compared the result by Yasuda's method (Table 3) with that of Kazawa's method (in Tables 1 and 2). Yasuda's method could accurately estimate manual scores. In particular, the Gap values of 0.023 by ROUGE-2 and by ROUGE-3 are smaller than those produced by Kazawa's method with a threshold value of 0.9 (Tables 1 and 2). This indicates that regression analysis used in Yasuda's method is superior to that used in Kazawa's method.

Table 3 Gap between the manual method and Yasuda's method

	Ratio		Average
	20%	40%	
Cosine	0.037	0.031	0.035
R-1	0.033	**0.022**	0.028
R-2	0.028	0.023	**0.025**
R-3	0.028	0.023	**0.025**
R-4	0.036	0.024	0.030
R-L	0.040	0.038	0.039
R-S(∞)	0.051	0.060	0.055
R-S4	0.025	0.040	0.033
R-S9	0.042	0.052	0.047
R-SU(∞)	0.027	0.055	0.041
R-SU4	**0.022**	0.037	0.029
R-SU9	0.023	0.048	0.036

Exp-4: Comparison of Yasuda's method with other automatic methods

We also evaluated Yasuda's method by comparison with other automatic methods in terms of Ranking. We evaluated 10 systems by Yasuda's method with ROUGE-3, which produced the best results in Exp-3. We also evaluated the systems by ROUGE and cosine distance, and compared the results. The results are shown in Table 4.

Table 4 Comparison between Yasuda's method and automatic methods

	Ratio		Average
	20%	40%	
Yasuda	**0.867**	**0.844**	**0.856**
Cosine	0.844	0.800	0.822
R-1	0.822	0.778	0.800
R-2	0.844	0.800	0.822
R-3	0.822	0.800	0.811
R-4	0.822	**0.844**	0.833
R-L	0.822	0.800	0.811
R-S(∞)	0.667	0.689	0.678
R-S4	0.800	0.756	0.778
R-S9	0.733	0.689	0.711
R-SU(∞)	0.711	0.711	0.711
R-SU4	0.800	0.822	0.811
R-SU9	0.756	0.711	0.733

As can be seen from Table 4, Yasuda's method produced the best results for the ratios of 20% and 40%. Of the automatic methods compared, ROUGE-4 was the best.

As evaluation scores by Yasuda's method were calculated based on ROUGE-3, there were no striking differences between Yasuda's method and the others except for the integration process of evaluation scores for each summary. Yasuda's method uses a regression analysis, whereas the other methods average the scores for each summary. Yasuda's method using ROUGE-3 outperformed the original ROUGE-3 for both ratios, 20% and 40%.

5 Conclusions

We have investigated an automatic method that uses several evaluation results from a manual method based on Kazawa's and Yasuda's methods. From the experimental results based on Kazawa's method, we found that limiting the number of pooled summaries could produce better results than using all the pooled summaries. However, the number of summaries that can be evaluated by this method was limited. To improve the Coverage of Kazawa's method, more summaries and their evaluation results are required, because the Coverage is critically dependent on the number and variety of pooled summaries.

We also investigated an automatic method based on Yasuda's method and found that the method using ROUGE-2 and -3 could accurately estimate manual scores, and could outperform Kazawa's method and the other automatic methods tested. From these results, we can conclude that the automatic method performed the best when ROUGE-2 or 3 is used as a similarity measure, and a regression analysis is used for combining manual method.

References

Robert L. Donaway, Kevin W. Drummey and Laura A. Mather. 2000. A Comparison of Rankings Produced by Summarization Evaluation Measures. Proceedings of the ANLP/NAACL 2000 Workshop on Automatic Summarization: 69–78.

Takahiro Fukushima and Manabu Okumura. 2001. Text Summarization Challenge/Text Summarization Evaluation at NTCIR Workshop2. *Proceedings of the Second NTCIR Workshop on Research in Chinese and Japanese Text Retrieval and Text Summarization*: 45–51.

Takahiro Fukushima, Manabu Okumura and Hidetsugu Nanba. 2002. Text Summarization Challenge 2/Text Summarization Evaluation at NTCIR Workshop3. *Working Notes of the 3rd NTCIR Workshop Meeting, PART V*: 1–7.

Tsutomu Hirao, Manabu Okumura, and Hideki Isozaki. 2005. Kernel-based Approach for Automatic Evaluation of Natural Language Generation Technologies: Application to Automatic Summarization. *Proceedings of HLT-EMNLP 2005*: 145–152.

Chiori Hori, Takaaki Hori, and Sadaoki Furui. 2003. Evaluation Methods for Automatic Speech Summarization. *Proceedings of Eurospeech 2003*: 2825–2828.

Hideto Kazawa, Thomas Arrigan, Tsutomu Hirao and Eisaku Maeda. 2003. An Automatic Evaluation Method of Machine-Generated Extracts. *IPSJ SIG Technical Reports, 2003-NL-158*: 25–30. (in Japanese).

Chin-Yew Lin and Eduard Hovy. 2003. Automatic Evaluation of Summaries Using N-gram Co-Occurrence Statistics. *Proceedings of the Human Language Technology Conference 2003*: 71–78.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. *Proceedings of the ACL-04 Workshop "Text Summarization Branches Out"*: 74–81.

Hidetsugu Nanba and Manabu Okumura. 2004. Comparison of Some Automatic and Manual Methods for Summary Evaluation Based on the Text Summarization Challenge 2. *Proceedings of the Fourth International Conference on Language Resources and Evaluation*: 1029–1032.

Ani Nenkova and Rebecca Passonneau, 2004. Evaluating Content Selection in Summarization: The Pyramid Method. *Proceedings of HLT-NAACL 2004*: 145–152.

Kishore Papineni, Salim Roukos, Todd Ward and Wei-Jing Zhu. 2001. BLEU: A Method for Automatic Evaluation of Machine Translation. *IBM Research Report,* RC22176 (W0109-022).

Keh-Yih Su, Ming-Wen Wu, and Jing-Shin Chang. 1992. A New Quantitative Quality Measure for Machine Translation Systems. *Proceedings of the 14th International Conference on Computational Linguistics*: 433–439.

Simone Teufel and Hans van Halteren. 2004. Evaluating Information Content by Factoid Analysis: Human Annotation and Stability. *Proceedings of EMNLP 2004*: 419–426.

Kenji Yasuda, Fumiaki Sugaya, Toshiyuki Takezawa, Seiichi Yamamoto and Masuzo Yanagida. 2003. Applications of Automatic Evaluation Methods to Measuring a Capability of Speech Translation System. *Proceedings of the Tenth Conference of the European Chapter of the Association for Computational Linguistics*: 371–378.

Examining the Role of Linguistic Knowledge Sources in the Automatic Identification and Classification of Reviews

Vincent Ng and **Sajib Dasgupta** and **S. M. Niaz Arifin**
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083-0688
{vince,sajib,arif}@hlt.utdallas.edu

Abstract

This paper examines two problems in document-level sentiment analysis: (1) determining whether a given document is a review or not, and (2) classifying the polarity of a review as positive or negative. We first demonstrate that review identification can be performed with high accuracy using only unigrams as features. We then examine the role of four types of simple linguistic knowledge sources in a polarity classification system.

1 Introduction

Sentiment analysis involves the identification of positive and negative opinions from a text segment. The task has recently received a lot of attention, with applications ranging from multi-perspective question-answering (e.g., Cardie et al. (2004)) to opinion-oriented information extraction (e.g., Riloff et al. (2005)) and summarization (e.g., Hu and Liu (2004)). Research in sentiment analysis has generally proceeded at three levels, aiming to identify and classify opinions from *documents*, *sentences*, and *phrases*. This paper examines two problems in document-level sentiment analysis, focusing on analyzing a particular type of opinionated documents: *reviews*.

The first problem, *polarity classification*, has the goal of determining a review's polarity — *positive* ("thumbs up") or *negative* ("thumbs down"). Recent work has expanded the polarity classification task to additionally handle documents expressing a *neutral* sentiment. Although studied fairly extensively, polarity classification remains a challenge to natural language processing systems.

We will focus on an important linguistic aspect of polarity classification: examining the role of a variety of simple, yet under-investigated, linguistic knowledge sources in a learning-based polarity classification system. Specifically, we will show how to build a high-performing polarity classifier by exploiting information provided by (1) high order n-grams, (2) a lexicon composed of adjectives manually annotated with their polarity information (e.g., *happy* is annotated as positive and *terrible* as negative), (3) dependency relations derived from dependency parses, and (4) objective terms and phrases extracted from *neutral* documents.

As mentioned above, the majority of work on document-level sentiment analysis to date has focused on polarity classification, assuming as input a set of reviews to be classified. A relevant question is: what if we don't know that an input document is a review in the first place? The second task we will examine in this paper — *review identification* — attempts to address this question. Specifically, review identification seeks to determine whether a given document is a review or not.

We view both *review identification* and *polarity classification* as a classification task. For review identification, we train a classifier to distinguish movie reviews and movie-related non-reviews (e.g., movie ads, plot summaries) using only unigrams as features, obtaining an accuracy of over 99% via 10-fold cross-validation. Similar experiments using documents from the book domain also yield an accuracy as high as 97%. An analysis of the results reveals that the high accuracy can be attributed to the difference in the vocabulary employed in reviews and non-reviews: while reviews can be composed of a mixture of subjective and objective language, our non-review documents rarely contain subjective expressions.

Next, we learn our polarity classifier using positive and negative reviews taken from two movie

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 611–618,
Sydney, July 2006.

review datasets, one assembled by Pang and Lee (2004) and the other by ourselves. The resulting classifier, when trained on a feature set derived from the four types of linguistic knowledge sources mentioned above, achieves a 10-fold cross-validation accuracy of 90.5% and 86.1% on Pang et al.'s dataset and ours, respectively. To our knowledge, our result on Pang et al.'s dataset is one of the best reported to date. Perhaps more importantly, an analysis of these results show that the various types of features interact in an interesting manner, allowing us to draw conclusions that provide new insights into polarity classification.

2 Related Work

2.1 Review Identification

As noted in the introduction, while a review can contain both subjective and objective phrases, our non-reviews are essentially factual documents in which subjective expressions can rarely be found. Hence, review identification can be viewed as an instance of the broader task of classifying whether a document is *mostly factual/objective* or *mostly opinionated/subjective*. There have been attempts on tackling this so-called *document-level subjectivity classification* task, with very encouraging results (see Yu and Hatzivassiloglou (2003) and Wiebe et al. (2004) for details).

2.2 Polarity Classification

There is a large body of work on classifying the polarity of a document (e.g., Pang et al. (2002), Turney (2002)), a sentence (e.g., Liu et al. (2003), Yu and Hatzivassiloglou (2003), Kim and Hovy (2004), Gamon et al. (2005)), a phrase (e.g., Wilson et al. (2005)), and a specific object (such as a product) mentioned in a document (e.g., Morinaga et al. (2002), Yi et al. (2003), Popescu and Etzioni (2005)). Below we will center our discussion of related work around the four types of features we will explore for polarity classification.

Higher-order n-grams. While n-grams offer a simple way of capturing context, previous work has rarely explored the use of n-grams as features in a polarity classification system beyond unigrams. Two notable exceptions are the work of Dave et al. (2003) and Pang et al. (2002). Interestingly, while Dave et al. report good performance on classifying reviews using bigrams or trigrams alone, Pang et al. show that bigrams are not useful features for the task, whether they are used in isolation or in conjunction with unigrams. This motivates us to take a closer look at the utility of higher-order n-grams in polarity classification.

Manually-tagged term polarity. Much work has been performed on learning to identify and classify *polarity* terms (i.e., terms expressing a positive sentiment (e.g., *happy*) or a negative sentiment (e.g., *terrible*)) and exploiting them to do polarity classification (e.g., Hatzivassiloglou and McKeown (1997), Turney (2002), Kim and Hovy (2004), Whitelaw et al. (2005), Esuli and Sebastiani (2005)). Though reasonably successful, these (semi-)automatic techniques often yield lexicons that have either high coverage/low precision or low coverage/high precision. While manually constructed positive and negative word lists exist (e.g., General Inquirer[1]), they too suffer from the problem of having low coverage. This prompts us to manually construct our own polarity word lists[2] and study their use in polarity classification.

Dependency relations. There have been several attempts at extracting features for polarity classification from dependency parses, but most focus on extracting specific types of information such as *adjective-noun relations* (e.g., Dave et al. (2003), Yi et al. (2003)) or *nouns* that enjoy a dependency relation with a polarity term (e.g., Popescu and Etzioni (2005)). Wilson et al. (2005) extract a larger variety of features from dependency parses, but unlike us, their goal is to determine the polarity of a *phrase*, not a document. In comparison to previous work, we investigate the use of a larger set of dependency relations for classifying reviews.

Objective information. The objective portions of a review do not contain the author's opinion; hence features extracted from objective sentences and phrases are irrelevant with respect to the polarity classification task and their presence may complicate the learning task. Indeed, recent work has shown that benefits can be made by first separating *facts* from *opinions* in a document (e.g, Yu and Hatzivassiloglou (2003)) and classifying the polarity based solely on the subjective portions of the document (e.g., Pang and Lee (2004)). Motivated by the work of Koppel and Schler (2005), we identify and extract objective material from *non-reviews* and show how to exploit such information in polarity classification.

[1] `http://www.wjh.harvard.edu/~inquirer/spreadsheet_guid.htm`

[2] Wilson et al. (2005) have also manually tagged a list of terms with their polarity, but this list is not publicly available.

Finally, previous work has also investigated features that do not fall into any of the above categories. For instance, instead of representing the polarity of a term using a binary value, Mullen and Collier (2004) use Turney's (2002) method to assign a real value to represent term polarity and introduce a variety of numerical features that are aggregate measures of the polarity values of terms selected from the document under consideration.

3 Review Identification

Recall that the goal of review identification is to determine whether a given document is a review or not. Given this definition, two immediate questions come to mind. First, should this problem be addressed in a domain-specific or domain-independent manner? In other words, should a review identification system take as input documents coming from the same domain or not?

Apparently this is a design question with no definite answer, but our decision is to perform domain-specific review identification. The reason is that the primary motivation of review identification is the need to identify reviews for further analysis by a polarity classification system. Since polarity classification has almost exclusively been addressed in a domain-specific fashion, it seems natural that its immediate upstream component — review identification — should also assume domain specificity. Note, however, that assuming domain specificity is not a self-imposed limitation. In fact, we envision that the review identification system will have as its upstream component a text classification system, which will classify documents by topic and pass to the review identifier only those documents that fall within its domain.

Given our choice of domain specificity, the next question is: which documents are non-reviews? Here, we adopt a simple and natural definition: a non-review is any document that belongs to the given domain but is not a review.

Dataset. Now, recall from the introduction that we cast review identification as a classification task. To train and test our review identifier, we use 2000 reviews and 2000 non-reviews from the movie domain. The 2000 reviews are taken from Pang et al.'s polarity dataset (version 2.0)[3], which consists of an equal number of positive and negative reviews. We collect the non-reviews for the movie domain from the Internet Movie Database website[4], randomly selecting any documents from this site that are on the movie topic but are not reviews themselves. With this criterion in mind, the 2000 non-review documents we end up with are either movie ads or plot summaries.

Training and testing the review identifier. We perform 10-fold cross-validation (CV) experiments on the above dataset, using Joachims' (1999) SVM^{light} package[5] to train an SVM classifier for distinguishing reviews and non-reviews. All learning parameters are set to their default values.[6] Each document is first tokenized and downcased, and then represented as a vector of unigrams with length normalization.[7] Following Pang et al. (2002), we use frequency as presence. In other words, the ith element of the document vector is 1 if the corresponding unigram is present in the document and 0 otherwise. The resulting classifier achieves an accuracy of 99.8%.

Classifying neutral reviews and non-reviews. Admittedly, the high accuracy achieved using such a simple set of features is somewhat surprising, although it is consistent with previous results on document-level subjectivity classification in which accuracies of 94-97% were obtained (Yu and Hatzivassiloglou, 2003; Wiebe et al., 2004). Before concluding that review classification is an easy task, we conduct an additional experiment: we train a review identifier on a new dataset where we keep the same 2000 non-reviews but replace the positive/negative reviews with 2000 *neutral* reviews (i.e., reviews with a mediocre rating). Intuitively, a neutral review contains fewer terms with strong polarity than a positive/negative review. Hence, this additional experiment would allow us to investigate whether the lack of strong polarized terms in neutral reviews would increase the difficulty of the learning task.

Our neutral reviews are randomly chosen from Pang et al.'s pool of 27886 unprocessed movie reviews[8] that have either a rating of 2 (on a 4-point scale) or 2.5 (on a 5-point scale). Each review then undergoes a semi-automatic preprocessing stage

[3] Available from `http://www.cs.cornell.edu/people/pabo/movie-review-data`.

[4] See `http://www.imdb.com`.

[5] Available from `svmlight.joachims.org`.

[6] We tried polynomial and RBF kernels, but none yields better performance than the default linear kernel.

[7] We observed that not performing length normalization hurts performance slightly.

[8] Also available from Pang's website. See Footnote 3.

where (1) HTML tags and any header and trailer information (such as date and author identity) are removed; (2) the document is tokenized and downcased; (3) the rating information extracted by regular expressions is removed; and (4) the document is manually checked to ensure that the rating information is successfully removed. When trained on this new dataset, the review identifier also achieves an accuracy of 99.8%, suggesting that this learning task isn't any harder in comparison to the previous one.

Discussion. We hypothesized that the high accuracies are attributable to the different vocabulary used in reviews and non-reviews. As part of our verification of this hypothesis, we plot the learning curve for each of the above experiments.[9] We observe that a 99% accuracy was achieved in all cases even when only 200 training instances are used to acquire the review identifier. The ability to separate the two classes with such a small amount of training data seems to imply that features strongly indicative of one or both classes are present. To test this hypothesis, we examine the "informative" features for both classes. To get these informative features, we rank the features by their weighted log-likelihood ratio (WLLR)[10]:

$$P(w_t|c_j) \log \frac{P(w_t|c_j)}{P(w_t|\neg c_j)},$$

where w_t and c_j denote the tth word in the vocabulary and the jth class, respectively. Informally, a feature (in our case a unigram) w will have a high rank with respect to a class c if it appears frequently in c and infrequently in other classes. This correlates reasonably well with what we think an informative feature should be. A closer examination of the feature lists sorted by WLLR confirms our hypothesis that each of the two classes has its own set of distinguishing features.

Experiments with the book domain. To understand whether these good review identification results only hold true for the movie domain, we conduct similar experiments with book reviews and non-reviews. Specifically, we collect 1000 book reviews (consisting of a mixture of positive, negative, and neutral reviews) from the Barnes and Noble website[11], and 1000 non-reviews that are on the book topic (mostly book summaries) from Amazon.[12] We then perform 10-fold CV experiments using these 2000 documents as before, achieving a high accuracy of 96.8%. These results seem to suggest that automatic review identification can be achieved with high accuracy.

4 Polarity Classification

Compared to review identification, polarity classification appears to be a much harder task. This section examines the role of various linguistic knowledge sources in our learning-based polarity classification system.

4.1 Experimental Setup

Like several previous work (e.g., Mullen and Collier (2004), Pang and Lee (2004), Whitelaw et al. (2005)), we view polarity classification as a supervised learning task. As in review identification, we use SVM^{light} with default parameter settings to train polarity classifiers[13], reporting all results as 10-fold CV accuracy.

We evaluate our polarity classifiers on two movie review datasets, each of which consists of 1000 positive reviews and 1000 negative reviews. The first one, which we will refer to as Dataset A, is the Pang et al. polarity dataset (version 2.0). The second one (Dataset B) was created by us, with the sole purpose of providing additional experimental results. Reviews in Dataset B were randomly chosen from Pang et al.'s pool of 27886 unprocessed movie reviews (see Section 3) that have either a positive or a negative rating. We followed exactly Pang et al.'s guideline when determining whether a review is positive or negative.[14] Also, we took care to ensure that reviews included in Dataset B do not appear in Dataset A. We applied to these reviews the same four pre-processing steps that we did to the neutral reviews in the previous section.

4.2 Results

The baseline classifier. We can now train our baseline polarity classifier on each of the two

[9] The curves are not shown due to space limitations.

[10] Nigam et al. (2000) show that this metric is effective at selecting good features for text classification. Other commonly-used feature selection metrics are discussed in Yang and Pedersen (1997).

[11] www.barnesandnoble.com

[12] www.amazon.com

[13] We also experimented with polynomial and RBF kernels when training polarity classifiers, but neither yields better results than linear kernels.

[14] The guidelines come with their polarity dataset. Briefly, a positive review has a rating of ≥ 3.5 (out of 5) or ≥ 3 (out of 4), whereas a negative review has a rating of ≤ 2 (out of 5) or ≤ 1.5 (out of 4).

System Variation	Dataset A	Dataset B
Baseline	87.1	82.7
Adding bigrams and trigrams	89.2	84.7
Adding dependency relations	89.0	84.5
Adding polarity info of adjectives	90.4	**86.2**
Discarding objective materials	**90.5**	86.1

Table 1: Polarity classification accuracies.

datasets. Our baseline classifier employs as features the k highest-ranking unigrams according to WLLR, with k/2 features selected from each class. Results with k = 10000 are shown in row 1 of Table 1.[15] As we can see, the baseline achieves an accuracy of 87.1% and 82.7% on Datasets A and B, respectively. Note that our result on Dataset A is as strong as that obtained by Pang and Lee (2004) via their subjectivity summarization algorithm, which retains only the subjective portions of a document.

As a sanity check, we duplicated Pang et al.'s (2002) baseline in which all unigrams that appear four or more times in the training documents are used as features. The resulting classifier achieves an accuracy of 87.2% and 82.7% for Datasets A and B, respectively. Neither of these results are significantly different from our baseline results.[16]

Adding higher-order n-grams. The negative results that Pang et al. (2002) obtained when using bigrams as features for their polarity classifier seem to suggest that high-order n-grams are not useful for polarity classification. However, recent research in the related (but arguably simpler) task of text classification shows that a bigram-based text classifier outperforms its unigram-based counterpart (Peng et al., 2003). This prompts us to re-examine the utility of high-order n-grams in polarity classification.

In our experiments we consider adding bigrams and trigrams to our baseline feature set. However, since these higher-order n-grams significantly outnumber the unigrams, adding all of them to the feature set will dramatically increase the dimensionality of the feature space and may undermine the impact of the unigrams in the resulting classifier. To avoid this potential problem, we keep the number of unigrams and higher-order n-grams equal. Specifically, we augment the baseline feature set (consisting of 10000 unigrams) with 5000 bigrams and 5000 trigrams. The bigrams and trigrams are selected based on their WLLR computed over the positive reviews and negative reviews in the training set for each CV run.

Results using this augmented feature set are shown in row 2 of Table 1. We see that accuracy rises significantly from 87.1% to 89.2% for Dataset A and from 82.7% to 84.7% for Dataset B. This provides evidence that polarity classification can indeed benefit from higher-order n-grams.

Adding dependency relations. While bigrams and trigrams are good at capturing local dependencies, dependency relations can be used to capture non-local dependencies among the constituents of a sentence. Hence, we hypothesized that our n-gram-based polarity classifier would benefit from the addition of dependency-based features.

Unlike most previous work on polarity classification, which has largely focused on exploiting adjective-noun (AN) relations (e.g., Dave et al. (2003), Popescu and Etzioni (2005)), we hypothesized that subject-verb (SV) and verb-object (VO) relations would also be useful for the task. The following (one-sentence) review illustrates why.

> *While I really like the actors, the plot is rather uninteresting.*

A unigram-based polarity classifier could be confused by the simultaneous presence of the positive term *like* and the negative term *uninteresting* when classifying this review. However, incorporating the VO relation ($like$, *actors*) as a feature may allow the learner to learn that the author likes the actors and not necessarily the movie.

In our experiments, the SV, VO and AN relations are extracted from each document by the MINIPAR dependency parser (Lin, 1998). As with n-grams, instead of using all the SV, VO and AN relations as features, we select among them the best 5000 according to their WLLR and retrain the polarity classifier with our n-gram-based feature set augmented by these 5000 dependency-based features. Results in row 3 of Table 1 are somewhat surprising: the addition of dependency-based features does not offer any improvements over the simple n-gram-based classifier.

[15] We experimented with several values of k and obtained the best result with k = 10000.

[16] We use two-tailed paired t-tests when performing significance testing, with p set to 0.05 unless otherwise stated.

Incorporating manually tagged term polarity. Next, we consider incorporating a set of features that are computed based on the polarity of adjectives. As noted before, we desire a high-precision, high-coverage lexicon. So, instead of exploiting a learned lexicon, we manually develop one.

To construct the lexicon, we take Pang et al.'s pool of unprocessed documents (see Section 3), remove those that appear in either Dataset A or Dataset B[17], and compile a list of adjectives from the remaining documents. Then, based on heuristics proposed in psycholinguistics[18], we hand-annotate each adjective with its *prior polarity* (i.e., polarity in the absence of context). Out of the 45592 adjectives we collected, 3599 were labeled as positive, 3204 as negative, and 38789 as neutral. A closer look at these adjectives reveals that they are by no means domain-dependent despite the fact that they were taken from movie reviews.

Now let us consider a simple procedure P for deriving a feature set that incorporates information from our lexicon: (1) collect all the bigrams from the training set; (2) for each bigram that contains at least one adjective labeled as positive or negative according to our lexicon, create a new feature that is identical to the bigram except that each adjective is replaced with its polarity label[19]; (3) merge the list of newly generated features with the list of bigrams[20] and select the top 5000 features from the merged list according to their WLLR.

We then repeat procedure P for the trigrams and also the dependency features, resulting in a total of 15000 features. Our new feature set comprises these 15000 features as well as the 10000 unigrams we used in the previous experiments.

Results of the polarity classifier that incorporates term polarity information are encouraging (see row 4 of Table 1). In comparison to the classifier that uses only n-grams and dependency-based features (row 3), accuracy increases significantly (p = .1) from 89.2% to 90.4% for Dataset A, and from 84.7% to 86.2% for Dataset B. These results suggest that the classifier has benefited from the use of features that are less sparse than n-grams.

Using objective information. Some of the 25000 features we generated above correspond to n-grams or dependency relations that do not contain subjective information. We hypothesized that not employing these "objective" features in the feature set would improve system performance. More specifically, our goal is to use procedure P again to generate 25000 "subjective" features by ensuring that the objective ones are not selected for incorporation into our feature set.

To achieve this goal, we first use the following rote-learning procedure to identify objective material: (1) extract all unigrams that appear in objective documents, which in our case are the 2000 non-reviews used in review identification [see Section 3]; (2) from these "objective" unigrams, we take the best 20000 according to their WLLR computed over the non-reviews and the reviews in the training set for each CV run; (3) repeat steps 1 and 2 separately for bigrams, trigrams and dependency relations; (4) merge these four lists to create our 80000-element list of objective material.

Now, we can employ procedure P to get a list of 25000 "subjective" features by ensuring that those that appear in our 80000-element list are not selected for incorporation into our feature set.

Results of our classifier trained using these subjective features are shown in row 5 of Table 1. Somewhat surprisingly, in comparison to row 4, we see that our method for filtering objective features does not help improve performance on the two datasets. We will examine the reasons in the following subsection.

4.3 Discussion and Further Analysis

Using the four types of knowledge sources previously described, our polarity classifier significantly outperforms a unigram-based baseline classifier. In this subsection, we analyze some of these results and conduct additional experiments in an attempt to gain further insight into the polarity classification task. Due to space limitations, we will simply present results on Dataset A below, and show results on Dataset B only in cases where a different trend is observed.

The role of feature selection. In all of our experiments we used the best k features obtained via WLLR. An interesting question is: how will these results change if we do not perform feature selection? To investigate this question, we conduct two

[17] We treat the test documents as unseen data that should not be accessed for any purpose during system development.

[18] `http://www.sci.sdsu.edu/CAL/wordlist`

[19] Neutral adjectives are not replaced.

[20] A newly generated feature could be misleading for the learner if the *contextual polarity* (i.e., polarity in the presence of context) of the adjective involved differs from its prior polarity (see Wilson et al. (2005)). The motivation behind merging with the bigrams is to create a feature set that is more robust in the face of potentially misleading generalizations.

experiments. First, we train a polarity classifier using all unigrams from the training set. Second, we train another polarity classifier using all unigrams, bigrams, and trigrams. We obtain an accuracy of 87.2% and 79.5% for the first and second experiments, respectively.

In comparison to our baseline classifier, which achieves an accuracy of 87.1%, we can see that using all unigrams does not hurt performance, but performance drops abruptly with the addition of all bigrams and trigrams. These results suggest that feature selection is critical when bigrams and trigrams are used in conjunction with unigrams for training a polarity classifier.

The role of bigrams and trigrams. So far we have seen that training a polarity classifier using only unigrams gives us reasonably good, though not outstanding, results. Our question, then, is: would bigrams alone do a better job at capturing the sentiment of a document than unigrams? To answer this question, we train a classifier using all bigrams (without feature selection) and obtain an accuracy of 83.6%, which is significantly worse than that of a unigram-only classifier. Similar results were also obtained by Pang et al. (2002).

It is possible that the worse result is due to the presence of a large number of irrelevant bigrams. To test this hypothesis, we repeat the above experiment except that we only use the best 10000 bigrams selected according to WLLR. Interestingly, the resulting classifier gives us a lower accuracy of 82.3%, suggesting that the poor accuracy is not due to the presence of irrelevant bigrams.

To understand why using bigrams alone does not yield a good classification model, we examine a number of test documents and find that the feature vectors corresponding to some of these documents (particularly the short ones) have all zeroes in them. In other words, none of the bigrams from the training set appears in these reviews. This suggests that the main problem with the bigram model is likely to be data sparseness. Additional experiments show that the trigram-only classifier yields even worse results than the bigram-only classifier, probably because of the same reason.

Nevertheless, these higher-order n-grams play a non-trivial role in polarity classification: we have shown that the addition of bigrams and trigrams selected via WLLR to a unigram-based classifier significantly improves its performance.

The role of dependency relations. In the previous subsection we see that dependency relations do not contribute to overall performance on top of bigrams and trigrams. There are two plausible reasons. First, dependency relations are simply not useful for polarity classification. Second, the higher-order n-grams and the dependency-based features capture essentially the same information and so using either of them would be sufficient.

To test the first hypothesis, we train a classifier using only 10000 unigrams and 10000 dependency-based features (both selected according to WLLR). For Dataset A, the classifier achieves an accuracy of 87.1%, which is statistically indistinguishable from our baseline result. On the other hand, the accuracy for Dataset B is 83.5%, which is significantly better than the corresponding baseline (82.7%) at the p = .1 level. These results indicate that dependency information is somewhat useful for the task when bigrams and trigrams are not used. So the first hypothesis is not entirely true.

So, it seems to be the case that the dependency relations do not provide useful knowledge for polarity classification only in the presence of bigrams and trigrams. This is somewhat surprising, since these n-grams do not capture the non-local dependencies (such as those that may be present in certain SV or VO relations) that should intuitively be useful for polarity classification.

To better understand this issue, we again examine a number of test documents. Our initial investigation suggests that the problem might have stemmed from the fact that MINIPAR returns dependency relations in which all the verb inflections are removed. For instance, given the sentence *My cousin Paul really likes this long movie*, MINIPAR will return the VO relation ($like$, $movie$). To see why this can be a problem, consider another sentence *I like this long movie*. From this sentence, MINIPAR will also extract the VO relation ($like$, $movie$). Hence, this same VO relation is capturing two different situations, one in which the author himself likes the movie, and in the other, the author's cousin likes the movie. The overgeneralization resulting from these "stemmed" relations renders dependency information not useful for polarity classification. Additional experiments are needed to determine the role of dependency relations when stemming in MINIPAR is disabled.

The role of objective information. Results from the previous subsection suggest that our method for extracting objective materials and removing them from the reviews is not effective in terms of improving performance. To determine the reason, we examine the n-grams and the dependency relations that are extracted from the non-reviews. We find that only in a few cases do these extracted objective materials appear in our set of 25000 features obtained in Section 4.2. This explains why our method is not as effective as we originally thought. We conjecture that more sophisticated methods would be needed in order to take advantage of objective information in polarity classification (e.g., Koppel and Schler (2005)).

5 Conclusions

We have examined two problems in document-level sentiment analysis, namely, review identification and polarity classification. We first found that review identification can be achieved with very high accuracies (97-99%) simply by training an SVM classifier using unigrams as features. We then examined the role of several linguistic knowledge sources in polarity classification. Our results suggested that bigrams and trigrams selected according to the weighted log-likelihood ratio as well as manually tagged term polarity information are very useful features for the task. On the other hand, no further performance gains are obtained by incorporating dependency-based information or filtering objective materials from the reviews using our proposed method. Nevertheless, the resulting polarity classifier compares favorably to state-of-the-art sentiment classification systems.

References

C. Cardie, J. Wiebe, T. Wilson, and D. Litman. 2004. Low-level annotations and summary representations of opinions for multi-perspective question answering. In *New Directions in Question Answering*. AAAI Press/MIT Press.

K. Dave, S. Lawrence, and D. M. Pennock. 2003. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proc. of WWW*, pages 519–528.

A. Esuli and F. Sebastiani. 2005. Determining the semantic orientation of terms through gloss classification. In *Proc. of CIKM*, pages 617–624.

M. Gamon, A. Aue, S. Corston-Oliver, and E. K. Ringger. 2005. Pulse: Mining customer opinions from free text. In *Proc. of the 6th International Symposium on Intelligent Data Analysis*, pages 121–132.

V. Hatzivassiloglou and K. McKeown. 1997. Predicting the semantic orientation of adjectives. In *Proc. of the ACL/EACL*, pages 174–181.

M. Hu and B. Liu. 2004. Mining and summarizing customer reviews. In *Proc. of KDD*, pages 168–177.

T. Joachims. 1999. Making large-scale SVM learning practical. In *Advances in Kernel Methods - Support Vector Learning*, pages 44–56. MIT Press.

S.-M. Kim and E. Hovy. 2004. Determining the sentiment of opinions. In *Proc. of COLING*, pages 1367–1373.

M. Koppel and J. Schler. 2005. Using neutral examples for learning polarity. In *Proc. of IJCAI (poster)*.

D. Lin. 1998. Dependency-based evaluation of MINIPAR. In *Proc. of the LREC Workshop on the Evaluation of Parsing Systems*, pages 48–56.

H. Liu, H. Lieberman, and T. Selker. 2003. A model of textual affect sensing using real-world knowledge. In *Proc. of Intelligent User Interfaces (IUI)*, pages 125–132.

S. Morinaga, K. Yamanishi, K. Tateishi, and T. Fukushima. 2002. Mining product reputations on the web. In *Proc. of KDD*, pages 341–349.

T. Mullen and N. Collier. 2004. Sentiment analysis using support vector machines with diverse information sources. In *Proc. of EMNLP*, pages 412–418.

K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. 2000. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134.

B. Pang and L. Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proc. of the ACL*, pages 271–278.

B. Pang, L. Lee, and S. Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *Proc. of EMNLP*, pages 79–86.

F. Peng, D. Schuurmans, and S. Wang. 2003. Language and task independent text categorization with simple language models. In *HLT/NAACL: Main Proc.* , pages 189–196.

A.-M. Popescu and O. Etzioni. 2005. Extracting product features and opinions from reviews. In *Proc. of HLT-EMNLP*, pages 339–346.

E. Riloff, J. Wiebe, and W. Phillips. 2005. Exploiting subjectivity classification to improve information extraction. In *Proc. of AAAI*, pages 1106–1111.

P. Turney. 2002. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proc. of the ACL*, pages 417–424.

C. Whitelaw, N. Garg, and S. Argamon. 2005. Using appraisal groups for sentiment analysis. In *Proc. of CIKM*, pages 625–631.

J. M. Wiebe, T. Wilson, R. Bruce, M. Bell, and M. Martin. 2004. Learning subjective language. *Computational Linguistics*, 30(3):277–308.

T. Wilson, J. M. Wiebe, and P. Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proc. of EMNLP*, pages 347–354.

Y. Yang and J. O. Pedersen. 1997. A comparative study on feature selection in text categorization. In *Proc. of ICML*, pages 412–420.

J. Yi, T. Nasukawa, R. Bunescu, and W. Niblack. 2003. Sentiment analyzer: Extracting sentiments about a given topic using natural language processing techniques. In *Proc. of the IEEE International Conference on Data Mining (ICDM)*.

H. Yu and V. Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proc. of EMNLP*, pages 129–136.

Semantic parsing with Structured SVM Ensemble Classification Models

Le-Minh Nguyen, Akira Shimazu, and **Xuan-Hieu Phan**
Japan Advanced Institute of Science and Technology (JAIST)
Asahidai 1-1, Nomi, Ishikawa, 923-1292 Japan
{nguyenml,shimazu,hieuxuan}@jaist.ac.jp

Abstract

We present a learning framework for structured support vector models in which boosting and bagging methods are used to construct ensemble models. We also propose a selection method which is based on a switching model among a set of outputs of individual classifiers when dealing with natural language parsing problems. The switching model uses subtrees mined from the corpus and a boosting-based algorithm to select the most appropriate output. The application of the proposed framework on the domain of semantic parsing shows advantages in comparison with the original large margin methods.

1 Introduction

Natural language semantic parsing is an interesting problem in NLP (Manning and Schutze, 1999) as it would very likely be part of any interesting NLP applications (Allen, 1995). For example, the necessary of semantic parsing for most of NLP application and the ability to map natural language to a formal query or command language is critical for developing more user-friendly interfaces.

Recent approaches have focused on using structured prediction for dealing with syntactic parsing (B. Taskar et. al., 2004) and text chunking problems (Lafferty et al, 2001). For semantic parsing, Zettlemoyer and Collins (2005) proposed a method for mapping a NL sentence to its logical form by structured classification using a log-linear model that represents a distribution over syntactic and semantic analyses conditioned on the input sentence. Taskar et al (B. Taskar et. al., 2004) present a discriminative approach to parsing inspired by the large-margin criterion underlying support vector machines in which the loss function is factorized analogous to the decoding process. Tsochantaridis et al (Tsochantaridis et al., 2004) propose a large-margin models based on SVMs for structured prediction (SSVM) in general and apply it for syntactic parsing problem so that the models can adapt to overlap features, kernels, and any loss functions.

Following the successes of the SSVM algorithm to structured prediction, in this paper we exploit the use of SSVM to the semantic parsing problem by modifying the loss function, feature representation, maximization algorithm in the original algorithm for structured outputs (Tsochantaridis et al., 2004).

Beside that, forming committees or ensembles of learned systems is known to improve accuracy and bagging and boosting are two popular ensemble methods that typically achieve better accuracy than a single classifier (Dietterich, 2000). This leads to employing ensemble learning models for SSVM is worth to investigate. The first problem of forming an ensemble learning for semantic parsing is how to obtain individual parsers with respect to the fact that each individual parser performs well enough as well as they make different types of errors. The second one is that of combining outputs from individual semantic parsers. The natural way is to use the majority voting strategy that the semantic tree with highest frequency among the outputs obtained by individual parsers is selected. However, it is not sure that the majority voting technique is effective for combining complex outputs such as a logical form structure. Thus, a better combination method for semantic tree output should be investigated.

To deal with these problems, we proposed an

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 619–626,
Sydney, July 2006.

ensemble method which consists of learning and averaging phases in which the learning phases are either a boosting or a bagging model, and the averaging phase is based on a switching method on outputs obtained from all individual SSVMs. For the averaging phase, the switching model is used subtrees mined from the corpus and a boosting-based algorithm to select the most appropriate output.

Applications of SSVM ensemble in the semantic parsing problem show that the proposed SSVM ensemble is better than the SSVM in term of the F-measure score and accuracy measurements.

The rest of this paper are organized as follows: Section 2 gives some background about the structured support vector machine model for structured predictions and related works. Section 3 proposes our ensemble method for structured SVMs on the semantic parsing problem. Section 4 shows experimental results and Section 5 discusses the advantage of our methods and describes future works.

2 Backgrounds

2.1 Related Works

Zelle and Mooney initially proposed the empirically based method using a corpus of NL sentences and their formal representation for learning by inductive logic programming (Zelle, 1996). Several extensions for mapping a NL sentence to its logical form have been addressed by (Tang, 2003). Transforming a natural language sentence to a logical form was formulated as the task of determining a sequence of actions that would transform the given input sentence to a logical form (Tang, 2003). The main problem is how to learn a set of rules from the corpus using the ILP method. The advantage of the ILP method is that we do not need to design features for learning a set of rules from corpus. The disadvantage is that it is quite complex and slow to acquire parsers for mapping sentences to logical forms. Kate et al presented a method (Kate et al., 2005) that used transformation rules to transform NL sentences to logical forms. Those transformation rules are learnt using the corpus of sentences and their logical forms. This method is much simpler than the ILP method, while it can achieve comparable result on the CLANG (Coach Language) and Query corpus. The transformation based method has the condition that the formal language should be in the form of LR grammar.

Ge and Mooney also presented a statistical method (Ge and Mooney, 2005) by merging syntactic and semantic information. Their method relaxed the condition in (Kate et al., 2005) and achieved a state-of the art performance on the CLANG and query database corpus. However the distinction of this method in comparison with the method presented in (Kate et al., 2005) is that Ge and Mooney require training data to have SAPTs, while the transforation based method only needs the LR grammar for the formal language.

The work proposed by (Zettlemoyer and Collins, 2005) that maps a NL sentence to its logical form by structured classification, using a log-linear model that represents a distribution over syntactic and semantic analyses conditioned on the input sentence. This work is quite similar to our work in considering the structured classification problem. The difference is that we used the kernel based method instead of a log-linear model in order to utilize the advantage of handling a very large number of features by maximizing the margin in the learning process.

2.2 Structured Support Vector Models

Structured classification is the problem of predicting y from x in the case where y has a meaningful internal structure. Elements $y \in Y$ may be, for instance, sequences, strings, labelled trees, lattices, or graphs.

The approach we pursue is to learn a discriminant function $F : X \times Y \rightarrow R$ over $< input, output >$ pairs from which we can derive a prediction by maximizing F over the response variable for a specific given input x. Hence, the general form of our hypotheses f is

$$f(x; w) = \arg\max_{y \in Y} F(x; y; w)$$

where w denotes a parameter vector.

As the principle of the maximum-margin presented in (Vapnik, 1998), in the structured classification problem, (Tsochantaridis et al., 2004) proposed several maximum-margin optimization problems.

For convenience, we define

$$\delta\psi_i(y) \equiv \psi(x_i, y_i) - \psi(x_i, y)$$

where (x_i,y_i) is the training data.

The hard-margin optimization problem is:

$$\text{SVM}_0 : min_w \frac{1}{2} \|w\|^2 \quad (1)$$

$$\forall i, \forall y \in Y \backslash y_i : \langle w, \delta\psi_i(y) \rangle > 0 \quad (2)$$

where $\langle w, \delta\psi_i(y) \rangle$ is the linear combination of feature representation for input and output.

The soft-margin criterion was proposed (Tsochantaridis et al., 2004) in order to allow errors in the training set, by introducing slack variables.

$$\mathrm{SVM}_1 : \min \frac{1}{2} \|w\|^2 + \frac{C}{n} \sum_{i=1}^{n} \xi_i, \text{s.t.} \forall i, \xi_i \geq 0 \quad (3)$$

$$\forall i, \forall y \in Y \backslash y_i : \langle w, \delta\psi_i(y) \rangle \geq 1 - \xi_i \quad (4)$$

Alternatively, using a quadratic term $\frac{C}{2n} \sum_i \xi_i^2$ to penalize margin violations, we obtained SVM_2. Here $C > 0$ is a constant that control the trade-off between training error minimization and margin maximization.

To deal with problems in which $|Y|$ is very large, such as semantic parsing, (Tsochantaridis et al., 2004) proposed two approaches that generalize the formulation SVM_0 and SVM_1 to the cases of arbitrary loss function. The first approach is to re-scale the slack variables according to the loss incurred in each of the linear constraints.

$$\mathrm{SVM}^{\Delta_s} : \underbrace{\min}_{w,\xi} \frac{1}{2} \|w\|^2 + \frac{C}{n} \sum_{i=1}^{n} \xi_i, \text{s.t.} \forall i, \xi_i \geq 0 \quad (5)$$

$$\forall i, \forall y \in Y \backslash y_i : \langle w, \delta\psi_i(y) \rangle \geq \frac{1 - \xi_i}{\Delta(y_i, y)} \quad (6)$$

The second approach to include loss function is to re-scale the margin as a special case of the Hamming loss. The margin constraints in this setting take the following form:

$$\forall i, \forall y \in Y \backslash y_i : \langle w, \delta\psi_i(y) \rangle \geq \Delta(y_i, y) - \xi_i \quad (7)$$

This set of constraints yields an optimization problem, namely $\mathrm{SVM}_1^{\Delta_m}$.

2.3 Support Vector Machine Learning

The support vector learning algorithm aims to find a small set of active constraints that ensures a sufficiently accurate solution. The detailed algorithm, as presented in (Tsochantaridis et al., 2004) can be applied to all SVM formulations mentioned above. The only difference between them is the cost function in the following optimization problems:

$$\mathrm{SVM}_1^{\Delta_s} : H(y) \equiv (1 - \langle \delta\psi_i(y), w \rangle)\Delta(y_i, y)$$
$$\mathrm{SVM}_2^{\Delta_s} : H(y) \equiv (1 - \langle \delta\psi_i(y), w \rangle)\sqrt{\Delta(y_i, y)}$$
$$\mathrm{SVM}_1^{\Delta_m} : H(y) \equiv (\Delta(y_i, y) - \langle \delta\psi_i(y), w \rangle)$$
$$\mathrm{SVM}_2^{\Delta_m} : H(y) \equiv (\sqrt{\Delta(y_i, y)} - \langle \delta\psi_i(y), w \rangle)$$

Typically, the way to apply structured SVM is to implement feature mapping $\psi(x, y)$, the loss function $\Delta(y_i, y)$, as well as the maximization algorithm. In the following section, we apply a structured support vector machine to the problem of semantic parsing in which the mapping function, the maximization algorithm, and the loss function are introduced.

3 SSVM Ensemble for Semantic Parsing

Although the bagging and boosting techniques have known to be effective for improving the performance of syntactic parsing (Henderson and Brill, 2000), in this section we focus on our ensemble learning of SSVM for semantic parsing and propose a new effective switching model for either bagging or boosting model.

3.1 SSVM for Semantic Parsing

As discussed in (Tsochantaridis et al., 2004), the major problem for using the SSVM is to implement the feature mapping $\psi(x, y)$, the loss function $\Delta(y_i, y)$, as well as the maximization algorithm. For semantic parsing, we describe here the method of structure representation, the feature mapping, the loss function, and the maximization algorithm.

3.1.1 Structure representation

A tree structure representation incorporated with semantic and syntactic information is named semantically augmented parse tree (SAPT) (Ge and Mooney, 2005). As defined in (Ge and Mooney, 2005), in an SAPT, each internal node in the parse tree is annotated with a semantic label. Figure 1 shows the SAPT for a simple sentence in the CLANG domain. The semantic labels which are shown after dashes are concepts in the domain. Some concepts refer to predicates and take an ordered list of arguments. Concepts such as "team" and "unum" might not have arguments. A special semantic label, "null", is used for a node that does not correspond to any concept in the domain.

3.1.2 Feature mapping

For semantic parsing, we can choose a mapping function to get a model that is isomorphic to a probabilistic grammar in which each rule within the grammar consists of both a syntactic rule and a semantic rule. Each node in a parse tree y for a sentence x corresponds to a grammar rule g_j with a score w_j.

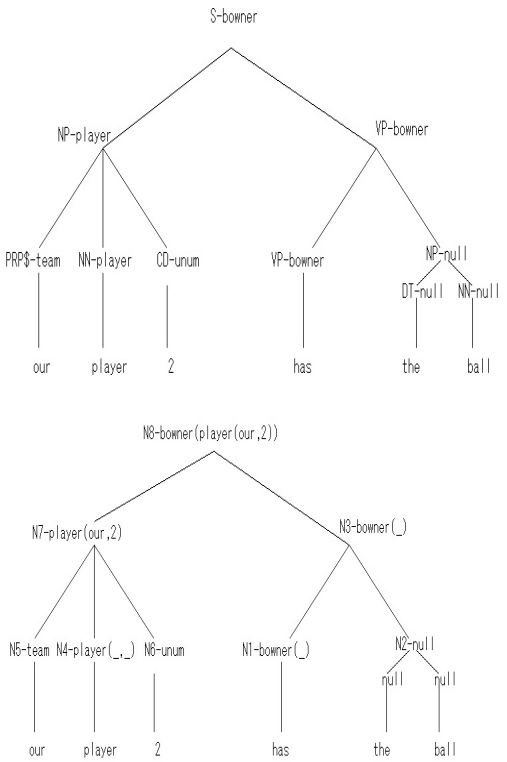


Figure 1: An Example of tree representation in SAPT

All valid parse trees y for a sentence x are scored by the sum of the w_j of their nodes, and the feature mapping $\psi(x, y)$ is a history gram vector counting how often each grammar rule g_j occurs in the tree y. Note that the grammar rules are lexicalized. The example shown in Figure 2 clearly explains the way features are mapped from an input sentence and a tree structure.

3.1.3 Loss function

Let z and z_i be two semantic tree outputs and $|z_i|$ and $|z_i|$ be the number of brackets in z and z_i, respectively. Let n be the number of common brackets in the two trees. The loss function between z_i and z is computed as bellow.

$$F - loss(z_i, z) = 1 - \frac{2 \times n}{|z_i| + |z|} \quad (8)$$

$$\text{zero} - \text{one}(z_i, z) = \begin{cases} 1 \text{ if } z_i \neq z \\ 0 \text{ otherwise} \end{cases} \quad (9)$$

3.1.4 Maximization algorithm

Note that the learning function can be efficiently computed by finding a structure $y \in Y$ that maximizes $F(x, y; w)$=$\langle w, \delta\psi_i(y)\rangle$ via a maximization algorithm. Typically we used a variant of

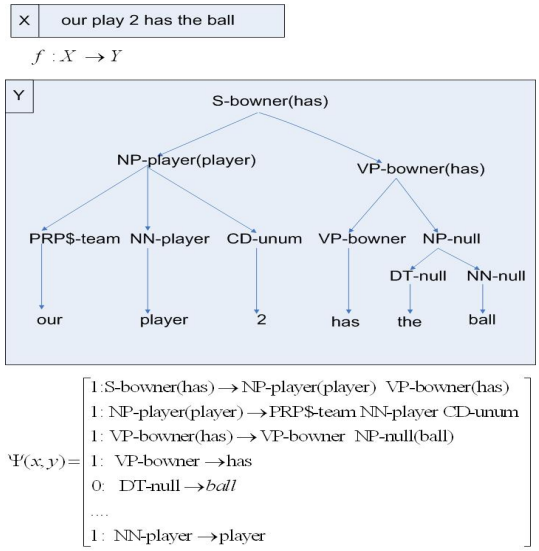


Figure 2: Example of feature mapping using tree representation

CYK maximization algorithm which is similar to the one for the syntactic parsing problem (Johnson,1999). There are two phases in our maximization algorithm for semantic parsing. The first is to use a variant of CYK algorithm to generate a SAPT tree. The second phase then applies a deterministic algorithm to output a logical form. The score of the maximization algorithm is the same with the obtained value of the CYK algorithm.

The procedure of generating a logical form using a SAPT structure originally proposed by (Ge and Mooney, 2005) and it is expressed as Algorithm 1. It generates a logical form based on a knowledge database K for given input node N. The predicate argument knowledge, K, specifies, for each predicate, the semantic constraints on its arguments. Constraints are specified in terms of the concepts that can fill each argument, such as player(team, unum) and bowner(player).

The GETsemanticHEAD determines which of node's children is its semantic head based on they having matching semantic labels. For example, in Figure 1 $N3$ is determined to be the semantic head of the sentence since it matches N8's semantic label. ComposeMR assigns their meaning representation (MR) to fill the arguments in the head's MR to construct the complete MR for the node. Figure 1 shows an example of using BuildMR to generate a semantic tree to a logical form.

Input: The root node N of a SAPT
Predicate knowledge K
Notation: X_{MR} is the MR of node X
Output: N_{MR}
Begin
C_i= the ith child node of N
C_h= GETsemanticHEAD(N)
$C_{h_{MR}}$ =BuildMR(C_h, K)
for *each other child* C_i *where* $i \neq h$ **do**
  $C_{i_{MR}}$ =BuildMR(C_i, K)
  ComposeMR($C_{h_{MR}}$,$C_{i_{MR}}, K$)
end
N_{MR}=$C_{h_{MR}}$
End

Algorithm 1: BuildMR(N, K): Computing a logical form form an SAPT(Ge and Mooney, 2005)

Input: $S = (x_i; y_i; z_i), i = 1, 2, ..., l$ in which x_i is the sentence and y_i, z_i is the pair of tree structure and its logical form
Output: SSVM model
repeat
  for $i = 1$ *to* n **do**

$\text{SVM}_1^{\Delta_s} : H(y, z) \equiv (1 - \langle \delta\psi_i(y), w \rangle)\Delta(z_i, z)$
$\text{SVM}_2^{\Delta_s} : H(y, z) \equiv (1 - \langle \delta\psi_i(y), w \rangle)\sqrt{\Delta(z_i, z)}$
$\text{SVM}_1^{\Delta_m} : H(y, z) \equiv (\Delta(z_i, z) - \langle \delta\psi_i(y), w \rangle)$
$\text{SVM}_2^{\Delta_m} : H(y, z) \equiv (\sqrt{\Delta(z_i, z)} - \langle \delta\psi_i(y), w \rangle)$

    compute $< y^*, z^* >= \arg\max_{y,z \in Y,Z} H(Y, Z)$;
    compute $\xi_i = \max\{0, \max_{y,z \in S_i} H(y, z)\}$;
    if $H(y^*, z^*) > \xi_i + \varepsilon$ **then**
      $S_i \leftarrow S_i \cup y^*, z^*$;
      solving optimization with SVM;
    end
  end
until *no* S_i *has changed during iteration*;

Algorithm 2: Algorithm of SSVM learning for semantic parsing. The algorithm is based on the original algorithm(Tsochantaridis et al., 2004)

3.1.5 SSVM learning for semantic parsing

As mentioned above, the proposed maximization algorithm includes two parts: the first is to parse the given input sentence to the SAPT tree and the second part (BuildMR) is to convert the SAPT tree to a logical form. Here, the score of maximization algorithm is the same with the score to generate a SAPT tree and the loss function should be the measurement based on two logical form outputs. Algorithm 2 shows our generation of SSVM learning for the semantic parsing problem which the loss function is based on the score of two logical form outputs.

3.2 SSVM Ensemble for semantic parsing

The structured SVM ensemble consists of a training and a testing phase. In the training phase, each individual SSVM is trained independently by its own replicated training data set via a bootstrap method. In the testing phase, a test example is applied to all SSVMs simultaneously and a collective decision is obtained based on an aggregation strategy.

3.2.1 Bagging for semantic parsing

The bagging method (Breiman, 1996) is simply created K bootstrap with sampling m items from the training data of sentences and their logical forms with replacement. We then applied the SSVM learning in the K generated training data to create K semantic parser. In the testing phase, a given input sentence is parsed by K semantic parsers and their outputs are applied a switching model to obtain an output for the SSVM ensemble parser.

3.2.2 Boosting for semantic parsing

The representative boosting algorithm is the AdaBoost algorithm (Schapire, 1999). Each SSVM is trained using a different training set. Assuming that we have a training set $TR = (x_i; y_i)|i = 1, 2, ..., l$ consisting of l samples and each sample in the TR is assigned to have the same value of weight $p_0(x_i) = 1/l$. For training the kth SSVM classifier, we build a set of training samples

$TR_{boost_k} = (x_i; y_i)|i = 1, 2, .., l'$ that is obtained by selecting $l'(< l)$ samples among the whole data set TR according to the weight value $p_{k-1}(x_i)$ at the (k-1)th iteration. This training samples is used for training the kth SSVM classifier. Then, we obtained the updated weight values $p_k(x_i)$ of the training samples as follows. The weight values of the incorrectly classified samples are increased but the weight values of the correctly classified samples are decreased. This shows that the samples which are hard to classify are selected more frequently. These updated weight values will be used for building the training samples $TR_{boost_{k+1}} = (x_i; y_i)|i = 1, 2, ..., l'$ of the (k+1)th SSVM classifier. The sampling procedure will be repeated until K training samples set has been built for the Kth SSVM classifier.

3.2.3 The proposed SSVM ensemble model

We construct a SSVM ensemble model by using different parameters for each individual SSVM together with bagging and boosting models. The parameters we used here including the kernel function and the loss function as well as features used in a SSVM. Let N and K be the number of different parameters and individual semantic parsers in a SSVM ensemble, respectively. The motivation is to create individual parsers with respect to the fact that each individual parser performs well enough as well as they make different types of errors. We firstly create N ensemble models using either boosting or bagging models to obtain $N \times K$ individual parsers. We then select the top T parsers so that their errors on the training data are minimized and in different types. After forming an ensemble model of SSVMs, we need a process for aggregating outputs of individual SSVM classifiers. Intuitively, a simplest way is to use a voting method to select the output of a SSVM ensemble. Instead, we propose a switching method using subtrees mining from the set of trees as follows.

Let $t_1, t_2, ..., t_K$ be a set of candidate parse trees produced by an ensemble of K parsers. From the set of tree $t_1, t_2, ..., t_K$ we generated a set of training data that maps a tree to a label +1 or -1, where the tree t_j received the label +1 if it is an corrected output. Otherwise t_j received the label -1. We need to define a learning function for classifying a tree structure to two labels +1 and -1.

For this problem, we can apply a boosting technique presented in (Kudo and Matsumoto, 2004). The method is based on a generation of Adaboost (Schapire, 1999) in which subtrees mined from the training data are severed as weak decision stump functions.

The technique for mining these subtrees is presented in (Zaki, 2002) which is an efficient method for mining a large corpus of trees. Table 1 shows an example of mining subtrees on our corpus. One

Table 1: Subtrees mined from the corpus

Frequency	Subtree
20	(and(bowner)(bpos))
4	(and(bowner)(bpos(right)))
4	(bpos(circle(pt(playerour11))))
15	(and(bpos)(not(bpos)))
8	(and(bpos(penalty-areaour)))

problem for using the boosting subtrees algorithm (BT) in our switching models is that we might obtain several outputs with label +1. To solve this, we evaluate a score for each value +1 obtained by the BT and select the output with the highest score. In the case of there is no tree output received the value +1, the output of the first individual semantic parser will be the value of our switching model.

4 Experimental Results

For the purpose of testing our SSVM ensembles on semantic parsing, we used the CLANG corpus which is the RoboCup Coach Language (www.robocup.org). In the Coach Competition, teams of agents compete on a simulated soccer field and receive advice from a team coach in a formal language. The CLANG consists of 37 non-terminal and 133 productions; the corpus for CLANG includes 300 sentences and their structured representation in SAPT (Kate et al., 2005), then the logical form representations were built from the trees. Table 2 shows the statistic on the CLANG corpus.

Table 2: Statistics on CLANG corpus. The average length of an NL sentence in the CLANG corpus is 22.52 words. This indicates that CLANG is the hard corpus. The average length of the MRs is also large in the CLANG corpus.

Statistic	CLANG
No.of. Examples	300
Avg. NL sentence length	22.5
Avg. MR length (tokens)	13.42
No. of non-terminals	16
No. of productions	102
No. of unique NL tokens	337

Table 3: Training accuracy on CLANG corpus

Parameter	Training Accuracy
linear+F-loss(Δ_s)	83.9%
polynomial(d=2)+F-loss (Δ_m)	90.1%
polynomial(d=2)+F-loss(Δ_s)	98.8%
polynomial(d=2)+F-loss(Δ_m)	90.2%
RBF+F-loss(Δ_s)	86.3%

To create an ensemble learning with SSVM, we used the following parameters with the linear kernel, the polynomial kernel, and RBF kernel, respectively. Table 3 shows that they obtained different accuracies on the training corpus, and their accuracies are good enough to form a SSVM ensemble. The parameters in Table 3 is used to form our proposed SSVM model.

The following is the performance of the SSVM[1], the boosting model, the bagging model, and the models with different parameters on the

[1]The SSVM is obtained via http://svmlight.joachims.org/

CLANG corpus[2]. Note that the numbers of individual SSVMs in our ensemble models are set to 10 for boosting and bagging, and each individual SSVM can be used the zero-one and F1 loss function. In addition, we also compare the performance of the proposed ensemble SSVM models and the conventional ensemble models to assert that our models are more effective in forming SSVM ensemble learning.

We used the standard 10-fold cross validation test for evaluating the methods. To train a BT model for the switching phase in each fold test, we separated the training data into 10-folds. We keep 9/10 for forming a SSVM ensemble, and 1/10 for producing training data for the switching model. In addition, we mined a subset of subtrees in which a frequency of each subtree is greater than 2, and used them as weak functions for the boosting tree model. Note that in testing the whole training data in each fold is formed a SSVM ensemble model to use the BT model estimated above for selecting outputs obtained by the SSVM ensemble.

To evaluate the proposed methods in parsing NL sentences to logical form, we measured the number of test sentences that produced complete logical forms, and the number of those logical forms that were correct. For CLANG, a logical form is correct if it exactly matches the correct representation, up to reordering of the arguments of commutative operators. We used the evaluation method presented in (Kate et al., 2005) as the formula below.

$$precision = \frac{\#\text{correct}-\text{representation}}{\#\text{completed}-\text{representation}}$$
$$recall = \frac{\#\text{correct}-\text{representation}}{\#\text{sentences}}$$

Table 4 shows the results of SSVM, the SCSISSOR system (Ge and Mooney, 2005), and the SILT system (Kate et al., 2005) on the CLANG corpus, respectively. It shows that SCSISSOR obtained approximately 89% precision and 72.3% recall while on the same corpus our best single SSVM method [3] achieved a recall (74.3%) and lower precision (84.2%). The SILT system achieved approximately 83.9% precision and 51.3% recall [4] which is lower than the best single SSVM.

[2]We set N to 5 and K to 6 for the proposed SSVM.

[3]The parameter for SSVM is the **polynomial(d=2)+(Δ_s)**

[4]Those figures for precision and recall described in (Kate et al., 2005) showed approximately this precision and recall of their method in this paper

Table 4: Experiment results with CLANG corpus. Each SSVM ensemble consists of 10 individual SSVM. SSVM bagging and SSVM boosting used the voting method. P-SSVM boosting and P-SSVM bagging used the switching method (BT) and voting method (VT).

System	Methods	Precision	Recall
1	SSVM	84.2%	74.3%
1	SCSISSOR	89.0%	72.3%
1	SILT	83.9%	51.3%
10	SSVM Bagging	85.7%	72.4%
10	SSVM Boosting	85.7%	72.4%
10	**P-SSVM Boosting(BT)**	**88.4%**	**79.3%**
10	**P-SSVM Bagging(BT)**	**86.5%**	**79.3%**
10	P-SSVM Boosting(VT)	86.5%	75.8%
10	P-SSVM Bagging(VT)	84.6%	75.8%

Table 4 also shows the performance of Bagging, Boosting, and the proposed SSVM ensemble models with bagging and boosting models. It is important to note that the switching model using a boosting tree method (BT) to learn the outputs of individual SSVMs within the SSVM ensemble model.

It clearly indicates that our proposed ensemble method can enhance the performance of the SSVM model and the proposed methods are more effective than the conventional ensemble method for SSVM. This was because the output of each SSVM is complex (i.e a logical form) so it is not sure that the voting method can select a corrected output. In other words, the boosting tree algorithms can utilize subtrees mined from the corpus to estimate the good weight values for subtrees, and then combines them to determine whether or not a tree is selected. In our opinion, with the boosting tree algorithm we can have a chance to obtain more accurate outputs. These results in Table 4 effectively support for this evidence.

Moreover, Table 4 depicts that the proposed ensemble method using different parameters for either bagging and boosting models can effectively improve the performance of bagging and boosting in term of precision and recall. This was because the accuracy of each individual parser in the model with different parameters is better than each one in either the boosting or the bagging model. In addition, when performing SSVM on the test set, we might obtain some 'NULL' outputs since the grammar generated by SSVM could not derive this sentence. Forming a number of individual SSVMs to an ensemble model is the way to handle this case, but it could make the numbers of completed outputs and corrected outputs increase. Ta-

ble 4 indicates that the proposed SSVM ensemble model obtained 88.4% precision and 79.3% recall. Therefore it shows substantially a better F1 score in comparison with previous work on the CLANG corpus.

Summarily, our method achieved the best recall result and a high precision on CLANG corpus. The proposed ensemble models outperformed the original SSVM on CLANG corpus and its performances also is better than that of the best published result.

5 Conclusions

This paper presents a structured support vector machine ensemble model for semantic parsing problem by employing it on the corpus of sentences and their representation in logical form.

We also draw a novel SSVM ensemble model in which the forming ensemble strategy is based on a selection method on various parameters of SSVM, and the aggregation method is based on a switching model using subtrees mined from the outputs of a SSVM ensemble model.

Experimental results show substantially that the proposed ensemble model is better than the conventional ensemble models for SSVM. It can also effectively improve the performance in term of precision and recall in comparison with previous works.

Acknowledgments

The work on this paper was supported by a Monbukagakusho 21st COE Program.

References

J. Allen. 1995. Natural Language Understanding (2nd Edition). *Mento Park, CA: Benjaming/Cumming*.

L. Breiman. 1996. Bagging predictors. *Machine Learning* 24, 123-140.

T.G. Dietterich. 2000. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning 40* (2) 139-158.

M. Johnson 1999. PCFG models of linguistic tree representation. *Computational Linguistics.*

R. Ge and R.J. Mooney. 2005. A Statistical Semantic Parser that Integrates Syntax and Semanics. *In proceedings of CONLL 2005.*

J.C. Henderson and E. Brill 2000. Bagging and Boosting a Treebank Parser. *In proceedings ANLP 2000*: 34-41

R.J. Kate et al. 2005. Learning to Transform Natural to Formal Languages. *Proceedings of AAAI 2005*, page 825-830.

T. Kudo, Y. Matsumoto. A Boosting Algorithm for Classification of Semi-Structured Text. *In proceeding EMNLP 2004*.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *In Proc. of ICML 2001*.

D.C. Manning and H. Schutze. 1999. Foundation of Statistical Natural Language Processing. *Cambridge, MA: MIT Press.*

L.S. Zettlemoyer and M. Collins. 2005. Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars. *In Proceedings of UAI*, pages 825–830.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. 2004. Support Vector Machine Learning for Interdependent and Structured Output Spaces. *In proceedings ICML 2004.*

V. Vapnik. 1995. *The Nature of Statistical Learning Theory. Springer*, N.Y., 1995.

L.R. Tang. 2003. Integrating Top-down and Bottom-up Approaches in Inductive Logic Programming: Applications in Natural Language Processing and Relation Data Mining. *Ph.D. Dissertation*, University of Texas, Austin, TX, 2003.

B. Taskar, D. Klein, M. Collins, D. Koller, and C.D. Manning. 2004. Max-Margin Parsing. *In proceedings of EMNLP, 2004*.

R.E. Schapire. 1999. A brief introduction to boosting. *Proceedings of IJCAI 99*

M.J. Zaki. 2002. Efficiently Mining Frequent Trees in a Forest. *In proceedings 8th ACM SIGKDD 2002.*

J.M. Zelle and R.J. Mooney. 1996. Learning to parse database queries using inductive logic programming. *In Proceedings AAAI-96, 1050-1055.*

Whose thumb is it anyway? Classifying author personality from weblog text

Jon Oberlander
School of Informatics
University of Edinburgh
2 Buccleuch Place
Edinburgh, EH8 9LW
j.oberlander@ed.ac.uk

Scott Nowson
School of Informatics
University of Edinburgh
2 Buccleuch Place
Edinburgh, EH8 9LW
s.nowson@ed.ac.uk

Abstract

We report initial results on the relatively novel task of automatic classification of author personality. Using a corpus of personal weblogs, or 'blogs', we investigate the accuracy that can be achieved when classifying authors on four important personality traits. We explore both binary and multiple classification, using differing sets of n-gram features. Results are promising for all four traits examined.

1 Introduction

There is now considerable interest in affective language processing. Work focusses on analysing subjective features of text or speech, such as sentiment, opinion, emotion or point of view (Pang et al., 2002; Turney, 2002; Dave et al., 2003; Liu et al., 2003; Pang and Lee, 2005; Shanahan et al., 2005). Discussing affective computing in general, Picard (1997) notes that phenomena vary in duration, ranging from short-lived feelings, through emotions, to moods, and ultimately to long-lived, slowly-changing personality characteristics.

Within computational linguistics, most work has focussed on sentiment and opinion concerning specific entities or events, and on binary classifications of these. For instance, both Pang and Lee (2002) and Turney (2002) consider the thumbs up/thumbs down decision: is a film review positive or negative? However, Pang and Lee (2005) point out that ranking items or comparing reviews will benefit from finer-grained classifications, over multiple ordered classes: is a film review two- or three- or four-star? And at the same time, some work now considers longer-term affective states. For example, Mishne (2005) aims to classify the primary mood of weblog postings; the study encompasses both fine-grained (but non-ordered) multiple classification (frustrated/loved/etc.) and coarse-grained binary classification (active/passive, positive/negative).

This paper is about the move to finer-grained multiple classifications; and also about weblogs. But it is also about even more persistent affective states; in particular, it focusses on classifying *author personality*. We would argue that ongoing work on sentiment analysis or opinion-mining stands to benefit from progress on personality-classification. The reason is that people vary in personality, and they vary in how they appraise events—and hence, in how strongly they phrase their praise or condemnation. Reiter and Sripada (2004) suggest that lexical choice may sometimes be determined by a writer's idiolect—their personal language preferences. We suggest that while idiolect can be a matter of accident or experience, it may also reflect systematic, personality-based differences. This can help explain why, as Pang and Lee (2005) note, one person's four star review is another's two-star. To put it more bluntly, if you're not a very outgoing sort of person, then your thumbs up might be mistaken for someone else's thumbs down. But how do we distinguish such people? Or, if we spot a thumbs-up review, how can we tell whose thumb it is, anyway?

The paper is structured as follows. It introduces trait theories of personality, notes work to date on personality classification, and raises some questions. It then outlines the weblog corpus and the experiments, which compare classification accuracies for four personality dimensions, seven tasks, and five feature selection policies. We discuss the implications of the results, and related work, and end with suggestions for next steps.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 627–634,
Sydney, July 2006.

2 Background: traits and language

Cattell's pioneering work led to the isolation of 16 primary personality factors, and later work on secondary factors led to Costa and McCrae's five-factor model, closely related to the 'Big Five' models emerging from lexical research (Costa and McCrae, 1992). Each factor gives a continuous dimension for personality scoring. These are: Extraversion; Neuroticism; Openness; Agreeableness; and Conscientiousness (Matthews et al., 2003). Work has also investigated whether scores on these dimensions correlate with language use (Scherer, 1979; Dewaele and Furnham, 1999). Building on the earlier work of Gottschalk and Gleser, Pennebaker and colleagues secured significant results using the Linguistic Inquiry and Word Count text analysis program (Pennebaker et al., 2001). This primarily counts relative frequencies of word-stems in pre-defined semantic and syntactic categories. It shows, for instance, that high Neuroticism scorers use: more first person singular and negative emotion words; and fewer articles and positive emotion words (Pennebaker and King, 1999).

So, can a text classifier trained on such features predict the author personality? We know of only one published study: Argamon et al. (2005) focussed on Extraversion and Neuroticism, dividing Pennebaker and King's (1999) population into the top- and bottom-third scorers on a dimension, and discarding the middle third. For both dimensions, using a restricted feature set, they report binary classification accuracy of around 58%: an 8% absolute improvement over their baseline. Although mood is more malleable, work on it is also relevant (Mishne, 2005). Using a more typical feature set (including n-grams of words and parts-of-speech), the best mood classification accuracy was 66%, for 'confused'. At a coarser grain, moods could be classified with accuracies of 57% (active vs. passive), and 60% (positive vs. negative).

So, Argamon et al. used a restricted feature set for binary classification on two dimensions: Extraversion and Neuroticism. Given this, we now pursue three questions. (1) Can we improve performance on a similar binary classification task? (2) How accurate can classification be on the *other* dimensions? (3) How accurate can multiple—three-way or five-way—classification be?

3 The weblog corpus

3.1 Construction

A corpus of personal weblog ('blog') text has been gathered (Nowson, 2006). Participants were recruited directly via e-mail to suitable candidates, and indirectly by word-of-mouth: many participants wrote about the study in their blogs. Participants were first required to answer sociobiographic and personality questionnaires. The personality instrument has specifically been validated for online completion (Buchanan, 2001). It was derived from the 50-item IPIP implementation of Costa and McCrae's (1992) revised NEO personality inventory; participants rate themselves on 41-items using a 5-point Likert scale. This provides scores for Neuroticism, Extraversion, Openness, Agreeableness and Conscientiousness.

After completing this stage, participants were requested to submit one month's worth of prior weblog postings. The month was pre-specified so as to reduce the effects of an individual choosing what they considered their 'best' or 'preferred' month. Raw submissions were marked-up using XML so as to automate extraction of the desired text. Text was also marked-up by post type, such as purely personal, commentary reporting of external matters, or direct posting of internet memes such as quizzes. The corpus consisted of 71 participants (47 females, 24 males; average ages 27.8 and 29.4, respectively) and only the text marked as 'personal' from each weblog, approximately 410,000 words. To eliminate undue influence of particularly verbose individuals, the size of each weblog file was truncated at the mean word count plus 2 standard deviations.

3.2 Personality distribution

It might be thought that bloggers are more Extravert than most (because they express themselves in public); or perhaps that they are *less* Extravert (because they keep diaries in the first place). In fact, plotting the Extraversion scores for the corpus authors gives an apparently normal distribution; and the same applies for three other dimensions. However, scores for Openness to experience are not normally distributed. Perhaps bloggers are more Open than average; or perhaps there is response bias. Without a comparison sample of matched non-bloggers, one cannot say, and Openness is not discussed further in this paper.

4 Experiments

We are thus confined to classifying on four personality dimensions. However, a number of other variables remain: different learning algorithms can be employed; authors in the corpus can be grouped in several ways, leading to various classification tasks; and more or less restricted linguistic feature sets can be used as input to the classifier.

4.1 Algorithms

Support Vector Machines (SVM) appear to work well for binary sentiment classification tasks, so Argamon et al. (2003) and Pang and Lee (2005) consider One-vs-All, or All-vs-All, variants on SVM, to permit multiple classifications. Choice of algorithm is *not* our focus, but it remains to be seen whether SVM outperforms Naïve Bayes (NB) for personality classification. Thus, we will use both on the binary Tasks 1 to 3 (defined in section 4.2.1), for each of the personality dimensions, and each of the manually-selected feature sets (Levels I to IV, defined in section 4.3). Whichever performs better overall is then reported in full, and used for the multiple Tasks 4 to 7 (defined in section 4.2.2). Both approaches are applied as implemented in the WEKA toolkit (Witten and Frank, 1999) and use 10-fold cross validation.

4.2 Tasks

For any blog, we have available the scores, on continuous scales, of its author on four personality dimensions. But for the classifier, the task can be made more or less easy, by grouping authors on each of the dimensions. The simplest tasks are, of course, binary: given the sequence of words from a blog, the classifier simply has to decide whether the author is (for instance) high or low in Agreeableness. Binary tasks vary in difficulty, depending on whether authors scoring in the middle of a dimension are left out, or not; and if they are left out, what proportion of authors are left out.

More complex tasks will also vary in difficulty depending on who is left out. But in the cases considered here, middle authors are now included. For a three-way task, the classifier must decide if an author is high, medium or low; and those authors known to score between these categories may, or may not, be left out. In the most challenging five-way task, no-one is left out. The point of considering such tasks is to gradually approximate the most challenging task of all: continuous rating.

4.2.1 Binary classification tasks

In these task variants, the goal is to classify authors as either high or low scorers on a dimension:

1. The easiest approach is to keep the high and low groups as far apart as possible: high scorers (H) are those whose scores fall above 1 SD above the mean; low scorers (L) are those whose scores fall below 1 SD below the mean.

2. Task-1 creates distinct groups, at the price of excluding over 50% of the corpus from the analysis. To include more of the corpus, parameters are relaxed: the high group (HH) includes anyone whose score is above .5 SD above the mean; the low group (LL) is similarly placed below.

3. The most obvious task (but not the easiest) arises from dividing the corpus in half about the mean score. This creates high (HHH) and low (LLL) groups, covering the entire population. Inevitably, some HHH scorers will actually have scores much closer to those of LLL scorers than to other HHH scorers.

These sub-groups are tabulated in Table 1, giving the size of each group within each trait. Note that in Task-2, the standard-deviation-based divisions contain very nearly the top third and bottom third of the population for each dimension. Hence, Task-2 is closest in proportion to the division by thirds used in Argamon et al. (2005).

	Lowest	…	Highest
1	L	–	H
2	LL	–	HH
3	LLL		HHH
N1	12	–	13
N2	25	–	22
N3	39		32
E1	11	–	12
E2	23	–	24
E3	32		39
A1	11	–	13
A2	22	–	21
A3	34		37
C1	11	–	14
C2	17	–	27
C3	30		41

Table 1: Binary task groups: division method and author numbers. N = Neuroticism; E = Extraversion; A = Agreeableness; C = Conscientiousness.

4.2.2 Multiple classification tasks

4. Takes the greatest distinction between high (H) and low (L) groups from Task-1, and adds a medium group, but attempts to reduce the possibility of inter-group confusion by including only the smaller medium (m) group omitted from Task-2. Not all subjects are therefore included in this analysis. Since the three groups to be classified are completely distinct, this should be the easiest of the four multiple-class tasks.

5. Following Task-4, this uses the most distinct high (H) and low (L) groups, but now considers all remaining subjects medium (M).

6. Following Task-2, this uses the larger high (hH) and low (Ll) groups, with all those in between forming the medium (m) group.

7. Using the distinction between the high and low groups of Task-5 and -6, this creates a 5-way split: highest (H), relatively high (h), medium (m), relatively low (l) and lowest (L). With the greatest number of classes, this task is the hardest.

These sub-groups are tabulated in Table 2, giving the size of each group within each trait.

	Lowest		...		Highest
4	L	–	m	–	H
5	L		M		H
6	Ll		m	hH	
7	L	l	m	h	H
N4	12	–	24	–	13
N5	12		46		13
N6	25		24	22	
N7	12	13	24	9	13
E4	11	–	24	–	12
E5	11		48		12
E6	23		24	24	
E7	11	12	24	12	12
A4	11	–	28	–	13
A5	11		47		13
A6	22		28	21	
A7	11	11	28	8	13
C4	11	–	27	–	14
C5	11		46		14
C6	17		27	27	
C7	11	6	27	13	14

Table 2: 3-way/5-way task groups: division method and author numbers. N = Neuroticism; E = Extraversion; A = Agreeableness; C = Conscientiousness.

4.3 Feature selection

There are many possible features that can be used for automatic text classification. These experiments use essentially word-based bi- and trigrams. It should be noted, however, that some generalisations have been made: all proper nouns were identified via CLAWS tagging using the WMatrix tool (Rayson, 2003), and replaced with a single marker (NP1); punctuation was collapsed into a single marker (<*p*>); and additional tags correspond to non-linguistic features of blogs—for instance, <*SOP*> and <*EOP*> were used the mark the start and end of individual blogs posts. Word n-gram approaches provide a large feature space with which to work. But in the general interest of computational tractability, it is useful to reduce the size of the feature set. There are many automatic approaches to feature selection, exploiting, for instance, information gain (Quinlan, 1993). However, 'manual' methods can offer principled ways of both reducing the size of the set and avoiding overfitting. We therefore explore the effect of different levels of restriction on the feature sets, and compare them with automatic feature selection. The levels of restriction are as follows:

I The least restricted feature set consists of the n-grams most commonly occurring within the blog corpus. Therefore, the feature set for each personality dimension is to be drawn from the same pool. The difference lies in the number of features selected: the size of the set will match that of the next level of restriction.

II The next set includes only those n-grams which were distinctive for the two extremes of each personality trait. Only features with a corpus frequency $\geq$5 are included. This allows accurate log-likelihood G^2 statistics to be computed (Rayson, 2003). Distinct collocations are identified via a three way comparison between the H and L groups in Task-1 (see section 4.2.1) and a third, neutral group. This neutral group contains all those individuals who fell in the medium group (M) for *all four traits in the study*; the resulting group was of comparable size to the H and L groups for each trait. Hence, this approach selects features using only a *subset* of the corpus. N-gram software was used to identify and count collocations within a sub-corpus (Banerjee

and Pedersen, 2003). For each feature found, its frequency and relative frequency are calculated. This permits relative frequency ratios and log-likelihood comparisons to be made between High-Low, High-Neutral and Low-Neutral. Only features that prove distinctive for the H or L groups with a significance of $p < .01$ are included in the feature set.

III The next set takes into account the possibility that, for a group used in Level-II, an n-gram may be used relatively frequently, but only because a small number of authors in a group use it very frequently, while others in the same group use it not at all. To enter the Level-III set, an n-gram meeting the Level-II criteria must also be used by at least 50%[1] of the individuals within the subgroup for which it is reported to be distinctive.

IV While Level-III guards against excessive individual influence, it may abstract too far from the fine-grained variation *within* a personality trait. The final manual set therefore includes only those n-grams that meet the Level-II criteria with $p < .001$, meet the Level-III criteria, and also correlate significantly ($p < .05$) with individual personality trait scores.

V Finally, it is possible to allow the n-gram feature set to be selected automatically during training. The set to be selected from is the broadest of the manually filtered sets, those n-grams that meet the Level-II criteria. The approach adopted is to use the defaults within the WEKA toolkit: Best First search with the CfsSubsetEval evaluator (Witten and Frank, 1999).

Thus, a key question is when—if ever—a 'manual' feature selection policy outperforms the automatic selection carried out under Level-V. Levels-II and -III are of particular interest, since they contain features derived from a subset of the corpus. Since different sub-groups are considered for each personality trait, the feature sets which meet the increasingly stringent criteria vary in size. Table 3 contains the size of each of the four manually-determined feature sets for each of the four personality traits. Note again that the number of n-grams selected from the most frequent in the corpus for Level-I matches the size of the set for Level-II. In addition, the features automatically selected are task-dependent, so the Level-V sets vary in size; here, the Table shows the number of features selected for Task-2.

[1]Conservatively rounded down in the case of an odd number of subjects.

	I	II	III	IV	V
N	747	747	169	22	19
E	701	701	167	11	20
A	823	823	237	36	34
C	704	704	197	22	25

Table 3: Number of n-grams per set.

	Low	High
N	*[was that]* *[NP1 <p> NP1]* *[<p> after]* *[is that]*	*[this year]* *[to eat]* *[slowly <p>]* *[and buy]*
E	*[point in]* *[last night <p>]* *[it the]* *[is to]*	*[and he]* *[cool <p>]* *[<p> NP1]* *[to her]*
A	*[thank god]* *[have any]* *[have to]* *[turn up]*	*[this is not]* *[<p> it is]* *[<p> after]* *[not have]*
C	*[a few weeks]* *[case <p>]* *[okay <p>]* *[the game]*	*[by the way]* *[<p> i hope]* *[how i]* *[kind of]*

Table 4: Examples of significant Low and High n-grams from the Level-IV set.

To illustrate the types of n-grams in the feature sets, Table 4 contains four of the most significant n-grams from Level-IV for each personality class.

5 Results

For each of the 60 binary classification tasks (1 to 3), the performance of the two approaches was compared. Naïve Bayes outperformed Support Vector Machines on 41/60, with 14 wins for SVM and 5 draws. With limited space available, we therefore discuss only the results for NB, and use NB for Task-4 to -7. The results for the binary tasks are displayed in Table 5. Those for the multiple tasks are displayed in Table 6. Baseline is the majority classification. The most accurate performance of a feature set for each task is highlighted

Task	Base	Lv.I	Lv.II	Lv.III	Lv.IV	Lv.V
N1	52.0	52.0	*92.0*	84.0	**96.0**	*92.0*
N2	53.2	51.1	63.8	68.1	*83.6*	**85.1**
N3	54.9	54.9	60.6	53.5	*71.9*	**83.1**
E1	52.2	56.5	91.3	*95.7*	87.0	**100.0**
E2	51.1	44.7	*74.5*	72.3	66.0	**93.6**
E3	54.9	50.7	53.5	59.2	*64.8*	**85.9**
A1	54.2	62.5	**100.0**	**100.0**	*95.8*	**100.0**
A2	51.2	60.5	*81.4*	79.1	72.1	**97.7**
A3	52.1	53.5	60.6	*69.0*	66.2	**93.0**
C1	56.0	52.0	**100.0**	**100.0**	84.0	*92.0*
C2	61.2	54.5	77.3	*81.8*	72.7	**93.2**
C3	57.7	54.9	63.4	*71.8*	70.4	**84.5**

Table 5: Naïve Bayes performance on binary tasks. Raw % accuracy for 4 personality dimensions, 3 tasks, and 5 feature selection policies.

Task	Base	Lv.I	Lv.II	Lv.III	Lv.IV	Lv.V
N4	49.0	49.0	*81.6*	65.3	77.6	**85.7**
N5	64.8	60.6	*76.1*	67.6	67.6	**94.4**
N6	35.2	31.0	47.9	46.5	*66.2*	**70.4**
N7	33.8	31.0	**49.3**	38.0	42.3	47.9
E4	51.1	44.7	*74.5*	59.6	53.2	**78.7**
E5	67.6	60.6	*83.1*	67.6	54.9	**90.1**
E6	33.8	23.9	*53.5*	46.5	46.5	**56.3**
E7	33.8	**44.7**	39.4	29.6	38.0	40.8
A4	53.8	51.9	**90.4**	78.8	67.3	80.8
A5	66.2	59.2	*83.1*	**84.5**	74.6	80.3
A6	39.4	31.0	*67.6*	60.6	56.3	**85.9**
A7	39.4	33.8	**69.8**	*60.6*	50.7	47.9
C4	51.9	53.8	**92.3**	65.4	67.3	82.7
C5	64.8	62.0	*74.6*	69.0	62.0	**83.1**
C6	38.0	39.4	*59.2*	*59.2*	50.7	**78.9**
C7	38.0	36.6	**62.0**	45.1	45.1	49.3

Table 6: Naïve Bayes performance on multiple tasks. Raw % accuracy for 4 personality dimensions, 4 tasks, and 5 feature selection policies.

in **bold** while the second most accurate is marked *italic*.

6 Discussion

Let us consider the results as they bear in turn on the three main questions posed earlier: Can we improve on Argamon et al.'s (2005) performance on binary classification for the Extraversion and Neuroticism dimensions? How accurately can we classify on the four personality dimensions? And how does performance on multiple classification compare with that on binary classification?

Before addressing these questions, we note the relatively good performance of NB compared with 'vanilla' SVM on the binary classification tasks. We also note that automatic selection generally outperforms 'manual' selection; however overfitting is very likely when examining just 71 data points. Therefore, we do not discuss the Level-V results further.

6.1 Extraversion and Neuroticism

The first main question relates to the feature sets chosen, because the main issue is whether word n-grams can give reasonable results on the Extraversion and Neuroticism classification tasks. Of the current binary classification tasks, Task-2 is most closely comparable to Argamon et al.'s. Here, the best performance for Extraversion was returned by the 'manual' Level-II feature set, closely followed by Level-III. The accuracy of 74.5% represents a 23.4% absolute improvement over baseline (45.8% relative improvement; we report relative improvement over baseline because baseline accuracies vary between tasks). The best performance for Neuroticism was returned by Level-IV. The accuracy of 83.6% represents a 30.4% absolute improvement over baseline (57.1% relative improvement).

Argamon et al.'s feature set combined insights from computational stylometrics (Koppel et al., 2002; Argamon et al., 2003) and systemic-functional grammar. Their focus on function words and appraisal-related features was intended to provide more general and informative features than the usual n-grams. Now, it is unlikely that weblogs are easier to categorise than the genres studied by Argamon et al. So there are instead at least two reasons for the improvement we report.

First, although we did not use systemic-functional linguistic features, we did test n-grams selected according to more or less strict policies. So, considering the manual policies, it seems that the Level-IV was the best-performing set for Neuroticism. This might be expected, given that Level-IV potentially overfits, allowing features to be derived from the full corpus. However, in spite of this, Level-II pproved best for Extraversion. Secondly, in classifying an individual as high or low on some dimension, Argamon et al. had

(for some of their materials) 500 words from that individual, whereas we had approximately 5000 words. The availability of more words per individual is to likely to help greatly in training. Additionally, a greater volume of text increases the chances that a long term 'property' such as personality will emerge

6.2 Binary classification of all dimensions

The second question concerns the relative ease of classifying the different dimensions. Across each of Task-1 to -3, we find that classification accuracies for Agreeableness and Conscientiousness tend to be higher than those for Extraversion and Neuroticism. In all but two cases, the automatically generated feature set (V) performs best. Putting this to one side, of the manually constructed sets, the unrestricted set (I) performs worst, often below the baseline, while Level-IV is the best for classifying each task of Neuroticism. Overall, II and III are better than IV, although the difference is not large.

As tasks increase in difficulty—as high and low groups become closer together, and the left-out middle shrinks—performance drops. But accuracy is still respectable.

6.3 Beyond binary classification

The final question is about how classification accuracy suffers as the classification task becomes more subtle. As expected, we find that as we add more categories, the tasks are harder: compare the results in the Tables for Task-1, -5 and -7. And, as with the binary tasks, if fewer mid-scoring individuals are left out, the task is typically harder: compare results for Task-4 and 5. It does seem that some personality dimensions respond to task difficulty more robustly than others. For instance, on the hardest task, the best Extraversion classification accuracy is 10.9% absolute over the baseline (32.2% relative), while the best Agreeableness accuracy is 30.4% absolute over the baseline (77.2% relative). It is notable that the feature set which return the best results—bar the automatic set V—tends to be Level-II, excepting for Neuroticism on Task-6, where Level-IV considerably outperforms the other sets.

A supplementary question is how the best classifiers compare with human performance on this task. Mishne (2005) reports that, for general mood classification on weblogs, the accuracy of his automatic classifier is comparable to human performance. There are also general results on human personality classification performance in computer-mediated communication, which suggest that at least some dimensions can be accurately judged even when computer-mediated. Vazire and Gosling (2004) report that for personal websites, relative accuracy of judgment was, in descending order: Openness $>$ Extraversion $>$ Neuroticism $>$ Agreeableness $>$ Conscientiousness. Similarly, Gill et al. (2006) report that for personal e-mail, Extraversion is more accurately judged than Neuroticism. The current study does not have a set of human judgments to report. For now, it is interesting to note that the performance profile for the best classifiers, on the simplest tasks, appears to diverge from the general human profile, instead ranking on raw accuracy: Agreeableness $>$ Conscientiousness $>$ Neuroticism $>$ Extraversion.

7 Conclusion and next steps

This paper has reported the first stages of our investigations into classification of author personality from weblog text. Results are quite promising, and comparable across all four personality traits. It seems that even a small selection of features found to exhibit an empirical relationship with personality traits can be used to generate reasonably accurate classification results. Naturally, there are still many paths to explore. Simple regression analyses are reported in Nowson (2006); however, for classification, a more thorough comparison of different machine learning methodologies is required. A richer set of features besides n-grams should be checked, and we should not ignore the potential effectiveness of unigrams in this task (Pang et al., 2002). A completely new test set can be gathered, so as to further guard against overfitting, and to explore systematically the effects of the amount of training data available for each author. And as just discussed, comparison with human personality classification accuracy is potentially very interesting.

However, it does seem that we are making progress towards being able to deal with a realistic task: if we spot a thumbs-up review in a weblog, we should be able to check other text in that weblog, and tell whose thumb it is; or more accurately, what *kind* of person's thumb it is, anyway. And that in turn should help tell us how high the thumb is really being held.

8 Acknowledgements

We are grateful for the helpful advice of Mirella Lapata, and our three anonymous reviewers. The second author was supported by a studentship from the Economic and Social Research Council.

References

Shlomo Argamon, Marin Saric, and Sterling S. Stein. 2003. Style mining of electronic messages for multiple authorship discrimination: first results. In *Proceedings of SIGKDD*, pages 475–480.

Shlomo Argamon, Sushant Dhawle, Moshe Koppel, and James W. Pennebaker. 2005. Lexical predictors of personality type. In *Proceedings of the 2005 Joint Annual Meeting of the Interface and the Classification Society of North America*.

Satanjeev Banerjee and Ted Pedersen. 2003. The design, implementation, and use of the ngram statistics package. In *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*, pages 370–381, Mexico City.

Tom Buchanan. 2001. Online implementation of an IPIP five factor personality inventory [web page]. http://users.wmin.ac.uk/∼buchant/wwwffi/introduction.html [Accessed 25/10/05].

Paul T. Costa and Robert R. McCrae, 1992. *Revised NEO Personality Inventory (NEO-PI-R) and NEO Five-Factor Inventory (NEO-FFI): Professional Manual*. Odessa, FL: Psychological Assessment Resources.

Kushal Dave, Steve Lawrence, and David M. Pennock. 2003. Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *Proceedings of the 12th International Conference on World Wide Web*, pages 519–528. ACM Press.

Jean-Marc Dewaele and Adrian Furnham. 1999. Extraversion: The unloved variable in applied linguistic research. *Language Learning*, 49:509–544.

Alastair J. Gill, Jon Oberlander, and Elizabeth Austin. 2006. Rating e-mail personality at zero acquaintance. *Personality and Individual Differences*, 40:497–507.

Moshe Koppel, Shlomo Argamon, and Arat Shimoni. 2002. Automatically categorizing written texts by author gender. *Literary and Linguistic Computing*, 17(4):401–412.

Hugo Liu, Henry Lieberman, and Ted Selker. 2003. A model of textual affect sensing using real-world knowledge. In *Proceedings of the 7th International Conference on Intelligent User Interfaces*.

Gerald Matthews, Ian J. Deary, and Martha C. Whiteman. 2003. *Personality Traits*. Cambridge University Press, Cambridge, 2nd edition.

Gilad Mishne. 2005. Experiments with mood classification in blog posts. In *Proceedings of ACM SIGIR 2005 Workshop on Stylistic Analysis of Text for Information Access*.

Scott Nowson. 2006. *The Language of Weblogs: A study of genre and individual differences*. Ph.D. thesis, University of Edinburgh.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the ACL*, pages 115–124.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 79–86.

James W. Pennebaker and Laura King. 1999. Linguistic styles: Language use as an individual difference. *Journal of Personality and Social Psychology*, 77:1296–1312.

James W. Pennebaker, Martha E. Francis, and Roger J. Booth. 2001. *Linguistic Inquiry and Word Count 2001*. Lawrence Erlbaum Associates, Mahwah, NJ.

Rosalind W. Picard. 1997. *Affective Computing*. MIT Press, Cambridge, Ma.

J. Ross Quinlan. 1993. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Paul Rayson. 2003. *Wmatrix: A statistical method and software tool for linguistic analysis through corpus comparison*. Ph.D. thesis, Lancaster University.

Ehud Reiter and Somayajulu Sripada. 2004. Contextual influences on near-synonym choice. In *Proceedings of the Third International Conference on Natural Language Generation*.

Klaus Scherer. 1979. Personality markers in speech. In K. R. Scherer and H. Giles, editors, *Social Markers in Speech*, pages 147–209. Cambridge University Press, Cambridge.

James G. Shanahan, Yan Qu, and Janyce Weibe, editors. 2005. *Computing Attitude and Affect in Text*. Springer, Dordrecht, Netherlands.

Peter D. Turney. 2002. Thumbs up or thumbs down? semantic orientation applied to unspervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the ACL*, pages 417–424.

Simine Vazire and Sam D. Gosling. 2004. e-perceptions: Personality impressions based on personal websites. *Journal of Personality and Social Psychology*, 87:123–132.

Ian H. Witten and Eibe Frank. 1999. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann.

Analysis of Selective Strategies to Build a Dependency-Analyzed Corpus

Kiyonori Ohtake
National Institute of Information and Communications Technology (NICT),
ATR Spoken Language Communication Research Labs.
2-2-2 Hikaridai "Keihanna Science City" Kyoto 619-0288 Japan
kiyonori.ohtake [at] nict.go.jp

Abstract

This paper discusses sampling strategies for building a dependency-analyzed corpus and analyzes them with different kinds of corpora. We used the Kyoto Text Corpus, a dependency-analyzed corpus of newspaper articles, and prepared the IPAL corpus, a dependency-analyzed corpus of example sentences in dictionaries, as a new and different kind of corpus. The experimental results revealed that the length of the test set controlled the accuracy and that the longest-first strategy was good for an expanding corpus, but this was not the case when constructing a corpus from scratch.

1 Introduction

Dependency-structure analysis plays a very important role in natural language processing (NLP). Thus, so far, much research has been done on this subject, with many analyzers being developed such as rule-based analyzers and corpus-based analyzers that use machine-learning techniques. However, the maximum accuracy achieved by state-of-the art analyzers is almost 90% for newspaper articles; it seems very difficult to exceed this figure of 90%. To improve our analyzers, we have to write more rules for rule-based analyzers or prepare more corpora for corpus-based analyzers.

If we take a machine-learning approach, it is important to consider what features are used. However, there are several machine-learning techniques, such as support vector machines (SVMs) with a kernel function, that have strong generalization ability and are very robust for choosing the right features. If we use such machine-learning techniques, we will be free from choosing a feature set because it will be possible to use all possible features with little or no decline in performance. Actually, Sasano tried to expand the feature set for a Japanese dependency analyzer using SVMs in (Sasano, 2004), with a small improvement in accuracy.

To write rules for a rule-based analyzer, and to produce an analyzer using machine-learning techniques, it is crucial to construct a dependency-analyzed corpus. Such a corpus is very useful not only for constructing a dependency analyzer but also for other natural language processing applications. However, building this kind of resource is very expensive and labor-intensive because it is difficult to annotate a large amount of dependency-analyzed corpus in short time.

At present, one promising approach to mitigating the annotation bottleneck problem is to use selective sampling, a variant of active learning (Cohn et al., 1994; Fujii et al., 1998; Hwa, 2004). In general, selective sampling is an interactive learning method in which the machine takes the initiative in selecting unlabeled data for the human to annotate. Under this framework, the system has access to a large pool of unlabeled data, and it has to predict how much it can learn from each candidate in the pool if that candidate is labeled.

Most of the experiments that had been carried out in the previous works for selective sampling used an annotated corpus in a limited domain. The most typical corpus is WSJ of Penn Treebank. The reason why the domain was so limited is very simple; corpus annotation is very expensive. However, we want to know the effects of selective sampling for corpora in various domains because a dependency analyzer constructed from a corpus does not always analyze a text in limited domain.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 635–642,
Sydney, July 2006.

On the other hand, there is no clear guideline nor development strategy for constructing a dependency-analyzed corpus to produce a highly accurate dependency analyzer. Thus in this paper, we discuss fundamental sampling strategies for a dependency-analyzed corpus for corpus-based dependency analyzers with several types of corpora. This paper unveils the essential characteristics of basic sampling strategies for a dependency-analyzed corpus.

2 Dependency-Analyzed Corpora

We use two dependency-analyzed corpora. One is the Kyoto Text Corpus, which consists of newspaper articles, and the other one is the IPAL corpus, which contains sentences extracted from the "example of use" section of the enties in several dictionaries for computers. The IPAL corpus was recently annotated for this study as a different kind of corpus.

2.1 Kyoto Text Corpus

In this study we use Kyoto Text Corpus version 3.0. The corpus consists of newspaper articles from Mainichi Newspapers from January 1st to January 17th, 1995 (almost 20,000 sentences) and all editorials of the year 1995 (almost 20,000 sentences). All of the articles were analyzed by morphological analyzer JUMAN and dependency analyzer KNP[1]. After that, the analyzed results were manually corrected. Kyoto Text Corpus version 4.0 is now available, holding on additional 5,000 annotated sentences in the corpus to version 3.0 for case relations, anaphoric relations, omission information and co-reference information[2].

The original POS system used in the Kyoto Text Corpus is JUMAN's POS system. We converted the POS system used in the Kyoto Text Corpus into ChaSen's POS system because we used ChaSen, a Japanese morphological analyzer, and CaboCha[3] (Kudo and Matsumoto, 2002), a dependency analyzer incorporating SVMs, as a state-of-the art corpus-based Japanese dependency structure analyzer that prefers ChaSen's POS system to that of JUMAN. In addition, we modified some bunsetu segmentations because there were several inconsistencies in bunsetu segmentation.

Table 1 shows the details of the Kyoto Text Corpus.

	Kyoto Text Corpus	
	(General)	(Editorial)
# of sentences	19,669	18,714
# of bunsetu	192,154	171,461
# of morphemes	542,334	480,005
vocabulary size	29,542	17,730
bunsetu / sentence	9.769	9.162

Table 1: Kyoto Text Corpus

2.2 IPAL corpus

IPAL (IPA, Information-technology Promotion Agency, Lexicon of the Japanese language for computers) dictionaries consist of three dictionaries, the IPAL noun dictionary, the IPAL verb dictionary and the IPAL adjective dictionary. Each of the dictionaries includes example sentences. We extracted 7,720 sentences from IPAL Noun, 5,244 sentences from IPAL Verb, and 2,366 sentences from IPAL Adjective. We analyzed them using CaboCha and manually corrected the errors. We named this dependency-analyzed corpus the IPAL corpus. Table 2 presents the details of the IPAL corpus. One characteristic of the IPAL corpus is that the average sentence length is very short; in other words, the sentences in the IPAL corpus are very simple.

# of sentences	15,330
# of bunsetu	67,170
# of morphemes	156,131
vocabulary size	11,895
bunsetu / sentence	4.382

Table 2: IPAL corpus

3 Experiments

We carried out several experiments to determine the basic characteristics of several selective strategies for a Japanese dependency-analyzed corpus. First, we briefly introduce Japanese dependency structure. Second, we carry out basic experiments with our dependency-analyzed corpora and analyze the errors. Finally, we conduct simulations to

[1] http://www.kc.t.u-tokyo.ac.jp/nl-resource

[2] http://www.kc.t.u-tokyo.ac.jp/nl-resource/corpus.html

[3] http://chasen.org/~taku/software/cabocha/

ascertain the fundamental characteristics of these strategies.

3.1 Japanese dependency structure

The Japanese dependency structure is usually defined in terms of the relationship between phrasal units called *bunsetu* segments. Conventional methods of dependency analysis have assumed the following three syntactic constraints (Kurohashi and Nagao, 1994a):

1. All dependencies are directed from left to right.
2. Dependencies do not cross each other.
3. Each *bunsetu* segment, except the last one, depends on only one *bunsetu* segment.

Figure 1 shows examples of Japanese dependency structure.

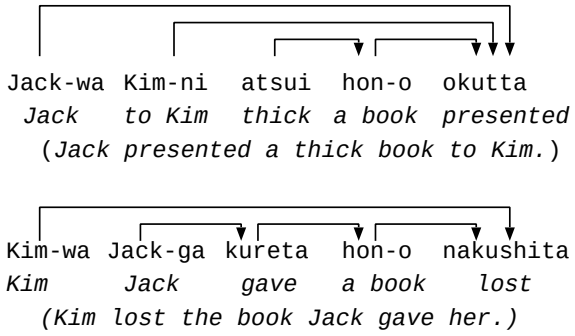


Figure 1: Examples of Japanese dependency structure

In this paper, we refer to the beginning of a dependency direction as a "modifier" and the end of that as a "head."

3.2 Analyzing errors

We performed a cross-validation test with our dependency-analyzed corpora by using the SVM-based dependency analyzer CaboCha. The feature set used for SVM in CaboCha followed the default settings of CaboCha.

First, we arbitrarily divided each corpus into two parts. General articles of the Kyoto Text Corpus were arbitrarily divided into KG0 and KG1, while editorials were also divided into ED0 and ED1. The IPAL corpus was arbitrarily divided into IPAL0 and IPAL1. Second, we carried out cross-validation tests on these divided corpora.

Table 3 shows the results of the cross-validation tests. We employed a polynomial kernel for the SVM of CaboCha, and tested with second- and third-degree polynomial kernels. The input data for each test were correct for morphological analysis and bunsetu segmentation, though in practical situations we have to expect some morphological analysis errors and bunsetu mis-segmentations.

In Table 3 "Learning" indicates the learning corpus, "Test" represents the test corpus, and "Degree" denotes the degree of the polynomial function. In addition, "Acc." indicates the accuracy of dependency-analyzed results and "S-acc." indicates the sentence accuracy that is the ratio of sentences that were analyzed without errors.

Learning	Test	Degree	Acc.(%)	S-acc.(%)
KG0	KG0	2	94.06	65.51
KG0	KG0	3	99.96	99.71
KG0	KG1	2	89.50	50.35
KG0	KG1	3	89.23	49.33
KG1	KG0	2	89.60	49.89
KG1	KG0	3	89.21	49.05
ED0	ED1	2	90.77	55.58
ED1	ED0	2	90.52	54.62
IPAL0	IPAL1	2	97.43	92.25
IPAL1	IPAL0	2	97.69	93.06
KG0	IPAL0	2	97.76	93.15
ED0	IPAL0	2	97.56	92.81

Table 3: Results of cross-validation tests

Table 3 also shows the biased evaluation (closed test; the test was the training set itself) results. In the cross-validation results of KG0 and KG1, the average accuracy of the second-degree kernel was 89.55 (154,455 / 172,485)% and the average sentence accuracy was 50.12 (9,858 / 19,669)%. In other words, there were 18,030 dependency errors in the cross validation test. We analyzed these errors.

Against the average length (9.769) of the corpus shown in Table 1, the average length of the sentences with errors in the cross-validation test is 12.53 (bunsetu / sentence). These results confirm that longer sentences tend to be analyzed incorrectly.

Next we analyzed modifier bunsetu that were mis-analyzed. Table 4 shows the top ten POS sequences that consisted of modifier mis-analyzed bunsetu.

We also analyzed the distance between modifier bunsetu and head bunsetu of the mis-analyzed dependencies. Table 5 shows top ten cases of the distance. In Table 5 "Err." indicates the distance between a modifier and a head bunsetu of mis-analyzed dependencies, "Correct" indicates

POS sequence	Frequency
noun, case marker	835
verb, comma	576
noun, topic marker	444
adverbial noun, comma	370
verb	336
number, numeral classifier, comma	318
noun, adnominal particle	304
adverb	304
verb, verbal auxiliary	281
verb, conjunctive particle, comma	265

Table 4: Modifier POS sequences of mis-analyzed dependencies and their frequencies in the cross-validation test (top 10)

the distance between a modifier and a correct (should modify) head bunsetu in each case of mis-analyzed dependencies, and "Freq." denotes their frequency.

Err.	Correct	Freq.	Err.	Correct	Freq.
1	2	3,117	2	4	478
2	1	1,362	3	2	436
3	1	919	4	1	434
1	3	863	4	2	379
2	3	482	1	4	329

Table 5: Frequencies of dependency distances at error and correct cases in the cross-validation test (top 10)

3.3 Selective sampling simulation

In this section, we discuss selective strategies through two simulations. One is expanding a dependency-analyzed corpus to construct a more accurate dependency analyzer, and the other is an initial situation just beginning to build a corpus.

3.3.1 Expanding situation

The situation is as follows. First, the corpus, Kyoto Text Corpus KG1, is given. Second, we expand the corpus using the editorials component of the Kyoto Text Corpus. Then we consider the following six strategies: (1) Longest first, (2) Maximizing vocabulary size first, (3) Maximizing unseen dependencies first, (4) Maximizing average distance of dependencies first, (5) Chronological order, and (6) Random.

We briefly introduce these six strategies as follows:

1. Longest first (Long)
   Since longer sentences tend to have complex structures and be analyzed incorrectly, we prepare the corpus in descending order of length. The length is measured by the number of bunsetu in a sentence.

2. Maximizing vocabulary size first (VSort)
   Unknown words cause unknown dependencies, thus we sort the corpus to maximize its vocabulary size.

3. Maximizing unseen dependencies first (UDep)
   This is similar to (2). However, we cannot know the true dependencies. The analyzed results by the dependency analyzer based on the current corpus are used to estimate the unseen dependencies. The accuracy of the estimated results was 90.25% and the sentence accuracy was 54.03%.

4. Maximizing average distance of dependencies first (ADist)
   It is difficult to analyze long-distance dependencies correctly. Thus, the average distance of dependencies is an approximation for the difficulty of analysis.

5. Chronological order (Chrono)
   Since there is a chronological order in newspaper articles, this strategy should feel quite natural.

6. Random (ED0)
   Chronological order seems natural, but newspaper articles also have cohesion. Thus, the vocabulary might be unbalanced when we consider the chronological order. We also try randomized order; actually, we used the corpus ED0 as the randomized corpus.

We sorted the editorial component of the Kyoto Text Corpus by each strategy mentioned above. After sorting, corpora were constructed by taking the top N sentences of each corpus sorted by each strategy. The size of each corpus was balanced with the number dependencies.

We constructed dependency analyzers based on each corpus, KG1 plus each prepared corpus, then tested them by using the following corpora: (a) K-mag, (b) IPAL0, and (c) KG0.

Corpus	# of sent.	# of bunsetu	vocabulary size	# of dependencies	# of bunsetu / sent.
Long	5,490	81,759	13,266	76,269	14.89
VSort	8,762	85,031	16,428	76,269	9.705
UDep	5,524	81,793	13,371	76,269	14.81
ADist	6,950	83,223	13,074	76,273	11.97
Chrono	9,342	85,609	13,278	76,267	9.164
ED0	9,357	85,628	13,561	76,271	9.151
K-mag	489	4,851	2,501	4,362	9.920
IPAL0	7,665	33,484	8,617	25,819	4.368
KG0	9,835	96,283	21,616	86,448	9.790
I-Long	5,523	91,972	20,068	86,449	16.65
I-VSort	8,437	94,881	28,867	86,444	11.25

Table 6: Detailed information of corpora

K-mag consists of articles from the Koizumi Cabinet's E-Mail Magazine. This magazine was first published on May 29th 1999 and is still released weekly. K-mag consists of articles of the magazine published from May 29th 1999 to July 19th 1999. In addition, since March 25th 2004 an English version of this E-Mail Magazine has been available. Thus, currently this E-mail Magazine is bilingual. The articles of this magazine were analyzed by the dependency analyzer CaboCha, and we manually corrected the errors.

K-mag includes a wide variety articles, and the average sentence length is longer than in newspapers. Basic information on K-mag is also provided in Table 6.

Learning corpus	Acc.(%)	S-acc.(%)
KG1	87.25	49.69
KG1+LONG	**87.67**	**51.53**
KG1+Vsort	87.25	50.10
KG1+UDep	87.57	51.12
KG1+ADist	**87.67**	50.72
KG1+Chrono	87.57	50.31
KG1+Rand	87.60	49.69

Table 7: Analyzed results of K-mag (which is different domain and has long average sentence length) with these learning corpora

3.3.2 Simulation for initial situation

The results revealed that the longest-first strategy seems the best way. Here, however, a question arises: "Does the longest-first strategy always provide good predictions?" We carried out an experiment to answer the question. The experimental

Learning corpus	Acc. (%)	S-acc.(%)
KG1	97.68	93.02
KG1+LONG	**97.75**	**93.22**
KG1+Vsort	97.70	93.06
KG1+UDep	**97.75**	93.18
KG1+ADist	97.70	93.10
KG1+Chrono	97.71	93.06
KG1+Rand	97.69	93.06

Table 8: Analyzed results of IPAL0 (which is different domain and has short average sentence length) with these learning corpora

results we presented above were simulations of an expanding corpus. On the other hand, it is also possible to consider an initial situation for building a dependency-analyzed corpus. In such a situation, which would be the best strategy to take?

We carried out a simulation experiment in which there was no annotated corpus; instead we began to construct a new one. We used general articles from the Kyoto Text Corpus and tried the following three strategies: (a) Random (actually, KG0 was used), (b) Longest first (I-Long), and (c) maximizing vocabulary size first (I-VSort). Three corpora were prepared by these strategies. Table 6 also shows the corpora information. In this experiment, the corpora were balanced with respect to the number of dependencies. We used CaboCha with these corpora and tested them with K-mag, ED0, and IPAL0. Table 10 shows the results of the experiment.

	K-mag		ED0		IPAL0	
Corpus	Acc. (%)	S-acc. (%)	Acc. (%)	S-acc. (%)	Acc. (%)	s-acc(%)
Random (KG0)	87.87	49.69	**90.17**	53.64	**97.76**	**93.15**
I-Long	87.41	49.28	90.11	52.96	97.66	92.94
I-VSort	**87.92**	**50.31**	90.14	**53.86**	97.72	93.06

Table 10: Results of initial situation experiment

Learning corpus	Acc. (%)	S-acc. (%)
KG1	89.60	49.89
KG1+LONG	**89.99**	51.25
KG1+Vsort	89.97	51.31
KG1+UDep	89.98	**51.39**
KG1+ADist	89.98	51.01
KG1+Chrono	89.86	51.09
KG1+Rand	89.95	51.20

Table 9: Analyzed results of KG0 (which is the same domain and has almost the same average sentence length) with these learning corpora

4 Discussion

4.1 Error analysis

To analyze corpora, we employed the dependency analyzer CaboCha, an SVM-based system. In general, when one attempts to solve a classification problem with kernel functions, it is difficult to know the kernel function that best fits the problem. To date, second- and third-degree polynomial kernels have been empirically used in Japanese dependency analysis with SVMs.

In the biased evaluation (the test corpus was the learning corpus), the third-degree polynomial kernel produced very accurate results, almost 100%. On the other hand, in the open test, however, the third-degree polynomial kernel did not produce results as good as the second-degree one. We conclude from these results that the third-degree polynomial kernel suffered the over-fitting problem.

The second-degree polynomial kernel produced on accuracy of almost 94% in the biased evaluation, and this can be considered as the upper bound for the second degree polynomial kernel to analyze Japanese dependency structure. The accuracy was stable when we adjusted the soft-margin parameter of the SVM. However, there were several annotation errors in the corpus. Thus, if we correct such annotation errors, the accuracy would improve.

Table 4 indicates that case elements consisting of nouns and case markers were frequently mis-analyzed. From a grammatical point of view, a case element should depend on a verb. However, the number of relations between verbs and case elements is combinatorial explosion. Thus, we can conclude that the learning data were not sufficient for relations between verbs and case elements to analyze unseen relations.

On the other hand, in Table 4, verbs take many places in comparison to their distribution in the test set corpus. These verbs tend to form conjunctive structures and it is known that analyzing conjunctive structure is difficult (Kurohashi and Nagao, 1994b). Particularly when a verb is a head of an adverbial clause, it seems very difficult to detect a head bunsetu, which is modified by the verb.

From Table 5, we can conclude that the analyzed errors centered on short-distance relations; the analyzer especially tends to mis-analyze the correct distance of two as one. Typical cases of such mis-analysis are "N_1-*no* N_2-*no* N_3" and "[adnominal clause] N_1-*no* N_2." In some cases, it is also difficult for humans to analyze these patterns correctly.

4.2 Selective sampling simulation

The results revealed very small differences between strategies possibly due to insufficient corpus size. However, there was an overall tendency that the accuracy depended heavily whether how many long sentences with very long dependencies were included in the test set. Table 3 shows a simple example of this. In the cross-validation tests the accuracy of the general articles, the average length of which was 9.769 bunsetu / sentence, was almost 1% lower than that of the editorial articles, whose average length was 9.162 bunsetu / sentence. The reason why sentence length controlled the accuracy was that an error in the long-distance dependency may have caused other errors in order to satisfy the condition that dependencies do not cross each other in Japanese dependencies. Thus,

many errors occurred in longer sentences. To improve the accuracy, it is vital to analyze very long-distance dependencies correctly.

From Tables 7, 8 and 9, the strategy of longest first appears good for the expanding situation even if the average length of the test set is very short like in IPAL0. However, in the initial situation, since there is no labeled data, the longest-first strategy is not a good method. Table 10 shows that the random strategy (KG0) and the strategy of maximizing vocabulary size first (I-VSort) were better than the longest-first strategy (I-Long). This is because the test sets comprised short sentences and we can imagine that there were dependencies included only in such short sentences. In other words, the longest-first strategy was heavily biased toward long sentences and the strategy could not cover the dependencies that were only included in short sentences.

On the other hand, the number of such dependencies that were only included in short sentences was quite small, and this number would soon be saturated when we built a dependency analyzed corpus. Thus, in the initial situation, the random strategy was better, whereas after we prepared a corpus to some extent, the longest-first strategy would be better because analyzing long sentences is difficult.

In the case of expansion, the longest-first strategy was good, though we have to consider the actual time required to annotate such long sentences because in general longer sentences tend to have more complex structures and introduce more opportunities for ambiguous parses. This means it is difficult for humans to annotate such long sentences.

5 Related works

To date, many works on selective sampling were conducted in the field related to natural language processing (Fujii et al., 1998; Hwa, 2004; Kamm and Meyer, 2002; Riccardi and Hakkani-Tür, 2005; Ngai and Yarowsky, 2000; Banko and Brill, 2001; Engelson and Dagan, 1996). The basic concepts are the same and it is important to predict the training utility value of each candidate with high accuracy. The work most closely related to this paper is Hwa's (Hwa, 2004), which proposed a sophisticated method for selective sampling for statistical parsing. However, the experiments carried out in that paper were done with just one corpus, WSJ Treebank. The study by Baldridge and Osborne (Baldridge and Osborne, 2004) is also very close to this paper. They used the Redwoods treebank environment (Oepen et al., 2002) and discussed the reduction in annotation cost by an active learning approach.

In this paper, we focused on the analysis of several fundamental sampling strategies for building a Japanese dependency-analyzed corpus. A complete estimating function of training utility value was not shown in this paper. However, we tested several strategies with different types of corpora, and these results can be used to design such a function for selective sampling.

6 Conclusion

This paper discussed several sampling strategies for Japanese dependency-analyzed corpora, testing them with the Kyoto Text Corpus and the IPAL corpus. The IPAL corpus was constructed especially for this study. In addition, although it was quite small, we prepared the K-mag corpus to test the strategies. The experimental results using these corpora revealed that the average length of a test set controlled the accuracy in case of expansion; thus the longest-first strategy outperformed other strategies. On the other hand, in the initial situation, the longest-first strategy was not suitable for any test set.

The current work points us in several future directions. First, we shall continue to build dependency-analyzed corpora. While newspaper articles may be sufficient for our purpose, other resources seem still inadequate. Second, while in this work we focused on analysis using several fundamental selective strategies for a dependency-analyzed corpus, it is necessary to provide a function to build a selective sampling framework to construct a dependency-analyzed corpus.

References

Jason Baldridge and Miles Osborne. 2004. Active learning and the total cost of annotation. In *Proceedings of EMNLP*.

Michele Banko and Eric Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL-2001)*, pages 26–33.

David A. Cohn, Les Atlas, and Richard E. Ladner.

1994. Improving generalization with active learning. *Machine Learning*, 15(2):201–221.

Sean P. Engelson and Ido Dagan. 1996. Minimizing manual annotation cost in supervised training from corpora. In *Proceedings of the 34th Annual meeting of Association for Computational Linguistics*, pages 319–326.

Atsushi Fujii, Kentaro Inui, Takenobu Tokunaga, and Hozumi Tanaka. 1998. Selective sampling for example-based word sense disambiguation. *Computational Linguistics*, 24(4):573–598.

Rebecca Hwa. 2004. Sample selection for statistical parsing. *Computational Linguistics*, 30(3):253–276.

Teresa M. Kamm and Gerard G. L. Meyer. 2002. Selective sampling of training data for speech recognition. In *Proceedings of Human Language Technology*.

Taku Kudo and Yuji Matsumoto. 2002. Japanese dependency analysis using cascaded chunking. In *CoNLL 2002: Proceedings of the 6th Conference on Natural Language Learning 2002 (COLING 2002 Post-Conference Workshops)*, pages 63–69.

Sadao Kurohashi and Makoto Nagao. 1994a. KN Parser: Japanese dependency/case structure analyzer. In *Proceedings of Workshop on Sharable Natural Language Resources*, pages 48–55.

Sadao Kurohashi and Makoto Nagao. 1994b. A syntactic analysis method of long Japanese sentences based on the detection of conjunctive structures. *Computational Linguistics*, 20(4):507–534.

Grace Ngai and David Yarowsky. 2000. Rule writing or annotation: Cost-effi cient resource usage for base noun phrase chunking. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 117–125.

Stephan Oepen, Kristina Toutanova, Stuart Shieber, Christopher Manning, Dan Flickinger, and Thorsten Brants. 2002. The LinGO Redwoods treebank: Motivation and preliminary applicatoins. In *Proceedings of COLING 2002*, pages 1–5.

Giuseppe Riccardi and Dilek Hakkani-Tür. 2005. Active learning: Theory and applications to automatic speech recognition. *IEEE Transactions on Speech and Audio Processing*, 13(4):504–511.

Manabu Sasano. 2004. Linear-time dependency analysis for Japanese. In *Proceedings of Coling 2004*, pages 8–14.

A Term Recognition Approach to Acronym Recognition

Naoaki Okazaki *
Graduate School of Information Science and Technology
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656 Japan
okazaki@mi.ci.i.u-tokyo.ac.jp

Sophia Ananiadou
National Centre for Text Mining
School of Informatics
Manchester University
PO Box 88, Sackville Street, Manchester M60 1QD United Kingdom
Sophia.Ananiadou@manchester.ac.uk

Abstract

We present a term recognition approach to extract acronyms and their definitions from a large text collection. Parenthetical expressions appearing in a text collection are identified as potential acronyms. Assuming terms appearing frequently in the proximity of an acronym to be the expanded forms (definitions) of the acronyms, we apply a term recognition method to enumerate such candidates and to measure the likelihood scores of the expanded forms. Based on the list of the expanded forms and their likelihood scores, the proposed algorithm determines the final acronym-definition pairs. The proposed method combined with a letter matching algorithm achieved 78% precision and 85% recall on an evaluation corpus with 4,212 acronym-definition pairs.

1 Introduction

In the biomedical literature the amount of terms (names of genes, proteins, chemical compounds, drugs, organisms, etc) is increasing at an astounding rate. Existing terminological resources and scientific databases (such as Swiss-Prot[1], SGD[2], FlyBase[3], and UniProt[4]) cannot keep up-to-date with the growth of neologisms (Pustejovsky et al., 2001). Although curation teams maintain terminological resources, integrating neologisms is very difficult if not based on systematic extraction and collection of terminology from literature. Term identification in literature is one of the major bottlenecks in processing information in biology as it faces many challenges (Ananiadou and Nenadic, 2006; Friedman et al., 2001; Bodenreider, 2004). The major challenges are due to term variation, e.g. spelling, morphological, syntactic, semantic variations (one term having different termforms), term synonymy and homonymy, which are all central concerns of any term management system.

Acronyms are among the most productive type of term variation. Acronyms (e.g. RARA) are compressed forms of terms, and are used as substitutes of the fully expanded termforms (e.g., *retinoic acid receptor alpha*). Chang and Schütze (2006) reported that, in MEDLINE abstracts, 64,242 new acronyms were introduced in 2004 with the estimated number being 800,000. Wren et al. (2005) reported that 5,477 documents could be retrieved by using the acronym *JNK* while only 3,773 documents could be retrieved by using its full term, *c-jun N-terminal kinase*.

In practice, there are no rules or exact patterns for the creation of acronyms. Moreover, acronyms are ambiguous, i.e., the same acronym may refer to different concepts (*GR* abbreviates both *glucocorticoid receptor* and *glutathione reductase*). Acronyms also have variant forms (e.g. NF kappa B, NF kB, NF-KB, NF-kappaB, NFKB factor for nuclear factor-kappa B). Ambiguity and variation present a challenge for any text mining system, since acronyms have not only to be recognised, but their variants have to be linked to the same canonical form and be disambiguated.

Thus, discovering acronyms and relating them to their expanded forms is important for terminology management. In this paper, we present a term recognition approach to construct an acronym dic-

*Research Fellow of the Japan Society for the Promotion of Science (JSPS)

[1] http://www.ebi.ac.uk/swissprot/
[2] http://www.yeastgenome.org/
[3] http://www.flybase.org/
[4] http://www.ebi.ac.uk/GOA/

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 643–650,
Sydney, July 2006.

tionary from a large text collection. The proposed method focuses on terms appearing frequently in the proximity of an acronym and measures the likelihood scores of such terms to be the expanded forms of the acronyms. We also describe an algorithm to combine the proposed method with a conventional letter-based method for acronym recognition.

2 Related Work

The goal of acronym identification is to extract pairs of short forms (acronyms) and long forms (their expanded forms or definitions) occurring in text[5]. Currently, most methods are based on letter matching of the acronym-definition pair, e.g., *hidden markov model (HMM)*, to identify short/-long form candidates. Existing methods of short-/long form recognition are divided into pattern matching approaches, e.g., exploring an efficient set of heuristics/rules (Adar, 2004; Ao and Takagi, 2005; Schwartz and Hearst, 2003; Wren and Garner, 2002; Yu et al., 2002), and pattern mining approaches, e.g., Longest Common Substring (LCS) formalization (Chang and Schütze, 2006; Taghva and Gilbreth, 1999).

Schwartz and Hearst (2003) implemented an algorithm for identifying acronyms by using parenthetical expressions as a marker of a short form. A character matching technique was used, i.e. all letters and digits in a short form had to appear in the corresponding long form in the same order, to determine its long form. Even though the core algorithm was very simple, the authors report 99% precision and 84% recall on the Medstract gold standard[6].

However, the letter-matching approach is affected by the expressions in the source text and sometimes finds incorrect long forms such as *acquired syndrome* and *a patient with human immunodeficiency syndrome*[7] instead of the correct one, *acquired immune deficiency syndrome* for the acronym *AIDS*. This approach also encounters difficulties finding a long form whose short form is arranged in a different word order, e.g., *beta 2 adrenergic receptor (ADRB2)*. To improve the accuracy of long/short form recognition, some methods measure the appropriateness of these candidates based on a set of rules (Ao and Takagi, 2005), scoring functions (Adar, 2004), statistical analysis (Hisamitsu and Niwa, 2001; Liu and Friedman, 2003) and machine learning approaches (Chang and Schütze, 2006; Pakhomov, 2002; Nadeau and Turney, 2005).

Chang and Schütze (2006) present an algorithm for matching short/long forms with a statistical learning method. They discover a list of abbreviation candidates based on parentheses and enumerate possible short/long form candidates by a dynamic programming algorithm. The likelihood of the recognized candidates is estimated as the probability calculated from a logistic regression with nine features such as the percentage of long-form letters aligned at the beginning of a word. Their method achieved 80% precision and 83% recall on the Medstract corpus.

Hisamitsu and Niwa (2001) propose a method for extracting useful parenthetical expressions from Japanese newspaper articles. Their method measures the co-occurrence strength between the inner and outer phrases of a parenthetical expression by using statistical measures such as mutual information, χ^2 test with Yate's correction, Dice coefficient, log-likelihood ratio, etc. Their method deals with generic parenthetical expressions (e.g., abbreviation, non abbreviation paraphrase, supplementary comments), not focusing exclusively on acronym recognition.

Liu and Friedman (2003) proposed a method based on mining collocations occurring before the parenthetical expressions. Their method creates a list of potential long forms from collocations appearing more than once in a text collection and eliminates unlikely candidates with three rules, e.g., "remove a set of candidates T_w formed by adding a prefix word to a candidate w if the number of such candidates T_w is greater than 3". Their approach cannot recognise expanded forms occurring only once in the corpus. They reported a precision of 96.3% and a recall of 88.5% for abbreviations recognition on their test corpus.

3 Methodology

3.1 Term-based long-form identification

We propose a method for identifying the long forms of an acronym based on a term extraction technique. We focus on terms appearing fre-

[5]This paper uses the terms "short form" and "long form" hereafter. "Long form" is what others call "definition", "meaning", "expansion", and "expanded form" of acronym.

[6]`http://www.medstract.org/`

[7]These examples are obtained from the actual MEDLINE abstracts submitted to Schwartz and Hearst's algorithm (2003). An author does not always write a proper definition with a parenthetic expression.

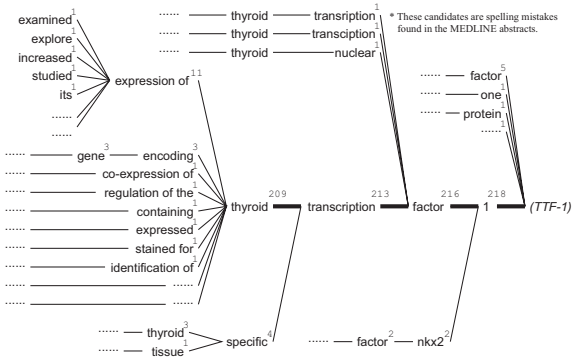


Figure 1: Long-form candidates for *TTF-1*.

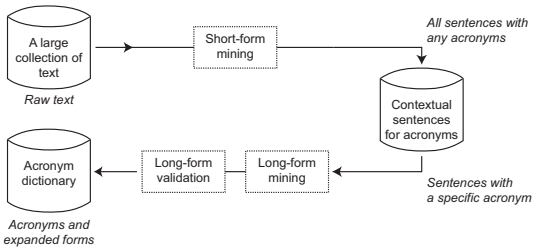


Figure 2: System diagram of acronym recognition

quently in the proximity of an acronym in a text collection. More specifically, if a word sequence co-occurs frequently with a specific acronym and not with other surrounding words, we assume that there is a relationship[8] between the acronym and the word sequence.

Figure 1 illustrates our hypothesis taking the acronym *TTF-1* as an example. The tree consists of expressions collected from all sentences with the acronym in parentheses and appearing before the acronym. A node represents a word, and a path from any node to *TTF-1* represents a long-form candidate[9]. The figure above each node shows the co-occurrence frequency of the corresponding long-form candidate. For example, long-form candidates *1*, *factor 1*, *transcription factor 1*, and *thyroid transcription factor 1* co-occur 218, 216, 213, and 209 times respectively with the acronym *TTF-1* in the text collection.

Even though long-form candidates *1*, *factor 1* and *transcription factor 1* co-occur frequently with the acronym *TTF-1*, we note that they also co-occur frequently with the word *thyroid*. Meanwhile, the candidate *thyroid transcription factor 1* is used in a number of contexts (e.g., *expression of thyroid transcription factor 1*, *expressed thyroid transcription factor 1*, *gene encoding thyroid transcription factor 1*, *etc.*). Therefore, we observe this to be the strongest relationship between acronym *TTF-1* and its long-form candidate *thyroid transcription factor 1* in the tree. We apply a number of validation rules (described later) to the candidate pair to make sure that it has an acronym-definition relation. In this example, the candidate pair is likely to be an acronym-definition relation because the long form *thyroid transcription factor 1* contains all alphanumeric letters in the short form *TTF-1*.

Figure 1 also shows another notable characteristic of long-form recognition. Assuming that the term *thyroid transcription factor 1* has an acronym *TTF-1*, we can disregard candidates such as *transcription factor 1*, *factor 1*, and *1* since they lack the necessary elements (e.g., *thyroid* for all candidates; *thyroid transcription* for candidates *factor 1* and *1*; etc.) to produce the acronym *TTF-1*. Similarly, we can disregard candidates such as *expression of thyroid transcription factor 1* and *encoding thyroid transcription factor 1* since they contain unnecessary elements (i.e., *expression of* and *encoding*) attached to the long-form. Hence, once *thyroid transcription factor 1* is chosen as the most likely long form of the acronym *TTF-1*, we prune the unlikely candidates: nested candidates (e.g., *transcription factor 1*); expansions (e.g., *expression of thyroid transcription factor 1*); and insertions (e.g., *thyroid specific transcription factor 1*).

3.2 Extracting acronyms and their contexts

Before describing in detail the formalization of long-form identification, we explain the whole process of acronym recognition. We divide the acronym extraction task into three steps (Figure 2):

1. **Short-form mining**: identifying and extracting short forms (i.e., acronyms) in a collection of documents

2. **Long-form mining**: generating a list of ranked long-form candidates for each short

[8] A sequence of words that co-occurs with an acronym does not always imply the acronym-definition relation. For example, the acronym *5-HT* co-occurs frequently with the term *serotonin*, but their relation is interpreted as a synonymous relation.

[9] The words with function words (e.g., *expression of*, *regulation of the*, etc.) are combined into a node. This is due to the requirement for a long-form candidate discussed later (Section 3.3).

Acronym	Contextual sentence
...	....
HML	Hard metal lung diseases *(HML)* are rare, and complex to diagnose.
HMM	Heavy meromyosin *(HMM)* from conditioned hearts had a higher Ca++-ATPase activity than from controls.
HMM	Heavy meromyosin *(HMM)* and myosin subfragment 1 (S1) were prepared from myosin by using low concentrations of alpha-chymotrypsin.
HMM	Hidden Markov model *(HMM)* techniques are used to model families of biological sequences.
HMM	Hexamethylmelamine *(HMM)* is a cytotoxic agent demonstrated to have broad antitumor activity.
HMN	Hereditary metabolic neuropathies *(HMN)* are marked by inherited enzyme or other metabolic defects.
...	... .

Table 1: An example of extracted acronyms and their contextual sentences.

form by using a term extraction technique

3. **Long-form validation**: extracting short/long form pairs recognized as having an acronym-definition relation and eliminating unnecessary candidates.

The first step, *short-form mining*, enumerates all short forms in a target text which are likely to be acronyms. Most studies make use of the following pattern to find candidate acronyms (Wren and Garner, 2002; Schwartz and Hearst, 2003):

long form '(' short form ')'

Just as the heuristic rules described in Schwartz and Hearst (Schwartz and Hearst, 2003), we consider short forms to be valid only if they consist of at most two words; their length is between two to ten characters; they contain at least an alphabetic letter; and the first character is alphanumeric. All sentences containing a short form in parenthesis are inserted into a database, which returns all contextual sentences for a short form to be processed in the next step. Table 1 shows an example of the database content.

3.3 Formalizing long-form mining as a term extraction problem

The second step, *long-form mining*, generates a list of long-form candidates and their likelihood scores for each short form. As mentioned previously, we focus on words or word sequences that co-occur frequently with a specific acronym and not with any other surrounding words. We deal with the problem of extracting long-form candidates from contextual sentences for an acronym in a similar manner as the term recognition task which extracts terms from the given text. For that purpose, we used a modified version of the C-value method (Frantzi and Ananiadou, 1999).

C-value is a domain-independent method for automatic term recognition (ATR) which combines linguistic and statistical information, emphasis being placed on the statistical part. The linguistic analysis enumerates all candidate terms in a given text by applying part-of-speech tagging, candidate extraction (e.g., extracting sequences of adjectives/nouns based on part-of-speech tags), and a stop-list. The statistical analysis assigns a termhood (likelihood to be a term) to a candidate term by using the following features: the frequency of occurrence of the candidate term; the frequency of the candidate term as part of other longer candidate terms; the number of these longer candidate terms; and the length of the candidate term.

The C-value approach is characterized by the extraction of *nested* terms which gives preference to terms appearing frequently in a given text but not as a part of specific longer terms. This is a desirable feature for acronym recognition to identify long-form candidates in contextual sentences. The rest of this subsection describes the method to extract long-form candidates and to assign scores to the candidates based on the C-value approach.

Given a contextual sentence as shown in Table 1, we tokenize a contextual sentence by non-alphanumeric characters (e.g., space, hyphen, colon, etc.) and apply Porter's stemming algorithm (Porter, 1980) to obtain a sequence of normalized words. We use the following pattern to extract long-form candidates from the sequence:

```
[:WORD:].*$        (1)
```

Therein: `[:WORD:]` matches a non-function word; `.*` matches an empty string or any word(s) of any length; and `$` matches a short form of the target acronym. The extraction pattern accepts a word or word sequence if the word or word sequence begins with any non-function word, and ends with any word just before the corresponding short form in the contextual sentence. We have defined 113 function words such as *a*, *the*, *of*, *we*, and *be* in an external dictionary so that long-form candidates cannot begin with these words.

Let us take the example of a contextual sentence, "we studied the expression of thyroid transcription factor-1 (TTF-1)". We extract the following substrings as long form candidates (words are stemmed): *1*; *factor 1*; *transcript factor 1*; *thyroid transcript factor 1*; *expression of thyroid transcript factor 1*; and *studi the expression of thyroid*

Candidate	Length	Freq	Score	Valid
adriamycin	1	727	721.4	o
adrenomedullin	1	247	241.7	o
abductor digiti minimi	3	78	74.9	o
doxorubicin	1	56	54.6	L
effect of adriamycin	3	25	23.6	E
adrenodemedullated	1	19	17.7	o
acellular dermal matrix	3	17	15.9	o
peptide adrenomedullin	2	17	15.1	E
effects of adrenomedullin	3	15	13.2	E
resistance to adriamycin	3	15	13.2	E
amyopathic dermatomyositis	2	14	12.8	o
vincristine (vcr) and adriamycin	4	11	10.0	E
drug adriamycin	2	14	10.0	E
brevis and abductor digiti minimi	5	11	9.8	E
minimi	1	83	5.8	N
digiti minimi	2	80	3.9	N
right abductor digiti minimi	4	4	2.5	E
automated digital microscopy	3	1	0.0	m
adrenomedullin concentration	2	1	0.0	N

Valid = { o: *valid*, m: *letter match*, L: *lacks necessary letters*, E: *expansion*, N: *nested*, B: *below the threshold* }

Table 2: Long-form candidates for *ADM*.

transcript factor 1. Substrings such as *of thyroid transcript factor 1* (which begins with a function word) and *thyroid transcript* (which ends prematurely before the short form) are not selected as long-form candidates.

We define the likelihood $\mathrm{LF}(w)$ for candidate w to be the long form of an acronym:

$$\mathrm{LF}(w) = \mathrm{freq}(w) - \sum_{t \in T_w} \mathrm{freq}(t) \times \frac{\mathrm{freq}(t)}{\mathrm{freq}(T_w)}. \quad (2)$$

Therein: w is a long-form candidate; $\mathrm{freq}(x)$ denotes the frequency of occurrence of a candidate x in the contextual sentences (i.e., co-occurrence frequency with a short form); T_w is a set of nested candidates, long-form candidates each of which consists of a preceding word followed by the candidate w; and $\mathrm{freq}(T_w)$ represents the total frequency of such candidates T_w.

The first term is equivalent to the co-occurrence frequency of a long-form candidate with a short form. The second term discounts the co-occurrence frequency based on the frequency distribution of nested candidates. Given a long-form candidate $t \in T_w$, $\frac{\mathrm{freq}(t)}{\mathrm{freq}(T_w)}$ presents the occurrence probability of candidate t in the nested candidate set T_w. Therefore, the second term of the formula calculates the expectation of the frequency of occurrence of a nested candidate accounting for the frequency of candidate w.

Table 2 shows a list of long-form candidates for acronym *ADM* extracted from 7,306,153 MEDLINE abstracts[10]. The long-form mining step extracted 10,216 unique long-form candidates from 1,319 contextual sentences containing the acronym *ADM* in parentheses. Table 2 arranges long-form candidates with their scores in desending order. Long-form candidates *adriamycin* and *adrenomedullin* co-occur frequently with the acronym *ADM*.

Note the huge difference in scores between the candidates *abductor digiti minimi* and *minimi*. Even though the candidate *minimi* co-occurs more frequently (83 times) than *abductor digiti minimi* (78 times), the co-occurrence frequency is mostly derived from the longer candidate, i.e., *digiti minimi*. In this case, the second term of Formula 2, the occurrence-frequency expectation of expansions for *minimi* (e.g., *digiti minimi*), will have a high value and will therefore lower the score of candidate *minimi*. This is also true for the candidate *digiti minimi*, i.e., the score of candidate *digiti minimi* is lowered by the longer candidate *abductor digiti minimi*. In contrast, the candidate *abductor digiti minimi* preserves its co-occurrence frequency since the second term of the formula is low, which means that each expansion (e.g, *brevis and abductor digiti minimi*, *right abductor digiti minimi*, ...) is expected to have a low frequency of occurrence.

3.4 Validation rules for long-form candidates

The final step of Figure 2 validates the extracted long-form candidates to generate a final set of short/long form pairs. According to the score in Table 2, *adriamycin* is the most likely long-form for acronym *ADM*. Since the long-form candidate *adriamycin* contains all letters in the acronym *ADM*, it is considered as an authentic long-form (marked as 'o' in the Valid field). This is also true for the second and third candidate (*adrenomedullin* and *abductor digiti minimi*).

The fourth candidate *doxorubicin* looks interesting, i.e., the proposed method assigns a high score to the candidate even though it lacks the letters *a* and *m*, which are necessary to form the corresponding short form. This is because *doxorubicin* is a synonymous term for *adriamycin* and described directly with its acronym *ADM*. In this paper, we deal with the acronym-definition relation although the proposed method would be applicable to mining other types of relations marked by parenthetical expressions. Hence, we introduce a constraint that a long form must cover all alphanu-

[10] 52GB XML files (from `medline05n0001.xml` to `medline05n0500.xml`)

```
# [Variables]
#   sf: the target short-form.
#   candidates: long-form candidates.
#   result: the list of decisive long-forms.
#   threshold: the threshold of cut-off.

# Sort long-form candidates in descending order
candidates.sort(                    # of scores.
    key=lambda lf:lf.score, reverse=True)

# Initialize result list as empty.
result = []

# Pick up a long form one by one from candidates.
for lf in candidates:
    # Apply a cut-off based on termhood score.
    # Allow candidates with letter matching.....(a)
    if lf.score < threshold and not lf.match:
        continue
    # A long-form must contain all letters......(b)
    if letter_recall(sf, lf) < 1:
        continue
    # Apply pruning of redundant long form......(c)
    if redundant(result, lf):
        continue
    # Insert this long form to the result list.
    result.append(lf)

# Output the decisive long-forms.
print result
```

Figure 3: Pseudo-code for long-form validation.

meric letters in the short form.

The fifth candidate *effect of adriamycin* is an expansion of a long form *adriamycin*, which has a higher score than *effect of adriamycin*. As we discussed previously, the candidate *effect of adriamycin* is skipped since it contains unnecessary word(s) to form an acronym. Similarly, we prune the candidate *minimi* because it forms a part of another long form *abductor digiti minimi*, which has a higher score than the candidate *minimi*. The likelihood score $LF(w)$ determines the most appropriate long-form among similar candidates sharing the same words or lacking some words.

We do not include candidates with scores below a given threshold. Therefore, the proposed method cannot extract candidates appearing rarely in the text collection. It depends on the application and considerations of the trade-off between precision and recall, whether or not an acronym recognition system should extract such rare long forms. When integrating the proposed method with e.g., Schwartz and Hearst's algorithm, we treat candidates recognized by the external method as if they pass the score cut-off. In Table 2, for example, candidate *automated digital microscopy* is inserted into the result set whereas candidate *adrenomedullin concentration* is skipped since it is nested by candidate *adrenomedullin*.

Figure 3 is a pseudo-code for the long-form validation algorithm described above. A long-form candidate is considered valid if the following conditions are met: *(a)* it has a score greater than a threshold or is nominated by a letter-matching algorithm; *(b)* it contains all letters in the corresponding short form; and *(c)* it is not nested, expansion, or insertion of the previously chosen long forms.

Rank	Parenthetic phrase	# contextual sentence	# unique long-forms
1	CT	30,982	171
2	PCR	25,387	39
3	HIV	19,566	13
4	LPS	18,071	51
5	MRI	16,966	18
6	ELISA	16,527	25
7	SD	15,760	165
8	BP	14,860	145
9	DA	14,518	129
10	CSF	14,035	34
11	CNS	13,573	47
12	IL	13,423	60
13	PKC	13,414	11
14	TNF-ALPHA	12,228	14
15	HPLC	12,211	16
16	ER	12,155	140
17	RT-PCR	12,153	21
18	TNF	12,145	13
19	LDL	11,960	24
20	5-HT	11,836	20
..	....	...	..
—	(overall 50 acronyms)	600,375	4,212

Table 3: Statistics on our evaluation corpus.

4 Evaluation

Several evaluation corpora for acronym recognition are available. The Medstract Gold Standard Evaluation Corpus, which consists of 166 alias pairs annotated to 201 MEDLINE abstracts, is widely used for evaluation (Chang and Schütze, 2006; Schwartz and Hearst, 2003). However, the amount of the text in the corpus is insufficient for the proposed method, which makes use of statistical features in a text collection. Therefore, we prepared an evaluation corpus with a large text collection and examined how the proposed algorithm extracts short/long forms precisely and comprehensively.

We applied the short-form mining described in Section 3 to 7,306,153 MEDLINE abstracts[10]. Out of 921,349 unique short-forms recognized by the short-form mining, top 50 acronyms[11] appearing frequently in the abstracts were chosen for our

[11] We have excluded several parenthetical expressions such as *II* (99,378 occurrences), *OH* (37,452 occurrences), and *P<0.05* (23,678 occurrences). Even though they are enclosed within parentheses, they do not introduce acronyms. We have also excluded a few acronyms such as *RA* (18,655 occurrences) and *AD* (15,540 occurrences) because they have many variations of their expanded forms to prepare the evaluation corpus manually.

evaluation corpus. We asked an expert in bioinformatics to extract long forms from 600,375 contextual sentences with the following criteria: a long form with minimum necessary elements (words) to produce its acronym is accepted; a long form with unnecessary elements, e.g., *magnetic resonance imaging unit (MRI)* or *computed x-ray tomography (CT)*, is not accepted; a misspelled long-form, e.g., *hidden markvov model (HMM)*, is accepted (to separate the acronym-recognition task from a spelling-correction task). Table 3 shows the top 20 acronyms in our evaluation corpus, the number of their contextual sentences, and the number of unique long-forms extracted.

Using this evaluation corpus as a gold standard, we examined precision, recall, and f-measure[12] of long forms recognized by the proposed algorithm and baseline systems. We compared five systems: the proposed algorithm with Schwartz and Hearst's algorithm integrated (PM+SH); the proposed algorithm without any letter-matching algorithm integrated (PM); the proposed algorithm but using the original C-value measure for long-form likelihood scores (CV+SH); the proposed algorithm but using co-occurrence frequency for long-form likelihood scores (FQ+SH); and Schwartz and Hearst's algorithm (SH). The threshold for the proposed algorithm was set to four.

Table 4 shows the evaluation result. The best-performing configuration of algorithms (PM+SH) achieved 78% precision and 85% recall. The Schwartz and Hearst's (SH) algorithm obtained a good recall (93%) but misrecognized a number of long-forms (56% precision), e.g., *the kinetics of serum tumour necrosis alpha (TNF-ALPHA)* and *infected mice lacking the gamma interferon (IFN-GAMMA)*. The SH algorithm cannot gather variations of long forms for an acronym, e.g., *ACE* as *angiotensin-converting enzyme level*, *angiotensin i-converting enzyme gene*, *angiotensin-1-converting enzyme*, *angiotensin-converting*, *angiotensin converting activity*, etc. The proposed method combined with the Schwartz and Hearst's algorithm remedied these misrecognitions based on the likelihood scores and the long-form validation algorithm. The PM+SH also outperformed other likelihood measures, CV+SH and FQ+SH.

[12]We count the number of unique long forms, i.e., count once even if short/long form pair ⟨*HMM*, *hidden markov model*⟩ occurs more than once in the text collection. The Porter's stemming algorithm was applied to long forms before comparing them with the gold standard.

Method	Precision	Recall	F-measure
PM+SH	0.783	0.849	0.809
CV+SH	0.722	0.838	0.765
FQ+SH	0.716	0.800	0.747
SH	0.555	0.933	0.681
PM	0.815	0.140	0.216

Table 4: Evaluation result of long-form recognition.

The proposed algorithm without Schwartz and Hearst's algorithm (PM) identified long forms the most precisely (81% precision) but misses a number of long forms in the text collection (14% recall). The result suggested that the proposed likelihood measure performed well to extract frequently used long-forms in a large text collection, but could not extract rare acronym-definition pairs. We also found the case where PM missed a set of long forms for acronym *ER* which end with *rate*, e.g., *eating rate*, *elimination rate*, *embolic rate*, etc. This was because the word *rate* was used with a variety of expansions (i.e., the likelihood score for *rate* was not reduced much) while it can be also interpreted as the long form of the acronym.

Even though the Medstract corpus is insufficient for evaluating the proposed method, we examined the number of long/short pairs extracted from 7,306,153 MEDLINE abstracts and also appearing in the Medstract corpus. We can neither calculate the precision from this experiment nor compare the recall directly with other acronym recognition methods since the size of the source texts is different. Out of 166 pairs in Medstract corpus, 123 (74%) pairs were exactly covered by the proposed method, and 15 (83% in total) pairs were partially covered[13]. The algorithm missed 28 pairs because: 17 (10%) pairs in the corpus were not acronyms but more generic aliases, e.g., *alpha tocopherol (Vitamin E)*; 4 (2%) pairs in the corpus were incorrectly annotated (e.g, long form in the corpus *embryo fibroblasts* lacks word *mouse* to form acronym *MEFS*); and 7 (4%) long forms are missed by the algorithm, e.g., the algorithm recognized pair *protein kinase (PKR)* while the correct pair in the corpus is *RNA-activated protein kinase (PKR)*.

[13]Medstract corpus leaves unnecessary elements attached to some long-forms such as *general transcription factor iib (TFIIB)*, whereas the proposed algorithm may drop the unnecessary elements (i.e. *general*) based on the frequency. We regard such cases as *partly* correct.

5 Conclusion

In this paper we described a term recognition approach to extract acronyms and their definitions from a large text collection. The main contribution of this study has been to show the usefulness of statistical information for recognizing acronyms in large text collections. The proposed method combined with a letter matching algorithm achieved 78% precision and 85% recall on the evaluation corpus with 4,212 acronym-definition pairs.

A future direction of this study would be to incorporate other types of relations expressed with parenthesis such as synonym, paraphrase, etc. Although this study dealt with the acronym-definition relation only, modelling these relations will also contribute to the accuracy of the acronym recognition, establishing a methodology to distinguish the acronym-definition relation from other types of relations.

References

Eytan Adar. 2004. SaRAD: A simple and robust abbreviation dictionary. *Bioinformatics*, 20(4):527–533.

Sophia Ananiadou and Goran Nenadic. 2006. Automatic terminology management in biomedicine. In Sophia Ananiadou and John McNaught, editors, *Text Mining for Biology and Biomedicine*, pages 67–97. Artech House, Inc.

Hiroko Ao and Toshihisa Takagi. 2005. ALICE: An algorithm to extract abbreviations from MEDLINE. *Journal of the American Medical Informatics Association*, 12(5):576–586.

Olivier Bodenreider. 2004. The Unified Medical Language System (UMLS): Integrating biomedical terminology. *Nucleic Acids Research*, 32:267–270.

Jeffrey T. Chang and Hinrich Schütze. 2006. Abbreviations in biomedical text. In S. Ananiadou and J. McNaught, editors, *Text Mining for Biology and Biomedicine*, pages 99–119. Artech House, Inc.

Katerina T. Frantzi and Sophia Ananiadou. 1999. The C-value / NC-value domain independent method for multi-word term extraction. *Journal of Natural Language Processing*, 6(3):145–179.

Carol Friedman, Hongfang Liu, Lyuda Shagina, Stephen Johnson, and George Hripcsak. 2001. Evaluating the UMLS as a source of lexical knowledge for medical language processing. In *AMIA Symposium*, pages 189–193.

Toru Hisamitsu and Yoshiki Niwa. 2001. Extracting useful terms from parenthetical expression by combining simple rules and statistical measures: A comparative evaluation of bigram statistics. In Didier Bourigault, Christian Jacquemin, and Marie-C L'Homme, editors, *Recent Advances in Computational Terminology*, pages 209–224. John Benjamins.

Hongfang Liu and Carol Friedman. 2003. Mining terminological knowledge in large biomedical corpora. In *8th Pacific Symposium on Biocomputing (PSB 2003)*, pages 415–426.

David Nadeau and Peter D. Turney. 2005. A supervised learning approach to acronym identification. In *8th Canadian Conference on Artificial Intelligence (AI'2005) (LNAI 3501)*, page 10 pages.

Serguei Pakhomov. 2002. Semi-supervised maximum entropy based approach to acronym and abbreviation normalization in medical texts. In *40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 160–167.

Youngja Park and Roy J. Byrd. 2001. Hybrid text mining for finding abbreviations and their definitions. In *2001 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 126–133.

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

James Pustejovsky, José Castaño, Brent Cochran, Maciej Kotecki, and Michael Morrell. 2001. Automatic extraction of acronym meaning pairs from MEDLINE databases. *MEDINFO 2001*, pages 371–375.

Ariel S. Schwartz and Marti A. Hearst. 2003. A simple algorithm for identifying abbreviation definitions in biomedical text. In *Pacific Symposium on Biocomputing (PSB 2003)*, number 8, pages 451–462.

Kazem Taghva and Jeff Gilbreth. 1999. Recognizing acronyms and their definitions. *International Journal on Document Analysis and Recognition (IJDAR)*, 1(4):191–198.

Jonathan D. Wren and Harold R. Garner. 2002. Heuristics for identification of acronym-definition patterns within text: towards an automated construction of comprehensive acronym-definition dictionaries. *Methods of Information in Medicine*, 41(5):426–434.

Jonathan D. Wren, Jeffrey T. Chang, James Pustejovsky, Eytan Adar, Harold R. Garner, and Russ B. Altman. 2005. Biomedical term mapping databases. *Database Issue*, 33:D289–D293.

Hong Yu, George Hripcsak, and Carol Friedman. 2002. Mapping abbreviations to full forms in biomedical articles. *Journal of the American Medical Informatics Association*, 9(3):262–272.

Combining Association Measures for Collocation Extraction

Pavel Pecina and **Pavel Schlesinger**
Institute of Formal and Applied Linguistics
Charles University, Prague, Czech Republic
{pecina,schlesinger}@ufal.mff.cuni.cz

Abstract

We introduce the possibility of combining lexical association measures and present empirical results of several methods employed in automatic collocation extraction. First, we present a comprehensive summary overview of association measures and their performance on manually annotated data evaluated by precision--recall graphs and mean average precision. Second, we describe several classification methods for combining association measures, followed by their evaluation and comparison with individual measures. Finally, we propose a feature selection algorithm significantly reducing the number of combined measures with only a small performance degradation.

1 Introduction

Lexical association measures are mathematical formulas determining the strength of association between two or more words based on their occurrences and cooccurrences in a text corpus. They have a wide spectrum of applications in the field of natural language processing and computational linguistics such as automatic collocation extraction (Manning and Schütze, 1999), bilingual word alignment (Mihalcea and Pedersen, 2003) or dependency parsing. A number of various association measures were introduced in the last decades. An overview of the most widely used techniques is given e.g. in Manning and Schütze (1999) or Pearce (2002). Several researchers also attempted to compare existing methods and suggest different evaluation schemes, e.g Kita (1994) and Evert (2001). A comprehensive study of statistical aspects of word cooccurrences can be found in Evert (2004) or Krenn (2000).

In this paper we present a novel approach to automatic collocation extraction based on combining multiple lexical association measures. We also address the issue of the evaluation of association measures by precision-recall graphs and mean average precision scores. Finally, we propose a stepwise feature selection algorithm that reduces the number of combined measures needed with respect to performance on held-out data.

The term *collocation* has both linguistic and lexicographic character. It has various definitions but none of them is widely accepted. We adopt the definition from Choueka (1988) who defines a *collocational expression* as "a syntactic and semantic unit whose exact and unambiguous meaning or connotation cannot be derived directly from the meaning or connotation of its components". This notion of collocation is relatively wide and covers a broad range of lexical phenomena such as idioms, phrasal verbs, light verb compounds, technological expressions, proper names, and stock phrases. Our motivation originates from machine translation: we want to capture all phenomena that may require special treatment in translation.

Experiments presented in this paper were performed on Czech data and our attention was restricted to two-word (*bigram*) collocations – primarily for the limited scalability of some methods to higher-order n-grams and also for the reason that experiments with longer word expressions would require processing of much larger corpus to obtain enough evidence of the observed events.

2 Reference data

The first step in our work was to create a reference data set. Krenn (2000) suggests that collocation extraction methods should be evaluated against a reference set of collocations manually extracted from the full candidate data from a corpus. To avoid the experiments to be biased by underlying data preprocessing (part-of-speech tagging, lemmatization, and parsing), we extracted the reference data from morphologically and syntactically annotated Prague Dependency Treebank 2.0 containing about 1.5 million words annotated on analytical layer (PDT 2.0, 2006). A corpus of this size is certainly not sufficient for real-world applications but we found it adequate for our evaluation purposes – a larger corpus would have made the manual collocation extraction task infeasible.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 651–658,
Sydney, July 2006.

Dependency trees from the corpus were broken down into *dependency bigrams* consisting of *lemmas* of the head word and its modifier, their *part-of-speech pattern*, and *dependency type*. From 87 980 sentences containing 1 504 847 words, we obtained a total of 635 952 different dependency bigrams types. Only 26 450 of them occur in the data more than five times. The less frequent bigrams do not meet the requirement of sufficient evidence of observations needed by some methods used in this work (they assume normal distribution of observations and become unreliable when dealing with rare events) and were not included in the evaluation. We, however, must agree with Moore (2004) arguing that these cases comprise majority of all the data (the Zipfian phenomenon) and thus should not be excluded from real-world applications. Finally, we filtered out all bigrams having such part-of-speech patterns that never form a collocation (conjunction–preposition, preposition–pronoun, etc.) and obtained a list consisting of 12 232 dependency bigrams, further called *collocation candidates*.

2.1 Manual annotation

The list of collocation candidates was manually processed by three trained linguists in parallel and independently with the aim of identifying collocations as defined by Choueka. To simplify and clarify the work they were instructed to select those bigrams that can be assigned to these categories:

* idiomatic expressions
  - *studená válka (cold war)*
  - *visí otazník (question mark is hanging ∼ open question)*
* technical terms
  - *předseda vlády (prime minister)*
  - *očitý svědek (eye witness)*
* support verb constructions
  - *mít pravdu (to be right)*
  - *učinit rozhodnutí (make decision)*
* names of persons, locations, and other entities
  - *Pražský hrad (Prague Castle)*
  - *Červený kříž (Red Cross)*
* stock phrases
  - *zásadní problém (major problem)*
  - *konec roku (end of the year)*

The first (expected) observation was that the interannotator agreement among all the categories was rather poor: the Cohen's κ between annotators ranged from 0.29 to 0.49, which demonstrates that the notion of collocation is very subjective, domain-specific, and somewhat vague. The reason that three annotators were used was to get a more precise and objective idea about what can be considered a collocation by combining outcomes from multiple annotators. Only those bigrams that *all* three annotators independently recognized as collocations (of any type) were considered true collocations. The reference data set contains 2 557 such bigrams, which is 20.9% of all. κ between these two categories reanged from 0.52 to 0.58.

The data was split into six stratified samples. Five folds were used for five-fold cross validation and average performance estimation. The remaining one fold was put aside and used as held-out data in experiments described in Section 5.

3 Association measures

In the context of collocation extraction, lexical association measures are formulas determining the degree of association between collocation components. They compute an *association score* for each collocation candidate extracted from a corpus. The scores indicate the potential for a candidate to be a collocation. They can be used for *ranking* (candidates with high scores at the top), or for *classification* (by setting a threshold and discarding all bigrams below this threshold).

If some words occur together more often than by chance, then this may be evidence that they have a special function that is not simply explained as a result of their combination (Manning and Schütze, 1999). This property is known in linguistics as *non-compositionality*. We think of a corpus as a randomly generated sequence of words that is viewed as a sequence of word pairs (dependency bigrams in our case). Occurrence frequencies and marginal frequencies are used in several association measures that reflect how much the word cooccurrence is accidental. Such measures include: estimation of joint and conditional bigram probabilities (Table 1, 1–3), mutual information and derived measures (4–9), statistical tests of independence (10–14), likelihood measures (15–16), and various other heuristic association measures and coefficients (17–55) originating in different research fields.

By determining the entropy of the *immediate context* of a word sequence (words immediately preceding or following the bigram), the association measures (56–60) rank collocations according to the assumption that they occur as (syntactic) units in a (information-theoretically) noisy environment (Shimohata et al., 1997). By comparing *empirical contexts* of a word sequence and of its components (open-class words occurring within

#	Name	Formula
1.	**Joint probability**	$P(xy)$
⋆2.	**Conditional probability**	$P(y\|x)$
3.	**Reverse conditional prob.**	$P(x\|y)$
4.	**Pointwise mutual inform.**	$\log \frac{P(xy)}{P(x*)P(*y)}$
5.	**Mutual dependency (*MD*)**	$\log \frac{P(xy)^2}{P(x*)P(*y)}$
6.	**Log frequency biased *MD***	$\log \frac{P(xy)^2}{P(x*)P(*y)} + \log P(xy)$
7.	**Normalized expectation**	$\frac{2f(xy)}{f(x*)+f(*y)}$
8.	**Mutual expectation**	$\frac{2f(xy)}{f(x*)+f(*y)} \cdot P(xy)$
⋆9.	**Salience**	$\log \frac{P(xy)^2}{P(x*)P(*y)} \cdot \log f(xy)$
10.	**Pearson's χ^2 test**	$\sum_{ij} \frac{(f_{ij}-\hat{f}_{ij})^2}{\hat{f}_{ij}}$
11.	**Fisher's exact test**	$\frac{f(x*)!f(\bar{x}*)!f(*y)!f(*\bar{y})!}{N!f(xy)!f(x\bar{y})!f(\bar{x}y)!f(\bar{x}\bar{y})!}$
12.	**t test**	$\frac{f(xy)-\hat{f}(xy)}{\sqrt{f(xy)(1-(f(xy)/N))}}$
13.	**z score**	$\frac{f(xy)-\hat{f}(xy)}{\sqrt{\hat{f}(xy)(1-(\hat{f}(xy)/N))}}$
14.	**Poison significance measure**	$\frac{f(xy)-f(xy)\log f(xy)+\log f(xy)!}{\log N}$
15.	**Log likelihood ratio**	$-2\sum_{ij} f_{ij} \log \frac{f_{ij}}{\hat{f}_{ij}}$
16.	**Squared log likelihood ratio**	$-2\sum_{ij} \frac{\log f_{ij}^2}{\hat{f}_{ij}}$
	Association coefficients:	
17.	**Russel-Rao**	$\frac{a}{a+b+c+d}$
18.	**Sokal-Michiner**	$\frac{a+d}{a+b+c+d}$
19.	**Rogers-Tanimoto**	$\frac{a+d}{a+2b+2c+d}$
20.	**Hamann**	$\frac{(a+d)-(b+c)}{a+b+c+d}$
21.	**Third Sokal-Sneath**	$\frac{b+c}{a+d}$
22.	**Jaccard**	$\frac{a}{a+b+c}$
⋆23.	**First Kulczynsky**	$\frac{a}{b+c}$
24.	**Second Sokal-Sneath**	$\frac{a}{a+2(b+c)}$
25.	**Second Kulczynski**	$\frac{1}{2}(\frac{a}{a+b}+\frac{a}{a+c})$
⋆26.	**Fourth Sokal-Sneath**	$\frac{1}{4}(\frac{a}{a+b}+\frac{a}{a+c}+\frac{d}{d+b}+\frac{d}{d+c})$
⋆27.	**Odds ratio**	$\frac{ad}{bc}$
28.	**Yulle's ω**	$\frac{\sqrt{ad}-\sqrt{bc}}{\sqrt{ad}+\sqrt{bc}}$
29.	**Yulle's Q**	$\frac{ad-bc}{ad+bc}$
30.	**Driver-Kroeber**	$\frac{a}{\sqrt{(a+b)(a+c)}}$
31.	**Fifth Sokal-Sneath**	$\frac{ad}{\sqrt{(a+b)(a+c)(d+b)(d+c)}}$
32.	**Pearson**	$\frac{ad-bc}{\sqrt{(a+b)(a+c)(d+b)(d+c)}}$
33.	**Baroni-Urbani**	$\frac{a+\sqrt{ad}}{a+b+c+\sqrt{ad}}$
⋆34.	**Braun-Blanquet**	$\frac{a}{\max(a+b,a+c)}$
⋆35.	**Simpson**	$\frac{a}{\min(a+b,a+c)}$
36.	**Michael**	$\frac{4(ad-bc)}{(a+d)^2+(b+c)^2}$
37.	**Mountford**	$\frac{2a}{2bc+ab+ac}$
38.	**Fager**	$\frac{a}{\sqrt{(a+b)(a+c)}} - \frac{1}{2}\max(b,c)$
39.	**Unigram subtuples**	$\log \frac{ad}{bc} - 3.29\sqrt{\frac{1}{a}+\frac{1}{b}+\frac{1}{c}+\frac{1}{d}}$
40.	***U* cost**	$\log(1+\frac{\min(b,c)+a}{\max(b,c)+a})$
41.	***S* cost**	$\log(1+\frac{\min(b,c)}{a+1})^{-\frac{1}{2}}$
42.	***R* cost**	$\log(1+\frac{a}{a+b})\cdot\log(1+\frac{a}{a+c})$
43.	***T* combined cost**	$\sqrt{U \times S \times R}$
44.	**Phi**	$\frac{P(xy)-P(x*)P(*y)}{\sqrt{P(x*)P(*y)(1-P(x*))(1-P(*y))}}$
45.	**Kappa**	$\frac{P(xy)+P(\bar{x}\bar{y})-P(x*)P(*y)-P(\bar{x}*)P(*\bar{y})}{1-P(x*)P(*y)-P(\bar{x}*)P(*\bar{y})}$
46.	***J* measure**	$\max[P(xy)\log\frac{P(y\|x)}{P(*y)}+P(x\bar{y})\log\frac{P(\bar{y}\|x)}{P(*\bar{y})}, P(xy)\log\frac{P(x\|y)}{P(x*)}+P(\bar{x}y)\log\frac{P(\bar{x}\|y)}{P(\bar{x}*)}]$

#	Name	Formula
47.	**Gini index**	$\max[P(x*)(P(y\|x)^2+P(\bar{y}\|x)^2)-P(*y)^2 - P(\bar{x}*)(P(y\|\bar{x})^2+P(\bar{y}\|\bar{x})^2)-P(*\bar{y})^2, P(*y)(P(x\|y)^2+P(\bar{x}\|y)^2)-P(x*)^2 - P(*\bar{y})(P(x\|\bar{y})^2+P(\bar{x}\|\bar{y})^2)-P(\bar{x}*)^2]$
48.	**Confidence**	$\max[P(y\|x), P(x\|y)]$
49.	**Laplace**	$\max[\frac{NP(xy)+1}{NP(x*)+2}, \frac{NP(xy)+1}{NP(*y)+2}]$
50.	**Conviction**	$\max[\frac{P(x*)P(*y)}{P(x\bar{y})}, \frac{P(\bar{x}*)P(*y)}{P(\bar{x}y)}]$
51.	**Piatersky-Shapiro**	$P(xy)-P(x*)P(*y)$
52.	**Certainity factor**	$\max[\frac{P(y\|x)-P(*y)}{1-P(*y)}, \frac{P(x\|y)-P(x*)}{1-P(x*)}]$
53.	**Added value (*AV*)**	$\max[P(y\|x)-P(*y), P(x\|y)-P(x*)]$
54.	**Collective strength**	$\frac{P(xy)+P(\bar{x}\bar{y})}{P(x*)P(y)+P(\bar{x}*)P(*y)} \cdot \frac{1-P(x*)P(*y)-P(\bar{x}*)P(*y)}{1-P(x*)-P(\bar{x}\bar{y})}$
⋆55.	**Klosgen**	$\sqrt{P(xy)} \cdot AV$
	Context measures:	
⋆56.	**Context entropy**	$-\sum_w P(w\|C_{xy}) \log P(w\|C_{xy})$
⋆57.	**Left context entropy**	$-\sum_w P(w\|C^l_{xy}) \log P(w\|C^l_{xy})$
58.	**Right context entropy**	$-\sum_w P(w\|C^r_{xy}) \log P(w\|C^r_{xy})$
59.	**Left context divergence**	$P(x*)\log P(x*)$ $-\sum_w P(w\|C^l_{xy}) \log P(w\|C^l_{xy})$
60.	**Right context divergence**	$P(*y)\log P(*y)$ $-\sum_w P(w\|C^r_{xy}) \log P(w\|C^r_{xy})$
61.	**Cross entropy**	$-\sum_w P(w\|C_x) \log P(w\|C_y)$
62.	**Reverse cross entropy**	$-\sum_w P(w\|C_y) \log P(w\|C_x)$
63.	**Intersection measure**	$\frac{2\|C_x \cap C_y\|}{\|C_x\|+\|C_y\|}$
⋆64.	**Euclidean norm**	$\sqrt{\sum_w (P(w\|C_x)-P(w\|C_y))^2}$
65.	**Cosine norm**	$\frac{\sum_w P(w\|C_x)P(w\|C_y)}{\sum_w P(w\|C_x)^2 \cdot \sum_w P(w\|C_y)^2}$
⋆66.	***L1* norm**	$\sum_w \|P(w\|C_x)-P(w\|C_y)\|$
67.	**Confusion probability**	$\sum_w \frac{P(x\|C_w)P(y\|C_w)P(w)}{P(x*)}$
⋆68.	**Reverse confusion prob.**	$\sum_w \frac{P(y\|C_w)P(x\|C_w)P(w)}{P(*y)}$
⋆69.	**Jensen-Shannon diverg.**	$\frac{1}{2}[D(p(w\|C_x)\|\|\frac{1}{2}(p(w\|C_x)+p(w\|C_y))) + D(p(w\|C_y)\|\|\frac{1}{2}(p(w\|C_x)+p(w\|C_y)))]$
⋆70.	**Cosine of pointwise *MI***	$\frac{\sum_w MI(w,x)MI(w,y)}{\sqrt{\sum_w MI(w,x)^2}\cdot\sqrt{\sum_w MI(w,y)^2}}$
71.	**KL divergence**	$\sum_w P(w\|C_x) \log \frac{P(w\|C_x)}{P(w\|C_y)}$
72.	**Reverse KL divergence**	$\sum_w P(w\|C_y) \log \frac{P(w\|C_y)}{P(w\|C_x)}$
⋆73.	**Skew divergence**	$D(p(w\|C_x)\|\|\alpha(w\|C_y)+(1-\alpha)p(w\|C_x))$
74.	**Reverse skew divergence**	$D(p(w\|C_y)\|\|\alpha p(w\|C_x)+(1-\alpha)p(w\|C_y))$
75.	**Phrase word coocurrence**	$\frac{1}{2}(\frac{f(x\|C_{xy})}{f(xy)}+\frac{f(y\|C_{xy})}{f(xy)})$
76.	**Word association**	$\frac{1}{2}(\frac{f(x\|C_y)-f(xy)}{f(xy)}+\frac{f(y\|C_x)-f(xy)}{f(xy)})$
	Cosine context similarity:	$\frac{1}{2}(\cos(\mathbf{c}_x,\mathbf{c}_{xy})+\cos(\mathbf{c}_y,\mathbf{c}_{xy}))$ $\mathbf{c}_z=(z_i);\ \cos(\mathbf{c}_x,\mathbf{c}_y)=\frac{\sum x_i y_i}{\sqrt{\sum {x_i}^2}\cdot\sqrt{\sum {y_i}^2}}$
⋆77.	**in boolean vector space**	$z_i = \delta(f(w_i\|C_z))$
78.	**in tf vector space**	$z_i = f(w_i\|C_z)$
79.	**in $tf{\cdot}idf$ vector space**	$z_i = f(w_i\|C_z)\cdot\frac{N}{df(w_i)}$; $df(w_i)=\|\{x : w_i \epsilon C_x\}\|$
	Dice context similarity:	$\frac{1}{2}(\mathrm{dice}(\mathbf{c}_x,\mathbf{c}_{xy})+\mathrm{dice}(\mathbf{c}_y,\mathbf{c}_{xy}))$ $\mathbf{c}_z=(z_i);\ \mathrm{dice}(\mathbf{c}_x,\mathbf{c}_y)=\frac{2\sum x_i y_i}{\sum {x_i}^2+\sum {y_i}^2}$
80.	**in boolean vector space**	$z_i = \delta(f(w_i\|C_z))$
81.	**in tf vector space**	$z_i = f(w_i\|C_z)$
82.	**in $tf{\cdot}idf$ vector space**	$z_i = f(w_i\|C_z)\cdot\frac{N}{df(w_i)}$; $df(w_i)=\|\{x : w_i \epsilon C_x\}\|$

$a=f(xy)$	$b=f(x\bar{y})$	$f(x*)$
$c=f(\bar{x}y)$	$d=f(\bar{x}\bar{y})$	$f(\bar{x}*)$
$f(*y)$	$f(*\bar{y})$	N

A contingency table contains observed frequencies and marginal frequencies for a bigram xy; $\bar{w}$ stands for any word except w; $*$ stands for any word; N is a total number of bigrams. The table cells are sometimes referred to as f_{ij}. Statistical tests of independence work with contingency tables of expected frequencies $\hat{f}(xy) = f(x*)f(*y)/N$.

C_w	empirical context of w
C_{xy}	empirical context of xy
C^l_{xy}	left immediate context of xy
C^r_{xy}	right immediate context of xy

Table 1: Lexical association measures used for bigram collocation extraction.
⋆denotes those selected by the model reduction algorithm discussed in Section 5.

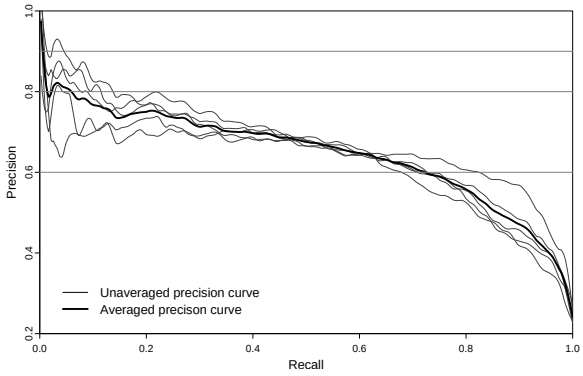


Figure 1: Vertical averaging of precision-recall curves. Thin curves represent individual non-averaged curves obtained by Pointwise mutual information (4) on five data folds.

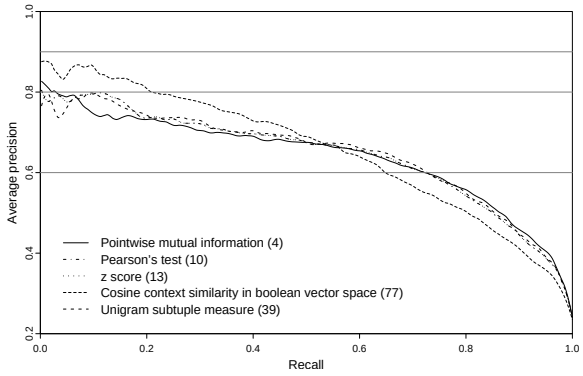


Figure 2: Crossvalidated and averaged precision-recall curves of selected association measures (numbers in brackets refer to Table 1).

a specified context window), the association measures rank collocations according to the assumption that semantically non-compositional expressions typically occur as (semantic) units in different contexts than their components (Zhai, 1997). Measures (61–74) have information theory background and measures (75–82) are adopted from the field of information retrieval.

3.1 Evaluation

Collocation extraction can be viewed as classification into two categories. By setting a threshold, any association measure becomes a binary classifier: bigrams with higher association scores fall into one class (collocations), the rest into the other class (non-collocations). Performance of such classifiers can be measured for example by *accuracy* – fraction of correct predictions. However, the proportion of the two classes in our case is far from equal and we want to distinguish classifier performance between them. In this case, several authors, e.g. Evert (2001), suggest using *precision* – fraction of positive predictions correct and *recall* – fraction of positives correctly predicted. The higher the scores the better the classification is.

3.2 Precision-recall curves

Since choosing a classification threshold depends primarily on the intended application and there is no principled way of finding it (Inkpen and Hirst, 2002), we can measure performance of association measures by precision–recall scores within the entire interval of possible threshold values. In this manner, individual association measures can be thoroughly compared by their two-dimensional *precision-recall curves* visualizing the quality of ranking without committing to a classification threshold. The closer the curve stays to the top and right, the better the ranking procedure is.

Precision-recall curves are very sensitive to data (see Figure 1). In order to obtain a good estimate of their shapes *cross validation* and *averaging* are necessary: all cross-validation folds with scores for each instance are combined and a single curve is drawn. Averaging can be done in three ways: *vertical* – fixing recall, averaging precision, *horizontal* – fixing precision, averaging recall, and *combined* – fixing threshold, averaging both precision and recall (Fawcett, 2003). Vertical averaging, as illustrated in Figure 1, worked reasonably well in our case and was used in all experiments.

3.3 Mean average precision

Visual comparison of precision-recall curves is a powerfull evaluation tool in many research fields (e.g. information retrieval). However, it has a serious weakness. One can easily compare two curves that never cross one another. The curve that predominates another one within the entire interval of recall seems obviously better. When this is not the case, the judgment is not so obvious. Also significance tests on the curves are problematic. Only well-defined one-dimensional quality measures can rank evaluated methods by their performance. We adopt such a measure from information retrieval (Hull, 1993). For each cross-validation data fold we define *average precision* (AP) as the expected value of precision for all possible values of recall (assuming uniform distribution) and *mean average precision* (MAP) as a mean of this measure computed for each data fold. Significance testing in this case can be realized by *paired t-test* or by more appropriate nonparametric *paired Wilcoxon test*.

Due to the unreliable precision scores for low recall and their fast changes for high recall, estimation of AP should be limited only to some narrower recall interval, e.g. $\langle 0.1,0.9 \rangle$

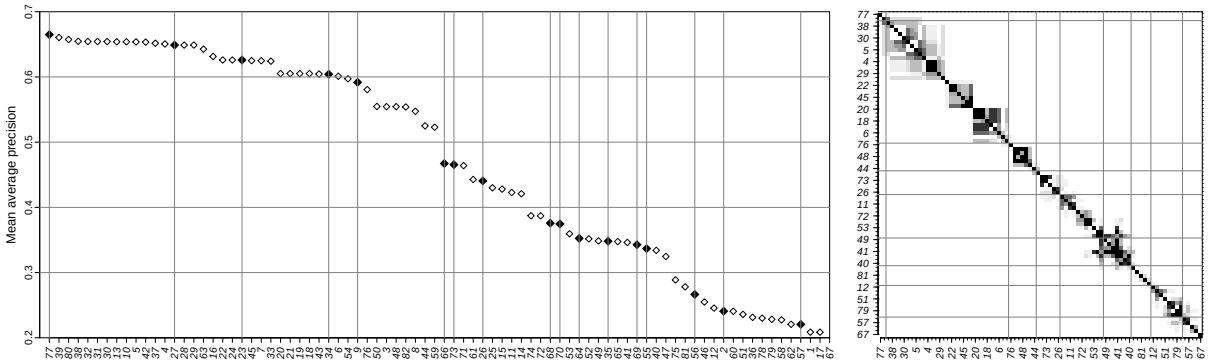


Figure 3: a) Mean average precision of all association measures in descending order. Methods are referred by numbers from Table 1. The solid points correspond to measures selected by the model reduction algorithm from Section 5. b) Visualization of p-values from the significance tests of difference between each method pair (order is the same for both graphs). The darker points correspond to p-values greater than $\alpha = 0.1$ and indicate methods with statistically indistinguishable performance (measured by paired Wilcoxon test on values of average precision obtained from five independent data folds).

3.4 Experiments and results

In the initial experiments, we implemented all 82 association measures from Table 1, processed all morphologically and syntactically annotated sentences from PDT 2.0, and computed scores of all the association measures for each dependency bigram in the reference data. For each association measure and each of the five evaluation data folds, we computed precision-recall scores and drew an averaged precision-recall curve. Curves of some well-performing methods are depicted in Figure 2. Next, for each association measure and each data fold, we estimated scores of average precision on narrower recall interval $\langle 0.1, 0.9 \rangle$, computed mean average precision, ranked the association measures according to MAP in descending order, and result depicted in Figure 3 a). Finally, we applied a paired Wilcoxon test, detected measures with statistically indistinguishable performance, and visualized this information in Figure 3 b).

A baseline system ranking bigrams randomly operates with average precision of 20.9%. The best performing method for collocation extraction measured by mean average precision is *cosine context similarity in boolean vector space* (77) (MAP 66.49%) followed by other 16 association measures with nearly identical performance (Figure 3 a). They include some popular methods well-known to perform reliably in this task, such as *pointwise mutual information* (4), *Pearson's χ^2 test* (10), *z score* (13), *odds ratio* (27), or *squared log likelihood ratio* (16).

The interesting point to note is that, in terms of MAP, context similarity measures, e.g. (77), slightly outperform measures based on simple occurence frequencies, e.g. (39). In a more thorough comparison by percision-recall curves, we observe that the former very significantly predominates the latter in the first half of the recall interval and vice versa in the second half (Figure 2). This is a case where the MAP is not a sufficient metric for comparison of association measure performance. It is also worth pointing out that even if two methods have the same precision-recall curves the actual bigram rank order can be very different. Existence of such *non-correlated* (in terms of ranking) measures will be essential in the following sections.

4 Combining association measures

Each collocation candidate x^i can be described by the *feature vector* $\mathbf{x}^i = (x^i_1, \ldots, x^i_{82})^T$ consisting of 82 association scores from Table 1 and assigned a label $y^i \in \{0, 1\}$ which indicates whether the bigram is considered to be a collocation ($y = 1$) or not ($y = 0$). We look for a *ranker* function $f(\mathbf{x}) \to \mathbb{R}$ that determines the strength of lexical association between components of bigram x and hence has the character of an association measure. This allows us to compare it with other association measures by the same means of precision-recall curves and mean average precision. Further, we present several classification methods and demonstrate how they can be employed for ranking, i.e. what function can be used as a ranker. For references see Venables and Ripley (2002).

4.1 Linear logistic regression

An additive model for binary response is represented by a generalized linear model (GLM) in a form of logistic regression:

$$\mathrm{logit}(\pi) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$$

method	*AP*			*MAP*	
	R=20	*R=50*	*R=80*	*R=⟨0.1,0.9⟩*	*+*
NNet (5 units)	89.56	82.74	70.11	80.81	21.53
NNet (3 units)	89.41	81.99	69.64	79.71	19.88
NNet (2 units)	86.92	81.68	68.33	78.77	18.47
SVM (linear)	85.72	79.49	63.86	75.66	13.79
LDA	84.72	77.18	62.90	75.11	12.96
SVM (quadratic)	84.29	79.54	64.24	74.53	12.09
NNet (1 unit)	77.98	76.83	66.75	73.25	10.17
GLM	82.45	76.26	58.61	71.88	8.11
Cosine similarity (77)	80.94	68.90	50.54	66.49	0.00
Unigram subtuples (39)	74.55	67.49	55.16	65.74	-

Table 2: Performance of methods combining all association measures: average precision (AP) for fixed recall values and mean average precision (MAP) on the narrower recall interval with relative improvement in the last column (values in %).

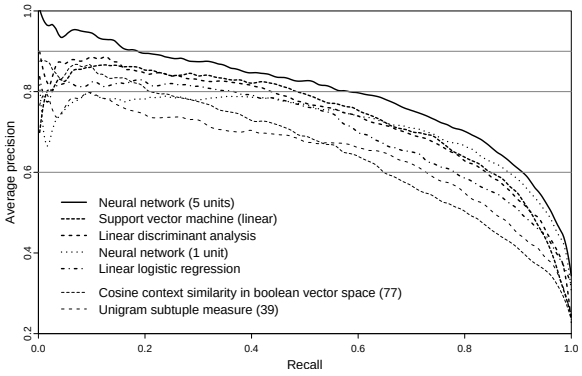


Figure 4: Precision-recall curves of selected methods combining all association measures compared with curves of two best measures employed individually on the same data sets.

where $\text{logit}(\pi) = \log(\pi/(1-\pi))$ is a canonical link function for odds-ratio and $\pi \in (0,1)$ is a conditional probability for positive response given a vector $\mathbf{x}$. The estimation of β_0 and $\boldsymbol{\beta}$ is done by maximum likelihood method which is solved by the *iteratively reweighted least squares algorithm*. The ranker function in this case is defined as the predicted value $\widehat{\pi}$, or equivalently (due to the monotonicity of logit link function) as the linear combination $\widehat{\beta}_0 + \widehat{\boldsymbol{\beta}}^T\mathbf{x}$.

4.2 Linear discriminant analysis

The basic idea of Fisher's linear discriminant analysis (LDA) is to find a one-dimensional projection defined by a vector $\mathbf{c}$ so that for the projected combination $\mathbf{c}^T\mathbf{x}$ the ratio of the *between* variance $\boldsymbol{B}$ to the *within* variance $\boldsymbol{W}$ is maximized:

$$\max_{\mathbf{c}} \frac{\mathbf{c}^T\boldsymbol{B}\mathbf{c}}{\mathbf{c}^T\boldsymbol{W}\mathbf{c}}$$

After projection, $\mathbf{c}^T\mathbf{x}$ can be directly used as ranker.

4.3 Support vector machines

For technical reason, let us now change the labels $y^i \in \{\text{-1,+1}\}$. The goal in support vector machines (SVM) is to estimate a function $f(\mathbf{x}) = \beta_0 + \boldsymbol{\beta}^T\mathbf{x}$ and find a classifier $y(\mathbf{x}) = \text{sign}\big(f(\mathbf{x})\big)$ which can be solved through the following convex optimization:

$$\min_{\beta_0,\boldsymbol{\beta}} \sum_{i=1}^{n} \left[1 - y^i(\beta_0 + \boldsymbol{\beta}^T\,\mathbf{x}^i)\right]^+ + \frac{\lambda}{2}||\boldsymbol{\beta}||^2$$

with λ as a regularization parameter. The *hinge loss function* $L(y, f(\mathbf{x})) = [1 - yf(\mathbf{x})]^+$ is active only for positive values (i.e. bad predictions) and therefore is very suitable for ranking models with $\widehat{\beta}_0 + \widehat{\boldsymbol{\beta}}^T\mathbf{x}$ as a ranker function. Setting the regularization parameter λ is crucial for both the estimators $\widehat{\beta}_0, \widehat{\boldsymbol{\beta}}$ and further classification (or ranking). As an alternative to a often inappropriate grid search, Hastie (2004) proposed an effective algorithm which fits the entire SVM regularization path $[\beta_0(\lambda), \boldsymbol{\beta}(\lambda)]$ and gave us the option to choose the optimal value of λ. As an objective function we used total amount of loss on training data.

4.4 Neural networks

Assuming the most common model of neural networks (NNet) with one hidden layer, the aim is to find inner weights w_{jh} and outer weights w_{hi} for

$$y^i = \phi_0\big(\alpha_0 + \sum w_{hi}\phi_h(\alpha_h + \sum w_{jh}x_j)\big)$$

where h ranges over units in the hidden layer. Activation functions ϕ_h and function ϕ_0 are fixed. Typically, ϕ_h is taken to be the logistic function $\phi_h(z) = \exp(z)/(1+\exp(z))$ and ϕ_0 to be the indicator function $\phi_0(z) = I(z > \Delta)$ with Δ as a classification threshold. For ranking we simply set $\phi_0(z) = z$. Parameters of neural networks are estimated by the *backpropagation algorithm*. The loss function can be based either on *least squares* or *maximum likehood*. To avoid problems with convergence of the algorithm we used the former one. The tuning parameter of a classifier is then the number of units in the hidden layer.

4.5 Experiments and results

To avoid incommensurability of association measures in our experiments, we used a common preprocessing technique for multivariate *standardization*: we centered values of each association measure towards zero and scaled them to unit variance.

Precision-recall curves of all methods were obtained by vertical averaging in five-fold cross validation on the same reference data as in the earlier experiments. Mean average precision was computed from average precision values estimated

on the recall interval $\langle 0.1,0.9 \rangle$. In each cross--validation step, four folds were used for training and one fold for testing.

All methods performed very well in comparison with individual measures. The best result was achieved by a neural network with five units in the hidden layer with 80.81% MAP, which is 21.53% relative improvement compared to the best individual associaton measure. More complex models, such as neural networks with more than five units in the hidden layer and support vector machines with higher order polynomial kernels, were highly overfitted on the training data folds and better results were achieved by simpler models. Detailed results of all experiment are given in Table 2 and precision-recall curves of selected methods depicted in Figure 4.

5 Model reduction

Combining association measures by any of the presented methods is reasonable and helps in the collocation extraction task. However, the combination models are too complex in number of predictors used. Some association measures are very similar (analytically or empirically) and as predictors perhaps even redundant. Such measures have no use in the models, make their training harder, and should be excluded. *Principal component analysis* applied to the evaluation data showed that 95% of its total variance is explained by only 17 principal components and 99.9% is explained by 42 of them. This gives us the idea that we should be able to significantly reduce the number of variables in our models with no (or relativelly small) degradation in their performance.

5.1 The algorithm

A straightforward, but in our case hardly feasible, approach is an exhaustive search through the space of all possible subsets of all association measures. Another option is a heuristic *step-wise* algorithm iteratively removing one variable at a time until some stopping criterion is met. Such algorithms are not very robust, they are sensitive to data and generally not very recommended. However, we tried to avoid these problems by initializing our step-wise algorithm by clustering similar variables and choosing one predictor from each cluster as a representative of variables with the same contribution to the model. Thus we remove the highly corelated predictors and continue with the step--wise procedure.

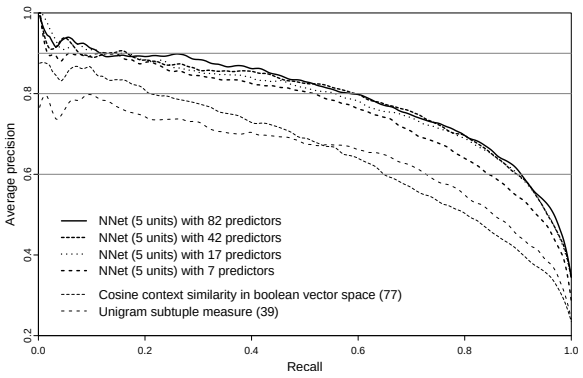


Figure 5: Precision-recall curves of four NNet models from the model reduction process with different number of predictors compared with curves of two best individual methods.

The algorithm starts with the hierarchical clustering of variables in order to group those with a similar contribution to the model, measured by the absolute value of *Pearson's correlation coefficient*. After $82-d$ iterations, variables are grouped into d non-empty clusters and one representative from each cluster is selected as a predictor into the initial model. This selection is based on individual predictor performance on held-out data.

Then, the algorithm continues with d predictors in the initial model and in each iteration removes a predictor causing minimal degradation of performance measured by MAP on held-out data. The algorithm stops when the difference becomes significant – either statistically (by paired Wilcoxon test) or practically (set by a human).

5.2 Experiments and results

We performed the model reduction experiment on the neural network with five units in the hidden layer (the best performing combination method). The similarity matrix for hierarchical clustering was computed on the held-out data and parameter d (number of initial predictors) was experimentally set to 60. In each iteration of the algorithm, we used four data folds (out of the five used in previous experiments) for fitting the models and the held-out fold to measure the performance of these models and to select the variable to be removed. The new model was cross-validated on the same five data-folds as in the previous experiments.

Precision-recall curves for some intermediate models are shown in Figure 5. We can conclude that we were able to reduce the NNet model to about 17 predictors without statistically significant difference in performance. The corresponding association measures are marked in Table 1 and highlighted in Figure 3a). They include measures from the entire range of individual mean average precision values.

6 Conclusions and discussion

We created and manually annotated a reference data set consisting of 12 232 Czech dependency bigrams. 20.9% of them were agreed to be a collocation by three annotators. We implemented 82 association measures, employed them for collocation extraction and evaluated them against the reference data set by averaged precision-recall curves and mean average precision in five-fold cross validation. The best result was achieved by a method measuring *cosine context similarity in boolean vector space* with mean average precision of 66.49%.

We exploit the fact that different subgroups of collocations have different sensitivity to certain association measures and showed that combining these measures aids in collocation extraction. All investigated methods significantly outperformed individual association measures. The best results were achieved by a simple neural network with five units in the hidden layer. Its mean average precision was 80.81% which is 21.53% relative improvement with respect to the best individual measure. Using more complex neural networks or a quadratic separator in support vector machines led to overtraining and did not improve the performace on test data.

We proposed a stepwise feature selection algorithm reducing the number of predictors in combination models and tested it with the neural network. We were able to reduce the number of its variables from 82 to 17 without significant degradation of its performance.

No attempt in our work has been made to select the "best universal method" for combining association measures nor to elicit the "best association measures" for collocation extraction. These tasks depend heavily on data, language, and notion of collocation itself. We demonstrated that combining association measures is meaningful and improves precission and recall of the extraction procedure and full performance improvement can be achieved by a relatively small number of measures combined.

Preliminary results of our research were already published in Pecina (2005). In the current work, we used a new version of the Prague Dependecy Treebank (PDT 2.0, 2006) and the reference data was improved by additional manual anotation by two linguists.

Acknowledgments

This work has been supported by the Ministry of Education of the Czech Republic, projects MSM 0021620838 and LC 536. We would like to thank our advisor Jan Hajič, our colleagues, and anonymous reviewers for their valuable comments.

References

Y. Choueka. 1988. Looking for needles in a haystack or locating interesting collocational expressions in large textual databases. In *Proceedings of the RIAO*.

S. Evert and B. Krenn. 2001. Methods for the qualitative evaluation of lexical association measures. In *Proceedings of the 39th Annual Meeting of the ACL*, Toulouse, France.

S. Evert. 2004. *The Statistics of Word Cooccurrences: Word Pairs and Collocations*. Ph.D. thesis, Univ. of Stuttgart.

T. Fawcett. 2003. ROC graphs: Notes and practical considerations for data mining researchers. Technical report, HPL-2003-4. HP Laboratories, Palo Alto, CA.

T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu. 2004. The entire regularization path for the support vector machine. *Journal of Machine Learning Research*, 5.

D. Hull. 1993. Using statistical testing in the evaluation of retrieval experiments. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, New York, NY.

D. Inkpen and G. Hirst. 2002. Acquiring collocations for lexical choice between near synonyms. In *SIGLEX Workshop on Unsupervised Lexical Acquisition, 40th meeting of the ACL*, Philadelphia.

K. Kita, Y. Kato, T. Omoto, and Y. Yano. 1994. A comparative study of automatic extraction of collocations from corpora: Mutual information vs. cost criteria. *Journal of Natural Language Processing*.

B. Krenn. 2000. *The Usual Suspects: Data-Oriented Models for Identification and Representation of Lexical Collocations*. Ph.D. thesis, Saarland University.

C. D. Manning and H. Schütze. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts.

R. Mihalcea and T. Pedersen. 2003. An evaluation exercise for word alignment. In *Proceedings of HLT-NAACL Workshop, Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*, Edmonton, Alberta.

R. C. Moore. 2004. On log-likelihood-ratios and the significance of rare events. In *Proceedings of the 2004 Conference on EMNLP*, Barcelona, Spain.

D. Pearce. 2002. A comparative evaluation of collocation extraction techniques. In *Third International Conference on language Resources and Evaluation*, Las Palmas, Spain.

P. Pecina. 2005. An extensive empirical study of collocation extraction methods. In *Proceedings of the ACL 2005 Student Research Workshop*, Ann Arbor, USA.

S. Shimohata, T. Sugio, and J. Nagata. 1997. Retrieving collocations by co-occurrences and word order constraints. In *Proc. of the 35th Meeting of ACL/EACL*, Madrid, Spain.

W. N. Venables and B. D. Ripley. 2002. *Modern Applied Statistics with S. 4th ed.* Springer Verlag, New York.

C. Zhai. 1997. Exploiting context to identify lexical atoms: A statistical view of linguistic context. In *International and Interdisciplinary Conf. on Modeling and Using Context*.

PDT 2.0. 2006. http://ufal.mff.cuni.cz/pdt2.0/.

Using Machine Learning to Explore Human Multimodal Clarification Strategies

Verena Rieser
Department of Computational Linguistics
Saarland University
Saarbrücken, D-66041
vrieser@coli.uni-sb.de

Oliver Lemon
School of Informatics
University of Edinburgh
Edinburgh, EH8 9LW, GB
olemon@inf.ed.ac.uk

Abstract

We investigate the use of machine learning in combination with feature engineering techniques to explore human multimodal clarification strategies and the use of those strategies for dialogue systems. We learn from data collected in a Wizard-of-Oz study where different wizards could decide whether to ask a clarification request in a multimodal manner or else use speech alone. We show that there is a uniform strategy across wizards which is based on multiple features in the context. These are generic runtime features which can be implemented in dialogue systems. Our prediction models achieve a weighted f-score of 85.3% (which is a 25.5% improvement over a one-rule baseline). To assess the effects of models, feature discretisation, and selection, we also conduct a regression analysis. We then interpret and discuss the use of the learnt strategy for dialogue systems. Throughout the investigation we discuss the issues arising from using small initial Wizard-of-Oz data sets, and we show that feature engineering is an essential step when learning from such limited data.

1 Introduction

Good clarification strategies in dialogue systems help to ensure and maintain mutual understanding and thus play a crucial role in robust conversational interaction. In dialogue application domains with high interpretation uncertainty, for example caused by acoustic uncertainties from a speech recogniser, multimodal generation and input leads to more robust interaction (Oviatt, 2002) and reduced cognitive load (Oviatt et al., 2004). In this paper we investigate the use of machine learning (ML) to explore human multimodal clarification strategies and the use of those strategies to decide, based on the current dialogue context, when a dialogue system's clarification request (CR) should be generated in a multimodal manner.

In previous work (Rieser and Moore, 2005) we showed that for spoken CRs in human-human communication people follow a context-dependent clarification strategy which systematically varies across domains (and even across Germanic languages). In this paper we investigate whether there exists a context-dependent "intuitive" human strategy for multimodal CRs as well. To test this hypothesis we gathered data in a Wizard-of-Oz (WOZ) study, where different wizards could decide when to show a screen output. From this data we build prediction models, using supervised learning techniques together with feature engineering methods, that may explain the underlying process which generated the data. If we can build a model which predicts the data quite reliably, we can show that there is a uniform strategy that the majority of our wizards followed in certain contexts.

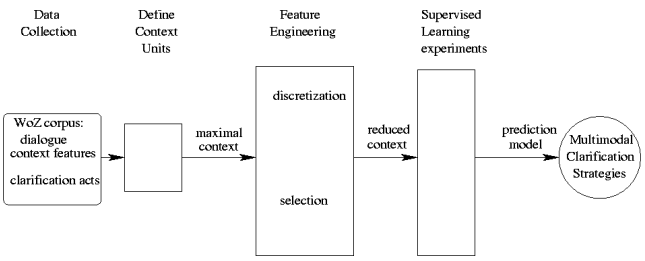


Figure 1: Methodology and structure

The overall method and corresponding structure of the paper is as shown in figure 1. We proceed

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 659–666,
Sydney, July 2006.

as follows. In section 2 we present the WOZ corpus from which we extract a potential context using "Information State Update" (ISU)-based features (Lemon et al., 2005), listed in section 3. We also address the question how to define a suitable "local" context definition for the wizard actions. We apply the feature engineering methods described in section 4 to address the questions of unique thresholds and feature subsets across wizards. These techniques also help to reduce the context representation and thus the feature space used for learning. In section 5 we test different classifiers upon this reduced context and separate out the independent contribution of learning algorithms and feature engineering techniques. In section 6 we discuss and interpret the learnt strategy. Finally we argue for the use of reinforcement learning to optimise the multimodal clarification strategy.

2 The WOZ Corpus

The corpus we are using for learning was collected in a multimodal WOZ study of German task-oriented dialogues for an in-car music player application, (Kruijff-Korbayová et al., 2005) . Using data from a WOZ study, rather than from real system interactions, allows us to investigate how humans clarify. In this study six people played the role of an intelligent interface to an MP3 player and were given access to a database of information. 24 subjects were given a set of predefined tasks to perform using an MP3 player with a multimodal interface. In one part of the session the users also performed a primary driving task, using a driving simulator. The wizards were able to speak freely and display the search results or the playlist on the screen by clicking on various pre-computed templates. The users were also able to speak, as well as make selections on the screen. The user's utterances were immediately transcribed by a typist. The transcribed user's speech was then corrupted by deleting a varying number of words, simulating understanding problems at the acoustic level. This (sometimes) corrupted transcription was then presented to the human wizard. Note that this environment introduces uncertainty on several levels, for example multiple matches in the database, lexical ambiguities, and errors on the acoustic level, as described in (Rieser et al., 2005). Whenever the wizard produced a CR, the experiment leader invoked a questionnaire window on a GUI, where the wizard classified their CR according to the primary source of the understanding problem, mapping to the categories defined by (Traum and Dillenbourg, 1996).

2.1 The Data

The corpus gathered with this setup comprises 70 dialogues, 1772 turns and 17076 words. Example 1 shows a typical multimodal clarification sub-dialogue, [1] concerning an uncertain reference (note that "Venus" is an album name, song title, and an artist name), where the wizard selects a screen output while asking a CR.

(1) **User:** Please play "Venus".
Wizard: Does this list contain the song?
[shows list with 20 DB matches]
User: Yes. It's number 4. *[clicks on item 4]*

For each session we gathered logging information which consists of e.g., the transcriptions of the spoken utterances, the wizard's database query and the number of results, the screen option chosen by the wizard, classification of CRs, etc. We transformed the log-files into an XML structure, consisting of sessions per user, dialogues per task, and turns.[2]

2.2 Data analysis:

Of the 774 wizard turns 19.6% were annotated as CRs, resulting in 152 instances for learning, where our six wizards contributed about equal proportions. A χ^2 test on multimodal strategy (i.e. showing a screen output or not with a CR) showed significant differences between wizards ($\chi^2(1) = 34.21, p < .000$). On the other hand, a Kruskal-Wallis test comparing user preference for the multimodal output showed no significant difference across wizards (H(5)=10.94, $p > .05$). [3] Mean performance ratings for the wizards' multimodal behaviour ranged from 1.67 to 3.5 on a five-point Likert scale. Observing significantly different strategies which are not significantly different in terms of user satisfaction, we conjecture that the wizards converged on strategies which were appropriate in certain *contexts*. To strengthen this

[1] Translated from German.

[2] Where a new "turn" begins at the start of each new user utterance after a wizard utterance, taking the user utterance as a most basic unit of dialogue progression as defined in (Paek and Chickering, 2005).

[3] The Kruskal-Wallis test is the non-parametric equivalent to a one-way ANOVA. Since the users indicated their satisfaction on a 5-point likert scale, an ANOVA which assumes normality would be invalid.

hypothesis we split the data by wizard and and performed a Kruskal-Wallis test on multimodal behaviour per session. Only the two wizards with the lowest performance score showed no significant variation across session, whereas the wizards with the highest scores showed the most varying behaviour. These results again indicate a context dependent strategy. In the following we test this hypothesis (that good multimodal clarification strategies are context-dependent) by building a prediction model of the strategy an *average* wizard took dependent on certain context features.

3 Context/Information-State Features

A state or context in our system is a dialogue information state as defined in (Lemon et al., 2005). We divide the types of information represented in the dialogue information state into *local features* (comprising low level and dialogue features), *dialogue history features*, and *user model features*. We also defined features reflecting the application environment (e.g. `driving`). All features are automatically extracted from the XML log-files (and are available at runtime in ISU-based dialogue systems). From these features we want to learn whether to generate a screen output (`graphic-yes`), or whether to clarify using speech only (`graphic-no`). The case that the wizard only used screen output for clarification did not occur.

3.1 Local Features

First, we extracted features present in the "local" context of a CR, such as the number of matches returned from the data base query (`DBmatches`), how many words were deleted by the corruption algorithm[4] (`deletion`), what problem source the wizard indicated in the pop-up questionnaire (`source`), the previous user speech act (`userSpeechAct`), and the delay between the last wizard utterance and the user's reply (`delay`). [5]

One decision to take for extracting these local features was how to define the "local" context of a CR. As shown in table 1, we experimented with a number of different context definitions. Context 1 defined the local context to be the current turn only, i.e. the turn containing the CR. Context 2 also considered the current turn and the turn following (and is thus not a "runtime" context). Context 3 considered the current turn and the previous turn. Context 4 is the maximal definition of a local context, namely the previous, current, and next turn (also not available at runtime). [6]

id	**Context (turns)**	**acc/ wf-score majority(%)**	**acc/ wf-score Naïve Bayes (%)**
1	only current turn	83.0/54.9	81.0/68.3
2	current and next	71.3/50.4	72.01/68.2
3	**current and previous**	60.50/59.8	76.0*/75.3
4	previous, current, next	67.8/48.9	76.9*/ 74.8

Table 1: Comparison of context definitions for local features (* denotes $p < .05$)

To find the context type which provides the richest information to a classifier, we compared the accuracy achieved in a 10-fold cross validation by a Naïve Bayes classifier (as a standard) on these data sets against the majority class baseline, using a paired t-test, we found that that for context 3 and context 4, Naïve Bayes shows a significant improvement (with $p < .05$ using Bonferroni correction). In table 1 we also show the weighted f-scores since they show that the high accuracy achieved using the first two contexts is due to over-prediction. We chose to use context 3, since these features will be available during system runtime and the learnt strategy could be implemented in an actual system.

3.2 Dialogue History Features

The history features account for events in the whole dialogue so far, i.e. all information gathered before asking the CR, such as the number of CRs asked (`CRhist`), how often the screen output was already used (`screenHist`), the corruption rate so far (`delHist`), the dialogue duration so far (`duration`), and whether the user reacted to the screen output, either by verbally referencing (`refHist`) , e.g. using expressions such as "It's item number 4", or by clicking (`clickHist`) as in example 1.

3.3 User Model Features

Under "user model features" we consider features reflecting the wizards' responsiveness to the be-

[4]Note that this feature is only an approximation of the ASR confidence score that we would expect in an automated dialogue system. See (Rieser et al., 2005) for full details.

[5]We introduced the `delay` feature to handle clarifications concerning contact.

[6]Note that dependent on the context definition a CR might get annotated differently, since placing the question and showing the graphic might be asynchronous events.

haviour and situation of the user. Each session comprised four dialogues with one wizard. The user model features average the user's behaviour in these dialogues so far, such as how responsive the user is towards the screen output, i.e. how often this user clicks (`clickUser`) and how frequently s/he uses verbal references (`refUser`); how often the wizard had already shown a screen output (`screenUser`) and how many CRs were already asked (`CRuser`); how much the user's speech was corrupted on average (`delUser`), i.e. an approximation of how well this user is recognised; and whether this user is currently driving or not (`driving`). This information was available to the wizard.

```
LOCAL FEATURES
        DBmatches: 20
        deletion: 0
        source: reference resolution
        userSpeechAct: command
        delay: 0

HISTORY FEATURES
        [CRhist, screenHist, delHist,
         refHist, clickHist]=0
        duration= 10s

USER MODEL FEATURES
        [clickUser,refUser,screenUser,
         CRuser]=0
        driving= true
```

Figure 2: Features in the context after the first turn in example 1.

3.4 Discussion

Note that all these features are generic over information-seeking dialogues where database results can be displayed on a screen; except for `driving` which only applies to hands-and-eyes-busy situations. Figure 2 shows a context for example 1, assuming that it was the first utterance by this user.

This potential feature space comprises 18 features, many of them taking numeric attributes as values. Considering our limited data set of 152 training instances we run the risk of severe data sparsity. Furthermore we want to explore which features of this potential feature space influenced the wizards' multimodal strategy. In the next two sections we describe feature engineering techniques, namely discretising methods for dimensionality reduction and feature selection methods, which help to reduce the feature space to a subset which is most predictive of multimodal clarification. For our experiments we use implementations of discretisation and feature selection methods provided by the WEKA toolkit (Witten and Frank, 2005).

4 Feature Engineering

4.1 Discretising Numeric Features

Global discretisation methods divide all continuous features into a smaller number of distinct ranges before learning starts. This has two advantages concerning the quality of our data for ML. First, discretisation methods take feature distributions into account and help to avoid sparse data. Second, most of our features are highly positively skewed. Some ML methods (such as the standard extension of the Naïve Bayes classifier to handle numeric features) assume that numeric attributes have a normal distribution. We use Proportional k-Interval (PKI) discretisation as a unsupervised method, and an entropy-based algorithm (Fayyad and Irani, 1993) based on the Minimal Description Length (MDL) principle as a supervised discretisation method.

4.2 Feature Selection

Feature selection refers to the problem of selecting an optimum subset of features that are most predictive of a given outcome. The objective of selection is two-fold: improving the prediction performance of ML models and providing a better understanding of the underlying concepts that generated the data. We chose to apply forward selection for all our experiments given our large feature set, which might include redundant features. We use the following feature filtering methods: correlation-based subset evaluation (CFS) (Hall, 2000) and a decision tree algorithm (rule-based ML) for selecting features before doing the actual learning. We also used a wrapper method called *Selective Naïve Bayes*, which has been shown to perform reliably well in practice (Langley and Sage, 1994). We also apply a correlation-based ranking technique since subset selection models inner-feature relations at the expense of saying less about individual feature performance itself.

4.3 Results for PKI and MDL Discretisation

Feature selection and discretisation influence one-another, i.e. feature selection performs differently on PKI or MDL discretised data. MDL discretisation reduces our range of feature values dramatically. It fails to discretise 10 of 14 numeric features and bars those features from playing a role in the final decision structure because the same discretised value will be given to all instances. However, MDL discretisation cannot replace proper feature selection methods since

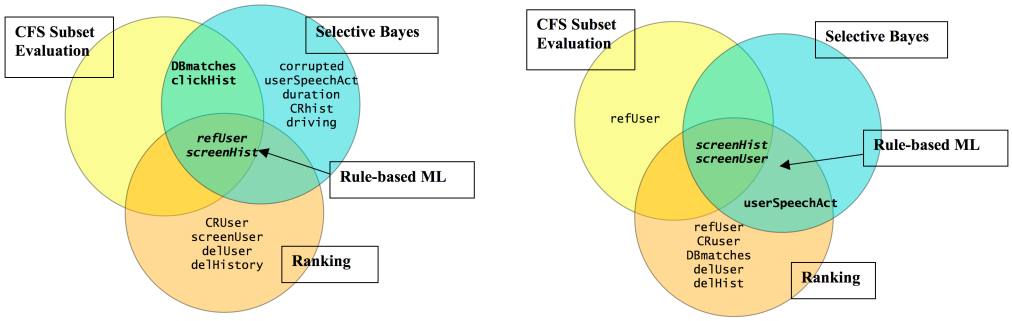


Table 2: Feature selection on PKI-discretised data (left) and on MDL-discretised data (right)

it doesn't explicitly account for redundancy between features, nor for non-numerical features. For the other 4 features which were discretised there is a binary split around one (fairly low) threshold: `screenHist` (.5), `refUser` (.375), `screenUser` (1.0), `CRUser` (1.25).

Table 2 shows two figures illustrating the different subsets of features chosen by the feature selection algorithms on discretised data. From these four subsets we extracted a fifth, using all the features which were chosen by at least two of the feature selection methods, i.e. the features in the overlapping circle regions shown in figure 2. For both data sets the highest ranking features are also the ones contained in the overlapping regions, which are `screenUser`, `refUser` and `screenHist`. For implementation dialogue management needs to keep track of whether the user already saw a screen output in a previous interaction (`screenUser`), or in the same dialogue (`screenHist`), and whether this user (verbally) reacted to the screen output (`refUser`).

5 Performance of Different Learners and Feature Engineering

In this section we evaluate the performance of feature engineering methods in combination with different ML algorithms (where we treat feature optimisation as an integral part of the training process). All experiments are carried out using 10-fold cross-validation. We take an approach similar to (Daelemans et al., 2003) where parameters of the classifier are optimised with respect to feature selection. We use a wide range of different multivariate classifiers which reflect our hypothesis that a decision is based on various features in the context, and compare them against two simple baseline strategies, reflecting deterministic contextual behaviour.

5.1 Baselines

The simplest baseline we can consider is to always predict the majority class in the data, in our case `graphic-no`. This yields a 45.6% wf-score. This baseline reflects a deterministic wizard strategy never showing a screen output.

A more interesting baseline is obtained by using a 1-rule classifier. It chooses the feature which produces the minimum error (which is `refUser` for the PKI discretised data set, and `screenHist` for the MDL set). We use the implementation of a one-rule classifier provided in the WEKA toolkit. This yields a 59.8% wf-score. This baseline reflects a deterministic wizard strategy which is based on a single feature only.

5.2 Machine Learners

For learning we experiment with five different types of supervised classifiers.We chose Naïve Bayes as a joint (generative) probabilistic model, using the WEKA implementation of (John and Langley, 1995)'s classifier; Bayesian Networks as a graphical generative model, again using the WEKA implementation; and we chose maxEnt as a discriminative (conditional) model, using the Maximum Entropy toolkit (Le, 2003). As a rule induction algorithm we used JRIP, the WEKA implementation of (Cohen, 1995)'s Repeated Incremental Pruning to Produce Error Reduction (RIPPER). And for decision trees we used the J4.8 classifier (WEKA's implementation of the C4.5 system (Quinlan, 1993)).

5.3 Comparison of Results

We experimented using these different classifiers on raw data, on MDL and PKI discretised data, and on discretised data using the different feature selection algorithms. To compare the classification outcomes we report on two measures: accuracy and wf-score, which is the weighted

Feature transformation/ (acc./ wf-score (%))	1-rule baseline	Rule Induction	Decision Tree	maxEnt	Naïve Bayes	Bayesian Network	Average
raw data	60.5/**59.8**	76.3/78.3	79.4/78.6	70.0/75.3	76.0/75.3	79.5/72.0	73.62/73.21
PKI + all features	60.5/ 64.6	67.1/66.4	77.4/76.3	70.7/76.7	77.5/81.6	77.3/82.3	71.75/74.65
PKI+ CFS subset	60.5/64.4	68.7/70.7	79.2/76.9	76.7/79.4	78.2/80.6	77.4/80.7	73.45/75.45
PKI+ rule-based ML	60.5/66.5	72.8/76.1	76.0/73.9	75.3/80.2	80.1/78.3	80.8/79.8	74.25/75.80
PKI+ selective Bayes	60.5/64.4	68.2/65.2	78.4/77.9	79.3/78.1	84.6/**85.3**	84.5/84.6	75.92/75.92
PKI+ subset overlap	60.5/64.4	70.9/70.7	75.9/76.9	76.7/78.2	84.0/80.6	83.7/80.7	75.28/75.25
MDL + all features	60.5/69.9	79.0/78.8	78.0/78.1	71.3/76.8	74.9/73.3	74.7/73.9	73.07/75.13
MDL + CFS subset	60.5/69.9	80.1/78.2	80.6/78.2	76.0/80.2	75.7/75.8	75.7/75.8	74.77/76.35
MDL + rule-based ML	60.5/75.5	80.4/81.6	78.7/80.2	79.3/78.8	82.7/82.9	82.7/82.9	77.38/80.32
MDL + select. Bayes	60.5/75.5	80.4/81.6	78.7/80.8	79.3/80.1	82.7/82.9	82.7/82.9	77.38/80.63
MDL + overlap	60.5/75.5	80.4/81.6	78.7/80.8	79.3/80.1	82.7/82.9	82.7/82.9	77.38/80.63
average	60.5/68.24	74.9/75.38	78.26/78.06	75.27/78.54	79.91/79.96	80.16/79.86	

Table 3: Average accuracy and wf-scores for models in feature engineering experiments .

sum (by class frequency in the data; 39.5% `graphic-yes`, 60.5% `graphic-no`) of the f-scores of the individual classes. In table 3 we see fairly stable high performance for Bayesian models with MDL feature selection. However, the best performing model is Naïve Bayes using wrapper methods (selective Bayes) for feature selection and PKI discretisation. This model achieves a wf-score of 85.3%, which is a 25.5% improvement over the 1-rule baseline.

We separately explore the models and feature engineering techniques and their impact on the prediction accuracy for each trial/cross-validation. In the following we separate out the independent contribution of models and features. To assess the effects of models, feature discretisation and selection on performance accuracy, we conduct a hierarchical regression analysis. The models alone explain 18.1% of the variation in accuracy ($R_2 = .181$) whereas discretisation methods only contribute 0.4% and feature selection 1% ($R_2 = .195$). All parameters, except for discretisation methods have a significant impact on modelling accuracy ($P < .001$), indicating that feature selection is an essential step for predicting wizard behaviour. The coefficients of the regression model lead us to the following hypotheses which we explore by comparing the group means for models, discretisation, and features selection methods. Applying a Kruskal-Wallis test with Mann-Whitney tests as a post-hoc procedure (using Bonferroni correction for multiple comparisons), we obtained the following results: [7]

- All ML algorithms are significantly better than the majority and one-rule baselines. All except maxEnt are significantly better than the Rule Induction algorithm. There is no significant difference in the performance of Decision Tree, maxEnt, Naïve Bayes, and Bayesian Network classifiers. Multivariate models being significantly better than the two baseline models indicates that we have a strategy that is based on context features.

- For discretisation methods we found that the classifiers were performing significantly better on MDL discretised data than on PKI or continuous data. MDL being significantly better than continuous data indicates that all wizards behaved as though using thresholds to make their decisions, and MDL being better than PKI supports the hypothesis that decisions were context dependent.

- All feature selection methods (except for CFS) lead to better performance than using all of the features. Selective Bayes and rule-based ML selection performed significantly better than CFS. Selective Bayes, rule-based ML, and subset-overlap showed no significant differences. These results show that wizards behaved as though specific features were important (but they suggest that inner-feature relations used by CFS are less important).

Discussion of results: These experimental results show two things. First, the results indicate that we can learn a good prediction model from our data. We conclude that our six wizards did not behave arbitrarily, but selected their strategy according to certain contextual features. By separating out the individual contributions of models and feature engineering techniques, we have shown that wizard behaviour is based on multiple features. In sum, Decision Tree, max-

[7] We cannot report full details here. Supplementary material is available at `www.coli.uni-saarland.de/~vrieser/acl06-supplementary.html`

Ent, Naïve Bayes, and Bayesian Network classifiers on MDL discretised data using Selective Bayes and Rule-based ML selection achieved the best results. The best performing feature subset was `screenUser,screenHist,` and `userSpeechAct`. The best performing model uses the richest feature space including the feature `driving`.

Second, the regression analysis shows that using these feature engineering techniques in combination with improved ML algorithms is an essential step for learning good prediction models from the small data sets which are typically available from multimodal WOZ studies.

6 Interpretation of the learnt Strategy

For interpreting the learnt strategies we discuss Rule Induction and Decision Trees since they are the easiest to interpret (and to implement in standard rule-based dialogue systems). For both we explain the results obtained by MDL and selective Bayes, since this combination leads to the best performance.

Rule induction: Figure 3 shows a reformulation of the rules from which the learned classifier is constructed. The feature `screenUser` plays a central role. These rules (in combination with the low thresholds) say that if you have already shown a screen output to this particular user in any previous turn (i.e. `screenUser > 1`), then do so again if the previous user speech act was a command (i.e. `userSpeechAct=command`) or if you have already shown a screen output in a previous turn in this dialogue (i.e. `screenHist>0.5`). Otherwise don't show screen output when asking a clarification.

Decision tree: Figure 4 shows the decision tree learnt by the classifier J4.8. The five rules contained in this tree also heavily rely on the user model as well as the previous screen history. The rules constructed by the first two nodes (`screenUser, screenHist`) may lead to a repetitive strategy since the right branch will result in the same action (`graphic-yes`) in all future actions. The only variation is introduced by the speech act, collapsing the tree to the same rule set as in figure 3. Note that this rule-set is based on domain independent features.

Discussion: Examining the classifications made by our best performing Bayesian models we found that the learnt conditional probability distributions produce similar feature-value mappings to the rules described above. The strategy learnt by the classifiers heavily depends on features obtained in previous interactions, i.e. user model features. Furthermore these strategies can lead to repetitive action, i.e. if a screen output was once shown to this user, and the user has previously used or referred to the screen, the screen will be used over and over again.

For learning a strategy which varies in context but adapts in more subtle ways (e.g. to the user model), we would need to explore many more strategies through interactions with users to find an optimal one. One way to reduce costs for building such an optimised strategy is to apply Reinforcement Learning (RL) with simulated users. In future work we will begin with the strategy learnt by supervised learning (which reflects sub-optimal average wizard behaviour) and optimise it for different user models and reward structures.

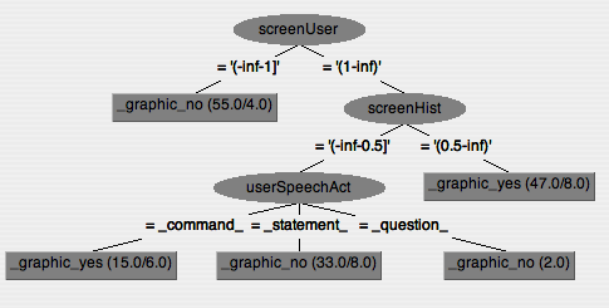


Figure 4: Five-rule tree from J4.8 ("inf" = ∞)

7 Summary and Future Work

We showed that humans use a context-dependent strategy for asking multimodal clarification requests by learning such a strategy from WOZ data. Only the two wizards with the lowest performance scores showed no significant variation across sessions, leading us to hypothesise that the better wizards converged on a context-dependent strategy. We were able to discover a runtime context based on which all wizards behaved uniformly, using feature discretisation methods and feature selection methods on dialogue context features. Based on these features we were able to predict how an 'average' wizard would behave in that context with an accuracy of 84.6% (wf-score of 85.3%, which is a 25.5% improvement over a one rule-based baseline). We explained the learned strategies and showed that they can be implemented in

```
IF screenUser>1 AND (userSpeechAct=command OR screenHist>0.5) THEN graphic=yes
ELSE graphic=no
```

Figure 3: Reformulation of the rules learnt by JRIP

rule-based dialogue systems based on domain independent features. We also showed that feature engineering is essential for achieving significant performance gains when using large feature spaces with the small data sets which are typical of dialogue WOZ studies. By interpreting the learnt strategies we found them to be sub-optimal. In current research, RL is applied to optimise strategies and has been shown to lead to dialogue strategies which are better than those present in the original data (Henderson et al., 2005). The next step towards a RL-based system is to add task-level and reward-level annotations to calculate reward functions, as discussed in (Rieser et al., 2005). We furthermore aim to learn more refined clarification strategies indicating the problem source and its severity.

Acknowledgements

The authors would like thank the ACL reviewers, Alissa Melinger, and Joel Tetreault for help and discussion. This work is supported by the TALK project, `www.talk-project.org`, and the International Post-Graduate College for Language Technology and Cognitive Systems, Saarbrücken.

References

William W. Cohen. 1995. Fast effective rule induction. In *Proceedings of the 12th ICML-95*.

Walter Daelemans, Véronique Hoste, Fien De Meulder, and Bart Naudts. 2003. Combined optimization of feature selection and algorithm parameter interaction in machine learning of language. In *Proceedings of the 14th ECML-03*.

Usama Fayyad and Keki Irani. 1993. Multi-interval discretization of continuousvalued attributes for classification learning. In *Proc. IJCAI-93*.

Mark Hall. 2000. Correlation-based feature selection for discrete and numeric class machine learning. In *Proc. 17th Int Conf. on Machine Learning*.

James Henderson, Oliver Lemon, and Kallirroi Georgila. 2005. Hybrid Reinforcement/Supervised Learning for Dialogue Policies from COMMUNICATOR data. In *IJCAI workshop on Knowledge and Reasoning in Practical Dialogue Systems,*.

George John and Pat Langley. 1995. Estimating continuous distributions in bayesian classifiers. In *Proceedings of the 11th UAI-95*. Morgan Kaufmann.

Ivana Kruijff-Korbayová, Nate Blaylock, Ciprian Gerstenberger, Verena Rieser, Tilman Becker, Michael Kaisser, Peter Poller, and Jan Schehl. 2005. An experiment setup for collecting data for adaptive output planning in a multimodal dialogue system. In *10th European Workshop on NLG*.

Pat Langley and Stephanie Sage. 1994. Induction of selective bayesian classifiers. In *Proceedings of the 10th UAI-94*.

Zhang Le. 2003. Maximum entropy modeling toolkit for Python and C++.

Oliver Lemon, Kallirroi Georgila, James Henderson, Malte Gabsdil, Ivan Meza-Ruiz, and Steve Young. 2005. Deliverable d4.1: Integration of learning and adaptivity with the ISU approach.

Sharon Oviatt, Rachel Coulston, and Rebecca Lunsford. 2004. When do we interact multimodally? Cognitive load and multimodal communication patterns. In *Proceedings of the 6th ICMI-04*.

Sharon Oviatt. 2002. Breaking the robustness barrier: Recent progress on the design of robust multimodal systems. In *Advances in Computers*. Academic Press.

Tim Paek and David Maxwell Chickering. 2005. The markov assumption in spoken dialogue management. In *Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue*.

Ross Quinlan. 1993. *C4.5: Programs for Machine Learning.* Morgan Kaufmann.

Verena Rieser and Johanna Moore. 2005. Implications for Generating Clarification Requests in Task-oriented Dialogues. In *Proceedings of the 43rd ACL*.

Verena Rieser, Ivana Kruijff-Korbayová, and Oliver Lemon. 2005. A corpus collection and annotation framework for learning multimodal clarification strategies. In *Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue*.

David Traum and Pierre Dillenbourg. 1996. Miscommunication in multi-modal collaboration. In *Proceedings of the Workshop on Detecting, Repairing, and Preventing Human-Machine Miscommunication*. AAAI-96.

Ian H. Witten and Eibe Frank. 2005. *Data Mining: Practical Machine Learning Tools and Techniques (Second Edition)*. Morgan Kaufmann.

URES : an Unsupervised Web Relation Extraction System

Benjamin Rosenfeld
Computer Science Department
Bar-Ilan University
Ramat-Gan, ISRAEL
grurgrur@gmail.com

Ronen Feldman
Computer Science Department
Bar-Ilan University
Ramat-Gan, ISRAEL
feldman@cs.biu.ac.il

Abstract

Most information extraction systems either use hand written extraction patterns or use a machine learning algorithm that is trained on a manually annotated corpus. Both of these approaches require massive human effort and hence prevent information extraction from becoming more widely applicable. In this paper we present URES (Unsupervised Relation Extraction System), which extracts relations from the Web in a totally unsupervised way. It takes as input the descriptions of the target relations, which include the names of the predicates, the types of their attributes, and several seed instances of the relations. Then the system downloads from the Web a large collection of pages that are likely to contain instances of the target relations. From those pages, utilizing the known seed instances, the system learns the relation patterns, which are then used for extraction. We present several experiments in which we learn patterns and extract instances of a set of several common IE relations, comparing several pattern learning and filtering setups. We demonstrate that using simple noun phrase tagger is sufficient as a base for accurate patterns. However, having a named entity recognizer, which is able to recognize the types of the relation attributes significantly, enhances the extraction performance. We also compare our approach with KnowItAll's fixed generic patterns.

1 Introduction

The most common preprocessing technique for text mining is information extraction (IE). It is defined as the task of extracting knowledge out of textual documents. In general, IE is divided into two main types of extraction tasks – *Entity tagging* and *Relation extraction*.

The main approaches used by most information extraction systems are the knowledge engineering approach and the machine learning approach. The knowledge engineering (mostly rule based) systems traditionally were the top performers in most IE benchmarks, such as MUC (Chinchor, Hirschman et al. 1994), ACE and the KDD CUP (Yeh and Hirschman 2002). Recently though, the machine learning systems became state-of-the-art, especially for simpler tagging problems, such as named entity recognition (Bikel, Miller et al. 1997), or field extraction (McCallum, Freitag et al. 2000). The general idea is that a domain expert labels the target concepts in a set of documents. The system then learns a model of the extraction task, which can be applied to new documents automatically.

Both of these approaches require massive human effort and hence prevent information extraction from becoming more widely applicable. In order to minimize the huge manual effort involved with building information extraction systems, we have designed and developed URES (Unsupervised Relation Extraction System) which learns a set of patterns to extract relations from the web in a totally unsupervised way. The system takes as input the names of the target relations, the types of its arguments, and a small set of seed instances of the relations. It then uses a large set of unlabeled documents downloaded from the Web in order to build extraction patterns. URES patterns currently have two modes of operation. One is based upon a generic shallow parser, able to extract noun phrases and their

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 667–674,
Sydney, July 2006.

heads. Another mode builds patterns for use by TEG (Rosenfeld, Feldman et al. 2004). TEG is a hybrid rule-based and statistical IE system. It utilizes a trained labeled corpus in order to complement and enhance the performance of a relatively small set of manually-built extraction rules. When it is used with URES, the relation extraction rules and training data are not built manually but are created automatically from the URES-learned patterns. However, URES does not built rules and training data for entity extraction. For those, we use the grammar and training data we developed separately.

It is important to note that URES is not a classic IE system. Its purpose is to extract as many as possible different instances of the given relations while maintaining a high precision. Since the goal is to extract *instances* and not *mentions*, we are quite willing to miss a particular sentence containing an instance of a target relation – if the instance can be found elsewhere. In contrast, the classical IE systems extract *mentions* of entities and relations from the input documents. This difference in goals leads to different ways of measuring the performance of the systems.

The rest of the paper is organized as follows: in Section 2 we present the related work. In Section 3 we outline the general design principles of URES and the architecture of the system and then describe the different components of URES in details while giving examples to the input and output of each component. In Section 4 we present our experimental evaluation and then wrap up with conclusions and suggestions for future work.

2 Related Work

Information Extraction (IE) is a sub-field of NLP, aims at aiding people to sift through large volume of documents by automatically identifying and tagging key entities, facts and events mentioned in the text.

Over the years, much effort has been invested in developing accurate and efficient IE systems. Some of the systems are rule-based (Fisher, Soderland et al. 1995; Soderland 1999), some are statistical (Bikel, Miller et al. 1997; Collins and Miller 1998; Manning and Schutze 1999; Miller, Schwartz et al. 1999) and some are based on inductive-logic-based (Zelle and Mooney. 1996; Califf and Mooney 1998). Recent IE research with bootstrap learning (Brin 1998; Riloff and Jones 1999; Phillips and Riloff 2002; Thelen and Riloff 2002) or learning from documents tagged as relevant (Riloff 1996; Sudo, Sekine et al. 2001) has decreased, but not eliminated hand-tagged training.

Snowball (Agichtein and Gravano 2000) is an unsupervised system for learning relations from document collections. The system takes as input a set of seed examples for each relation, and uses a clustering technique to learn patterns from the seed examples. It does rely on a full fledges Named Entity Recognition system. Snowball achieved fairly low precision figures (30-50%) on relations such as merger and acquisition on the same dataset used in our experiments.

KnowItAll system is a direct predecessor of URES. It is developed at University of Washington by Oren Etzioni and colleagues (Etzioni, Cafarella et al. 2005). KnowItAll is an autonomous, domain-independent system that extracts facts from the Web. The primary focus of the system is on extracting entities (unary predicates). The input to KnowItAll is a set of entity classes to be extracted, such as "city", "scientist", "movie", etc., and the output is a list of entities extracted from the Web. KnowItAll uses a set of manually-built generic rules, which are instantiated with the target predicate names, producing queries, patterns and discriminator phrases. The queries are passed to a search engine, the suggested pages are downloaded and processed with patterns. Every time a pattern is matched, the extraction is generated and evaluated using Web statistics – the number of search engine hits of the extraction alone and the extraction together with discriminator phrases. KnowItAll has also a pattern learning module (PL) that is able to learn patterns for extracting entities. However, it is unsuitable for learning patterns for relations. Hence, for extracting relations KnowItAll currently uses only the generic hand written patterns.

3 Description of URES

The goal of URES is extracting instances of relations from the Web without human supervision. Accordingly, the input of the system is limited to (reasonably short) definition of the target relations. The output of the system is a large list of relation instances, ordered by confidence. The system consists of several largely independent components. The *Sentence Gatherer* generates (e.g., downloads from the Web) a large set of sentences that may contain target instances. The *Pattern Learner* uses a small number of known seed instances to learn likely patterns of relation

occurrences. The *Sentence Classifier* filters the set of sentences, removing those that are unlikely to contain instances of the target relations. The *Instance Extractor* extracts the attributes of the instances from the sentences, and generates the output of the system.

3.1 Sentence Gatherer

The Sentence Gatherer is currently implemented in a very simple way. It gets a set of keywords as input, and proceeds to download all documents that contain one of those keywords. From the documents, it extracts all sentences that contain at least one of the keywords.

The keywords for a relation are the words that are indicative of instances of the relation. The keywords are given to the system as part of the relation definition. Their number is usually small. For instance, the set of keywords for *Acquisition* in our experiments contains two words – "acquired" and "acquisition". Additional keywords (such as "acquire", "purchased", and "hostile takeover") can be added automatically by using WordNet (Miller 1995).

3.2 Pattern Learner

The task of the Pattern Learner is to learn the patterns of occurrence of relation instances. This is an inherently supervised task, because at least some occurrences must be known in order to be able to find patterns among them. Consequently, the input to the Pattern Learner includes a small set (10-15 instances) of known instances for each target relation. Our system assumes that the seeds are a part of the target relation definition. However, the seeds need not be created manually. Instead, they can be taken from the top-scoring results of a high-precision low-recall unsupervised extraction system, such as KnowItAll. The seeds for our experiments were produced in exactly this way.

The Pattern Learner proceeds as follows: first, the gathered sentences that contain the seed instances are used to generate the *positive* and *negative sets*. From those sets the pattern are learned. Then, the patterns are post-processed and filtered. We shall now describe those steps in detail.

Preparing the positive and negative sets

The positive set of a predicate (the terms *predicate* and *relation* are interchangeable in our work) consists of sentences that contain a known instance of the predicate, with the instance attributes changed to "*<AttrN>*", where N is the attribute index. For example, assuming there is a seed instance *Acquisition*(*Oracle, PeopleSoft*), the sentence

> *The Antitrust Division of the U.S. Department of Justice evaluated the likely competitive effects of Oracle's proposed acquisition of PeopleSoft.*

will be changed to

> *The Antitrust Division… …of <Attr1>'s proposed acquisition of <Attr2>.*

The positive set of a predicate P is generated straightforwardly, using substring search.

The negative set of a predicate consists of similarly modified sentences with known false instances of the predicate. We build the negative set as a union of two subsets. The first subset is generated from the sentences in the positive set by changing the assignment of one or both attributes to some other suitable entity. In the first mode of operation, when only a shallow parser is available, any suitable noun phrase can be assigned to an attribute. Continuing the example above, the following sentences will be included in the negative set:

> *<Attr1> of <Attr2> evaluated the likely…*
> *<Attr2> of the U.S. … …acquisition of <Attr1>.*
> etc.

In the second mode of operation, when the NER is available, only entities of the correct type get assigned to an attribute.

The other subset of the negative set contains all sentences produced in a similar way from the positive sentences of all other target predicates. We assume without loss of generality that the predicates that are being extracted simultaneously are all disjoint. In addition, the definition of each predicate indicates whether the predicate is symmetric (like "merger") or antisymmetric (like "acquisition"). In the former case, the sentences produced by exchanging the attributes in positive sentences are placed into the positive set, and in the later case – into the negative set of the predicate.

The following pseudo code shows the process of generating the positive and negative sets in detail:

```
Let S be the set of gathered sentences.
For each predicate P
  For each s∈S containing a word from Keywords(P)
    For each known seed P(A1, A2) of the predicate P
      If A1 and A2 are each found exactly once inside s
        For all entities e1, e2 ∈ s, such that e2 ≠ e1, and
            Type(e1) = type of Attr1 of P, and
            Type(e2) = type of Attr2 of P
          Let s' := s with eN changed to "<AttrN>".
          If e1 = A1 and e2 = A2
            Add s' to the PositiveSet(P).
          Elseif e1 = A2 and e2 = A1 and symmetric(P)
            Add s' to the PositiveSet(P).
          Else
            Add s' to the NegativeSet(P).
For each predicate P
  For each predicate P2 ≠ P
    For each sentence s ∈ PositiveSet(P2)
      Put s into the NegativeSet(P).
```

Generating the patterns

The patterns for predicate *P* are generalizations of pairs of sentences from the positive set of *P*. The function *Generalize*(S_1, S_2) is applied to each pair of sentences S_1 and S_2 from the positive set of the predicate. The function generates a pattern that is the best (according to the objective function defined below) generalization of its two arguments. The following pseudo code shows the process of generating the patterns:

```
For each predicate P
  For each pair S1, S2 from PositiveSet(P)
    Let Pattern := Generalize(S1, S2).
    Add Pattern to PatternsSet(P).
```

The patterns are sequences of *tokens*, *skips* (denoted *), *limited skips* (denoted *?) and *slots*. The tokens can match only themselves, the skips match zero or more arbitrary tokens, and slots match instance attributes. The limited skips match zero or more arbitrary tokens, which must not belong to entities of the types equal to the types of the predicate attributes. The *Generalize*(s_1, s_2) function takes two patterns (note, that sentences in the positive and negative sets are patterns without skips) and generates the least (most specific) common generalization of both. The function does a dynamical programming search for the best match between the two patterns (Optimal String Alignment algorithm), with the cost of the match defined as the sum of costs of matches for all elements. We use the following numbers: two identical elements match at cost 0, a token matches a skip or an empty space at cost 10, a skip matches an empty space at cost 2, and different kinds of skip match at cost 3. All other combinations have infinite cost. After the best match is found, it is converted into a pattern by copying matched identical elements and adding skips where non-identical elements are matched. For example, assume the sentences are

Toward this end, <Attr1> in July acquired <Attr2>
Earlier this year, <Attr1> acquired <Attr2> from X

After the dynamical programming-based search, the following match will be found:

Table 1 - Best Match between Sentences

Toward		(cost 10)
	Earlier	(cost 10)
this	*this*	(cost 0)
end		(cost 10)
	year	(cost 10)
,	,	(cost 0)
<Attr1>	*<Attr1>*	(cost 0)
in July		(cost 20)
acquired	*acquired*	(cost 0)
<Attr2>	*<Attr2>*	(cost 0)
	from	(cost 10)
	X	(cost 10)

at total cost = 80. The match will be converted to the pattern (assuming the NER mode, so the only entity belonging to the same type as one of the attributes is "*X*"):

*? *? *this* *? *? , *<Attr1>* *? *acquired* *<Attr2>* *? *

which becomes, after combining adjacent skips,

*? *this* *? , *<Attr1>* *? *acquired* *<Attr2>* *

Note, that the generalization algorithm allows patterns with any kind of elements beside skips, such as *CapitalWord*, *Number*, *CapitalizedSequence*, etc. As long as the costs and results of matches are properly defined, the *Generalize* function is able to find the best generalization of any two patterns. However, in the present work we stick with the simplest pattern definition as described above.

Post-processing, filtering, and scoring

The number of patterns generated at the previous step is very large. Post-processing and filtering tries to reduce this number, keeping the most useful patterns and removing the too specific and irrelevant ones.

First, we remove from patterns all "stop words" surrounded by skips from both sides,

such as the word "*this*" in the last pattern in the previous subsection. Such words do not add to the discriminative power of patterns, and only needlessly reduce the pattern recall. The list of stop words includes all functional and very common English words, as well as puncuation marks. Note, that the stop words are removed only if they are surrounded by skips, because when they are adjacent to slots or non-stop words they often convey valuable information. After this step, the pattern above becomes

*? , *<Attr1>* *? *acquired* *<Attr2>* *

In the next step of filtering, we remove all patterns that do not contain *relevant* words. For each predicate, the list of relevant words is automatically generated from WordNet by following all links to depth at most 2 starting from the predicate keywords. For example, the pattern

* *<Attr1>* * *by* *<Attr2>* *

will be removed, while the pattern

* *<Attr1>* * *purchased* *<Attr2>* *

will be kept, because the word "*purchased*" can be reached from "*acquisition*" via synonym and derivation links.

The final (optional) filtering step removes all patterns, that contain slots surrounded by skips on both sides, keeping only the patterns, whose slots are adjacent to tokens or to sentence boundaries. Since both the shallow parser and the NER system that we use are far from perfect, they often place the entity boundaries incorrectly. Using only patterns with *anchored* slots significantly improves the precision of the whole system. In our experiments we compare the performance of anchored and unanchored patterns.

The filtered patterns are then scored by their performance on the positive and negative sets. Currently we use a simple scoring method – the score of a pattern is the number of positive matches divided by the number of negative matches plus one:

$$Score(Pattern) = \frac{|\{S \in PositiveSet : Pattern \text{ matches } S\}|}{|\{S \in NegativeSet : Pattern \text{ matches } S\}| + 1}$$

This formula is purely empirical and produces reasonable results. The threshold is applied to the set of patterns, and all patterns scoring less than the threshold (currently, it is set to 6) are discarded.

3.3 Sentence Classifier

The task of the Sentence Classifier is to filter out from the large pool of sentences produced by the Sentence Gatherer the sentences that do not contain the target predicate instances. In the current version of our system, this is only done in order to reduce the number of sentences that need to be processed by the Slot Extractor. Therefore, in this stage we just remove the sentences that do not match any of the regular expressions generated from the patterns. Regular expressions are generated from patterns by replacing slots with skips.

3.4 Instance Extractor

The task of the Instance Extractor is to use the patterns generated by the Pattern Learner on the sentences that were passed through by the Sentence Classifier. However, the patterns cannot be directly matched to the sentences, because the patterns only define the placeholders for instance attributes and cannot by themselves extract the values of the attributes.

We currently have two different ways to solve this problem – using a general-purpose shallow parser, which is able to recognize noun phrases and their heads, and using an information extraction system called TEG (Rosenfeld, Feldman et al. 2004), together with a trained grammar able to recognize the entities of the types of the predicates' attributes. We shall briefly describe the two modes of operation.

Shallow Parser mode

In the first mode of operation, the predicates may define attributes of two different types: *ProperName* and *CommonNP*. We assume that the values of the *ProperName* type are always heads of proper noun phrases. And the values of the
CommonNP type are simple common noun phrases (with possible proper noun modifiers, e.g. "*the Kodak camera*").

We use a Java-written shallow parser from the OpenNLP (http://opennlp.sourceforge.net/) package. Each sentence is tokenized, tagged with part-of-speech, and tagged with noun phrase boundaries. The pattern matching and extraction is straightforward.

TEG mode

TEG (Trainable Extraction Grammars) (Rosenfeld, Feldman et al. 2004) is general-

purpose hybrid rule-based and statistical IE system, able to extract entities and relations at the sentence level. It is adapted to any domain by writing a suitable set of rules, and training them using an annotated corpus. The TEG rule language is a straightforward extension of a context-free grammar syntax. A complete set of rules is compiled into a PCFG (Probabilistic Context Free Grammar), which is then trained upon the training corpus.

Some of the nonterminals inside the TEG grammar can be marked as *target concepts*. Wherever such nonterminal occurs in a final parse of a sentence, TEG generates an output label. The target concept rules may specify some of their parts as attributes. Then the concept is considered to be a relation, with the values of the attributes determined by the concept parse. Concepts without attributes are entities.

For the TEG-based instance extractor we utilize the NER ruleset of TEG and an internal training corpus called INC, as described in (Rosenfeld, Feldman et al. 2004). The ruleset defines a grammar with a set of concepts for *Person*, *Location*, and *Organization* entities. In addition, the grammar defines a generic *NounPhrase* concept, which can be used for capturing the entities that do not belong to any of the entity types above.

In order to do the extraction, the patterns generated by the Pattern Learner are converted to the TEG syntax and added to the pre-built NER grammar. This produces a grammar, which is able to extract relations. This grammar is trained upon the automatically labeled positive set from the Pattern Learning. The resulting trained model is applied to the sets of sentences produced by the Sentence Classifier.

4 Experimental Evaluation

In order to evaluate URES, we used five predicates

Acquisition(*BuyerCompany*, *BoughtCompany*),
Merger(*Company1*, *Company2*),
CEO_Of(*Company*, *Name*),
MayorOf(*City*, *Name*),
InventorOf(*InventorName*, *Invention*).

Merger is *symmetric* predicate, in the sense that the order of its attributes does not matter. *Acquisition* is *antisymmetric*, and the other three are tested as *bound* in the first attribute. For the bound predicates, we are only interested in the instances with particular prespecified values of the first attribute.

We test both modes of operation – using shallow parser and using TEG. In the shallow parser mode, the *Invention* attribute of the *InventorOf* predicate is of type *CommonNP*, and all other attributes are of type *ProperName*. In the TEG mode, the "*Company*" attributes are of type *Organization*, the "*Name*" attributes are of type *Person*, the "*City*" attribute is of type *Location*, and the "*Invention*" attribute is of type *NounPhrase*.

We evaluate our system by running it over a large set of sentences, counting the number of extracted instances, and manually checking a random sample of the instances to estimate precision. In order to be able to compare our results with KnowItAll-produced results, we used the set of sentences collected by the KnowItAll's crawler as if they were produced by the Sentence Gatherer.

The set of sentences for the *Acquisition* and *Merger* predicates contained around 900,000 sentences each. For the other three predicates, each of the sentences contained one of the 100 predefined values for the first attribute. The values (100 companies for *CEO_Of*, 100 cities for *MayorOf*, and 100 inventors for *InventorOf*) are entities collected by KnowItAll, half of them are frequent entities (>100,000 hits), and another half are rare (<10,000 hits).

In all of the experiments, we use ten top predicate instances extracted by KnowItAll for the relation seeds needed by the Pattern Learner.

The results of our experiments are summarized in the Table 2. The table displays the number of extracted instances and estimated precision for three different URES setups, and for the KnowItAll manually built patterns. Three results are shown for each setup and each relation – extractions supported by at least one, at least two, and at least three different sentences, respectively.

Several conclusions can be drawn from the results. First, both modes of URES significantly outperform KnowItAll in recall (number of extractions), while maintaining the same level of precision or improving it. This demonstrates utility of our pattern learning component. Second, it is immediately apparent, that using only anchored patterns significantly improves precision of NP Tagger-based URES, though at a high cost in recall. The NP tagger-based URES with anchored patterns performs somewhat worse than

Table 2 - Experimental results.

		Acquisition		*CEO_Of*		*InventorOf*		*MayorOf*		*Merger*	
	support	Count	Prec	Count	Prec	Count	Prec	Count	Prec	Count	Prec
NP Tagger All patterns	≥ 1	10587	0.74	545	0.7	1233	0.84	2815	0.6	25071	0.71
	≥ 2	815	0.87	221	0.92	333	0.92	717	0.74	2981	0.8
	≥ 3	234	0.9	133	0.94	185	0.96	442	0.84	1245	0.88
NP Tagger Anchored patterns	≥ 1	5803	0.84	447	0.8	1035	0.86	2462	0.65	17107	0.8
	≥ 2	465	0.96	186	0.94	284	0.92	652	0.78	2481	0.83
	≥ 3	148	0.98	123	0.96	159	0.96	411	0.88	1084	0.9
TEG All patterns	≥ 1	8926	0.82	618	0.83	2322	0.65	2434	0.85	15002	0.8
	≥ 2	1261	0.94	244	0.94	592	0.85	779	0.93	2932	0.86
	≥ 3	467	0.98	158	0.98	334	0.88	482	0.98	1443	0.9
KnowItAll	≥ 1	2235	0.84	421	0.81	604	0.8	725	0.76	3233	0.82
	≥ 2	257	0.98	190	0.98	168	0.92	308	0.92	352	0.92

TEG-based URES on all predicates except *InventorOf*, as expected. For the *InventorOf*, TEG performs worse, because of overly simplistic implementation of the *NounPhrase* concept inside the TEG grammar – it is defined as a sequence of zero or more adjectives followed by a sequence of nouns. Such definition often leads to only part of a correct invention name being extracted.

5 Conclusions and Future Work

We have presented the URES system for autonomously extracting relations from the Web. URES bypasses the bottleneck created by classic information extraction systems that either relies on manually developed extraction patterns or on manually tagged training corpus. Instead, the system relies upon learning patterns from a large unlabeled set of sentences downloaded from Web.

One of the topics we would like to further explore is the complexity of the patterns that we learn. Currently we use a very simple pattern language that just has 4 types of elements, slots, constants and two types of skips. We want to see if we can achieve higher precision with more complex patterns. In addition we would like to test URES on n-ary predicates, and to extend the system to handle predicates that are allowed to lack some of the attributes.

References

Agichtein, E. and L. Gravano (2000). Snowball: Extracting Relations from Large Plain-Text Collections. Proceedings of the 5th ACM International Conference on Digital Libraries (DL).

Bikel, D. M., S. Miller, et al. (1997). Nymble: a high-performance learning name-finder. Proceedings of ANLP-97**:** 194-201.

Brin, S. (1998). Extracting Patterns and Relations from the World Wide Web. WebDB Workshop, EDBT '98.

Califf, M. E. and R. J. Mooney (1998). Relational Learning of Pattern-Match Rules for Information Extraction. Working Notes of AAAI Spring Symposium on Applying Machine Learning to Discourse Processing. Menlo Park, CA, AAAI Press**:** 6-11.

Chinchor, N., L. Hirschman, et al. (1994). "Evaluating Message Understanding Systems: An Analysis of the Third Message Understanding Conference (MUC-3)." Computational Linguistics **3**(19): 409-449.

Collins, M. and S. Miller (1998). Semantic Tagging using a Probabilistic Context Free Grammar. Proceedings of the Sixth Workshop on Very Large Corpora.

Etzioni, O., M. Cafarella, et al. (2005). "Unsupervised named-entity extraction from the Web: An experimental study." Artificial Intelligence.

Fisher, D., S. Soderland, et al. (1995). Description of the UMass Systems as Used for MUC-6. 6th Message Understanding Conference**:** 127-140.

Manning, C. and H. Schutze (1999). Foundations of Statistical Natural Language Processing. Cambridge, US, The MIT Press.

McCallum, A., D. Freitag, et al. (2000). Maximum Entropy Markov Models for Information Extraction and Segmentation. Proc. 17th International Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA**:** 591-598.

Miller, D., R. Schwartz, et al. (1999). Named entity extraction from broadcast news. Proceedings of DARPA Broadcast News Workshop. Herndon, VA.

Miller, G. A. (1995). "WordNet: A lexical database for English." CACM **38**(11): 39-41.

Phillips, W. and E. Riloff (2002). Exploiting Strong Syntactic Heuristics and Co-Training to Learn Semantic Lexicons. Conference on Empirical Methods in Natural Language Processing (EMNLP 2002).

Riloff, E. (1996). Automatically Generating Extraction Patterns from Untagged Text. AAAI/IAAI, Vol. 2**:** 1044-1049.

Riloff, E. and R. Jones (1999). Learning Dictionaries for Information Extraction by Multi-level Bootstrapping. Proceedings of the Sixteenth National Conference on Artificial Intelligence, The AAAI Press/MIT Press**:** 1044-1049.

Rosenfeld, B., R. Feldman, et al. (2004). TEG: a hybrid approach to information extraction. CIKM 2004, Arlington, VA.

Soderland, S. (1999). "Learning Information Extraction Rules for Semi-Structured and Free Text." Machine Learning **34**(1-3): 233-272.

Sudo, K., S. Sekine, et al. (2001). Automatic pattern acquisition for Japanese information extraction. Human Language Technology Conference (HTL2001).

Thelen, M. and E. Riloff (2002). A Bootstrapping Method for Learning Semantic Lexicons using Extraction Pattern Contexts. Conference on Empirical Methods in Natural Language Processing (EMNLP 2002).

Yeh, A. and L. Hirschman (2002). "Background and overview for kdd cup 2002 task 1: Information extraction from biomedical articles." KDD Explorarions **4**(2): 87-89.

Zelle, J. M. and R. J. Mooney. (1996). Learning to parse database queries using inductive logic programming. 13th National Conference on Artificial Intelligence (AAAI-96).

Argumentative Feedback: A Linguistically-motivated Term Expansion for Information Retrieval

Patrick Ruch, Imad Tbahriti, Julien Gobeill
Medical Informatics Service
University of Geneva
24 Micheli du Crest
1201 Geneva
Switzerland
{patrick.ruch,julien.gobeill,imad.tbahriti}@hcuge.ch

Alan R. Aronson
Lister Hill Center
National Library of Medicine
8600 Rockville Pike
Bethesda, MD 20894
USA
alan@nlm.nih.gov

Abstract

We report on the development of a new automatic feedback model to improve information retrieval in digital libraries. Our hypothesis is that some particular sentences, selected based on argumentative criteria, can be more useful than others to perform well-known feedback information retrieval tasks. The argumentative model we explore is based on four disjunct classes, which has been very regularly observed in scientific reports: PURPOSE, METHODS, RESULTS, CONCLUSION. To test this hypothesis, we use the Rocchio algorithm as baseline. While Rocchio selects the features to be added to the original query based on statistical evidence, we propose to base our feature selection also on argumentative criteria. Thus, we restrict the expansion on features appearing only in sentences classified into one of our argumentative categories. Our results, obtained on the OHSUMED collection, show a significant improvement when expansion is based on PURPOSE (mean average precision = +23%) and CONCLUSION (mean average precision = +41%) contents rather than on other argumentative contents. These results suggest that argumentation is an important linguistic dimension that could benefit information retrieval.

1 Introduction

Information retrieval (IR) is a challenging endeavor due to problems caused by the underlying expressiveness of all natural languages. One of these problems, synonymy, is that authors and users frequently employ different words or expressions to refer to the same meaning (*accident* may be expressed as *event*, *incident*, *problem*, *difficulty*, *unfortunate situation*, *the subject of your last letter*, *what happened last week*, etc.) (Furnas et al., 1987). Another problem is ambiguity, where a specific term may have several (and sometimes contradictory) meanings and interpretations (e.g., the word *horse* as in *Trojan horse*, *light horse*, *to work like a horse*, *horse about*). In order to obtain better meaning-based matches between queries and documents, various propositions have been suggested, usually without giving any consideration to the underlying domain.

During our participation in different international evaluation campaigns such as the TREC Genomics track (Hersh, 2005), the BioCreative initiative (Hirschman et al., 2005), as well as in our attempts to deliver advanced search tools for biologists (Ruch, 2006) and healthcare providers (Ruch, 2002) (Ruch, 2004), we were more concerned with domain-specific information retrieval in which systems must return a ranked list of MEDLINE records in response to an expert's information request. This involved a set of available queries describing typical search interests, in which gene, protein names, and diseases were often essential for an effective retrieval. Biomedical publications however tend to generate new information very rapidly and also use a wide variation in terminology, thus leading to the current situation whereby a large number of names, symbols and synonyms are used to denote the same concepts. Current solutions to these issues can be classified into domain-specific strategies, such as thesaurus-based expansion, and domain-independent strategies, such as blind-feedback. By proposing to explore a third type of approach, which attempts to take advantage of argumentative specificities of scientific reports, our study initiates a new research direction for natural language processing applied to information retrieval.

The rest of this paper is organized as follows. Section 2 presents some related work in information retrieval and in argumentative parsing, while Section 3 depicts the main characteristics of our test collection and the metrics used in our experiments. Section 4 details the strategy

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 675–682,
Sydney, July 2006.

used to develop our improved feedback method. Section 5 reports on results obtained by varying our model and Section 6 contains conclusions on our experiments.

2 Related works

Our basic experimental hypothesis is that some particular sentences, selected based on argumentative categories, can be more useful than others to support well-known feedback information retrieval tasks. It means that selecting sentences based on argumentative categories can help focusing on content-bearing sections of scientific articles.

2.1 Argumentation

Originally inspired by corpus linguistics studies (Orasan, 2001), which suggests that scientific reports (in chemistry, linguistics, computer sciences, medicine...) exhibit a very regular logical distribution -confirmed by studies conducted on biomedical corpora (Swales, 1990) and by ANSI/ISO professional standards - the argumentative model we experiment is based on four disjunct classes: PURPOSE, METHODS, RESULTS, CONCLUSION.

Argumentation belongs to discourse analysis[1], with fairly complex computational models such as the implementation of the rhetorical structure theory proposed by (Marcu, 1997), which proposes dozens of rhetorical classes. More recent advances were applied to document summarization. Of particular interest for our approach, Teufel and Moens (Teufel and Moens, 1999) propose using a list of manually crafted triggers (using both words and expressions such as *we argued*, *in this article*, *the paper is an attempt to*, *we aim at*, etc.) to automatically structure scientific articles into a lighter model, with only seven categories: BACKGROUND, TOPIC, RELATED WORK, PURPOSE, METHOD, RESULT, and CONCLUSION.

More recently and for knowledge discovery in molecular biology, more elaborated models were proposed by (Mizuta and Collier, 2004) (Mizuta et al., 2005) and by (Lisacek et al., 2005) for novelty-detection. (McKnight and Srinivasan, 2003) propose a model very similar to our four-class model but is inspired by clinical trials. Preliminary applications were proposed for bibliometrics and related-article search (Tbahriti et al., 2004) (Tbahriti et al., 2005), information extraction and passage retrieval (Ruch et al., 2005b). In these studies, sentences were selected as the basic classification unit in order to avoid as far as possible co-reference issues (Hirst, 1981), which hinder readibity of automatically generated and extracted sentences.

[1] After Aristotle, discourses structured following an appropriate argumentative distribution belong to logics, while ill-defined ones belong to rhetorics.

2.2 Query expansion

Various query expansion techniques have been suggested to provide a better match between user information needs and documents, and to increase retrieval effectiveness. The general principle is to expand the query using words or phrases having a similar or related meaning to those appearing in the original request. Various empirical studies based on different IR models or collections have shown that this type of search strategy should usually be effective in enhancing retrieval performance. Scheme propositions such as this should consider the various relationships between words as well as term selection mechanisms and term weighting schemes (Robertson, 1990). The specific answers found to these questions may vary; thus a variety of query expansion approaches were suggested (Efthimiadis, 1996).

In a first attempt to find related search terms, we might ask the user to select additional terms to be included in a new query, e.g. (Velez et al., 1997). This could be handled interactively through displaying a ranked list of retrieved items returned by the first query. Voorhees (Voorhees, 1994) proposed basing a scheme based on the WordNet thesaurus. The author demonstrated that terms having a lexical-semantic relation with the original query words (extracted from a synonym relationship) provided very little improvement (around 1% when compared to the original unexpanded query).

As a second strategy for expanding the original query, Rocchio (Rocchio, 1971) proposed accounting for the relevance or irrelevance of top-ranked documents, according to the user's manual input. In this case, a new query was automatically built in the form of a linear combination of the term included in the previous query and terms automatically extracted from both the relevant documents (with a positive weight) and non-relevant items (with a negative weight). Empirical studies (e.g., (Salton and Buckley, 1990)) demonstrated that such an approach is usually quite effective, and could

be used more than once per query (Aalbersberg, 1992). Buckley et al. (Singhal et al., 1996b) suggested that we could assume, without even looking at them or asking the user, that the top k ranked documents are relevant. Denoted the pseudo-relevance feedback or blind-query expansion approach, this approach is usually effective, at least when handling relatively large text collections.

As a third source, we might use large text corpora to derive various term-term relationships, using statistically or information-based measures (Jones, 1971), (Manning and Schütze, 2000). For example, (Qiu and Frei, 1993) suggested that terms to be added to a new query could be extracted from a similarity thesaurus automatically built through calculating co-occurrence frequencies in the search collection. The underlying effect was to add idiosyncratic terms to the underlying document collection, related to the query terms by language use. When using such query expansion approaches, we can assume that the new terms are more appropriate for the retrieval of pertinent items than are lexically or semantically related terms provided by a general thesaurus or dictionary. To complement this global document analysis, (Croft, 1998) suggested that text passages (with a text window size of between 100 to 300 words) be taken into account. This local document analysis seemed to be more effective than a global term relationship generation.

As a forth source of additional terms, we might account for specific user information needs and/or the underlying domain. In this vein, (Liu and Chu, 2005) suggested that terms related to the user's intention or scenario might be included. In the medical domain, it was observed that users looking for information usually have an underlying scenario in mind (or a typical medical task). Knowing that the number of scenarios for a user is rather limited (e.g., *diagnosis*, *treatment*, *etiology*), the authors suggested automatically building a semantic network based on a domain-specific thesaurus (using the Unified Medical Language System (UMLS) in this case). The effectiveness of this strategy would of course depend on the quality and completeness of domain-specific knowledge sources. Using the well-known term frequency (tf)/inverse document frequency (idf) retrieval model, the domain-specific query-expansion scheme suggested by Liu and Chu (2005) produces better retrieval performance than a scheme based on statistics (MAP: 0.408 without query expansion, 0.433 using statistical methods and 0.452 with domain-specific approaches).

In these different query expansion approaches, various underlying parameters must be specified, and generally there is no single theory able to help us find the most appropriate values. Recent empirical studies conducted in the context of the TREC Genomics track, using the OHSUGEN collection (Hersh, 2005), show that neither blind expansion (Rocchio), nor domain-specific query expansion (thesaurus-based Gene and Protein expansion) seem appropriate to improve retrieval effectiveness (Aronson et al., 2006) (Abdou et al., 2006).

3 Data and metrics

To test our hypothesis, we used the OHSUMED collection (Hersh et al., 1994), originally developed for the TREC topic detection track, which is the most popular information retrieval collection for evaluating information search in library corpora. Alternative collections (cf. (Savoy, 2005)), such as the French Amaryllis collection, are usually smaller and/or not appropriate to evaluate our argumentative classifier, which can only process English documents. Other MEDLINE collections, which can be regarded as similar in size or larger, such as the TREC Genomics 2004 and 2005 collections are unfortunately more domain-specific since information requests in these collection are usually targeting a particular gene or gene product.

Among the 348,566 MEDLINE citations of the OHSUMED collection, we use the 233,455 records provided with an abstract. An example of a MEDLINE citation is given in Table 1: only Title, Abstract, MeSH and Chemical (RN) fields of MEDLINE records were used for indexing. Out of the 105 queries of the OHSUMED collection, only 101 queries have at least one positive relevance judgement, therefore we used only this subset for our experiments. The subset has been randomly split into a training set (75 queries), which is used to select the different parameters of our retrieval model, and a test set (26 queries), used for our final evaluation.

As usual in information retrieval evaluations, the mean average precision, which computes the precision of the engine at different levels (0%, 10%, 20%... 100%) of recall, will be used in our experiments. The precision of the top returned

Title: Computerized extraction of coded findings from free-text radiologic reports. Work in progress.
Abstract: A computerized data acquisition tool, the special purpose radiology understanding system (SPRUS), has been implemented as a module in the Health Evaluation through Logical Processing Hospital Information System. This tool uses semantic information from a diagnostic expert system to parse free-text radiology reports and to extract and encode both the findings and the radiologists' interpretations. These coded findings and interpretations are then stored in a clinical data base. The system recognizes both radiologic findings and diagnostic interpretations. Initial tests showed a true-positive rate of 87% for radiographic findings and a bad data rate of 5%. Diagnostic interpretations are recognized at a rate of 95% with a bad data rate of 6%. Testing suggests that these rates can be improved through enhancements to the system's thesaurus and the computerized medical knowledge that drives it. This system holds promise as a tool to obtain coded radiologic data for research, medical audit, and patient care.
MeSH Terms: *Artificial Intelligence*; Decision Support Techniques; Diagnosis, Computer-Assisted; Documentation; Expert Systems; Hospital Information Systems*; Human; Natural Language Processing*; Online Systems; Radiology Information Systems*.*

Table 1: MEDLINE records with, title, abstract and keyword fields as provided by MEDLINE librarians: major concepts are marked with *; Subheadings and checktags are removed.

document, which is obviously of major importance is also provided together with the total number of relevant retrieved documents for each evaluated run.

4 Methods

To test our experimental hypothesis, we use the Rocchio algorithm as baseline. In addition, we also provide the score obtained by the engine before the feedback step. This measure is necessary to verify that feedback is useful for querying the OHSUMED collection and to establish a strong baseline. While Rocchio selects the features to be added to the original queries based on pure statistical analysis, we propose to base our feature expansion also on argumentative criteria. That is, we overweight features appearing in sentences classified in a particular argumentative category by the argumentative categorizer.

4.1 Retrieval engine and indexing units

The easyIR system is a standard vector-space engine (Ruch, 2004), which computes state-of-the-art *tf.idf* and probabilistic weighting schema. All experiments were conducted with pivoted normalization (Singhal et al., 1996a), which has recently shown some effectiveness on MEDLINE corpora (Aronson et al., 2006). Query and document weighings are provided in Equation (1): the dtu formula is applied to the documents, while the dtn formula is applied to the query; t the number of indexing terms, df_j the number of documents in which the term t_j; pivot and slope are constants (fixed at pivot = 0.14, slope = 146).

$$\begin{array}{l}\text{dtu: } w_{ij} = \frac{(Ln(Ln(tf_{ij})+1)+1)\cdot idf_j}{(1-slope)\cdot pivot+slope\cdot nt_i} \\ \text{dtn: } w_{ij} = idf_j \cdot (Ln(Ln(tf_{if})+1)+1)\end{array} \quad (1)$$

As already observed in several linguistically-motivated studies (Hull, 1996), we observe that common stemming methods do not perform well on MEDLINE collections (Abdou et al., 2006), therefore indexing units are stored in the inverted file using a simple S-stemmer (Harman, 1991), which basically handles most frequent plural forms and exceptions of the English language such as *-ies*, *-es* and *-s* and exclude endings such as *-aies*, *-eies*, *-ss*, etc. This simple normalization procedure performs better than others and better than no stemming. We also use a slightly modified standard stopword list of 544 items, where strings such as *a*, which stands for *alpha* in chemistry and is relevant in biomedical expressions such as *vitamin a*.

4.2 Argumentative categorizer

The argumentative classifier ranks and categorizes abstract sentences as to their argumentative classes. To implement our argumentative categorizer, we rely on four binary Bayesian classifiers, which use lexical features, and a Markov model, which models the logical distribution of the argumentative classes in MEDLINE abstracts. A comprehensive description of the classifier with feature selection and comparative evaluation can be found in (Ruch et al., 2005a)

To train the classifier, we obtained 19,555 explicitly structured abstracts from MEDLINE. A

Abstract: PURPOSE: The overall prognosis for patients with congestive heart failure is poor. Defining specific populations that might demonstrate improved survival has been difficult [...] PATIENTS AND METHODS: We identified 11 patients with severe congestive heart failure (average ejection fraction 21.9 +/- 4.23% (+/- SD) who developed spontaneous, marked improvement over a period of follow-up lasting 4.25 +/- 1.49 years [...] RESULTS: During the follow-up period, the average ejection fraction improved in 11 patients from 21.9 +/- 4.23% to 56.64 +/- 10.22%. Late follow-up indicates an average ejection fraction of 52.6 +/- 8.55% for the group [...] CONCLUSIONS: We conclude that selected patients with severe congestive heart failure can markedly improve their left ventricular function in association with complete resolution of heart failure [...]

Table 2: MEDLINE records with explicit argumentative markers: PURPOSE, (PATIENTS and) METHODS, RESULTS and CONCLUSION.

	Bayesian classifier			
	PURP.	METH.	RESU.	CONC.
PURP.	80.65 %	0 %	3.23 %	16 %
METH.	8 %	78 %	8 %	6 %
RESU.	18.58 %	5.31 %	52.21 %	23.89 %
CONC.	18.18 %	0 %	2.27 %	79.55 %
	Bayesian classifier with Markov model			
	PURP.	METH.	RESU.	CONC.
PURP.	93.35 %	0 %	3.23 %	3 %
METH.	3 %	78 %	8 %	6 %
RESU.	12.73 %	2.07 %	57.15 %	10.01 %
CONC.	2.27 %	0 %	2.27 %	95.45 %

Table 3: Confusion matrix for argumentative classification. The harmonic means between recall and precision score (or F-score) is in the range of 85% for the combined system.

conjunctive query was used to combine the following four strings: *PURPOSE:*, *METHODS:*, *RESULTS:*, *CONCLUSION:*. From the original set, we retained 12,000 abstracts used for training our categorizer, and 1,200 were used for fine-tuning and evaluating the categorizer, following removal of explicit argumentative markers. An example of an abstract, structured with explicit argumentative labels, is given in Table 2. The per-class performance of the categorizer is given by a contingency matrix in Table 3.

4.3 Rocchio feedback

Various general query expansion approaches have been suggested, and in this paper we compared ours with that of Rocchio. In this latter case, the system was allowed to add m terms extracted from the k best-ranked abstracts from the original query. Each new query was derived by applying the following formula (Equation 2): $Q' = \alpha \cdot Q + (\beta/k) \cdot \sum kj = 1 w_{ij}$ (2), in which Q' denotes the new query built from the previous query Q, and w_{ij} denotes the indexing term weight attached to the term t_j in the document D_i. By direct use of the training data, we determine the optimal values of our model: m = 10, k = 15. In our experiments, we fixed α = 2.0, β = 0.75. Without feedback the mean average precision of the evaluation run is 0.3066, the Rocchio feedback (mean average precision = 0.353) represents an improvement of about 15% (cf. Table 5), which is statistically[2] significant ($p < 0.05$).

4.4 Argumentative selection for feedback

To apply our argumentation-driven feedback strategy, we first have to classify the top-ranked abstracts into our four argumentative moves: PURPOSE, METHODS, RESULTS, and CONCLUSION. For the argumentative feedback, different m and k values are recomputed on the training queries, depending on the argumentative category we want to over-weight. The basic segment is the sentence; therefore the abstract is split into a set of sentences before being processed by the argumentative classifier. The sentence splitter simply applies as set of regular expressions to locate sentence boundaries. The precision of this simple sentence splitter equals 97% on MEDLINE abstracts. In this setting only one argumentative category is attributed to each sentence, which makes the decision model binary.

Table 4 shows the output of the argumentative classifier when applied to an abstract. To determine the respective value of each argumentative contents for feedback, the argumentative categorizer parses each top-ranked abstract. These abstracts are then used to generate four groups of sentences. Each group corresponds to a unique argumentative class. Each argumentative index contains sentences classified in one of four argumentative classes. Because argumen-

[2]Tests are computed using a non-parametric signed test, cf. (Zobel, 1998) for more details.

CONCLUSION (00160116) The highly favorable pathologic stage (RI-RII, 58%) and the fact that the majority of patients were alive and disease-free suggested a more favorable prognosis for this type of renal cell carcinoma.
METHODS (00160119) Tumors were classified according to well-established histologic criteria to determine stage of disease; the system proposed by Robson was used.
METHODS (00162303) Of 250 renal cell carcinomas analyzed, 36 were classified as chromophobe renal cell carcinoma, representing 14% of the group studied.
PURPOSE (00156456) In this study, we analyzed 250 renal cell carcinomas to a) determine frequency of CCRC at our Hospital and b) analyze clinical and pathologic features of CCRCs.
PURPOSE (00167817) Chromophobe renal cell carcinoma (CCRC) comprises 5% of neoplasms of renal tubular epithelium. CCRC may have a slightly better prognosis than clear cell carcinoma, but outcome data are limited.
RESULTS (00155338) Robson staging was possible in all cases, and 10 patients were stage 1) 11 stage II; 10 stage III, and five stage IV.

Table 4: Output of the argumentative categorizer when applied to an argumentatively structured abstract after removal of explicit markers. For each row, the attributed class is followed by the score for the class, followed by the extracted text segment. The reader can compare this categorization with argumentative labels as provided in the original abstract (PMID 12404725).

tative classes are equally distributed in MEDLINE abstracts, each index contains approximately a quarter of the top-ranked abstracts collection.

5 Results and Discussion

All results are computed using the treceval program, using the top 1000 retrieved documents for each evaluation query. We mainly evaluate the impact of varying the feedback category on the retrieval effectiveness, so we separately expand our queries based a single category. Query expansion based on RESULTS or METHODS sentences does not result in any improvement. On the contrary, expansion based on PURPOSE sentences improve the Rocchio baseline by + 23%, which is again significant ($p < 0.05$). But the main improvement is observed when CONCLUSION sentences are used to generate the expansion, with a remarkable gain of 41% when compared to Rocchio. We also observe in Table 5 that other measures (top precision) and number of relevant retrieved articles do confirm this trend.

For the PURPOSE category, the optimal k parameter, computed on the test queries was 11. For the CONCLUSION category, the optimal k parameter, computed on the test queries was 10. The difference between the m values between Rocchio feedback and the argumentative feedback, respectively 15 vs. 11 and 10 for Rocchio, PURPOSE, CONCLUSION sentences can

No feeback		
Relevant retrieved	Top precision	Mean average precision
1020	0.3871	0.3066
Rocchio feedback		
Relevant retrieved	Top precision	Mean average precision
1112	0.4020	0.353
Argumentative feedback: PURPOSE		
Relevant retrieved	Top precision	Mean average precision
1136	0.485	0.4353
Argumentative feedback: CONCLUSION		
Relevant retrieved	Top precision	Mean average precision
1143	0.550	0.4999

Table 5: Results without feedback, with Rocchio and with argumentative feedback applied on PURPOSE and CONCLUSION sentences. The number of relevant document for all queries is 1178.

be explained by the fact that less textual material is available when a particular class of sentences is selected; therefore the number of words that should be added to the original query is more targeted.

From a more general perspective, the importance of CONCLUSION and PURPOSE sentences is consistent with other studies, which aimed at selecting highly content bearing sentences for information extraction (Ruch et al., 2005b). This result is also consistent with the state-of-the-art in automatic summarization, which tends to prefer sentences appearing at the beginning or at the end of documents to generate summaries.

6 Conclusion

We have reported on the evaluation of a new linguistically-motivated feedback strategy, which selects highly-content bearing features for expansion based on argumentative criteria. Our simple model is based on four classes, which have been reported very stable in scientific reports of all kinds. Our results suggest that argumentation-driven expansion can improve retrieval effectiveness of search engines by more than 40%. The proposed methods open new research directions and are generally promising for natural language processing applied to information retrieval, whose positive impact is still to be confirmed (Strzalkowski et al., 1998). Finally, the proposed methods are important from a theoretical perspective, if we consider

that it initiates a *genre-specific* paradigm as opposed to the usual information retrieval typology, which distinguishes between domain-specific and domain-independent approaches.

Acknowledgements

The first author was supported by a visiting faculty grant (ORAU) at the Lister Hill Center of the National Library of Medicine in 2005. We would like to thank Dina Demner-Fushman, Susanne M. Humphrey, Jimmy Lin, Hongfang Liu, Miguel E. Ruiz, Lawrence H. Smith, Lorraine K. Tanabe, W. John Wilbur for the fruitful discussions we had during our weekly TREC meetings at the NLM. The study has also been partially supported by the Swiss National Foundation (Grant 3200-065228).

References

I Aalbersberg. 1992. Incremental Relevance Feedback. In *SIGIR*, pages 11–22.

S Abdou, P Ruch, and J Savoy. 2006. General vs. Specific Blind Query Expansion for Biomedical Searches. In *TREC 2005*.

A Aronson, D Demner-Fushman, S Humphrey, J Lin, H Liu, P Ruch, M Ruiz, L Smith, L Tanabe, and J Wilbur. 2006. Fusion of Knowledge-intensive and Statistical Approaches for Retrieving and Annotating Textual Genomics Documents. In *TREC 2005*.

J Xu B Croft. 1998. Corpus-based stemming using cooccurrence of word variants. *ACM-Transactions on Information Systems*, 16(1):61–81.

E Efthimiadis. 1996. Query expansion. *Annual Review of Information Science and Technology*, 31.

G Furnas, T Landauer, L Gomez, and S Dumais. 1987. The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11).

D Harman. 1991. How effective is suffixing ? *JASIS*, 42 (1):7–15.

W Hersh, C Buckley, T Leone, and D Hickam. 1994. OHSUMED: An interactive retrieval evaluation and new large test collection for research. In *SIGIR*, pages 192–201.

W Hersh. 2005. Report on the trec 2004 genomics track. pages 21–24.

Lynette Hirschman, Alexander Yeh, Christian Blaschke, and Alfonso Valencia. 2005. Overview of BioCreAtIvE: critical assessment of information extraction for biology. *BMC Bioinformatics*, 6 (suppl. 1).

G Hirst. 1981. *Anaphora in Natural Language Understanding: A Survey*. Lecture Notes in Computer Science 119 - Springer.

D Hull. 1996. Stemming algorithms: A case study for detailed evaluation. *Journal of the American Society of Information Science*, 47(1):70–84.

K Sparck Jones. 1971. *Automatic Keyword Classification for Information Retrieval*. Butterworths.

F Lisacek, C Chichester, A Kaplan, and Sandor. 2005. Discovering Paradigm Shift Patterns in Biomedical Abstracts: Application to Neurodegenerative Diseases. In *Proceedings of the First International Symposium on Semantic Mining in Biomedicine (SMBM)*, pages 212–217. Morgan Kaufmann.

Z Liu and W Chu. 2005. Knowledge-based query expansion to support scenario-specific retrieval of medical free text. *ACM-SAC Information Access and Retrieval Track*, pages 1076–1083.

C Manning and H Schütze. 2000. *Foundations of Statistical Natural Language Processing*. MIT Press.

D Marcu. 1997. The Rhetorical Parsing of Natural Language Texts. pages 96–103.

L McKnight and P Srinivasan. 2003. Categorization of sentence types in medical abstracts. *AMIA Annu Symp Proc.*, pages 440–444.

Y Mizuta and N Collier. 2004. Zone identification in biology articles as a basis for information extraction. *Proceedings of the joint NLPBA/BioNLP Workshop on Natural Language for Biomedical Applications*, pages 119–125.

Y Mizuta, A Korhonen, T Mullen, and N Collier. 2005. Zone Analysis in Biology Articles as a Basis for Information Extraction. *International Journal of Medical Informatics*, to appear.

C Orasan. 2001. Patterns in Scientific Abstracts. In *Proceedings of Corpus Linguistics*, pages 433–445.

Y Qiu and H Frei. 1993. Concept based query expansion. *ACM-SIGIR*, pages 160–69.

S Robertson. 1990. On term selection for query expansion. *Journal of Documentation*, 46(4):359–364.

J Rocchio. 1971. *Relevance feedback in information retrieval in The SMART Retrieval System - Experiments in Automatic Document Processing*. Prentice-Hall.

P Ruch, R Baud, C Chichester, A Geissbühler, F Lisacek, J Marty, D Rebholz-Schuhmann, I Tbahriti, and AL Veuthey. 2005a. Extracting Key Sentences with Latent Argumentative Structuring. In *Medical Informatica Europe (MIE)*, pages 835–40.

P Ruch, L Perret, and J Savoy. 2005b. Features Combination for Extracting Gene Functions from MEDLINE. In *European Colloquium on Information Retrieval (ECIR)*, pages 112–126.

P Ruch. 2002. Using contextual spelling correction to improve retrieval effectiveness in degraded text collections. *COLING 2002*.

P Ruch. 2004. Query translation by text categorization. *COLING 2004*.

P Ruch. 2006. Automatic Assignment of Biomedical Categories: Toward a Generic Approach. *Bioinformatics*, 6.

G Salton and C Buckley. 1990. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4).

J Savoy. 2005. Bibliographic database access using free-text and controlled vocabulary: An evaluation. *Information Processing and Management*, 41(4):873–890.

A Singhal, C Buckley, and M Mitra. 1996a. Pivoted document length normalization. *ACM-SIGIR*, pages 21–29.

C Buckley A Singhal, M Mitra, and G Salton. 1996b. New retrieval approaches using smart. *In Proceedings of TREC-4*.

T Strzalkowski, G Stein, G Bowden Wise, J Perez Carballo, P Tapanainen, T Jarvinen, A Voutilainen, and J Karlgren. 1998. Natural language information retrieval: TREC-7 report. In *Text REtrieval Conference*, pages 164–173.

J Swales. 1990. *Genre Analysis: English in Academic and Research Settings*. Cambridge University Press.

I Tbahriti, C Chichester, F Lisacek, and P Ruch. 2004. Using Argumention to Retrieve Articles with Similar Citations from MEDLINE. *Proceedings of the joint NLPBA/BioNLP Workshop on Natural Language for Biomedical Applications*.

I Tbahriti, C Chichester, F Lisacek, and P Ruch. 2005. Using Argumentation to Retrieve Articles with Similar Citations: an Inquiry into Improving Related Articles Search in the MEDLINE Digital Library. *International Journal of Medical Informatics*, to appear.

S Teufel and M Moens. 1999. Argumentative Classification of Extracted Sentences as a First Step Towards Flexible Abstracting. *Advances in Automatic Text Summarization, MIT Press*, pages 155–171.

B Velez, R Weiss, M Sheldon, and D Gifford. 1997. Fast and effective query refinement. In *ACM SIGIR*, pages 6–15.

E Voorhees. 1994. Query expansion using lexical-semantic relations. In *ACM SIGIR*, pages 61–69.

J Zobel. 1998. How reliable are large-scale information retrieval experiments? *ACM-SIGIR*, pages 307–314.

Simultaneous English-Japanese Spoken Language Translation Based on Incremental Dependency Parsing and Transfer

Koichiro Ryu
Graduate School of Information Science, Nagoya University
Furo-cho, Chikusa-ku, Nagoya, 464-8601, Japan
ryu@el.itc.nagoya-u.ac.jp

Shigeki Matsubara
Information Technology Center, Nagoya University
Furo-cho, Chikusa-ku, Nagoya, 464-8601, Japan

Yasuyoshi Inagaki
Faculty of Information Science and Technology, Aichi Prefectural University
Nagakute-cho, Aichi-gun, Aichi-ken, 480-1198, Japan

Abstract

This paper proposes a method for incrementally translating English spoken language into Japanese. To realize simultaneous translation between languages with different word order, such as English and Japanese, our method utilizes the feature that the word order of a target language is flexible. To resolve the problem of generating a grammatically incorrect sentence, our method uses dependency structures and Japanese dependency constraints to determine the word order of a translation. Moreover, by considering the fact that the inversion of predicate expressions occurs more frequently in Japanese spoken language, our method takes advantage of a predicate inversion to resolve the problem that Japanese has the predicate at the end of a sentence. Furthermore, our method includes the function of canceling an inversion by restating a predicate when the translation is incomprehensible due to the inversion. We implement a prototype translation system and conduct an experiment with all 578 sentences in the ATIS corpus. The results indicate improvements in comparison to two other methods.

1 Introduction

Recently, speech-to-speech translation has become one of the important research topics in machine translation. Projects concerning speech translation such as TC-STAR (Hoge, 2002) and DARPA Babylon have been executed, and conferences on spoken language translation such as IWSLT have been held. Though some speech translation systems have been developed so far (Frederking et al., 2002; Isotani et al., 2003; Liu et al., 2003; Takezawa et al., 1998), these systems, because of their sentence-by-sentence translation, cannot start to translate a sentence until it has been fully uttered. The following problems may arise in cross-language communication:

- The conversation time become long since it takes much time to translate
- The listener has to wait for the translation since such systems increase the difference between the beginning time of the speaker's utterance and the beginning time of its translation

These problems are likely to cause some awkwardness in conversations. One effective method of improving these problems is that a translation system begins to translate the words without waiting for the end of the speaker's utterance, much as a simultaneous interpreter does. This has been verified as possible by a study on comparing simultaneous interpretation with consecutive interpretation from the viewpoint of efficiency and smoothness of cross-language conversations (Ohara et al., 2003).

There has also been some research on simultaneous machine interpretation with the aim of developing environments that support multilingual communication (Mima et al., 1998; Casacuberta et al., 2002; Matsubara and Inagaki, 1997).

To realize simultaneous translation between languages with different word order, such as English and Japanese, our method utilizes the feature that the word order of a target language is flexible. To resolve the problem that translation systems generates grammatically dubious sentence,

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 683–690,
Sydney, July 2006.

our method utilizes dependency structures and Japanese dependency constraints to determine the word order of a translation. Moreover, by considering the fact that the inversion of predicate expressions occurs more frequently in Japanese spoken language, our method employs predicate inversion to resolve the problem that Japanese has the predicate at the end of the sentence. Furthermore, our method features the function of canceling an inversion by restating a predicate when the translation is incomprehensible due to the inversion. In the research described in this paper, we implement a prototype translation system, and to evaluate it, we conduct an experiment with all 578 sentences in the ATIS corpus.

This paper is organized as follows: Section 2 discusses an important problem in English-Japanese simultaneous translation and explains the idea of utilizing flexible word order. Section 3 introduces our method for the generation in English-Japanese simultaneous translation, and Section 4 describes the configuration of our system. Section 5 reports the experimental results, and the paper concludes in Section 6.

2 Japanese in Simultaneous English-Japanese Translation

In this section, we describe the problem of the difference of word order between English and Japanese in incremental English-Japanese translation. In addition, we outline an approach of simultaneous machine translation utilizing linguistic phenomena, flexible word order, and inversion, characterizing Japanese speech.

2.1 Difference of Word Order between English and Japanese

Let us consider the following English:

(E1) I want to fly from San Francisco to Denver next Monday.

The standard Japanese for (E1) is

(J1) raishu-no (‘*next*’) getsuyobi-ni (‘*Monday*’) San Francisco-kara (‘*from*’) Denver-he (‘*to*’) tobi-tai-to omoi-masu (‘*want to fly*’).

Figure 1 shows the output timing when the translation is generated as incrementally as possible in consideration of the word alignments between (E1) and (J1). In Fig. 1, the flow of time is shown from top to bottom. In this study, we assume that the system translates input words chunk-by-chunk. We define a simple noun phrase (e.g. San Francisco, Denver and next Monday), a predicate (e.g. want to fly) and each other word (e.g. I, from, to) as a chunk. There is “raishu-no getsuyobi-ni” (‘*next Monday*’) at the beginning of the translation (J1), and there is “next Monday” corresponding to “raishu-no getsuyobi-ni” at the end of the sentence (E1). Thus, the system cannot output “raishu-no getsuyobi-ni” and its following translation until the whole sentence is uttered. This is a fatal flaw in incremental English-Japanese translation because there exists an essential difference between English and Japanese in the word order. It is fundamentally impossible to cancel these problems as long as we assume (J1) to be the translation of (E1).

Input	Output
I	
want to fly	
from	
San Francisco	
to	
Denver	
next Monday	raishu-no (‘*next*’) getsuyobi-ni (‘*Monday*’)
	San Francisco-kara (‘*from*’)
	Denver-he (‘*to*’)
	tobi-tai-to omoi-masu (‘*want to fly*’)

Figure 1: The output timing of the translation (J1)

Input	Output
I	
want to fly	
from	
San Francisco	San Francisco-kara (‘*from*’)
to	
Denver	Denver-he (‘*to*’) tobi-tai-to omoi-masu (‘*want to fly*’)
next Monday	raishu-no (‘*next*’) getsuyobi-ni (‘*Monday*’)

Figure 2: The output timing of the translation (J2)

2.2 Utilizing Flexible Word Order in Japanese

Japanese is a language with a relatively flexible word order. Thus, it is possible that a Japanese translation can be accepted even if it keeps the word order of an English sentence. Let us consider the following Japanese:

(J2) San Francisco-kara (‘*from*’) Denver-he (‘*to*’) tobi-tai-to omoi-masu (‘*want to fly*’) raishu-no (‘*next*’) getsuyobi-ni (‘*Monday*’).

(J2) can be accepted as the translation of the sentence (E1) and still keep the word order as close as possible to the sentence (E1). Figure 2 shows the output timing when the translation is generated as incrementally as possible in consideration of the word alignments between (E1) and (J2). The figure demonstrates that a translation system might

be able to output “San Francisco -kara (‘*from*’)” when “San Francisco” is input and “Denver-he (‘*to*’) tobi-tai-to omoi-masu (‘*want to fly*’)” when “Denver” is input. If a translation system outputs the sentence (J2) as the translation of the sentence (E1), the system can translate it incrementally. The translation (J2) is not necessarily an ideal translation because its word order differs from that of the standard translation and it has an inverted sentence structure. However the translation (J2) can be easily understood due to the high flexibility of word order in Japanese. Moreover, in spoken language machine translation, the high degree of incrementality is preferred to that of quality. Therefore, our study positively utilizes flexible word order and inversion to realize incremental English-Japanese translation while keeping the translation quality acceptable.

3 Japanese Generation based on Dependency Structure

When an English-Japanese translation system incrementally translates an input sentence by utilizing flexible word order and inversion, it is possible that the system will generate a grammatically incorrect Japanese sentence. Therefore, it is necessary for the system to generate the translation while maintaining the translation quality at an acceptable level as a correct Japanese sentence. In this section, we describe how to generate an English-Japanese translation that retains the word order of the input sentence as much as possible while keeping the quality acceptable.

3.1 Dependency Grammar in English and Japanese

Dependency grammar illustrates the syntactic structure of a sentence by linking individual words. In each link, modifiers (dependents) are connected to the word that they modify (head). In Japanese, the dependency structure is usually defined in terms of the relation between phrasal units called *bunsetsu*[1]. The Japanese dependency relations are satisfied with the following constraints (Kurohashi and Nagao, 1997):

- No dependency is directed from right to left.
- Dependencies do not cross each other.

[1]A *bunsetsu* is one of the linguistic units in Japanese, and roughly corresponds to a basic phrase in English. A bunsetsu consists of one independent word and more than zero ancillary words. A dependency is a modification relation between two bunsetsus.

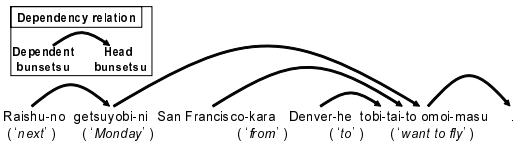


Figure 3: The dependency structures of translation (J1)

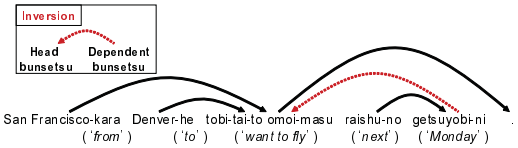


Figure 4: The dependency structures of translation (J2)

- Each bunsetsu, except the last one, depends on only one bunsetsu.

The translation (J1) is satisfied with these constraints as shown in Fig. 3. A sentence satisfying these constraints is deemed grammatically correct sentence in Japanese. To meet this requirement, our method parses the dependency relations between input chunks and generates a translation satisfying Japanese dependency constraints.

3.2 Inversion

In this paper, we call the dependency relations heading from right to left ”inversions”. Inversions occur more frequently in spontaneous speech than in written text in Japanese. That is to say, there are some sentences in Japanese spoken language that do not satisfy the constraint mentioned above. Translation (J2) does not satisfy this constraint, as shown in Fig. 4. We investigated the inversions using the CIAIR corpus (Ohno et al., 2003) and found the following features:

Feature 1 92.2% of the inversions are that the head bunsetsu of the dependency relation is a predicate. (predicate inversion)

Feature 2 The more the number of dependency relations that depend on a predicate increases, the more the frequency of predicate inversions increases.

Feature 3 There are not three or more inversions in a sentence.

From Feature 1, our method utilizes a predicate inversion to retain the word order of an input sentence. It also generates a predicate when the number of dependency relations that depend on a predicate exceeds the constant R (from Feature 2). If there are three or more inversions in the translation, the system cancels an inversion by restating a predicate (from Feature 3).

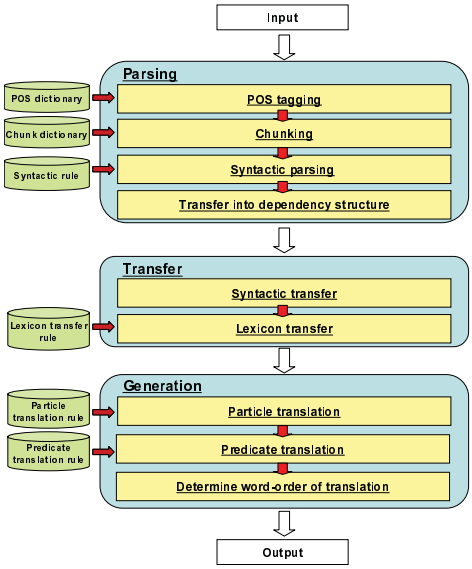


Figure 5: Configuration of our system

4 System Configuration

Figure 5 shows the configuration of our system. The system translates an English speech transcript into Japanese incrementally. It is composed of three modules: incremental parsing, transfer and generation. In the parsing module the parser determines the English dependency structure for input words incrementally. In the transfer module, structure and lexicon transfer rules transform the English dependency structure into the Japanese case structure. As for the generation module, the system judges whether the translation of each chunk can be output, and if so, outputs the translation of the chunk. Figure 6 shows the processing flow when the fragment "I want to fly from San Francisco to Denver" of （2.1）is input. In the following subsections we explain each module, referring to Fig. 6.

4.1 Incremental Dependency Parsing

First, the system performs POS tagging for input words and chunking (c.f. "Chunk" in Fig. 6).

Next, we explain how to parse the English phrase structure (c.f. "English phrase structure" in Fig. 6). When we parse the phrase structure for input words incrementally, there arises the problem of ambiguity; our method needs to determine only one parsing result at a time. To resolve this problem our system selects the phrase structure of the maximum likelihood at that time by using PCFG (Probabilistic Context-Free Grammar) rules. To resolve the problem of the processing time our system sets a cut-off value.

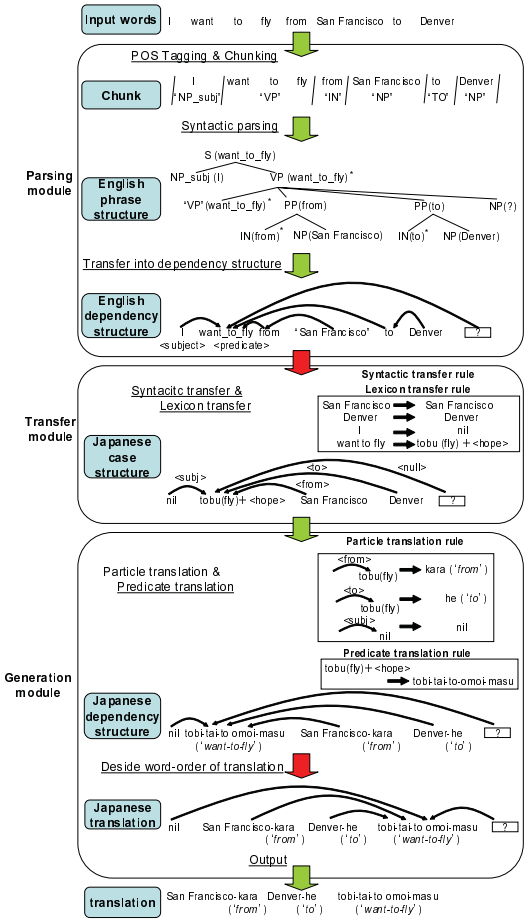


Figure 6: The translation flow for the fragment "I want to fly from San Francisco to Denver"

Furthermore, the system transforms the English phrase structure into an English dependency structure (c.f. "English dependency structure" in Fig. 6). The dependency structure for the sentence can be computed from the phrase structure for the input words by defining the category for each rule in CFG, called a "head child" (Collins, 1999). The head is indicated using an asterisk * in the phrase structure of Fig. 6. In the "English phrase structure," the chunk in parentheses at each node is the head chunk of the node that is determined by the head information of the syntax rules. If the head chunk (e.g. "from") of a child node (e.g. PP(from)) differs from that of its parent node (e.g. VP(want-to-fly)), the head chunk (e.g. "from") of the child node depends on the head chunk (e.g. "want-to-fly") of the parent node. Some syntax rules are also annotated with subject and object information. Our system uses such information to add Japanese function words to the translation of the subject chunk or the object chunk in the generation module. To use a predicate inversion in the

generation module the system has to recognize the predicate of an input sentence. This system recognizes the chunk (e.g. “want to fly”) on which the subject chunk (e.g. “I”) depends as a predicate.

4.2 Incremental Transfer

In the transfer module, structure and lexicon transfer rules transform the English dependency structure into the Japanese case structure (“Japanese case structure” in Fig. 6). In the structure transfer, the system adds a type of relation to each dependency relation according to the following rules.

- If the dependent chunk of a dependency relation is a subject or object (e.g. “I”), then the type of such dependency relation is “subj” or “obj”.
- If a chunk A (e.g. “San Francisco”) indirectly depends on another chunk B (e.g. “want-to-fly”) through a preposition (e.g. “from”), then the system creates a new dependency relation where A depends on B directly, and the type of the relation is the preposition.
- The type of the other relations is ”null”.

In the lexicon transfer, the system transforms each English chunk into its Japanese translation.

4.3 Incremental Generation

In the generation module, the system transforms the Japanese case structure into the Japanese dependency structure by translating a particle and a predicate. In attaching a particle (e.g. “kara” (from)) to the translation of a chunk (e.g. “San Francisco”), the system determines the attached particle (e.g. “kara” (from)) by particle translation rules. In translating a predicate (e.g. “want to fly”), the system translates a predicate by predicate translation rules, and outputs the translation of each chunk using the method described in Section 3.

4.4 Example of Translation Process

Figure 7 shows the processing flow for the English sentence, “I want to fly from San Francisco to Denver next Monday.” In Fig. 7 the underlined words indicate that they can be output at that time.

5 Experiment

5.1 Outline of Experiment

To evaluate our method, we conducted a translation experiment was made as follows. We implemented the system in Java language on a 1.0-GHz PentiumM PC with 512 MB of RAM. The OS was Windows XP. The experiment used all 578 sentences in the ATIS corpus with a parse tree, in the Penn Treebank (Marcus et al. 1993). In addition, we used 533 syntax rules, which were extracted from the corpus’ parse tree. The position of the head child in the grammatical rule was defined according to Collins’ method (Collins, 1999).

5.2 Evaluation Metric

Since an incremental translation system for spoken dialogues is required to realize a quick and informative response to support smooth communication, we evaluated the translation results of our system in terms of both simultaneity and quality.

To evaluate the translation quality of our system, each translation result of our system was assigned one of four ranks for translation quality by a human translator:

A (Perfect): no problems in either information or grammar

B (Fair): easy to understand but some important information is missing or it is grammatically flawed

C (Acceptable): broken but understandable with effort

D (Nonsense): important information has been translated incorrectly

To evaluate the simultaneity of our system, we calculated the average delay time for translating chunks using the following expression:

$$Average\ \ delay\ \ time = \frac{\sum_k d_k}{n}, \qquad (1)$$

where d_k is the virtual elapsed time from inputting the kth chunk until outputting its translated chunk. (When a repetition is used, d_k is the elapsed time from inputting the kth chunk until restate its translated chunk.) The virtual elapsed time increases by one unit of time whenever a chunk is input, n is the total number of chunks in all of the test sentences.

The average delay time is effective for evaluating the simultaneity of translation. However, it is difficult to evaluate whether our system actually improves the efficiency of a conversation. To do so, we measured “the speaker’ and the interpreter’s utterance time.” “The speaker’ and the interpreter ’utterance time” runs from the start time of a speaker’s utterance to the end time of its translation. We cannot actually measure actual “the

Table 1: Comparing our method (Y) with two other methods (X, Z)

Method	Quality			Average delay time	Speaker and interpreter utterance time (sec)
	A	A+B	A+B+C		
X	7 (1.2%)	48 (8.3%)	92 (15.9%)	0	4.7
Y	40 (6.9%)	358 (61.9%)	413 (71.5%)	2.79	6.0
Z				3.79	6.4

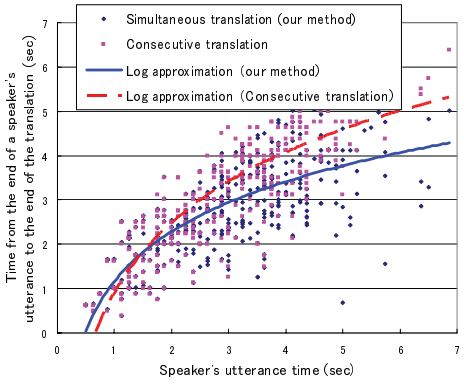


Figure 8: The relation between the speaker's utterance time and the time from the end time of the speaker's utterance to the end time of the translation

speaker' and the interpreter' utterance time" because our system does not include speech recognition and synthesis. Thus, the processing time of speech recognition and transfer text-to-speech synthesis is zero, and the speaker's utterance time and the interpreter's utterance time is calculated virtually by assuming that the speaker's and interpreter's utterance speed is 125 ms per mora.

5.3 Experiment Results

To evaluate the translation quality and simultaneity of our system, we compared the translation results of our method (Y) with two other methods. One method (X) translates the input chunks with no delay time. The other method (Z) translates the input chunks by waiting for the whole sentence to be input, in as consecutive translation. We could not evaluate the translation quality of the method Z because we have not implemented the method Z. And we virtually compute the delay time and the utterance time. Table 1 shows the estimation results of methods X, Y and Z. Note, however, that we virtually calculated the average delay time and the speaker's and interpreter's utterance times in method Z without translating the input sentence.

Table 1 indicates that our method Y achieved a 55.6% improvement over method X in terms of translation quality and a 1.0 improvement over method Z for the average delay time.

Figure 8 shows the relation between the speaker's utterance time and the time from the end time of the speaker's utterance to the end time of the translation. According to Fig. 8, the longer a speaker speaks, the more the system reduces the time from the end time of the speaker's utterance to the end time of the translation.

In Section 3, we explained the constant R. Table 2 shows increases in R from 0 to 4, with the results of the estimation of quality, the average delay time, the number of inverted sentences and the number of sentences with restatement. When we set the constant to $R = 2$, the average delay time improved by a 0.08 over that of method Y, and the translation quality did not decrease remarkably. Note, however, that method Y did not utilize any predicate inversions.

To ascertain the problem with our method, we investigated 165 sentences whose translations were assigned the level D when the system translated them by utilizing dependency constraints. According to the investigation, the system generated grammatically incorrect sentences in the following cases:

- There is an interrogative word (e.g. "what", "which") in the English sentence (64 sentences).
- There are two or more predicates in the English sentence (25 sentences).
- There is a coordinate conjunction (e.g. "and", "or") in the English sentence (21 sentences).

Other cases of decreases in the translation quality occurred when a English sentence was ill-formed or when the system fails to parse.

6 Conclusion

In this paper, we have proposed a method for incrementally translating English spoken language into Japanese. To realize simultaneous translation

Table 2: The results of each R ($0 \leq R \leq 4$)

R	Quality			Average delay time	Sentences with inversion	Sentences with restatement
	A	A+B	A+B+C			
0	8 (1.4%)	152 (26.3%)	363 (62.8%)	2.51	324	27
1	14 (2.4%)	174 (30.1%)	364 (63.0%)	2.53	289	29
2	36 (6.2%)	306 (52.9%)	396 (68.5%)	2.71	73	5
3	39 (6.7%)	344 (59.5%)	412 (71.3%)	2.79	28	2
4	40 (7.0%)	358 (61.9%)	412 (71.3%)	2.79	3	2

our method utilizes the feature that word order is flexible in Japanese, and determines the word order of a translation based on dependency structures and Japanese dependency constraints. Moreover, our method employs predicate inversion and repetition to resolve the problem that Japanese has a predicate at the end of a sentence. We implemented a prototype system and conducted an experiment with 578 sentences in the ATIS corpus. We evaluated the translation results of our system in terms of quality and simultaneity, confirming that our method achieved a 55.6% improvement over the method of translating by retaining the word order of an original with respect to translation quality, and a 1.0 improvement over the method of consecutive translation regarding average delay time.

Acknoledgments

The authors would like to thank Prof. Dr. Toshiki Sakabe. They also thank Yoshiyuki Watanabe, Atsushi Mizuno and translator Sachiko Waki for their contribution to our study.

References

F. Casacuberta, E. Vidal and J. M. Vilar. 2002. Architectures for speech-to-speech translation using finite-state models, *Proceedings of Workshop on Speech-to-Speech Translation: Algorithms and System*, pages 39-44.

M. Collins. 1999. Head-Driven Statistical Models for Natural Language Parsing, *Ph.D. Thesis, University of Pennsylvania*,

R. Frederking, A. Blackk, R. Brow, J. Moody, and E. Stein-brecher, 2002. Field Testing the Tongues Speech-to-Speech Machin Translation System, *Proceedings of the 3rd International Conference on Language Resources and Evaluation(LREC-2002)* pages 160-164.

H. Hoge. 2002. Project Proposal TC-STAR: Make Speech to Speech Translation Real, *Proceedings of the 3rd International Conference on Language Resources and Evaluation(LREC-2002)*, pages 136-141.

R. Isotani, K. Yamada, S. Ando, K. Hanazawa, S. Ishikawa and K. Iso. 2003. Speech-to-Speech Translation Software PDAs for Travel Conversation, *NEC Research and Development, 44, No.2* pages 197-202.

S. Kurohashi and M. Nagao. 1997. Building a Japanese Parsed Corpus while Improving the Parsing System, *Proceedings of 4th Natural Language Processing Pacific Rim Symposium*, pages 451-456.

F. Liu, Y. Gao, L. Gu and M. Picheny. 2003. Noise Robustness in Speech to Speech Translation, *IBM Tech Report RC22874*.

M. P. Marcus, B. Santorini and M. A. Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank, *Computational Linguistics*, 19(2):310-330.

S. Matsubara and Y. Inagaki. 1997. Incremental Transfer in English-Japanese Machine Translation, *IEICE Transactions on Information and Systems*, (11):1122-1129.

H. Mima, H. Iida and O. Furuse. 1998. Simultaneous Interpretation Utilizing Example-based Incremental Transfer, *Proceedings of 17th International Conference on Computational Linguistics and 36th Annual Meeting of Association for Computational Linguistics*, pages 855-861.

M. Ohara, S. Matsubara, K. Ryu, N. Kawaguchi and Y. Inagaki. 2003. Temporal Features of Cross-Lingual Communication Mediated by Simultaneous Interpreting: An Analysis of Parallel Translation Corpus in Comparison to Consecutive Interpreting, *The Journal of the Japan Association for Interpretation Studies* pages 35-53.

T. Ohno, S. Matsubrara, N. Kawaguchi and Y. Inagaki. 2003. Spiral Construction of Syntactically Annotated Spoken Language Corpus, *Proceedings of 2003 IEEE International Conference on Natural Language Processing and Knowledge Engineering*, pages 477-483.

T. Takezawa, T. Morimoto, Y. Sagisaka, N. Campbell, H. Iida, F. Sugaya, A. Yokoo and S. Yamamoto. 1998. A Japanese-to-English Speech Translation System:ATR-MATRIX, *Proceedings of 5th International Conference on Spoken Language Processing*, pages 957-960.

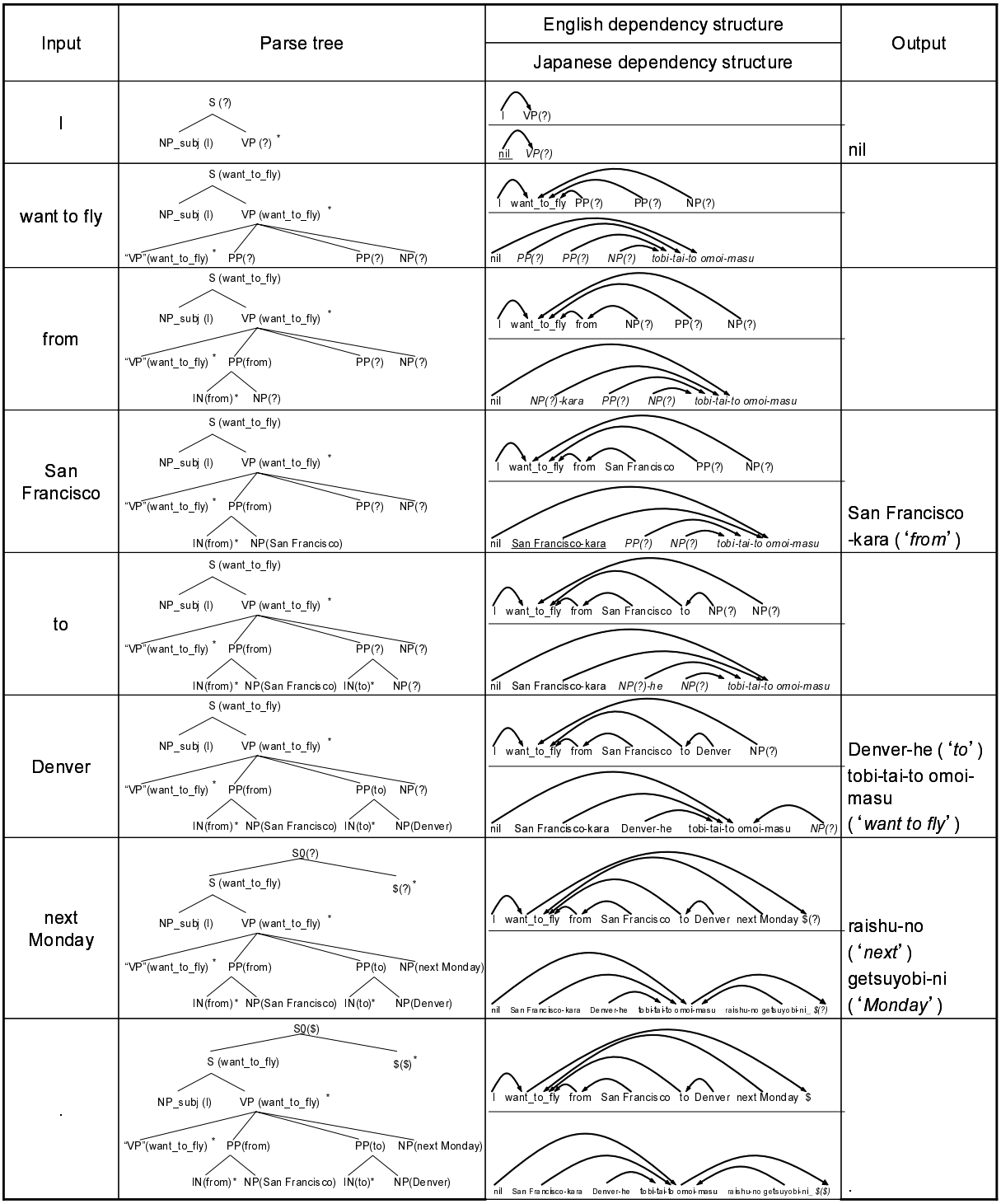


Figure 7: The translation flow for "I want to fly from San Francisco to Denver next Monday."

A Best-First Probabilistic Shift-Reduce Parser

Kenji Sagae and **Alon Lavie**
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213
{sagae,alavie}@cs.cmu.edu

Abstract

Recently proposed deterministic classifier-based parsers (Nivre and Scholz, 2004; Sagae and Lavie, 2005; Yamada and Matsumoto, 2003) offer attractive alternatives to generative statistical parsers. Deterministic parsers are fast, efficient, and simple to implement, but generally less accurate than optimal (or nearly optimal) statistical parsers. We present a statistical shift-reduce parser that bridges the gap between deterministic and probabilistic parsers. The parsing model is essentially the same as one previously used for deterministic parsing, but the parser performs a best-first search instead of a greedy search. Using the standard sections of the WSJ corpus of the Penn Treebank for training and testing, our parser has 88.1% precision and 87.8% recall (using automatically assigned part-of-speech tags). Perhaps more interestingly, the parsing model is significantly different from the generative models used by other well-known accurate parsers, allowing for a simple combination that produces precision and recall of 90.9% and 90.7%, respectively.

1 Introduction

Over the past decade, researchers have developed several constituent parsers trained on annotated data that achieve high levels of accuracy. Some of the more popular and more accurate of these approaches to data-driven parsing (Charniak, 2000; Collins, 1997; Klein and Manning, 2002) have been based on generative models that are closely related to probabilistic context-free grammars. Recently, classifier-based dependency parsing (Nivre and Scholz, 2004; Yamada and Matsumoto, 2003) has showed that deterministic parsers are capable of high levels of accuracy, despite great simplicity. This work has led to the development of deterministic parsers for constituent structures as well (Sagae and Lavie, 2005; Tsuruoka and Tsujii, 2005). However, evaluations on the widely used WSJ corpus of the Penn Treebank (Marcus et al., 1993) show that the accuracy of these parsers still lags behind the state-of-the-art.

A reasonable and commonly held assumption is that the accuracy of deterministic classifier-based parsers can be improved if determinism is abandoned in favor of a search over a larger space of possible parses. While this assumption was shown to be true for the parser of Tsuruoka and Tsujii (2005), only a moderate improvement resulted from the addition of a non-greedy search strategy, and overall parser accuracy was still well below that of state-of-the-art statistical parsers.

We present a statistical parser that is based on a shift-reduce algorithm, like the parsers of Sagae and Lavie (2005) and Nivre and Scholz (2004), but performs a best-first search instead of pursuing a single analysis path in deterministic fashion. The parser retains much of the simplicity of deterministic classifier-based parsers, but achieves results that are closer in accuracy to state-of-the-art statistical parsers. Furthermore, a simple combination of the shift-reduce parsing model with an existing generative parsing model produces results with accuracy that surpasses any that of any single (non-reranked) parser tested on the WSJ Penn Treebank, and comes close to the best results obtained with discriminative reranking (Charniak and John-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 691–698,
Sydney, July 2006.

son, 2005).

2 Parser Description

Our parser uses an extended version of the basic bottom-up shift-reduce algorithm for constituent structures used in Sagae and Lavie's (2005) deterministic parser. For clarity, we will first describe the deterministic version of the algorithm, and then show how it can be extended into a probabilistic algorithm that performs a best-first search.

2.1 A Shift-Reduce Algorithm for Deterministic Constituent Parsing

In its deterministic form, our parsing algorithm is the same single-pass shift-reduce algorithm as the one used in the classifer-based parser of Sagae and Lavie (2005). That algorithm, in turn, is similar to the dependency parsing algorithm of Nivre and Scholz (2004), but it builds a constituent tree and a dependency tree simultaneously. The algorithm considers only trees with unary and binary productions. Training the parser with arbitrary branching trees is accomplished by a simple procedure to transform those trees into trees with at most binary productions. This is done by converting each production with n children, where $n > 2$, into $n - 1$ binary productions. This binarization process is similar to the one described in (Charniak et al., 1998). Additional non-terminal nodes introduced in this conversion must be clearly marked. Transforming the parser's output into arbitrary branching trees is accomplished using the reverse process.

The deterministic parsing algorithm involves two main data structures: a stack S, and a queue W. Items in S may be terminal nodes (part-of-speech-tagged words), or (lexicalized) subtrees of the final parse tree for the input string. Items in W are terminals (words tagged with parts-of-speech) corresponding to the input string. When parsing begins, S is empty and W is initialized by inserting every word from the input string in order, so that the first word is in front of the queue.

The algorithm defines two types of parser actions, shift and reduce, explained below:

- Shift: A shift action consists only of removing (shifting) the first item (part-of-speech-tagged word) from W (at which point the next word becomes the new first item), and placing it on top of S.
- Reduce: Reduce actions are subdivided into unary and binary cases. In a unary reduction, the item on top of S is popped, and a new item is pushed onto S. The new item consists of a tree formed by a non-terminal node with the popped item as its single child. The lexical head of the new item is the same as the lexical head of the popped item. In a binary reduction, two items are popped from S in sequence, and a new item is pushed onto S. The new item consists of a tree formed by a non-terminal node with two children: the first item popped from S is the right child, and the second item is the left child. The lexical head of the new item may be the lexical head of its left child, or the lexical head of its right child.

If S is empty, only a shift action is allowed. If W is empty, only a reduce action is allowed. If both S and W are non-empty, either shift or reduce actions are possible, and the parser must decide whether to shift or reduce. If it decides to reduce, it must also choose between a unary-reduce or a binary-reduce, what non-terminal should be at the root of the newly created subtree to be pushed onto the stack S, and whether the lexical head of the newly created subtree will be taken from the right child or the left child of its root node. Following the work of Sagae and Lavie, we consider the complete set of decisions associated with a reduce action to be part of that reduce action. Parsing terminates when W is empty and S contains only one item, and the single item in S is the parse tree for the input string.

2.2 Shift-Reduce Best-First Parsing

A deterministic shift-reduce parser based on the algorithm described in section 2.1 does not handle ambiguity. By choosing a single parser action at each opportunity, the input string is parsed deterministically, and a single constituent structure is built during the parsing process from beginning to end (no other structures are even considered).

A simple extension to this idea is to eliminate determinism by allowing the parser to choose several actions at each opportunity, creating different paths that lead to different parse trees. This is essentially the difference between deterministic LR parsing (Knuth, 1965) and Generalized-LR parsing (Tomita, 1987; Tomita, 1990). Furthermore, if a probability is assigned to every parser action, the probability of a parse tree can be computed

simply as the product of the probabilities of each action in the path that resulted in that parse tree (the derivation of the tree). This produces a probabilistic shift-reduce parser that resembles a generalized probabilistic LR parser (Briscoe and Carroll, 1993), where probabilities are associated with an LR parsing table. In our case, although there is no LR table, the action probabilities are associated with several aspects of the current state of the parser, which to some extent parallel the information contained in an LR table. Instead of having an explicit LR table and pushing LR states onto the stack, the state of the parser is implicitly defined by the configurations of the stack and queue. In a way, there is a parallel between how modern PCFG-like parsers use markov grammars as a distribution that is used to determine the probability of any possible grammar rules, and the way a statistical model is used in our parser to assign a probability to any transition of parser states (instead of a symbolic LR table).

Pursuing every possible sequence of parser actions creates a very large space of actions for even moderately sized sentences. To find the most likely parse tree efficiently according to the probabilistic shift-reduce parsing scheme described so far, we use a best-first strategy. This involves an extension of the deterministic shift-reduce algorithm into a best-first shift-reduce algorithm. To describe this extension, we first introduce a new data structure T_i that represents a parser state, which includes a stack S_i and a queue W_i. In the deterministic algorithm, we would have a single parser state T that contains S and W. The best-first algorithm, on the other hand, has a heap H containing multiple parser states T_1 ... T_n. These states are ordered in the heap according to their probabilities, so that the state with the highest probability is at the top. State probabilities are determined by multiplying the probabilities of each of the actions that resulted in that state. Parser actions are determined from and applied to a parser state T_i popped from the top of H. The parser actions are the same as in the deterministic version of the algorithm. When the item popped from the top of the heap H contains a stack S_i with a single item and an empty queue (in other words, meets the acceptance criteria for the deterministic version of the algorithm), the item on top of S_i is the tree with the highest probability. At that point, parsing terminates if we are searching for the most probable parse. To obtain a list of n-best parses, we simply continue parsing once the first parse tree is found, until either n trees are found, or H is empty.

We note that this approach does not use dynamic programming, and relies only on the best-first search strategy to arrive at the most probable parse efficiently. Without any pruning of the search space, the distribution of probability mass among different possible actions for a parse state has a large impact on the behavior of the search. We do not use any normalization to account for the size (in number of actions) of different derivations when calculating their probabilities, so it may seem that shorter derivations usually have higher probabilities than longer ones, causing the best-first search to approximate a breadth-first search in practice. However, this is not the case if for a given parser state only a few actions (or, ideally, only one action) have high probability, and all other actions have very small probabilities. In this case, only likely derivations would reach the top of the heap, resulting in the desired search behavior. The accuracy of deterministic parsers suggest that this may in fact be the types of probabilities a classifier would produce given features that describe the parser state, and thus the context of the parser action, specifically enough. The experiments described in section 4 support this assumption.

2.3 Classifier-Based Best-First Parsing

To build a parser based on the deterministic algorithm described in section 2.1, a classifier is used to determine parser actions. Sagae and Lavie (2005) built two deterministic parsers this way, one using support vector machines, and one using k-nearest neighbors. In each case, the set of features and classes used with each classifier was the same. Items 1 – 13 in figure 1 shows the features used by Sagae and Lavie. The classes produced by the classifier encode every aspect of a parser action. Classes have one of the following forms:

SHIFT : represents a shift action;

REDUCE-UNARY-*XX* : represents a unary reduce action, where the root of the new subtree pushed onto S is of type *XX* (where *XX* is a non-terminal symbol, typically NP, VP, PP, for example);

REDUCE-LEFT-*XX* : represents a binary reduce action, where the root of the new sub-

tree pushed onto S is of non-terminal type *XX*. Additionally, the head of the new subtree is the same as the head of the left child of the root node;

REDUCE-RIGHT-*XX* : represents a binary reduce action, where the root of the new subtree pushed onto S is of non-terminal type *XX*. Additionally, the head of the new subtree is the same as the head of the right child of the root node.

To implement a parser based on the best-first algorithm, instead of just using a classifier to determine one parser action given a stack and a queue, we need a classification approach that provides us with probabilities for different parser actions associated with a given parser state. One such approach is maximum entropy classification (Berger et al., 1996), which we use in the form of a library implemented by Tsuruoka[1] and used in his classifier-based parser (Tsuruoka and Tsujii, 2005). We used the same classes and the same features as Sagae and Lavie, and an additional feature that represents the previous parser action applied the current parser state (figure 1).

3 Related Work

As mentioned in section 2, our parsing approach can be seen as an extension of the approach of Sagae and Lavie (2005). Sagae and Lavie evaluated their deterministic classifier-based parsing framework using two classifiers: support vector machines (SVM) and k-nearest neighbors (kNN). Although the kNN-based parser performed poorly, the SVM-based parser achieved about 86% precision and recall (or 87.5% using gold-standard POS tags) on the WSJ test section of the Penn Treebank, taking only 11 minutes to parse the test set. Sagae and Lavie's parsing algorithm is similar to the one used by Nivre and Scholz (2004) for deterministic dependency parsing (using kNN). Yamada and Matsumoto (2003) have also presented a deterministic classifier-based (SVM-based) dependency parser, but using a different parsing algorithm, and using only unlabeled dependencies.

Tsuruoka and Tsujii (2005) developed a classifier-based parser that uses the chunk-parsing algorithm and achieves extremely high parsing speed, but somewhat low recall. The algorithm is based on reframing the parsing task as several sequential chunking tasks.

Finally, our parser is in many ways similar to the parser of Ratnaparkhi (1997). Ratnaparkhi's parser uses maximum-entropy models to determine the actions of a parser based to some extent on the shift-reduce framework, and it is also capable of pursuing several paths and returning the top-n highest scoring parses for a sentence. However, in addition to using different features for parsing, Ratnaparkhi's parser uses a different, more complex algorithm. The use of a more involved algorithm allows Ratnaparkhi's parser to work with arbitrary branching trees without the need of the binarization transform employed here. It breaks the usual reduce actions into smaller pieces (CHECK and BUILD), and uses two separate passes (not including the part-of-speech tagging pass) for determining chunks and higher syntactic structures separately. Instead of keeping a stack, the parser makes multiple passes over the input string, like the dependency parsing algorithm used by Yamada and Matsumoto. Our parser, on the other hand, uses a simpler stack-based shift-reduce (LR-like) algorithm for trees with only unary and binary productions.

4 Experiments

We evaluated our classifier-based best-first parser on the Wall Street Journal corpus of the Penn Treebank (Marcus et al., 1993) using the standard split: sections 2-21 were used for training, section 22 was used for development and tuning of parameters and features, and section 23 was used for testing. Every experiment reported here was performed on a Pentium4 3.2GHz with 2GB of RAM.

Each tree in the training set had empty-node and function tag information removed, and the trees were lexicalized using the same head-table rules as in the Collins (1999) parser (these rules were taken from Bikel's (2002) implementation of the Collins parser). The trees were then converted into trees containing only unary and binary productions, using the binarization transform described in section 2. Classifier training instances of features paired with classes (parser actions) were extracted from the trees in the training set, and the total number of training instances was about 1.9 million. It is interesting to note that the procedure of training the best-first parser is identical to the training of a deterministic version of the parser: the deterministic

[1]The SS MaxEnt library is publicly available from http://www-tsujii.is.s.u-tokyo.ac.jp/ tsuruoka/maxent/.

Let:

$S(n)$ denote the nth item from the top of the stack S, and
$W(n)$ denote the nth item from the front of the queue W.

Features:

1. The head-word (and its POS tag) of: $S(0)$, $S(1)$, $S(2)$, $and S(3)$
2. The head-word (and its POS tag) of: $W(0)$, $W(1)$, $W(2)$ and $W(3)$
3. The non-terminal node of the root of: $S(0)$, and $S(1)$
4. The non-terminal node of the left child of the root of: $S(0)$, and $S(1)$
5. The non-terminal node of the right child of the root of: $S(0)$, and $S(1)$
6. The POS tag of the head-word of the left child of the root of: $S(0)$, and $S(1)$
7. The POS tag of the head-word of the right child of the root of: $S(0)$, and $S(1)$
8. The linear distance (number of words apart) between the head-words of $S(0)$ and $S(1)$
9. The number of lexical items (words) that have been found (so far) to be dependents of the head-words of: $S(0)$, and $S(1)$
10. The most recently found lexical dependent of the head-word of $S(0)$ that is to the left of $S(0)$'s head
11. The most recently found lexical dependent of the head-word of $S(0)$ that is to the right of $S(0)$'s head
12. The most recently found lexical dependent of the head-word of $S(1)$ that is to the left of $S(1)$'s head
13. The most recently found lexical dependent of the head-word of $S(1)$ that is to the right of $S(1)$'s head
14. The previous parser action applied to the current parser state

Figure 1: Features used for classification, with features 1 to 13 taken from Sagae and Lavie (2005). The features described in items 1 – 7 are more directly related to the lexicalized constituent trees that are built during parsing, while the features described in items 8 – 13 are more directly related to the dependency structures that are built simultaneously to the constituent structures.

algorithm is simply run over all sentences in the training set, and since the correct trees are known in advance, we can simply record the features and correct parser actions that lead to the construction of the correct tree.

Training the maximum entropy classifier with such a large number (1.9 million) of training instances and features required more memory than was available (the maximum training set size we were able to train with 2GB of RAM was about 200,000 instances), so we employed the training set splitting idea used by Yamada and Matsumoto (2003) and Sagae and Lavie (2005). In our case, we split the training data according to the part-of-speech (POS) tag of the head-word of the item on top of the stack, and trained each split of the training data separately. At run-time, every trained classifier is loaded, and the choice of classifier to use is made by looking at the head-word of the item on top of the stack in the current parser state. The total training time (a single machine was used and each classifier was trained in series) was slightly under nine hours. For comparison, Sagae and Lavie (2005) report that training support vector machines for one-against-all multi-class classification on the same set of features for their deterministic parser took 62 hours, and training a k-nearest neighbors classifier took 11 minutes.

When given perfectly tagged text (gold part-of-speech tags extracted from the Penn Treebank), our parser has labeled constituent precision and recall of 89.40% and 88.79% respectively over all sentences in the test set, and 90.01% and 89.32% over sentences with length of at most 40 words. These results are at the same level of accuracy as those obtained with other state-of-the-art statistical parsers, although still well below the best published results for this test set (Bod, 2003; Charniak and Johnson, 2005). Although the parser is quite accurate, parsing the test set took 41 minutes. By implementing a very simple pruning strategy, the parser can be made much faster. Pruning the search space is done by only adding a new parser state to the heap if its probability is greater than $1/b$ of the probability of the most likely state in the heap that has had the same number of parser actions. By setting b to 50, the parser's accuracy is only affected minimally, and we obtain 89.3% precision and 88.7% recall, while parsing the test set in slightly under 17 minutes and taking less than 60 megabytes of RAM. Under the same conditions, but using automatically assigned part-of-speech tags (at 97.1% accuracy) using the SVMTool tagger (Gimenez and Marquez, 2004), we obtain 88.1% precision and 87.8% recall. It is likely that the deterioration in accuracy is aggravated by the training set splitting scheme based on POS tags.

A deterministic version of our parser, obtained by simply taking the most likely parser action as the only action at each step (in other words, by setting b to 1), has precision and recall of 85.4% and 84.8%, respectively (86.5% and 86.0% using gold-standard POS tags). More interestingly, it parses all 2,416 sentences (more than 50,000 words) in only 46 seconds, 10 times faster than the deterministic SVM parser of Sagae and Lavie (2005). The parser of Tsuruoka and Tsujii (Tsuruoka and Tsujii, 2005) has comparable speed, but we obtain more accurate results. In addition to being fast, our deterministic parser is also lean, requiring only about 25 megabytes of RAM.

A summary of these results is shown in table 1, along with the results obtained with other parsers for comparison purposes. The figures shown in table 1 only include experiments using automatically assigned POS tags. Results obtained with gold-standard POS tags are not shown, since they serve little purpose in a comparison with existing parsers. Although the time figures reflect the performance of each parser at the stated level of accuracy, all of the search-based parsers can trade accuracy for increased speed. For example, the Charniak parser can be made twice as fast at the cost of a 0.5% decrease in precision/recall, or ten times as fast at the cost of a 4% decrease in precision/recall (Roark and Charniak, 2002).

4.1 Reranking with the Probabililstic Shift-Reduce Model

One interesting aspect of having an accurate parsing model that is significantly different from other well-known generative models is that the combination of two accurate parsers may produce even more accurate results. A probabilistic shift-reduce LR-like model, such as the one used in our parser, is different in many ways from a lexicalized PCFG-like model (using markov a grammar), such as those used in the Collins (1999) and Charniak (2000) parsers. In the probabilistic LR model, probabilities are assigned to tree

	Precision	Recall	F-score	Time (min)
Best-First Classifier-Based (this paper)	88.1	87.8	87.9	17
Deterministic (MaxEnt) (this paper)	85.4	84.8	85.1	< 1
Charniak & Johnson (2005)	91.3	90.6	91.0	Unk
Bod (2003)	90.8	90.7	90.7	145*
Charniak (2000)	89.5	89.6	89.5	23
Collins (1999)	88.3	88.1	88.2	39
Ratnaparkhi (1997)	87.5	86.3	86.9	Unk
Tsuruoka & Tsujii (2005): deterministic	86.5	81.2	83.8	< 1*
Tsuruoka & Tsujii (2005): search	86.8	85.0	85.9	2*
Sagae & Lavie (2005)	86.0	86.1	86.0	11*

Table 1: Summary of results on labeled precision and recall of constituents, and time required to parse the test set. We first show results for the parsers described here, then for four of the most accurate or most widely known parsers, for the Ratnaparkhi maximum entropy parser, and finally for three recent classifier-based parsers. For the purposes of direct comparisons, only results obtained with automatically assigned part-of-speech tags are shown (tags are assigned by the parser itself or by a separate part-of-speech tagger). * Times reported by authors running on different hardware.

derivations (not the constituents themselves) based on the sequence of parser shift/reduce actions. PCFG-like models, on the other hand, assign probabilities to the trees directly. With models that differ in such fundamental ways, it is possible that the probabilities assigned to different trees are independent enough that even a very simple combination of the two models may result in increased accuracy.

We tested this hypothesis by using the Charniak (2000) parser in n-best mode, producing the top 10 trees with corresponding probabilities. We then rescored the trees produced by the Charniak parser using our probabilistic LR model, and simply multiplied the probabilities assigned by the Charniak model and our LR model to get a combined score for each tree[2]. On development data this resulted in a 1.3% absolute improvement in f-score over the 1-best trees produced by the Charniak parser. On the test set (WSJ Penn Treebank section 23), this reranking scheme produces precision of 90.9% and recall of 90.7%, for an f-score of 90.8%.

[2]The trees produced by the Charniak parser may include the part-of-speech tags AUX and AUXG, which are not part of the original Penn Treebank tagset. See (Charniak, 2000) for details. These are converted deterministically into the appropriate Penn Treebank verb tags, possibly introducing a small number of minor POS tagging errors. Gold-standard tags or the output of a separate part-of-speech tagger are not used at any point in rescoring the trees.

5 Conclusion

We have presented a best-first classifier-based parser that achieves high levels of precision and recall, with fast parsing times and low memory requirements. One way to view the parser is as an extension of recent work on classifier-based deterministic parsing. It retains the modularity between parsing algorithms and learning mechanisms associated with deterministic parsers, making it simple to understand, implement, and experiment with. Another way to view the parser is as a variant of probabilistic GLR parsers without an explicit LR table.

We have shown that our best-first strategy results in significant improvements in accuracy over deterministic parsing. Although the best-first search makes parsing slower, we have implemented a beam strategy that prunes much of the search space with very little cost in accuracy. This strategy involves a parameter that can be used to control the trade-off between accuracy and speed. At one extreme, the parser is very fast (more than 1,000 words per second) and still moderately accurate (about 85% f-score, or 86% using gold-standard POS tags). This makes it possible to apply parsing to natural language tasks involving very large amounts of text (such as question-answering or information extraction with large corpora). A less aggressive pruning setting results in an f-score of about 88% (or 89%, using gold-standard POS tags), taking 17 minutes to parse the WSJ test set.

Finally, we have shown that by multiplying the probabilities assigned by our maximum entropy shift-reduce model to the probabilities of the 10-best trees produced for each sentence by the Charniak parser, we can rescore the trees to obtain more accurate results than those produced by either model in isolation. This simple combination of the two models produces an f-score of 90.8% for the standard WSJ test set.

Acknowledgements

We thank John Carroll for insightful discussions at various stages of this work, and the reviewers for their detailed comments. This work was supported in part by the National Science Foundation under grant IIS-0414630.

References

A. Berger, S. A. Della Pietra, and V. J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

D. Bikel. 2002. Design of a multi-lingual, parallel-processing statistical parsing engine. In *Proceedings of HLT2002*. San Diego, CA.

R. Bod. 2003. An efficient implementation of a new dop model. In *Proceedings of the European chapter of the 2003 meeting of the Association for Computational Linguistics*. Budapest, Hungary.

E. Briscoe and J. Carroll. 1993. Generalised probabilistic lr parsing of natural language (corpora) with unification-based grammars. *Computational Linguistics*, 19(1):25–59.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd meeting of the Association for Computational Linguistics*. Ann Arbor, MI.

Eugene Charniak, Sharon Goldwater, and Mark Johnson. 1998. Edge-based best-first chart parsing. In *Proceedings of the Sixth Workshop on Very Large Corpora*. Montreal, Canada.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the First Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 132–139. Seattle, WA.

Michael Collins. 1997. Three generative, lexicalized models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 16–23.

M. Collins. 1999. *Head-Driven Models for Natural Language Parsing*. Phd thesis, University of Pennsylvania.

J. Gimenez and L. Marquez. 2004. Svmtool: A general pos tagger generator based on support vector machines. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*. Lisbon, Portugal.

Dan Klein and Christopher D. Manning. 2002. Fast exact inference with a factored model for natural language parsing. In *Advances in Neural Information Processing Systems 15 (NIPS 2002)*. Vancouver, BC.

D. E. Knuth. 1965. On the translation of languages from left to right. *Information and Control*, 8(6):607–639.

M. P. Marcus, B. Santorini, and M. A. Marcinkiewics. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19.

Joakim Nivre and Mario Scholz. 2004. Deterministic dependency parsing of english text. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 64–70. Geneva, Switzerland.

Adwait Ratnaparkhi. 1997. A linear observed time statistical parser based on maximum entropy models. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*. Providence, RI.

B. Roark and E. Charniak. 2002. Measuring efficiency in high-accuracy, broad coverage statistical parsing. In *Proceedings of the Efficiency in Large-scale Parsing Systems Workshop at COLING-2000*. Luxembourg.

Kenji Sagae and Alon Lavie. 2005. A classifier-based parser with linear run-time complexity. In *Proceedings of the Ninth International Workshop on Parsing Technologies*. Vancouver, BC.

Masaru Tomita. 1987. An efficient augmented context-free parsing algorithm. *Computational Linguistics*, 13:31–46.

Masaru Tomita. 1990. The generalized lr parser/compiler - version 8.4. In *Proceedings of the International Conference on Computational Linguistics (COLING'90)*, pages 59–63. Helsinki, Finland.

Y. Tsuruoka and K. Tsujii. 2005. Chunk parsing revisited. In *Proceedings of the Ninth International Workshop on Parsing Technologies*. Vancouver, Canada.

H. Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis using support vector machines. In *Proceedings of the Eighth International Workshop on Parsing Technologies*. Nancy, France.

Implementing a Characterization of Genre for Automatic Genre Identification of Web Pages

Marina Santini
NLTG
University of Brighton
UK
M.Santini@brighton.ac.uk

Richard Power
Computing Department
Open University
UK
r.power@open.ac.uk

Roger Evans
NLTG
University of Brighton
UK
R.P.Evans@brighton.ac.uk

Abstract

In this paper, we propose an implementable characterization of genre suitable for automatic genre identification of web pages. This characterization is implemented as an inferential model based on a modified version of Bayes' theorem. Such a model can deal with genre hybridism and individualization, two important forces behind genre evolution. Results show that this approach is effective and is worth further research.

1 Introduction

The term 'genre' is employed in virtually all cultural fields: literature, music, art, architecture, dance, pedagogy, hypermedia studies, computer-mediated communication, and so forth. As has often been pointed out, it is hard to pin down the concept of genre from a unified perspective (cf. Kwasnik and Crowston, 2004). This lack is also experienced in the more restricted world of non-literary or non-fictional document genres, such as professional or instrumental genres, where variation due to personal style is less pronounced than in literary genres. In particular, scholars working with practical genres focus upon a specific environment. For instance Swales (1990) develops his notion of genre in academic and research settings, Bathia (1993) in professional settings, and so on. In automatic genre classification studies, genres have often been seen as non-topical categories that could help reduce information overload (e.g. Mayer zu Eissen and Stein, 2004; Lim et al., 2005).

Despite the lack of an agreed theoretical notion, genre is a well-established term, intuitively understood in its vagueness. What humans intuitively perceive is that there are categories created within a culture, a society or a community which are used to group documents that share some conventions. Each of these groups is a genre, i.e. a cultural object or artefact, purposely made to meet and streamline communicative needs. Genres show sets of standardized or conventional characteristics that make them recognizable, and this identity raises specific expectations.

Together with conventions and expectations, genres have many other traits. We would like to focus on three traits, namely hybridism, individualization and evolution. Genres are not mutually exclusive and different genres can be merged into a single document, generating hybrid forms. Also, genres allow a certain freedom of variation and consequently can be individualized. Finally, genre repertoires are dynamic, i.e. they change over time, thus triggering genre change and evolution. It is also important to notice that before genre conventions become fully standardized, a genre does not have an official name. A genre name becomes acknowledged when the genre itself has an active role and a communicative function in a community or society (Swales, 1990). Before this acknowledgement, a genre shows hybrid or individualized forms, and indistinct functions.

Putting all these traits together, we suggest the following broad theoretical characterization of genre of written texts: genres are named communication artefacts characterized by conventions, raising expectations, showing hybridism and individualization, and undergoing evolution.

This characterization is flexible enough to encompass not only paper genres (both literary and practical genres), but also digital genres, and more specifically web genres. Web genres or cybergenres (Shepherd and Watters 1998) are those genres created by the combination of the use of the computer and the Internet.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 699–706,
Sydney, July 2006.

Genre hybridism and individualization are very evident on the web. In fact, web pages are often very hybrid because of the wider intra-genre variation and the smaller inter-genre differentiation. They can also be highly individualized because of the creative freedom provided by HTML tags (the building blocks of web pages) or programming languages such as Javascript. We suggest that genre hybridism and individualization can be seen as forces acting behind genre evolution. They allow the upgrade of existing genres and the creation of novel genres.

The change of genre repertoire and the creation of new genres were well illustrated by Crowston and Williams (2000) and Shepherd and Watters (1998). Both these studies describe a similar process. Web genres can start either as reproductions or as unprecedented types of documents. In the first case, existing genres are gradually upgraded and modified to adapt to potentials offered by the web. These variants might become very different from the original genres with time passing by. In the second case, novel genres can be generated from specific needs and requirements of the web. Crowston and Williams (2000) have traced this evolution through a manual qualitative survey of 1000 web pages. Shepherd and Watters (1998) have proposed a fuzzy taxonomy for web genres.

We would like to add a new force in this scenario, namely emerging genres. Emerging genre are those genres still in formation, not fully standardized and without any name or fixed function. For example, before 1998 web logs (or blogs) were already present on the web, but they were not yet identified as a genre. They were just "web pages", with similar characteristics and functions. In 1999, suddenly a community sprang up using this new genre (Blood, 2000). Only at this point, the genre "web log" or "blog" started being spread and being recognized.

Emerging genres may account for all those web pages, which remain unclassified or unclassifiable (cf. Crowston and Williams, 2000) because they show genre mixture or no genre at all. Authors often point out that assigning a genre to a web page might be difficult and controversial (e.g. Roussinov et al., 2001; Meyer zu Eissen and Stein, 2004; Shepherd et al., 2004) because web pages can appear hybrid or peculiar. Genre-mixed web pages or web pages without any evident genre can represent the antecedent of a future genre, but currently they might be considered as belonging to a genre still in formation. It is also important to highlight, however, that since the acknowledgement of genre relies on social acceptance, it is impossible to define the exact point at which a new genre emerges (Crowston and Williams 2000). The multi-facetted model capable of hosting new genres wished for by Kwasnik and Crowston (2004), and the adaptive learning system that can identify genre as they emerge announced by Shepherd et al. (2004) are hard to implement. For this reason, the focus of the method proposed below is not to detect emerging genres, but to show a flexible approach capable of giving account of genre hybridism and individualization.

Flexible genre classification systems are uncommon in automatic genre classification studies. Apart from two notable exceptions, namely Kessler et al. (1997) and Rehm (2006) whose implementations require extensive manual annotation (Kessler et al., 1997) or analysis (Rehm, 2006), genres are usually classified as single-label discrete entities, relying on the simplified assumption that a document can be assigned to only one genre.

In this paper, we propose a tuple representation that maps onto the theoretical characterization of genre suggested above and that can be implemented without much overhead. The implementable tuple includes the following attributes:

(genre(s)) of web pages=<***linguistic features, HTML, text types, [...]***>

This tuple means that web pages can have zero, one or more genres (`(genre(s)) of web pages`) and that this situation can be captured by a number of attributes. For the time being these attributes are limited to `linguistic features, HTML tags, text types`, but in future other attributes can be added (`[...]`). The attributes of the tuple can capture the presence of textual conventions or their absence. The presence of conventions brings about expectations, and can be used to identify acknowledged genres. The absence of conventions brings about hybridism and individualisation and can be interpreted in terms of emerging genres and genre evolution.

In this paper we present a simple model that implement the tuple and can deal with this complex situation. This model is based on statistical inference, performs automatic text analysis and has a classification scheme that includes zero labels, one label or multiple labels. More specifically, in addition to the traditional single-label classification, a zero-label

classification is useful when, for example, a web page is so peculiar from a textual point of view that it does not show any similarity with the genres included in the model. Conversely, a multi-label classification is useful when web pages show several genres at the same time. As there is no standard evaluation metrics for a comprehensive evaluation of such a model, we defer to further research the assessment of the model as a whole. In this paper, we report a partial evaluation based on single-label classification accuracy and predictions.

From a theoretical point of view, the inferential model makes a clear-cut separation between the concepts of 'text types' and 'genres'. Text types are rhetorical/discourse patterns dictated by the purposes of a text. For example, when the purpose of a text producer is to narrate, the narration text type is used. On the contrary, genres are cultural objects created by a society or a community, characterized by a set of linguistic and non-linguistic conventions, which can be fulfilled, personalized, transgressed, colonized, etc., but that are nonetheless recognized by the members of the society and community that have created them, raising predictable expectations. For example, what we expect from a personal blog is diary-form narration of the self, where opinions and comments are freely expressed.

The model presented here is capable of inferring text types from web pages using a modified form of Bayes' theorem, and derive genres through *if-then* rules.

With this model, emerging genres can be hypothesized through the analysis of unexpected combinations of text types and/or other traits in a large number of web pages. However, this potential will be investigated in future work. The results presented here are just a first step towards a more dynamic view of a genre classification system.

Automatic identification of text types and genres represents a great advantage in many fields because manual annotation is expensive and time-consuming. Apart from the benefits that it could bring to information retrieval, information extraction, digital libraries and so forth, automatic identification of text types and genres could be particularly useful for problems that natural language processing (NLP) is concerned with. For example, parsing accuracy could be increased if parsers were tested on different text types or genres, as certain constructions may occur only in certain types of texts. The same is true for Part-of-Speech (POS) tagging and word sense disambiguation. More accurate NLP tools could in turn be beneficial for automatic genre identification, because many features used for this task are extracted from the output of taggers and parsers, such as POS frequencies and syntactic constructions.

The paper is organized as follows: Section 2 reports previous characterization that have been implemented as statistical or computational models; Section 3 illustrates the attributes of the tuple; Section 4 describes the inferential model and reports evaluation; finally in Section 5 we draw some conclusions and outline points for future work.

2 Background

Although both Crowston and Williams (2000) and Shepherd and Watters (1998) have well described the evolution of genres on the web, when it comes to the actual genre identification of web pages (Roussinov et al., 2001; and Shepherd et al., 2004, respectively), they set aside the evolutionary aspect and consider genre from a static point of view. For Crowston and Williams (2000) and the follow-up Roussinov et al. (2001) most genres imply a combination of <purpose/function, form, content>, and, as they are complex entities, a multi-facetted classification seems appropriate (Kwasnik and Crowston, 2004). For Shepherd and Watters (1998) and the practical implementation Shepherd et al. (2004), cybergenres or web genres are characterized by the triple <content, form, functionality>, where functionality is a key evolutionary aspect afforded by the web. Crowston and co-workers have not yet implemented the combination of <purpose/function, form, content> together with the facetted classification in any automatic classification model, but the tuple <content, form, function> has been employed by Rehm (2006) for an original approach to single-web genre analysis, the personal home pages in the domain of academia. Rehm (2006) describes the relationship between HTML and web genres and depicts the evolutionary processes that shape and form web genres. In the practical implementation, however, he focuses only on a single web genre, the academic's personal home page, that is seen from a static point of view. As far as we know, Boese and Howe (2005) is the only study that tries to implement a diachronic view on genre of web pages using the triple

<style, form, content>. This study has the practical aim of finding out whether feature sets for genre identification need to be changed or updated because of genre evolution. They tried to detect the change through the use of a classifier on two parallel corpora separated by a six-year gap. Although this study does not focus on how to detect newly created web genres or how to deal with difficult web pages, it is an interesting starting point for traditional diachronic analysis applied to automatic genre classification.

In contrast, the model described in this paper aims at pointing out genre hybridism and individualisation in web pages. These two phenomena can be interpreted in terms of genre evolution in future investigations.

3 Attributes of the Tuple

The attributes *<linguistic features, HTML tags, text types>* of the tuple represent the computationally tractable version of the combination <purpose, form> often used to define the concept of genre (e.g. cf. Roussinov et al. 2001).

In our view, the purpose corresponds to text types, i.e. the rhetorical patterns that indicate what a text has been written for. For example, a text can be produced to narrate, instruct, argue, etc. Narration, instruction, and argumentation are examples of text types. As stressed earlier, text types are usually considered separate entities from genres (cf. Biber, 1988; Lee, 2001).

Form is a more heterogeneous attribute. Form can refer to linguistic form and to the shape (layout etc.). From an automatic point of view, linguistic form is represented by linguistic features, while shape is represented by HTML tags. Also the functionality attribute introduced by Shepherd and Watters (1998) can be seen in terms of HTML tags (e.g. tags for links and scripts). While content words or terms show some drawbacks for automatic genre identification (cf. Boese and Howe, 2005), there are several types of linguistic features that return good results, for instance, Biberian features (Biber, 1988). In the model presented here we use a mixture of Biberian features and additional syntactic traits. The total number of features used in this implementation of the model is 100. These features are available online at: http://www.nltg.brighton.ac.uk/home/Marina.Santini/

4 Inferential Model

The inferential model presented here (partially discussed in Santini (2006a) combines the advantages of deductive and inductive approaches. It is deductive because the co-occurrence and the combination of features in text types is decided a priori by the linguist on the basis on previous studies, and not derived by a statistical procedure, which is too biased towards high frequencies (some linguistic phenomena can be rare, but they are nonetheless discriminating). It is also inductive because the inference process is corpus-based, which means that it is based on a pool of data used to predict some text types. A few handcrafted *if-then* rules combine the inferred text types with other traits (mainly layout and functionality tags) in order to suggest genres. These rules are worked out either on the basis of previous genre studies or of a cursory qualitative analysis. For example, rules for personal home pages are based on the observations by Roberts (1998), Dillon and Gushrowski (2000). When previous studies were not available, as in the cases of eshops or search pages, the author of this paper has briefly analysed these genres to extract generalizations useful to write few rules.

It is important to stress that there is no hand-coding in the model. Web pages were randomly downloaded from genre-specific portals or archives without any further annotation. Web pages were parsed, linguistic features were automatically extracted and counted from the parsed outputs, while frequencies of HTML tags were automatically counted from the raw web pages. All feature frequencies were normalized by the length of web pages (in tokens) and then submitted to the model.

As stated earlier, the inferential model makes a clear-cut separation between text types and genres. The four text types included in this implementation are: *descriptive_narrative, expository_informational, argumentative_persuasive,* and *instructional*. The linguistic features for these text types come from previous (corpus-)linguistic studies (Werlich 1976; Biber, 1988; etc.), and are not extracted from the corpus using statistical methods. For each web page the model returns the probability of belonging to the four text types. For example, a web page can have 0.9 probabilities of being argumentative_persuasive, 0.7 of being instructional and so on. Probabilities are interpreted in terms of degree or gradation. For example, a web page with 0.9 probabilities

of being argumentative_persuasive shows a high gradation of argumentation. Gradations/ probabilities are ranked for each web page.
The computation of text types as intermediate step between linguistic and non-linguistic features and genres is useful if we see genres as conventionalised and standardized cultural objects raising expectations. For example, what we expect from an editorial is an 'opinion' or a 'comment' by the editor, which represents, broadly speaking, the view of the newspaper or magazine. Opinions are a form of 'argumentation'. Argumentation is a rhetorical pattern, or text type, expressed by a combination of linguistic features. If a document shows a high probability of being argumentative, i.e. it has a high gradation of argumentation, this document has a good chance of belonging to argumentative genres, such as editorials, sermons, pleadings, academic papers, etc. It has less chances of being a story, a biography, etc. We suggest that the exploitation of this knowledge about the textuality of a web page can add flexibility to the model and this flexibility can capture hybridism and individualization, the key forces behind genre evolution.

4.1 The Web Corpus

The inferential model is based on a corpus representative of the web. In this implementation of the model we approximated one of the possible compositions of a random slice of the web, statistically supported by reliable standard error measures. We built a web corpus with four BBC web genres (editorial, Do-It-Yourself (DIY) mini-guide, short biography, and feature), seven novel web genres (blog, eshop, FAQs, front page, listing, personal home page, search page), and 1,000 unclassified web pages from SPIRIT collection (Joho and Sanderson, 2004). The total number of web pages is 2,480. The four BBC genres represent traditional genres adapted to the functionalities of the web, while the seven genres are novel web genres, either unprecedented or showing a loose kinship with paper genres. Proportions are purely arbitrary and based on the assumption that at least half of web users tend to use recognized genre patterns in order to achieve felicitous communication. We consider the sampling distribution of the sample mean as approximately normal, following the Central Limit Theorem. This allows us to make inferences even if the population distribution is irregular or if variables are very skewed or highly discrete. The web corpus is available at: http://www.nltg.brighton.ac.uk/home/Marina.Santini/

4.2 Bayesian Inference: Inferring with Odds-Likelihood

The inferential model is based on a modified version of Bayes' theorem. This modified version uses a form of Bayes' theorem called *odds-likelihood* or *subjective Bayesian method* (Duda and Reboh, 1984) and is capable of solving more complex reasoning problems than the basic version. Odds is a number that tells us how much more likely one hypothesis is than the other. Odds and probabilities contain exactly the same information and are interconvertible. The main difference with original Bayes' theorem is that in the modified version much of the effort is devoted to weighing the contributions of different pieces of evidence in establishing the match with a hypothesis. These weights are confidence measures: Logical Sufficiency (LS) and Logical Necessity (LN). LS is used when the evidence is known to exist (larger value means greater sufficiency), while LN is used when evidence is known NOT to exist (a smaller value means greater necessity). LS is typically a number > 1, and LN is typically a number < 1. Usually LS*LN=1. In this implementation of the model, LS and LN were set to 1.25 and 0.8 respectively, on the basis of previous studies and empirical adjustments. Future work will include more investigation on the tuning of these two parameters.
The steps included in the model are the following:

1) Representation of the web in a corpus that is approximately normal.
2) Extraction, count and normalization of genre-revealing features.
3) Conversion of normalized counts into z-scores, which represent the deviation from the 'norm' coming out from the web corpus. The concept of "gradation" is based on these deviations from the norm.
4) Conversion of z-scores into probabilities, which means that feature frequencies are seen in terms of probabilities distribution.
5) Calculation of prior odds from prior probabilities of a text type. The prior probability for each of the four text types was set to 0.25 (all text types were given an equal chance to appear in a web page). Prior odds are calculated with the formula:

   `prOdds(H)=prProb(H)/1-prProb(H)`

6) Calculation of weighted features, or multipliers (M_n). If a feature or piece of evidence (E) has a

probability >=0.5, LS is applied, otherwise LN is applied. Multipliers are calculated with the following formulae:

```
if  Prob (E)>=0.5 then
    M(E)=1+(LS-1)(Prob(E)-0.5)/0.25
if Prob (E)<0.5 then
    M(E)=1-(1-LN)(0.5-Prob(E))/0.25
```

7) Multiplication of weighted probabilities together, according to the co-occurrence decided by the analyst on the basis of previous studies in order to infer text types. In this implementation the feature co-occurrence was decided following Werlich (1976) and Biber (1988).
8) Posterior odds for the text type is then calculated by multiplying prior odds (step 5) with co-occurrence of weighted features (step 7).
9) Finally, posterior odds is re-converted into a probability value with the following formula:

```
Prob(H)=Odds(H)/1+Odds(H)
```

Although odds contains exactly the same information as probability values, they are not constrained in 0-1 range, like probabilities.

Once text types have been inferred, *if-then* rules are applied for determining genres. In particular, for each of the seven web genre included in this implementation, few hand-crafted rules combine the two predominant text types per web genre with additional traits. For example, the actual rules for deriving a blog are as simple as the following ones:

```
if (text_type_1=descr_narrat_1|argum_pers_1)
if (text_type_2=descr_narrat_2|argum_pers_2)
if (page_length=LONG)
if (blog_words >= 0.5 probabilities)
then good blog candidate.
```

That is, if a web page has description_narration and argumentation_persuasion as the two predominant text types, and the page length is > 500 words (LONG), and the probability value for blog words is >=0.5 (blog words are terms such as *web log, weblog, blog, journal, diary, posted by, comments, archive* plus names of the days and months), then this web page is a good blog candidate.

For other web genres, the number of rules is higher, but it is worth saying that in the current implementation, rules are useful to understand how features interact and correlate.

One important thing to highlight is that each genre is computed independently for each web page. Therefore a web page can be assigned to different genres (Table 1) or to none (Table 2).
Multi-label and no-label classification cannot be evaluated with standard metrics and their evaluation requires further research. In the next subsection we present the evaluation of the single label classification returned by the inferential model.

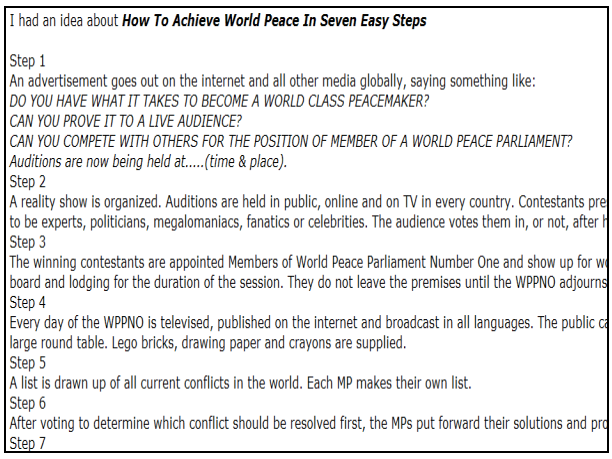
4.3 Evaluation of the Results

Single-label classification. For the seven web genres we compared the classification accuracy of the inferential model with the accuracy of classifiers. Two standard classifiers – SVM and Naive Bayes from Weka Machine Learning Workbench (Witten, Frank, 2005) – were run on the seven web genres. The stratified cross-validated accuracy returned by these classifiers for one seed is ca. 89% for SVM and ca. 67% for Naïve Bayes. The accuracy achieved by the inferential model is ca. 86%.

An accuracy of 86% is a good achievement for a first implementation, especially if we consider that the standard Naïve Bayes classifier returns an accuracy of about 67%. Although slightly lower than SVM, an accuracy of 86% looks promising because this evaluation is only on a single label. Ideally the inferential model could be more accurate than SVM if more labels could be taken into account. For example, the actual classification returned by the inferential model is shown in Table 1. The web pages in Table 1 are blogs but they also contain either sequences of questions and answers or are organized like a how-to document, like in the snippet in Figure 1

blog augustine 0000024	**GOOD blog**	BAD eshop	**GOOD faq**	BAD frontpage	BAD listing	BAD php	BAD spage
blog britblog 00000107	**GOOD blog**	BAD eshop	**GOOD faq**	BAD frontpage	BAD listing	BAD php	BAD spage

Table 1. Examples of multi-label classification


I had an idea about ***How To Achieve World Peace In Seven Easy Steps***

Step 1
An advertisement goes out on the internet and all other media globally, saying something like:
DO YOU HAVE WHAT IT TAKES TO BECOME A WORLD CLASS PEACEMAKER?
CAN YOU PROVE IT TO A LIVE AUDIENCE?
CAN YOU COMPETE WITH OTHERS FOR THE POSITION OF MEMBER OF A WORLD PEACE PARLIAMENT?
Auditions are now being held at.....(time & place).
Step 2
A reality show is organized. Auditions are held in public, online and on TV in every country. Contestants pre
to be experts, politicians, megalomaniacs, fanatics or celebrities. The audience votes them in, or not, after h
Step 3
The winning contestants are appointed Members of World Peace Parliament Number One and show up for w
board and lodging for the duration of the session. They do not leave the premises until the WPPNO adjourns
Step 4
Every day of the WPPNO is televised, published on the internet and broadcast in all languages. The public ca
large round table. Lego bricks, drawing paper and crayons are supplied.
Step 5
A list is drawn up of all current conflicts in the world. Each MP makes their own list.
Step 6
After voting to determine which conflict should be resolved first, the MPs put forward their solutions and pro
Step 7

Figure 1. Snippet *blog_augustine_0000024*

The snippet shows an example of genre colonization, where the vocabulary and text forms of one genre (FAQs/How to in this case) are inserted in another (cf. Beghtol, 2001). These strategies are frequent on the web and might give rise to new web genres. The model also captures a situation where the genre labels available in the system are not suitable for the web page under analysis, like in the example in Table 2.

SPRT_010_049_112_0055685	BAD blog	BAD eshop	BAD faq	BAD frontpage	BAD listing	BAD php	BAD spage

Table 2. Example of zero label classification

This web page (shown in Figure 2) from the unannotated SPIRIT collection (see Section 4.1) does not receive any of the genre labels currently available in the system.

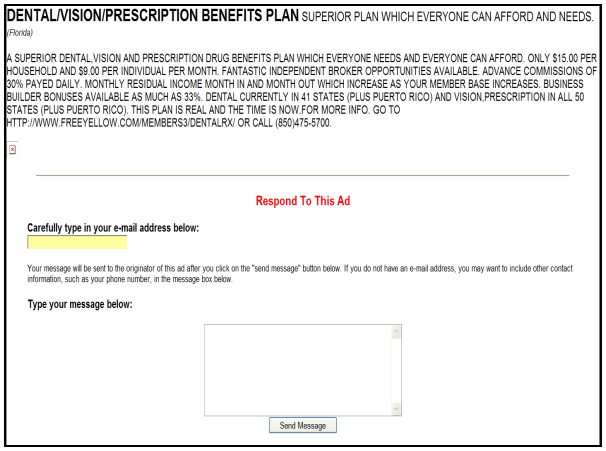


Figure 2. SPRT_010_049_112_0055685

If the pattern shown in Figure 2 keeps on recurring even when more web genres are added to the system, a possible interpretation could be that this pattern might develop into a stable web genre in future. If this happens, the system will be ready to host such a novelty. In the current implementation, only a few rules need to be added. In future implementations hand-crafted rules can be replaced by other methods. For example, an interesting adaptive solution has been explored by Segal and Kephart (2000).

Predictions. Precision of predictions on one web genre is used as an additional evaluation metric. The predictions on the eshop genre issued by the inferential model are compared with the predictions returned by two SVM models built with two different web page collections, Meyer-zu-Eissen collection and the 7-web-genre collection (Santini, 2006). Only the predictions on eshops are evaluated, because eshop is the only web genre shared by the three models. The number of predictions is shown in Table 3.

Models	Total Predictions	Correct Predictions	Incorrect Predictions and Uncertain
Meyer-zu-Eissen and SVM	6	3	3
7-web-genre and SVM	11	3	8
Web corpus and inferential model	17	6	11

Table 3. Predictions on eshops

The number of retrieved web pages (Total Predictions) is higher when the inferential model is used. Also the value of precision (Correct Predictions) is higher. The manual evaluation of the predictions is available online at: http://www.nltg.brighton.ac.uk/home/Marina.Santini/

5 Conclusions and Future Work

From a technical point of view, the inferential model presented in this paper is a simple starting point for reflection on a number of issues in automatic identification of genres in web pages. Although parameters need a better tuning and text type and genre palettes need to be enlarged, it seems that the inferential approach is effective, as shown by the preliminary evaluation reported in Section 4.3.

More importantly, this model instantiates a theoretical characterization of genre that includes hybridism and individualization, and interprets these two elements as the forces behind genre evolution. It is also worth noticing that the inclusion of the attribute 'text types' in the tuple gives flexibility to the model. In fact, the model can assign not only a single genre label, as in previous approaches to genre, but also multiple labels or no label at all. Ideally other computationally tractable attributes can be added to the tuple to increase flexibility and provide a multi-facetted classification, for example register or layout analysis.

However, other issues remain open. First, the possibility of a comprehensive evaluation of the model is to be explored. So far, only tentative evaluation schemes exist for multi-label classification (e.g. McCallum, 1999). Further research is still needed.

Second, in this model the detection of emerging genres can be done indirectly through the analysis of an unexpected combination of text types and/or genres. Other possibilities can be explored in future. Also the objective evaluation

of emerging genres requires further research and discussion.

More feasible in the short term is an investigation of the scalability of the model, when additional web pages, classified or not classified by genre, are added to the web corpus.

Also the possibility of replacing hand-crafted rules with some learning methodology can be explored in the near future. Apart from the approach suggested by Segal and Kephart (2000) mentioned above, many other pieces of experience are now available on adaptive learning (for example those reported in the EACL 2006 on Workshop on Adaptive Text Extraction and Mining).

References

Bathia V. 1993. *Analysing Genre. Language Use in Professional Settings*. Longman, London-NY.

Beghtol C. 2001. The Concept of Genre and Its Characteristics. *Bulletin of The American Society for Inform. Science and Technology*, Vol. 27 (2).

Biber D. 1988. *Variations across speech and writing*. Cambridge University Press, Cambridge.

Blood, R. 2000. Weblogs: A History and Perspective, Rebecca's Pocket.

Boese E. and Howe A. 2005. Effects of Web Document Evolution on Genre Classification. *CIKM 2005*, Germany.

Crowston K. and Williams M. 2000. Reproduced and Emergent Genres of Communication on the World-Wide Web, *The Information Society*, 16(3), 201-216.

Dillon, A. and Gushrowski, B. 2000. Genres and the Web: is the personal home page the first uniquely digital genre?, *JASIS*, 51(2).

Duda R. and Reboh R. 1984. AI and decision making: The PROSPECTOR experience. In Reitman, W. (Ed.), *Artificial Intelligence Applications for Business*, Norwood, NJ.

Joho H. and Sanderson M. 2004. The SPIRIT collection: an overview of a large web collection, *SIGIR Forum*, December 2004, Vol. 38(2).

Kessler B., Numberg G. and Shütze H. (1997), Automatic Detection of Text Genre, *Proc. 35 ACL and 8 EACL*.

Kwasnik B and Crowston K. 2004. A Framework for Creating a Facetted Classification for Genres: Addressing Issues of Multidimensionality. *Proc. 37 Hawaii Intern. Conference on System Science*.

Lee D. 2001. Genres, Registers, Text types, Domains, and Styles: Clarifying the concepts and navigating a path through the BNC Jungle. *Language Learning and Technology*, 5, 37-72.

Lim, C., Lee, K. and Kim G. 2005. Automatic Genre Detection of Web Documents, in Su K., Tsujii J., Lee J., Kwong O. Y. (eds.) *Natural Language Processing*, Springer, Berlin.

Meyer zu Eissen S. and Stein B. 2004. Genre Classification of Web Pages: User Study and Feasibility Analysis, in Biundo S., Fruhwirth T., Palm G. (eds.), *Advances in Artificial Intelligence*, Springer, Berlin, 256-269.

McCallum A. 1999. Multi-Label Text Classification with a Mixture Model Trained by EM, *AAAI'99 Workshop on Text Learning*.

Rehm G. 2006. Hypertext Types and Markup Languages. In Metzing D. and Witt A. (eds.), *Linguistic Modelling of Information and Markup Languages*. Springer, 2006 (in preparation).

Roberts, G. 1998. The Home Page as Genre: A Narrative Approach, *Proc. 31 Hawaii Intern. Conference on System Sciences*.

Roussinov D., Crowston K., Nilan M., Kwasnik B., Cai J., Liu X. 2001. Genre Based Navigation on the Web, *Proc. 34 Hawaii Intern. Conference on System Sciences*.

Santini M. 2006a. Identifying Genres of Web Pages, *TALN 06 - Actes de la 13 Conference sur le Traitement Automatique des Langues Naturelles*, Vol. 1, 307-316.

Santini M. 2006b. Some issues in Automatic Genre Classification of Web Pages, *JADT 06 – Actes des 8 Journées internationales d'analyse statistiques des donnés textuelles*, Vol 2, 865-876.

Segal R. and Kephart J. 2000. Incremental Learning in SwiftFile. *Proc. 17 Intern. Conf. on Machine Learning*.

Shepherd M. and Watters C. 1998. The Evolution of Cybergenre, *Proc. 31 Hawaii Intern. Conference on System Sciences*.

Shepherd M., Watters C., Kennedy A. 2004. Cybergenre: Automatic Identification of Home Pages on the Web. *Journal of Web Engineering*, Vol. 3(3-4), 236-251.

Swales, J. *Genre Analysis. English in academic and research settings*, Cambridge University Press, Cambridge, 1990.

Werlich E. (1976). *A Text Grammar of English*. Quelle & Meyer, Heidelberg.

Translating HPSG-style Outputs of a Robust Parser into Typed Dynamic Logic

Manabu Sato† **Daisuke Bekki**‡ **Yusuke Miyao**† **Jun'ichi Tsujii**†∗
† Department of Computer Science, University of Tokyo
Hongo 7-3-1, Bunkyo-ku, Tokyo 113-0033, Japan
‡ Center for Evolutionary Cognitive Sciences, University of Tokyo
Komaba 3-8-1, Meguro-ku, Tokyo 153-8902, Japan
∗School of Informatics, University of Manchester
PO Box 88, Sackville St, Manchester M60 1QD, UK
∗SORST, JST (Japan Science and Technology Corporation)
Honcho 4-1-8, Kawaguchi-shi, Saitama 332-0012, Japan
† {sa-ma, yusuke, tsujii}@is.s.u-tokyo.ac.jp
‡ bekki@ecs.c.u-tokyo.ac.jp

Abstract

The present paper proposes a method by which to translate outputs of a robust HPSG parser into semantic representations of Typed Dynamic Logic (TDL), a dynamic plural semantics defined in typed lambda calculus. With its higher-order representations of contexts, TDL analyzes and describes the inherently inter-sentential nature of quantification and anaphora in a strictly lexicalized and compositional manner. The present study shows that the proposed translation method successfully combines robustness and descriptive adequacy of contemporary semantics. The present implementation achieves high coverage, approximately 90%, for the real text of the Penn Treebank corpus.

1 Introduction

Robust parsing technology is one result of the recent fusion between symbolic and statistical approaches in natural language processing and has been applied to tasks such as information extraction, information retrieval and machine translation (Hockenmaier and Steedman, 2002; Miyao et al., 2005). However, reflecting the field boundary and unestablished interfaces between syntax and semantics in formal theory of grammar, this fusion has achieved less in semantics than in syntax.

For example, a system that translates the output of a robust CCG parser into semantic representations has been developed (Bos et al., 2004). While its corpus-oriented parser attained high coverage with respect to real text, the expressive power of the resulting semantic representations is confined to first-order predicate logic.

The more elaborate tasks tied to discourse information and plurality, such as resolution of anaphora antecedent, scope ambiguity, presupposition, topic and focus, are required to refer to 'deeper' semantic structures, such as dynamic semantics (Groenendijk and Stokhof, 1991).

However, most dynamic semantic theories are not equipped with large-scale syntax that covers more than a small fragment of target languages. One of a few exceptions is Minimal Recursion Semantics (MRS) (Copestake et al., 1999), which is compatible with large-scale HPSG syntax (Pollard and Sag, 1994) and has affinities with UDRS (Reyle, 1993). For real text, however, its implementation, as in the case of the ERG parser (Copestake and Flickinger, 2000), restricts its target to the static fragment of MRS and yet has a lower coverage than corpus-oriented parsers (Baldwin, to appear).

The lack of transparency between syntax and discourse semantics appears to have created a tension between the robustness of syntax and the descriptive adequacy of semantics.

In the present paper, we will introduce a robust method to obtain dynamic semantic representations based on Typed Dynamic Logic (TDL) (Bekki, 2000) from real text by translating the outputs of a robust HPSG parser (Miyao et al., 2005). Typed Dynamic Logic is a dynamic plural semantics that formalizes the structure underlying the semantic interactions between quantification, plurality, bound variable/E-type anaphora

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 707–714,
Sydney, July 2006.

$$
\begin{array}{lcl}
r^{e\times\cdots\times e\mapsto t}x_1^i\cdots x_n^i & \equiv & \lambda G^{(i\mapsto e)\mapsto t}.\lambda g^{i\mapsto e}.g\in G\wedge r\left\langle gx_1,\ldots,gx_m\right\rangle \\
\sim\phi^{prop} & \equiv & \lambda G^{(i\mapsto e)\mapsto t}.\lambda g^{i\mapsto e}.g\in G\wedge\neg\exists h^{i\mapsto e}.h\in\phi G \\
\left[\begin{array}{c}\phi^{prop}\\ \vdots \\ \varphi^{prop}\end{array}\right] & \equiv & \lambda G^{(i\mapsto e)\mapsto t}.(\varphi\cdots(\phi G)) \\
ref\left(x^i\right)[\phi^{prop}][\varphi^{prop}] & \equiv & \lambda G^{(i\mapsto e)\mapsto t}.\left\{\begin{array}{l}if\quad G/_x=\phi G/_x\\ then\ \lambda g^{i\mapsto e}.g\in\varphi G\wedge\ G/_x=\varphi G/_x\\ otherwise\ undefined\end{array}\right\} \\
\left(\begin{array}{lcl} where\quad prop & \equiv & ((i\mapsto e)\mapsto t)\mapsto(i\mapsto e)\mapsto t \\ g^\alpha\in G^{\alpha\mapsto t} & \equiv & Gg \\ G^{(i\mapsto e)\mapsto t}/_{x^i} & \equiv & \lambda d^e.\exists g^{i\mapsto e}.g\in G\wedge gx=d\end{array}\right) &&
\end{array}
$$

Figure 1: Propositions of TDL (Bekki, 2005)

and presuppositions. All of this complex discourse/plurality-related information is encapsulated within higher-order structures in TDL, and the analysis remains strictly lexical and compositional, which makes its interface with syntax transparent and straightforward. This is a significant advantage for achieving robustness in natural language processing.

2 Background

2.1 Typed Dynamic Logic

Figure 1 shows a number of propositions defined in (Bekki, 2005), including atomic predicate, negation, conjunction, and anaphoric expression. Typed Dynamic Logic is described in typed lambda calculus (Gödel's System T) with four ground types: *e*(entity), *i*(index), *n*(natural number), and *t*(truth). While assignment functions in static logic are functions in meta-language from type e variables (in the case of first-order logic) to objects in the domain D_e, assignment functions in TDL are functions in object-language from indices to entities. Typed Dynamic Logic defines the notion *context* as a set of assignment functions (an object of type $(i\mapsto e)\mapsto t$) and a *proposition* as a function from context to context (an object of type $((i\mapsto e)\mapsto t)\mapsto(i\mapsto e)\mapsto t$). The conjunctions of two propositions are then defined as composite functions thereof. This setting conforms to the view of "propositions as information flow", which is widely accepted in dynamic semantics.

Since all of these higher-order notions are described in lambda terms, the path for compositional type-theoretic semantics based on functional application, functional composition and type raising is clarified. The derivations of TDL semantic representations for the sentences "A boy ran. He tumbled." are exemplified in Figure 2 and Figure 3. With some instantiation of variables, the semantic representations of these two sentences are simply conjoined and yield a single representation, as shown in (1).

$$
(1)\quad \left[\begin{array}{l} boy'x_1s_1 \\ run'e_1s_1 \\ agent'e_1x_1 \\ ref\,(x_2)\,[]\left[\begin{array}{l} tumble'e_2s_2 \\ agent'e_2x_2\end{array}\right]\end{array}\right]
$$

The propositions $boy'x_1s_1$, $run'e_1s_1$ and $agent'e_1x_1$ roughly mean "the entity referred to by x_1 is a boy in the situation s_1", "the event referred to by e_1 is a running event in the situation s_1", and "the agent of event e_1 is x_1", respectively.

The former part of (1) that corresponds to the first sentence, filtering and testing the input context, returns the updated context schematized in (2). The updated context is then passed to the latter part, which corresponds to the second sentence as its input.

(2)

…	x_1	s_1	e_1	…
	john *john*	$situation_1$ $situation_2$	$running_1$ $running_2$	
	⋮	⋮	⋮	

This mechanism makes anaphoric expressions, such as "He" in "He tumbles", accessible to its preceding context; namely, the descriptions of their presuppositions can refer to the preceding context compositionally. Moreover, the referents of the anaphoric expressions are correctly calculated as a result of previous filtering and testing.

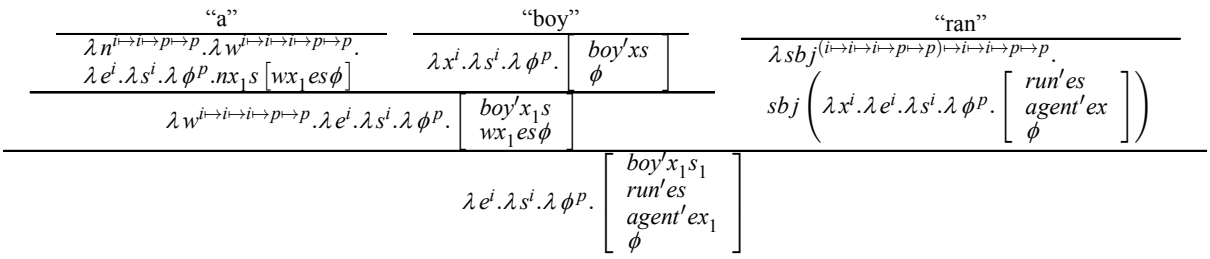


Figure 2: Derivation of a TDL semantic representation of "A boy ran".

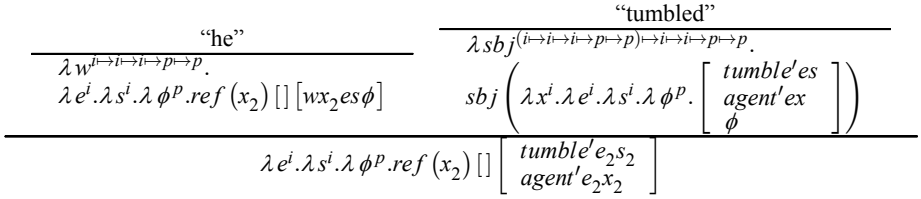


Figure 3: Derivation of TDL semantic representation of "He tumbled".

Although the antecedent for x_2 is not determined in this structure, the possible candidates can be enumerated: x_1, s_1 and e_1, which precede x_2. Since TDL seamlessly represents linguistic notions such as "entity", "event" and "situation", by indices, the anaphoric expressions, such as "the event" and "that case", can be treated in the same manner.

2.2 Head-driven Phrase Structure Grammar

Head-driven Phrase Structure Grammar (Pollard and Sag, 1994) is a kind of lexicalized grammar that consists of lexical items and a small number of composition rules called schema. Schemata and lexical items are all described in typed feature structures and the unification operation defined thereon.

(3)
$$\begin{bmatrix} PHON & \text{"boy"} \\ SYN & \begin{bmatrix} HEAD & \begin{bmatrix} noun \\ MOD & \langle\ \rangle \end{bmatrix} \\ VAL & \begin{bmatrix} SUBJ & \langle\ \rangle \\ COMPS & \langle\ \rangle \\ SPR & \langle det \rangle \end{bmatrix} \\ SLASH & \langle\ \rangle \end{bmatrix} \\ SEM & \end{bmatrix}$$

Figure 4 is an example of a parse tree, where the feature structures marked with the same boxed numbers have a shared structure. In the first stage of the derivation of this tree, lexical items are assigned to each of the strings, "John" and "runs." Next, the mother node, which dominates the two items,

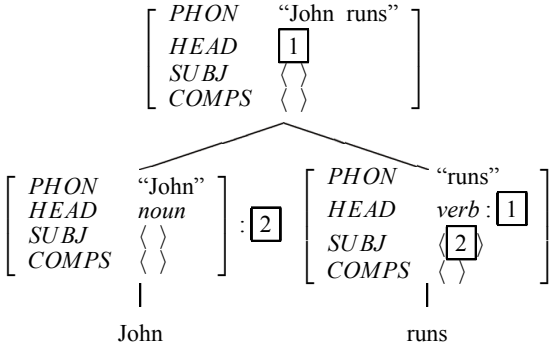


Figure 4: An HPSG parse tree

is generated by the application of Subject-Head Schema. The recursive application of these operations derives the entire tree.

3 Method

In this section, we present a method to derive TDL semantic representations from HPSG parse trees, adopting, in part, a previous method (Bos et al., 2004). Basically, we first assign TDL representations to lexical items that are terminal nodes of a parse tree, and then compose the TDL representation for the entire tree according to the tree structure (Figure 5). One problematic aspect of this approach is that the composition process of TDL semantic representations and that of HPSG parse trees are not identical. For example, in the HPSG

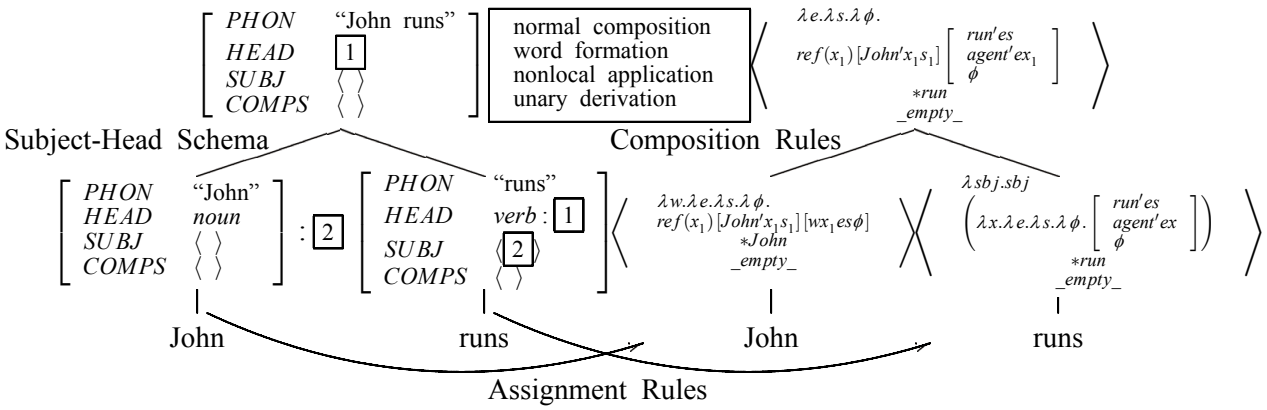


Figure 5: Example of the application of the rules

parser, a compound noun is regarded as two distinct words, whereas in TDL, a compound noun is regarded as one word. Long-distance dependency is also treated differently in the two systems. Furthermore, TDL has an operation called unary derivation to deal with empty categories, whereas the HPSG parser does not have such an operation.

In order to overcome these differences and realize a straightforward composition of TDL representations according to the HPSG parse tree, we defined two extended composition rules, **word formation rule** and **non-local application rule**, and redefined TDL unary derivation rules for the use in the HPSG parser. At each step of the composition, one composition rule is chosen from the set of rules, based on the information of the schemata applied to the HPSG tree and TDL representations of the constituents. In addition, we defined extended TDL semantic representations, referred to as **TDL Extended Structures (TDLESs)**, to be paired with the extended composition rules.

In summary, the proposed method is comprised of TDLESs, assignment rules, composition rules, and unary derivation rules, as will be elucidated in subsequent sections.

3.1 Data Structure

A TDLES is a tuple $\langle T, p, n \rangle$, where T is an extended TDL term, which can be either a TDL term or a special value ω. Here, ω is a value used by the **word formation rule**, which indicates that the word is a *word modifier* (See Section 3.3). In addition, p and n are the necessary information for extended composition rules, where p is a matrix predicate in T and is used by the **word formation rule**, and n is a nonlocal argument, which takes either a variable occurring in T or an empty value. This element corresponds to the SLASH feature in HPSG and is used by the **nonlocal application rule**.

The TDLES of the common noun "boy" is given in (4). The contents of the structure are T, p and n, beginning at the top. In (4), T corresponds to the TDL term of "boy" in Figure 2, p is the predicate *boy*, which is identical to a predicate in the TDL term (the identity relation between the two is indicated by "*"). If either T or p is changed, the other will be changed accordingly. This mechanism is a part of the **word formation rule**, which offers advantages in creating a new predicate from multiple words. Finally, n is an empty value.

(4) $\left\langle \begin{array}{c} \lambda x.\lambda s.\lambda\phi. \left[\begin{array}{l} *boy'xs \\ \phi \end{array} \right] \\ *boy \\ \_empty\_ \end{array} \right\rangle$

3.2 Assignment Rules

We define assignment rules to associate HPSG lexical items with corresponding TDLESs. For closed class words, such as "a", "the" or "not", assignment rules are given in the form of a template for each word as exemplified below.

(5) $\left[\begin{array}{ll} PHON & \text{"a"} \\ HEAD & det \\ SPEC & \langle noun \rangle \end{array} \right]$
$\Downarrow$
$\left\langle \begin{array}{c} \lambda x.\lambda s.\lambda\phi. \left[\begin{array}{l} \lambda n.\lambda w.\lambda e.\lambda s.\lambda\phi. \\ nx_1 s\,[wx_1 es\phi] \end{array} \right] \\ \_empty\_ \\ \_empty\_ \end{array} \right\rangle$

Shown in (5) is an assignment rule for the indefinite determiner "a". The upper half of (5) shows a template of an HPSG lexical item that specifies its phonetic form as "a", where POS is a determiner and specifies a noun. A TDLES is shown in the lower half of the figure. The TDL term slot of this structure is identical to that of "a" in Figure 2, while slots for the matrix predicate and nonlocal argument are empty.

For open class words, such as nouns, verbs, adjectives, adverbs and others, assignment rules are defined for each syntactic category.

$$(6)\quad \begin{bmatrix} PHON & P \\ HEAD & noun \\ MOD & \langle\rangle \\ SUBJ & \langle\rangle \\ COMPS & \langle\rangle \\ SPR & \langle det\rangle \end{bmatrix} \Downarrow \left\langle \begin{array}{c} \lambda x.\lambda s.\lambda\phi.\begin{bmatrix} *P'xs \\ \phi \end{bmatrix} \\ *P \\ \_empty\_ \end{array} \right\rangle$$

The assignment rule (6) is for common nouns. The HPSG lexical item in the upper half of (6) specifies that the phonetic form of this item is a variable, P, that takes no arguments, does not modify other words and takes a specifier. Here, POS is a noun. In the TDLES assigned to this item, an actual input word will be substituted for the variable P, from which the matrix predicate P' is produced. Note that we can obtain the TDLES (4) by applying the rule of (6) to the HPSG lexical item of (3).

As for verbs, a base TDL semantic representation is first assigned to a verb root, and the representation is then modified by lexical rules to reflect an inflected form of the verb. This process corresponds to HPSG lexical rules for verbs. Details are not presented herein due to space limitations.

3.3 Composition Rules

We define three composition rules: **the function application rule**, **the word formation rule**, and **the nonlocal application rule**. Hereinafter, let $S_L = \langle T_L, p_L, n_L\rangle$ and $S_R = \langle T_R, p_R, n_R\rangle$ be TDLESs of the left and the right daughter nodes, respectively. In addition, let S_M be TDLESs of the mother node.

Function application rule: The composition of TDL terms in the TDLESs is performed by function application, in the same manner as in the original TDL, as explained in Section 2.1.

Definition 3.1 (function application rule). *If* $Type(T_L) = \alpha$ *and* $Type(T_R) = \alpha \mapsto \beta$ *then*

$$S_M = \left\langle \begin{array}{c} T_R T_L \\ p_R \\ union(n_L, n_R) \end{array} \right\rangle$$

Else if $Type(T_L) = \alpha \mapsto \beta$ *and* $Type(T_R) = \alpha$ *then*

$$S_M = \left\langle \begin{array}{c} T_L T_R \\ p_L \\ union(n_L, n_R) \end{array} \right\rangle$$

In Definition 3.1, $Type(T)$ is a function that returns the type of TDL term T, and $union(n_L, n_R)$ is defined as:

$$union(n_L, n_R) = \begin{cases} empty & if\ n_L = n_R = \_empty\_ \\ n & if\ n_L = n,\ n_R = \_empty\_ \\ n & if\ n_L = \_empty\_,\ n_R = n \\ undefined & if\ n_L \neq \_empty\_,\ n_R \neq \_empty\_ \end{cases}$$

This function corresponds to the behavior of the union of SLASH in HPSG. The composition in the right-hand side of Figure 5 is an example of the application of this rule.

Word formation rule: In natural language, it is often the case that a new word is created by combining multiple words, for example, "orange juice". This phenomenon is called *word formation*. Typed Dynamic Logic and the HPSG parser handle this phenomenon in different ways. Typed Dynamic Logic does not have any rule for word formation and regards "orange juice" as a single word, whereas most parsers treat "orange juice" as the separate words "orange" and "juice". This requires a special composition rule for word formation to be defined. Among the constituent words of a compound word, we consider those that are not HPSG heads as *word modifiers* and define their value for T as ω. In addition, we apply the **word formation rule** defined below.

Definition 3.2 (word formation rule). *If* $Type(T_L) = \omega$ *then*

$$S_M = \left\langle \begin{array}{c} T_R \\ concat(p_L, p_R) \\ n_R \end{array} \right\rangle$$

Else if $Type(T_R) = \omega$ *then*

$$S_M = \left\langle \begin{array}{c} T_L \\ concat(p_L, p_R) \\ n_L \end{array} \right\rangle$$

$concat(p_L, p_R)$ in Definition 3.2 is a function that returns a concatenation of p_L and p_R. For example, the composition of a word modifier "orange" (7) and and a common noun "juice" (8) will generate the TDLES (9).

(7) $\left\langle \begin{array}{c} \omega \\ orange \\ \_empty\_ \end{array} \right\rangle$

(8) $\left\langle \begin{array}{c} \lambda x.\lambda s.\lambda \phi.\left[\begin{array}{c} *juice'xs \\ \phi \end{array} \right] \\ *juice \\ \_empty\_ \end{array} \right\rangle$

(9) $\left\langle \begin{array}{c} \lambda x.\lambda s.\lambda \phi.\left[\begin{array}{c} *orange\_juice'xs \\ \phi \end{array} \right] \\ *orange\_juice \\ \_empty\_ \end{array} \right\rangle$

Nonlocal application rule: Typed Dynamic Logic and HPSG also handle the phenomenon of wh-movement differently. In HPSG, a wh-phrase is treated as a value of SLASH, and the value is kept until the *Filler-Head Schema* are applied. In TDL, however, wh-movement is handled by the functional composition rule.

In order to resolve the difference between these two approaches, we define the **nonlocal application rule**, a special rule that introduces a slot relating to HPSG SLASH to TDLESs. This slot becomes the third element of TDLESs. This rule is applied when the *Filler-Head Schema* are applied in HPSG parse trees.

Definition 3.3 (nonlocal application rule). *If* $Type(T_L) = (\alpha \mapsto \beta) \mapsto \gamma$, $Type(T_R) = \beta$, $Type(n_R) = \alpha$ *and the Filler-Head Schema are applied in HPSG, then*

$$S_M = \left\langle \begin{array}{c} T_L(\lambda n_R.T_R) \\ p_L \\ \_empty\_ \end{array} \right\rangle$$

3.4 Unary Derivation Rules

In TDL, type-shifting of a word or a phrase is performed by composition with an empty category (a category that has no phonetic form, but has syntactic/semantic functions). For example, the phrase "this year" is a noun phrase at the first stage and can be changed into a verb modifier when combined with an empty category. Since many of the type-shifting rules are not available in HPSG, we defined unary derivation rules in order to provide an equivalent function to the type-shifting rules of TDL. These unary rules are applied independently with HPSG parse trees. (10) and (11) illustrate the unary derivation of "this year". (11) is derived from (10) using a unary derivation rule.

Table 1: Number of implemented rules

assignment rules	
HPSG-TDL template	51
for closed words	16
for open words	35
verb lexical rules	27

composition rules	
binary composition rules	3
function application rule	
word formation rule	
nonlocal application rule	
unary derivation rules	12

(10) $\left\langle \begin{array}{c} \lambda w.\lambda e.\lambda s.\lambda \phi.ref(x_1)[*year'x_1 s_1][wx_1 es\phi] \\ *year \\ \_empty\_ \end{array} \right\rangle$

(11) $\left\langle \begin{array}{c} \lambda v.\lambda e.\lambda s.\lambda \phi. \\ ref(x_1)[*year'x_1 s_1]\left[ves\left[\begin{array}{c} mod'ex_1 \\ \phi \end{array} \right]\right] \\ *year \\ \_empty\_ \end{array} \right\rangle$

4 Experiment

The number of rules we have implemented is shown in Table 1. We used the Penn Treebank (Marcus, 1994) Section 22 (1,527 sentences) to develop and evaluate the proposed method and Section 23 (2,144 sentences) as the final test set.

We measured the coverage of the construction of TDL semantic representations, in the manner described in a previous study (Bos et al., 2004). Although the best method for strictly evaluating the proposed method is to measure the agreement between the obtained semantic representations and the intuitions of the speaker/writer of the texts, this type of evaluation could not be performed because of insufficient resources. Instead, we measured the rate of successful derivations as an indicator of the coverage of the proposed system.

The sentences in the test set were parsed by a robust HPSG parser (Miyao et al., 2005), and HPSG parse trees were successfully generated for 2,122 (98.9%) sentences. The proposed method was then applied to these parse trees. Table 2 shows that 88.3% of the un-

Table 2: Coverage with respect to the test set

covered sentences	88.3 %
uncovered sentences	11.7 %
assignment failures	6.2 %
composition failures	5.5 %
word coverage	99.6 %

Table 3: Error analysis: the development set

# assignment failures	103
# unimplemented words	61
# TDL unsupporting words	17
# nonlinguistic HPSG lexical items	25
# composition failures	72
# unsupported compositions	20
# invalid assignments	36
# nonlinguistic parse trees	16

```
ref($1)[]
   [lecture($2,$3) &
    past($3) &
    agent($2,$1) &
    content($2,$4) &
    ref($5)[]
       [every($6)[ball($6,$4)]
            [see($7,$4) &
             present($4) &
             agent($7,$5) &
             theme($7,$6) &
             tremendously($7,$4) &
             ref($8)[]
                [ref($9)[groove($9,$10)]
                    [be($11,$4) &
                     present($4) &
                     agent($11,$8) &
                     in($11,$9) &
                     when($11,$7)]]]]]
```

Figure 6: Output for the sentence: *"When you're in the groove, you see every ball tremendously," he lectured.*

seen sentences are assigned TDL semantic representations. Although this number is slightly less than 92.3%, as reported by Bos et al., (2004), it seems reasonable to say that the proposed method attained a relatively high coverage, given the expressive power of TDL.

The construction of TDL semantic representations failed for 11.7% of the sentences. We classified the causes of the failure into two types. One of which is application failure of the assignment rules (assignment failure); that is, no assignment rules are applied to a number of HPSG lexical items, and so no TDLESs are assigned to these items. The other is application failure of the composition rules (composition failure). In this case, a type mismatch occurred in the composition, and so a TDLES was not derived.

Table 3 shows further classification of the causes categorized into the two classes. We manually investigated all of the failures in the development set.

Assignment failures are caused by three factors. Most assignment failures occurred due to the limitation in the number of the assignment rules (as indicated by "unimplemented words" in the table). In this experiment, we did not implement rules for infrequent HPSG lexical items. We believe that this type of failure will be resolved by increasing the number of assignment rules. The second factor in the table, "TDL unsupported words", refers to expressions that are not covered by the current theory of TDL. In order to resolve this type of failure, the development of TDL is required. The third factor, "nonlinguistic HPSG lexical items" includes a small number of cases in which TDLESs are not assigned to the words that are categorized as nonlinguistic syntactic categories by the HPSG parser. This problem is caused by ill-formed outputs of the parser.

The composition failures can be further classified into three classes according to their causative factors. The first factor is the existence of HPSG schemata for which we have not yet implemented composition rules. These failures will be fixed by extending of the definition of our composition rules. The second factor is type mismatches due to the unintended assignments of TDLESs to lexical items. We need to further elaborate the assignment rules in order to deal with this problem. The third factor is parse trees that are linguistically invalid.

The error analysis given above indicates that we can further increase the coverage through the improvement of the assignment/composition rules.

Figure 6 shows an example of the output for a sentence in the development set. The variables `$1, ...,$11` are indices that

represent entities, events and situations. For example, `$3` represents a situation and `$2` represents the lecturing event that exists in `$3`. `past($3)` requires that the situation is past. `agent($2,$1)` requires that the entity `$1` is the agent of `$2`. `content($2,$4)` requires that `$4` (as a set of possible worlds) is the content of `$2`. `be($11,$4)` refers to `$4`. Finally, `every($6)[ball($6,$4)][see($7,$4) ...]` represents a generalized quantifier "every ball". The index `$6` serves as an antecedent both for bound-variable anaphora within its scope and for E-type anaphora outside its scope. The entities that correspond to the two occurrences of "you" are represented by `$8` and `$5`. Their unification is left as an anaphora resolution task that can be easily solved by existing statistical or rule-based methods, given the structural information of the TDL semantic representation.

5 Conclusion

The present paper proposed a method by which to translate HPSG-style outputs of a robust parser (Miyao et al., 2005) into dynamic semantic representations of TDL (Bekki, 2000). We showed that our implementation achieved high coverage, approximately 90%, for real text of the Penn Treebank corpus and that the resulting representations have sufficient expressive power of contemporary semantic theory involving quantification, plurality, inter/intra-sentential anaphora and presupposition.

In the present study, we investigated the possibility of achieving robustness and descriptive adequacy of semantics. Although previously thought to have a trade-off relationship, the present study proved that robustness and descriptive adequacy of semantics are not intrinsically incompatible, given the transparency between syntax and discourse semantics.

If the notion of robustness serves as a criterion not only for the practical usefulness of natural language processing but also for the validity of linguistic theories, then the compositional transparency that penetrates all levels of syntax, sentential semantics, and discourse semantics, beyond the superficial difference between the laws that govern each of the levels, might be reconsidered as an essential principle of linguistic theories.

References

Timothy Baldwin, John Beavers, Emily M. Bender, Dan Flickinger, Ara Kim and Stephan Oepen (to appear) Beauty and the Beast: What running a broad-coverage precision grammar over the BNC taught us about the grammar ? and the corpus, In *Linguistic Evidence: Empirical, Theoretical, and Computational Perspectives*, Mouton de Gruyter.

Daisuke Bekki. 2000. Typed Dynamic Logic for Compositional Grammar, Doctoral Dissertation, University of Tokyo.

Daisuke Bekki. 2005. Typed Dynamic Logic and Grammar: the Introduction, manuscript, University of Tokyo,

Johan Bos, Stephen Clark, Mark Steedman, James R. Curran and Julia Hockenmaier. 2004. Wide-Coverage Semantic Representations from a CCG Parser, In *Proc. COLING '04*, Geneva.

Ann Copestake, Dan Flickinger, Ivan A. Sag and Carl Pollard. 1999. Minimal Recursion Semantics: An introduction, manuscript.

Ann Copestake and Dan Flickinger. 2000. An open-source grammar development environment and broad-coverage English grammar using HPSG In *Proc. LREC-2000*, Athens.

Jeroen Groenendijk and Martin Stokhof. 1991. Dynamic Predicate Logic, In *Linguistics and Philosophy 14*, pp.39-100.

Julia Hockenmaier and Mark Steedman. 2002. Acquiring Compact Lexicalized Grammars from a Cleaner Treebank, In *Proc. LREC-2002*, Las Palmas.

Mitch Marcus. 1994. The Penn Treebank: A revised corpus design for extracting predicate-argument structure. In *Proceedings of the ARPA Human Language Technolog Workshop*, Princeton, NJ.

Yusuke Miyao, Takashi Ninomiya and Jun'ichi Tsujii. 2005. Corpus-oriented Grammar Development for Acquiring a Head-driven Phrase Structure Grammar from the Penn Treebank, in *IJC-NLP 2004, LNAI3248*, pp.684-693. Springer-Verlag.

Carl Pollard and Ivan A. Sag. 1994. Head-Driven Phrase Structure Grammar, *Studies in Contemporary Linguistics*. University of Chicago Press, Chicago, London.

Uwe Reyle. 1993. Dealing with Ambiguities by Underspecification: Construction, Representation and Deduction, In *Journal of Semantics 10*, pp.123-179.

ATLAS – a new text alignment architecture

Bettina Schrader
Institute of cognitive Science
University of Osnabrück
49069 Osnabrück
bschrade@uos.de

Abstract

We are presenting a new, hybrid alignment architecture for aligning bilingual, linguistically annotated parallel corpora. It is able to align simultaneously at paragraph, sentence, phrase and word level, using statistical and heuristic cues, along with linguistics-based rules. The system currently aligns English and German texts, and the linguistic annotation used covers POS-tags, lemmas and syntactic constitutents. However, as the system is highly modular, we can easily adapt it to new language pairs and other types of annotation.

The hybrid nature of the system allows experiments with a variety of alignment cues to find solutions to word alignment problems like the correct alignment of rare words and multiwords, or how to align despite syntactic differences between two languages.

First performance tests are promising, and we are setting up a gold standard for a thorough evaluation of the system.

1 Introduction

Aligning parallel text, i.e. automatically setting the sentences or words in one text into correspondence with their equivalents in a translation, is a very useful preprocessing step for a range of applications, including but not limited to machine translation (Brown et al., 1993), cross-language information retrieval (Hiemstra, 1996), dictionary creation (Smadja et al., 1996) and induction of NLP-tools (Kuhn, 2004). Aligned corpora can be also be used in translation studies (Neumann and Hansen-Schirra, 2005).

The alignment of sentences can be done sufficiently well using cues such as sentence length (Gale and Church, 1993) or cognates (Simard et al., 1992). Word alignment, however, is almost exclusively done using statistics (Brown et al., 1993; Hiemstra, 1996; Vogel et al., 1999; Toutanova et al., 2002).

Hence it is difficult to align so-called rare events, i.e. tokens with a frequency below 10. This is a considerable drawback, as rare events make up more than half of the vocabulary of any corpus. Another problem is the correct alignment of multiword units like idioms. Then, differences in word order are not modelled well by the statistical algorithms.

In order to find solutions to these problems, we have developed a hybrid alignment architecture: it uses statistical information extracted directly from a corpus, and rules or heuristics based on the linguistic information as given by the corpus' annotation. Additionally, it is not necessary to compute sentence alignment prior to aligning at the word level. Instead, the system is capable of interactively and incrementally computing sentence and word alignment, along with alignment at the paragraph and phrase level. The simultaneous alignment at different levels of granularity imposes restrictions on the way text alignment is computed: we are using a constrained best-first strategy for this purpose.

Although we are currently developing and testing the alignment system for the language pair English-German, we have made sure that it can easily be extended to new language pairs. In fact, we are currently adding Swedish and French to the set of supported languages.

First performance tests have been promising, and we are currently setting up a gold standard

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 715–722,
Sydney, July 2006.

of 242 manually aligned sentence pairs in English and German for a thorough evaluation.

In the following, we give an overview on standard approaches to sentence and word alignment, and discuss their advantages and shortcomings. Then, we describe the design of our alignment architecture. In the next two sections, we are describing the data on which we test our system, and our evaluation strategy. Finally, we sum up and describe further work.

2 Related work

Research on text alignment has largely focused on aligning either sentences or words, i.e. most approaches either compute which sentences of a source and a target language form a translation pair, or they use sentence alignment as a preprocessing step to align on the word level.

Additionally, emphasis was laid on the development of *language-independent* algorithms. Ideally, such algorithms would not be tailored to align a specific language pair, but would be applicable to any two languages. Language-independence has also been favoured with respect to linguistic resources in that alignment should do without e.g. using pre-existing dictionaries. Hence there is a dominance of purely statistical approaches.

2.1 Sentence Alignment

Sentence alignment strategies fall roughly into three categories: length-based approaches (Gale and Church, 1991; Gale and Church, 1993) are based on the assumption that the length proportions of a sentence and its translation are roughly the same. Anchor-based algorithms align sentences based on cues like corpus-specific markup and orthographic similarity (Simard et al., 1992). The third approach uses bilingual lexical information, e.g. estimated from the corpus (Kay and Röscheisen, 1993; Fung and Church, 1994; Fung and McKeown, 1994).

Hybrid methods (Tschorn, 2002) combine these standard approaches such that the shortcomings of one approach are counterbalanced by the strength of another component: length-based methods are very sensitive towards deletions in that a single omission can cause the alignment to go on a wrong track from the point where it occurred to the end of the corpus. Strategies that assume that orthographic similarity entails translational equivalence rely on the relatedness of the language pair in question. In closely-related languages like English and French, the amount of orthographically similar words that share the same meaning is higher than in unrelated languages like English and Chinese, were orthographic or even phonetic similarity may only indicate translational equivalence for names. Strategies that use system-external dictionaries, finally, can only be used if a large-enough dictionary exists for a specific language pair.

2.2 Word Alignment

Aligning below the sentence level is usually done using statistical models for machine translation (Brown et al., 1991; Brown et al., 1993; Hiemstra, 1996; Vogel et al., 1999) where any word of the target language is taken to be a possible translation for each source language word. The probability of some target language word to be a translation of a source language word then depends on the frequency with which both co-occur at the same or similar positions in the parallel corpus.

The probabilities are estimated from the using the EM-algorithm[1], and a Viterbi search is carried out to compute the most probable sequence of word translation pairs. Word order differences between the two languages are modelled by using statistical weights, and multiword units are similarly treated.

Another approach to word alignment is presented by Tiedemann (2003), where alignment probabilities are computed using a combination of features like e.g. co-occurrence, cognateness, syntactic category membership. However, although the alignment is partly based on linguistic features, its computation is entirely statistical. Other word alignment strategies (Toutanova et al., 2002; Cherry and Lin, 2003) have also begun to incorporate linguistic knowledge. Unfortunately, the basic, statistical, assumptions have not been changed, and hence no sufficient solution to the shortcomings of the early alignment models have been found.

3 Shortcomings of the statistical alignment approaches

While sentence alignment can be done successfully using a combination of the existing algorithms, word alignment quality suffers due to three problematic phenomena: the amount of *rare*

[1] see (Manning and Schütze, 1999), chapter 14.2.2 for a general introduction

words typically found in corpora, *word order differences* between the to languages to be aligned, and the existence of *multiword units*

3.1 Rare Words

Approximately half of a corpus' vocabulary consists of so-called *hapax legomena*, i.e. types that occur exactly once in a text. Most other words fall into the range of so-called *rare events*, which we define here as types with occurrences between 2 and 10. Both hapax legomena and rare events obviously do not provide sufficient information for statistical analysis.

In the case of word alignment, it is easy to see that they are hard to align: there is virtually no frequency or co-occurrence data with which to compute the alignment. On the other hand, five to ten percent of a corpus' vocabulary consists of highly frequent words, i.e. words with frequencies of 100 or above. These types have the advantage of occurring frequently enough for statistical analysis, however, as they occur at virtually every position in a corpus, they can correspond to anything if alignment decisions are taken on the basis of statistics only.

One solution to this problem would be to use statistics-free rules for alignment, i.e. rules that are insensitive to the rarity or frequency of a word. However, this means that statistical models either have to be abandoned completely, or that effort has to be put in finding a means to combine both alignment approaches into one single, hybrid system.

An alternative would be to design a statistical alignment model that is better suited for the Zipfian frequency distributions in the source and target language texts. Research in this direction would greatly benefit from large amounts of high quality example alignments, e.g. taken from the parallel treebanks that are currently being built (Volk and Samuelsson, 2004; Neumann and Hansen-Schirra, 2005).

3.2 Word Order Differences

Another problem that has been noticed as early as 1993 with the first research on word alignment (Brown et al., 1993) concerns the differences in word order between source and target language.

While simple statistical alignment models like IBM-1 (Brown et al., 1993) and the symmetric alignment approach by Hiemstra (1996) treat sentences as unstructured bags of words, the more sophisticated IBM-models by Brown et al. (1993) approximates word order differences using a statistical *distortion* factor. Vogel et al. (1999), on the other hand, treat word order differences as a local phenomenon that can be modelled within a window of no more than three words. Recently, researchers like Cherry and Lin (2003) have begun to use syntactic analyses to guide and restrict the word alignment process.

The advantage of using available syntactic information for word alignment is that it helps to overcome data sparseness: although a token may be rare, its syntactic category may not, and hence there may be sufficient statistical information to align at the phrase level. Subsequently, the phrase level information can be used to compute alignments for the tokens within the aligned phrases. The syntactic function of a token as modifier, head etc. can equally simplify and guide the alignment process considerably. However, it is unclear whether such an approach performs well for language pairs where syntactic and functional differences are greater than between e.g. English and French.

3.3 Multiword alignment

Like syntactic differences, n:m correspondences, i.e. alignments that involve multiword expressions, have soon been noted as being difficult for statistical word alignment: Brown et al. (1993) modelled *fertility*, as they called it, statistically in the more sophisticated IBM-models. Other approaches adopt again a normalizing procedure: in a preprocessing step, multiwords are either recognized as such and subsequently treated as if they were a single token (Tiedemann, 1999), or, reversely, the tokens they align to may be split into their components, with the components being aligned to the parts of the corresponding multiword expression on a 1:1 basis.

The latter approach is clearly insufficient for word alignment quality: it assumes that compositionality holds for both the multiword unit and its translation, i.e. that the meaning of the whole unit is made up of the meaning of its part. This clearly need not be the case, and further problems arise when a multiword unit and its translation contain different numbers of elements.

The former approach, i.e. of recognizing multiword units as such and treating them as a single token, depends on the kind of recognition procedure adopted, and on the way their alignment is

computed: if it is based on statistics, again, the approach will hardly perform well for rare expressions.

To sum up, aligning at the sentence level can be done with success using a combination of language-independent methods. Word alignment, on the other hand, still leaves room for improvement: current models do not suffice to align rare words and multiword units, and syntactic differences between source and target languages, too, still present a challenge for most word alignment strategies.

4 An alternative text alignment system

In order to address these problems, we have designed an *alternative text alignment system*, called ATLAS, that computes text alignment based on a combination of linguistically informed rules and statistical computation. It takes a linguistically annotated corpus as input[2]. The output of the text alignment system consists of the corpus alignment information and a bilingual dictionary.

During the alignment process, hypotheses on translation pairs are computed by different *alignment modules*, and assigned a *confidence value*. These hypotheses may be about paragraphs, sentences, words, or phrases.

All hypotheses are reused to refine and complete the text alignment, and in a final filtering step, implausible hypotheses are filtered out. The remaining hypotheses constitute the final overall text alignment and are used to generate a bilingual dictionary (see figure 1 for an illustration).

4.1 Core Component

The alignment process is controlled by a core component: it manages all knowledge bases, i.e.

- information contained in a system-internal dictionary,
- corpus information like the positions of tokens and their annotations, and
- the set of alignment hypotheses.

[2]The linguistic annotation currently supported includes lemmas, parts of speech, and syntactic phrases, along with information on sentence or paragraph boundaries. The annotation may include sentence alignment information, and a bilingual dictionary may be used, too.

Additionally, the core component triggers the different *alignment modules* depending on the type of a hypothesis: if, for example, a hypothesis is about a sentence pair, then the word alignment modules of ATLAS are started in order to find translation pairs within the sentence pair.

The alignment modules are run simultaneously, but independently of each other, i.e. an alignment hypothesis may be generated several times, based on cues used by different alignment modules. A word pair e.g. may be aligned based on orthographic similarity by one module, and based on syntactic information by another module.

Each hypothesis is assigned a confidence value by the alignment module that generated it, and then returned to the core component. The confidence value of each hypothesis is derived from i) its probability or similarity value, and ii) the confidence value of the parent hypothesis.

The core component may change the confidence value of a hypothesis, e.g. if it was generated multiple times by different alignment modules, based on different alignment cues. This multiple generation of the same hypothesis is taken as indication that the hypothesis is more reliable than if it had been generated by only one alignment module, and hence its confidence value is increase.

The core component adds all new information to its knowledge bases, and hands it over to appropriate alignment modules for further computation.

The process is iterated until no new hypotheses are found. Then, the core component assembles the best hypotheses to compute a final hypothesis set: starting with the hypothesis that has the highest confidence, each next-best hypothesis is tested whether it fits into the final set; if there is a contradiction between the hypotheses already in the set and the next-best, the latter is discarded from the knowledge base. If not, then it is added to the final set. This process is iterated until all hypotheses have been either added to the final hypothesis set, or have been discarded.

Cleaning-up procedures ensure that corpus items left unaligned are either aligned to null, or can be aligned based on a process of elimination: if two units a and b are contained within the same textual unit, e.g. within the same paragraph, and aligning them would not cause a contradiction with the hypotheses in the final set, then they are aligned. Finally, all remaining hypothesis are used to generate the overall text alignment, and to com-

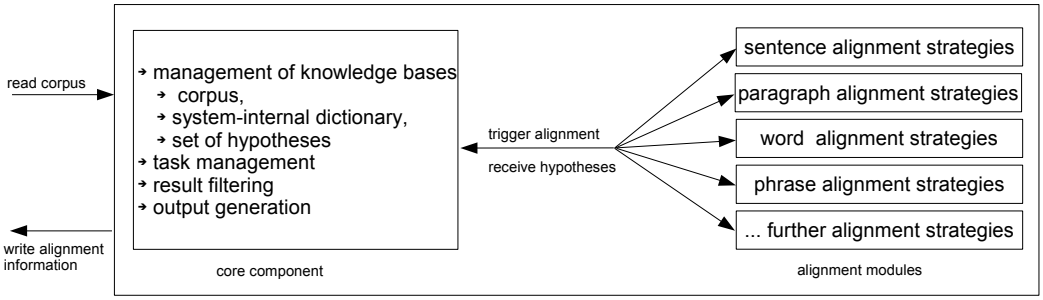


Figure 1: A schema of the text alignment architecture

pute a bilingual dictionary.

4.2 Alignment modules

Each alignment module receives a parent hypothesis as input that covers certain units of the corpus, i.e. a hypothesis on a sentence pair covers those tokens along with their annotations that are contained within the sentence pair. It uses this information to compute child hypotheses within the units of the parent hypothesis, assigns each child hypothesis a confidence value that indicates how reliable it is, and returns the set of children hypotheses to the core component.

In the case of a statistics-based alignment module, the confidence value corresponds to the probability with which a translation pair may be aligned. In other, non-statistical alignment modules, the confidence value is derived from the similarity value computed for a specific translation pair.

The alignment modules that are currently used by our the system are modules for aligning sentences or paragraphs based on the strategies that have been proposed in the literature (see overview in section 2.1), but also strategies that we have experimented with for aligning words based on linear ordering, parts of speech, dictionary lookup etc (see section 5). No statistical word alignment procedure has yet been added to the system, but we are experimenting with using statistical co-occurrence measures for deriving word correspondences. One language independent alignment strategy is based on *inheritance*: if two units a and b are aligned, then this information is used to derive alignment hypotheses for the elements within a and b as well as for the textual units that contain a and b.

5 Advantages of the hybrid architecture

As our alignment architecture is hybrid and hence need not rely on statistial information alone, it can be used to successfully address word alignment problems. Note that although linguistically informed alignment strategies are used, the system is not restricted to statistics-free computation: it is still possible to compute word co-occurrence statistics and derive alignment hypotheses.

5.1 Rare words

Linguistically-informed rules that compute alignments based on corpus annotation, *but not* on statistics, can be used to overcome data sparseness. Syntactic categories e.g. give reliable alignment cues as lexical categories such as nouns and verbs are not commonly changed during the translation process. Even if category changes occur, it is likely that the categorial class stays the same. Ideally, a noun e.g. will be translated as a noun, or if it is not, it is highly probable that it is translated as an adjective or verb, but not as a functional class member like a preposition.

Likewise, dictionary lookup may be used, and is used by or system, to align words within sentences or phrases. We have also implemented a module that aligns sentences and words based on string similarity constrained by syntactic categories: the module exploits the part of speech annotation to align sentences and words based on string similarity between nouns, adjectives, and verbs, thus modifying the classic approach by Simard et al (1992). The advantage of the modification is that the amount of cognates within lexical class words will be considerably higher than between prepositions, determiners, etc., hence filtering by word

category yields good results.

5.2 Word Order Differences

As ATLAS supports the alignment of phrases, mismatches between the linear orderings of source and target language words become irrelevant. Additionally, phrase alignment can considerably narrow down the search space within which to find the translation of a word. If e.g. a noun phrase has already been aligned to its equivalent in the other language, aligning its daughter nodes on the basis of their syntactic categories, without any further constraints or statistical information, can be sufficient.

Furthermore, if parts of the phrase can be aligned using the system-internal dictionary, aligning the remaining words could be done by process of elimination.

5.3 Multiword alignment

Multiwords are traditionally hardest to align, one reason being that they are hard to recognize statistically. With our text alignment system, however, it is possible to write i) language-specific rules that detect multiwords and define ii) a similarity measure that aligns the detected multiwords to their translations. This similarity measure may be language-pair specific, or it may be defined globally, i.e. it will be used for any language pair.

We have already tested such a procedure for aligning English nominal multiwords with their German translations: In this procedure, English nominals are detected based on their typical part-of-speech patterns, and aligned to German nouns if the two expressions are roughly of the same length, counted in characters. The results are encouraging, indicating that nominals can be aligned reliably irrespective of their frequencies in the corpus (Schrader, 2006).

6 Data

As development corpus, we are using *Europarl*, a corpus of European Parliament debates (Koehn, 2005). *Europarl* consists of roughly 30 million tokens per language and is tokenized and sentence-aligned. For the purposes of testing ATLAS, we have POS-tagged and lemmatized the German, English, and French parts of the corpus using the freely available *tree-tagger* (Schmid, 1994). Additionally, we have chunked the German and English texts with an extension of this tool (Schmid, unpublished). Table 1 shows the number of tokens and types of the corpus for all three languages. It also shows the percentages of hapax legomena, rare events[3], and all other types of the corpus.

7 Evaluation

For evaluating of our text alignment system, we are currently setting up an English-German gold standard: we have randomly chosen a debate protocol of the *Europarl* corpus that contains approximately 100,000 tokens per language (see table 2), and we corrected its sentence alignment manually. The correction was done by two annotaters independently of each other, and remaining sentence alignment differences after the corrections were resolved.

In a second step, we have chosen 242 sentence pairs from this reference set to create a word alignment gold standard. Some sentence pairs of this set have been chosen randomly, the others are taken from two text passages in the protocol. We had considered choosing sentence pairs that were distributed randomly over the reference set, however, we decided for taking complete text passages in order to make manual annotation easier. This way, the annotators can easily access the context of a sentence pair to resolve alignment ambiguities.

Additionally, we have created word alignment guidelines based on those already given by Melamed (1998) and Merkel (1999). We have annotated all 242 sentence pairs twice, and annotation differences are currently being resolved.

As this gold standard can only be used to evaluate the performance of English-German word alignment, we will also evaluate our system on the Stockholm parallel treebank (Volk and Samuelsson, 2004). Evaluating against this manually constructed treebank has the advantage that we can evaluate phrase alignment quality, and that we can gather evaluation data for the language pairs English-Swedish and Swedish-German.

We have decided to use the evaluation metrics precision, recall and the *alignment error rate* (AER) proposed by Och and Ney (2000) in order to compare results to those of other alignment systems.

[3]We define rare events here as types occurring 2 to 10 times

Language	Tokens	Types	Hapax Legomena	Rare Events	Frequent Types
English	29.077,024	101,967	39,200 (38.44%)	35,608 (34.92%)	27,159 (26.64%)
German	27.643,792	286,330	140,826 (49.18%)	98,126 (34.27%)	47,378 (16.55%)
French	32.439,353	114,891	42,114 (36.66%)	41,194 (35.84%)	31,583 (27.49%)

Table 1: Corpus characteristics of the Europarl corpus

Language	Tokens	Types	Hapax Legomena	Rare Events	Frequent Types
English	111,222	7,657	3,474 (45.37%)	3,027 (39.53%)	1,156 (15.10%)
German	91,054	11,237	6,336 (56.39%)	3,973 (35.36%)	928 (8.26%)

Table 2: Characteristics of the evaluation suite

8 Summary

Summing up, we have presented a new text alignment architecture that makes use of multiple sources of information, partly statistical, partly linguistics-based, to align bilingual, parallel texts. Its input is a linguistically annotated parallel corpus, and corpus annotation may include information on syntactic constituency, syntactic category membership, lemmas, etc. Alignment is done on various levels of granularity, i.e. the system aligns simultaneously at the paragraph, sentence, phrase, and word level. A constrained best-first search is used to filter out errors, and the output of the system is corpus alignment information along with a bilingual dictionary, generated on the basis of the text alignment.

As our system need not rely on statistics alone, the alignment of hapax legomena and other rare events is not a problem. Additionally, specific strategies have been implemented, and further can be added, to deal with various kinds of multiword units. Finally, as the system allows phrase alignment, it stands on equal footing with other phrase alignment approaches.

Currently, the system is tested on the English-German parts of the *Europarl corpus*, but as it is highly modular, it can easily be extended to new language pairs, types of information, and different alignment strategies.

First performance test have been promising, and we are setting up a gold standard alignment for a thorough evaluation.

9 Further work

We are currently adding Swedish and French to the set of supported languages, such that our system will be able to align all possible pairings with the languages German, English, French and Swedish. If possible, we want to conduct experiments that involve further languages and additional kinds of corpus annotation, like e.g. detailed morphological information as annotated e.g. within the *CroCo* project (Neumann and Hansen-Schirra, 2005).

At the same time, we are constantly extending the set of available alignment strategies, e.g. with strategies for specific syntactic categories or strategies that compute alignments based on statistical co-occurrence.

A first evaluation of our text alignment system will have been completed by autumn 2006, and we plan to make our gold standard as well as our guidelines available to the research community.

Acknowledgement

We thank Judith Degen for annotation help with the gold standard.

References

Peter F. Brown, Jennifer C. Lai, and Robert L. Mercer. 1991. Aligning sentences in parallel corpora. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 169–176, Berkeley, California, USA.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Colin Cherry and Dekang Lin. 2003. A probability model to improve word alignment. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 88–95, Sapporo, Japan.

Pascale Fung and Kenneth W. Church. 1994. K-vec: a new approach for aligning parallel texts. In *Proceedings of the 15th International Conference on*

Computational Linguistics (COLING), pages 1096–1102, Kyoto, Japan.

Pascale Fung and Kathleen McKeown. 1994. Aligning noisy parallel corpora across language groups: word pair feature matching by dynamic time warping. In *Proceedings of the First Conference of the Association for Machine Translation in the Americas (AMTA-94)*, pages 81–88, Columbia, Maryland, USA.

William A. Gale and Kenneth W. Church. 1991. A program for aligning sentences in bilingual corpora. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 177–184, Berkeley, California, USA. Reprinted 1993 in Computational Linguistics.

William A. Gale and Kenneth W. Church. 1993. A program for aligning sentences in bilingual corpora. *Computational Linguistics*, 19(1):75–102.

D. Hiemstra. 1996. Using statistical methods to create a bilingual dictionary. Master's thesis, Universiteit Twente.

Martin Kay and Martin Röscheisen. 1993. Text-translation alignment. *Computational Linguistics*, 19(1):121–142.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT Summit*.

Jonas Kuhn. 2004. Exploiting parallel corpora for monolingual grammar induction – a pilot study. In *Workshop proceedings of the 4th International Conference on Language Resources and Evaluation (LREC)*, pages 54–57, Lisbon, Portugal. LREC Workshop: The Amazing Utility of Parallel and Comparable Corpora.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of statistical natural language processing*. MIT Press, Cambridge, Massachusetts, London.

I. Dan Melamed. 1998. Annotation style guide for the BLINKER project. Technical Report 98-06, Institute for Research in Cognitive Science, University of Pennsylvania.

Magnus Merkel. 1999. Annotation style guide for the PLUG link annotator. Technical report, Linköping university, Linköping, March. PLUG report.

Stella Neumann and Silvia Hansen-Schirra. 2005. The CroCo project. Cross-linguistic corpora for the investigateon of explicitation in translation. In *Proceedings of the Corpus Linguistics Conference*, Birmingham, UK.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 440–447, Hong Kong, China.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, pages 44–49, Manchester, England.

Helmut Schmid. unpublished. The IMS Chunker. unpublished manuscript.

Bettina Schrader. 2006. Non-probabilistic alignment of rare German and English nominal expressions. In *To appear in: Proceedings of the Fifth Language Resources and Evaluation Conference (LREC)*, Genoa, Italy. to appear.

Michel Simard, G. F. Foster, and P. Isabelle. 1992. Using cognates to align sentences in bilingual corpora. In *Proceedings of the Fourth International conference on theoretical and methodological issues in Machine translation*, pages 67–81, Montreal, Canada.

Frank Smadja, Kathleen R. McKeown, and Vasileios Hatzivassiloglou. 1996. Translating collocations for bilingual lexicons: A statistical approach. *Computational Linguistics*, 22(1):1–38.

Jörg Tiedemann. 1999. Word alignment - step by step. In *Proceedings of the 12th Nordic Conference on Computational Linguistics*, pages 216–227, Trondheim, Norway.

Jörg Tiedemann. 2003. Combining clues for word alignment. In *Proceedings of the 10th Conference of the European Chapter of the ACL (EACL03)*, pages 339 – 346, Budapest, Hungary.

Kristina Toutanova, H. Tolga Ilhan, and Christopher D. Manning. 2002. Extensions to HMM-based statistical word alignment models. In *Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 87–94, Philadelphia, USA.

Patrick Tschorn. 2002. Automatically aligning English-German parallel texts at sentence level using linguistic knowledge. Master's thesis, Universität Osnabrück.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1999. HMM-based word alignment in statistical translation. In *Proceedings of the International Conference on Computational Linguistics*, pages 836–841, Copenhagen, Denmark.

Martin Volk and Yvonne Samuelsson. 2004. Bootstrapping parallel treebanks. In *Proceedings of the Workshop on Linguistically Interpreted Corpora (LINC) at COLING*, Geneva, Switzerland.

Continuous Space Language Models for Statistical Machine Translation

Holger Schwenk and **Daniel Dchelotte** and **Jean-Luc Gauvain**
LIMSI-CNRS, BP 133
91403 Orsay cedex, FRANCE
{schwenk,dechelot,gauvain}@limsi.fr

Abstract

Statistical machine translation systems are based on one or more translation models and a language model of the target language. While many different translation models and phrase extraction algorithms have been proposed, a standard word n-gram back-off language model is used in most systems.

In this work, we propose to use a new statistical language model that is based on a continuous representation of the words in the vocabulary. A neural network is used to perform the projection and the probability estimation. We consider the translation of European Parliament Speeches. This task is part of an international evaluation organized by the TC-STAR project in 2006. The proposed method achieves consistent improvements in the BLEU score on the development and test data.

We also present algorithms to improve the estimation of the language model probabilities when splitting long sentences into shorter chunks.

1 Introduction

The goal of statistical machine translation (SMT) is to produce a target sentence $\mathbf{e}$ from a source sentence $\mathbf{f}$. Among all possible target sentences the one with maximal probability is chosen. The classical Bayes relation is used to introduce a target language model (Brown et al., 1993):

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e}} \Pr(\mathbf{e}|\mathbf{f}) = \arg\max_{\mathbf{e}} \Pr(\mathbf{f}|\mathbf{e}) \Pr(\mathbf{e})$$

where $\Pr(\mathbf{f}|\mathbf{e})$ is the translation model and $\Pr(\mathbf{e})$ is the target language model. This approach is usually referred to as the *noisy source-channel* approach in statistical machine translation.

Since the introduction of this basic model, many improvements have been made, but it seems that research is mainly focused on better translation and alignment models or phrase extraction algorithms as demonstrated by numerous publications on these topics. On the other hand, we are aware of only a small amount of papers investigating new approaches to language modeling for statistical machine translation. Traditionally, statistical machine translation systems use a simple 3-gram back-off language model (LM) during decoding to generate n-best lists. These n-best lists are then rescored using a log-linear combination of feature functions (Och and Ney, 2002):

$$\hat{\mathbf{e}} \approx \arg\max_{\mathbf{e}} \Pr(\mathbf{e})^{\lambda_1} \Pr(\mathbf{f}|\mathbf{e})^{\lambda_2} \qquad (1)$$

where the coefficients λ_i are optimized on a development set, usually maximizing the BLEU score. In addition to the standard feature functions, many others have been proposed, in particular several ones that aim at improving the modeling of the target language. In most SMT systems the use of a 4-gram back-off language model usually achieves improvements in the BLEU score in comparison to the 3-gram LM used during decoding. It seems however difficult to improve upon the 4-gram LM. Many different feature functions were explored in (Och et al., 2004). In that work, the incorporation of part-of-speech (POS) information gave only a small improvement compared to a 3-gram back-off LM. In another study, a factored LM using POS information achieved the same results as the 4-gram LM (Kirchhoff and Yang, 2005). Syntax-based LMs were investigated in (Charniak et al.,

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 723–730,
Sydney, July 2006.

2003), and reranking of translation hypothesis using structural properties in (Hasan et al., 2006).

An interesting experiment was reported at the NIST 2005 MT evaluation workshop (Och, 2005): starting with a 5-gram LM trained on 75 million words of Broadcast News data, a gain of about 0.5 point BLEU was observed each time when the amount of LM training data was doubled, using at the end 237 billion words of texts. Most of this additional data was collected by Google on the Internet. We believe that this kind of approach is difficult to apply to other tasks than Broadcast News and other target languages than English. There are many areas where automatic machine translation could be deployed and for which considerably less *appropriate in-domain* training data is available. We could for instance mention automatic translation of medical records, translation systems for tourism related tasks or even any task for which Broadcast news and Web texts is of limited help.

In this work, we consider the translation of European Parliament Speeches from Spanish to English, in the framework of an international evaluation organized by the European TC-STAR project in February 2006. The training data consists of about 35M words of aligned texts that are also used to train the target LM. In our experiments, adding more than 580M words of Broadcast News data had no impact on the BLEU score, despite a notable decrease of the perplexity of the target LM. Therefore, we suggest to use more complex statistical LMs that are expected to take better advantage of the limited amount of appropriate training data. Promising candidates are random forest LMs (Xu and Jelinek, 2004), random cluster LMs (Emami and Jelinek, 2005) and the neural network LM (Bengio et al., 2003). In this paper, we investigate whether the latter approach can be used in a statistical machine translation system.

The basic idea of the neural network LM, also called continuous space LM, is to project the word indices onto a continuous space and to use a probability estimator operating on this space. Since the resulting probability functions are smooth functions of the word representation, better generalization to unknown n-grams can be expected. A neural network can be used to simultaneously learn the projection of the words onto the continuous space and to estimate the n-gram probabilities. This is still a n-gram approach, but the LM posterior probabilities are "interpolated" for any possible context of length n-1 instead of backing-off to shorter contexts. This approach was successfully used in large vocabulary speech recognition (Schwenk and Gauvain, 2005), and we are interested here if similar ideas can be applied to statistical machine translation.

This paper is organized as follows. In the next section we first describe the baseline statistical machine translation system. Section 3 presents the architecture of the continuous space LM and section 4 summarizes the experimental evaluation. The paper concludes with a discussion of future research directions.

2 Statistical Translation Engine

A word-based translation engine is used based on the so-called IBM-4 model (Brown et al., 1993). A brief description of this model is given below along with the decoding algorithm.

The search algorithm aims at finding what target sentence $\mathbf{e}$ is most likely to have produced the observed source sentence $\mathbf{f}$. The translation model $\Pr(\mathbf{f}|\mathbf{e})$ is decomposed into four components:

1. a fertility model;
2. a lexical model of the form $t(f|e)$, which gives the probability that the target word e translates into the source word f;
3. a distortion model, that characterizes how words are reordered when translated;
4. and probabilities to model the insertion of source words that are not aligned to any target words.

An A* search was implemented to find the best translation as predicted by the model, when given enough time and memory, i.e., provided pruning did not eliminate it. The decoder manages partial hypotheses, each of which translates a subset of source words into a sequence of target words. Expanding a partial hypothesis consists of covering one extra source position (in random order) and, by doing so, appending one, several or possibly zero target words to its target word sequence. For details about the implemented algorithm, the reader is referred to (Déchelotte et al., 2006).

Decoding uses a 3-gram back-off target language model. Equivalent hypotheses are merged, and only the best scoring one is further expanded. The decoder generates a lattice representing the

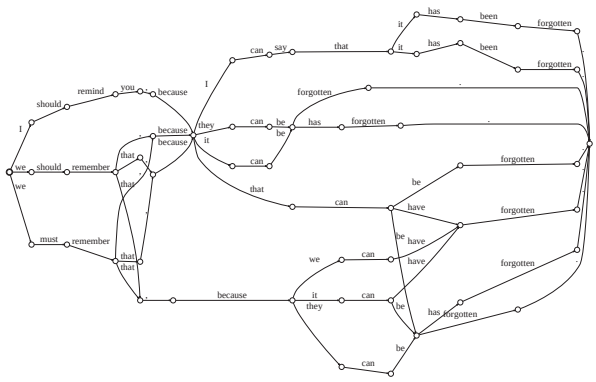


Figure 1: Example of a translation lattice. Source sentence: *"conviene recordarlo , porque puede que se haya olvidado ."*, Reference 1: *"it is appropriate to remember this , because it may have been forgotten ."* Reference 2: *"it is good to remember this , because maybe we forgot it ."*

explored search space. Figure 1 shows an example of such a search space, here heavily pruned for the sake of clarity.

2.1 Sentence Splitting

The execution complexity of our SMT decoder increases non-linearly with the length of the sentence to be translated. Therefore, the source text is split into smaller chunks, each one being translated separately. The chunks are then concatenated together. Several algorithms have been proposed in the literature that try to find the best splits, see for instance (Berger et al., 1996). In this work, we first split long sentences at punctuation marks, the remaining segments that still exceed the allowed length being split linearly. In a second pass, adjoining very short chunks are merged together.

During decoding, target LM probabilities of the type $\Pr(w_1|\texttt{<s>})$ and $\Pr(\texttt{</s>}|w_{n-1}w_n)$ will be requested at the beginning and at the end of the hypothesized target sentence respectively.[1] This is correct when a whole sentence is translated, but leads to wrong LM probabilities when processing smaller chunks. Therefore, we define a sentence break symbol, <b>, that is used at the beginning and at the end of a chunk. During decoding a 3-gram back-off LM is used that was trained on text where sentence break symbols have been added.

Each chunk is translated and a lattice is generated. The individual lattices are then joined, omitting the sentence break symbols. Finally, the resulting lattice is rescored with a LM that was trained on text *without* sentence breaks. In that way we find the best junction of the chunks. Section 4.1 provides comparative results of the different algorithms to split and join sentences.

2.2 Parameter Tuning

It is nowadays common practice to optimize the coefficients of the log-linear combination of feature functions by maximizing the BLEU score on the development data (Och and Ney, 2002). This is usually done by first creating n-best lists that are then reranked using an iterative optimization algorithm.

In this work, a slightly different procedure was used that operates directly on the translation lattices. We believe that this is more efficient than reranking n-best lists since it guarantees that always all possible hypotheses are considered. The decoder first generates large lattices using the current set of parameters. These lattices are then processed by a separate tool that extracts the best path, given the coefficients of six feature functions (translations, distortion, fertility, spontaneous insertion, target language model probability, and a sentence length penalty). Then, the BLEU score of the extracted solution is calculated. This tool is called in a loop by the public numerical optimization tool Condor (Berghen and Bersini, 2005). The solution vector was usually found after about 100 iterations. In our experiments, only two cycles of lattice generation and parameter optimization were necessary (with a very small difference in the BLEU score).

In all our experiments, the 4-gram back-off and the neural network LM are used to calculate language model probabilities that replace those of the default 3-gram LM. An alternative would be to define each LM as a feature function and to combine them under the log-linear model framework, using maximum BLEU training. We believe that this would not make a notable difference in our experiments since we do interpolate the individual LMs, the coefficients being optimized to minimize perplexity on the development data. However, this raises the interesting question whether the two criteria lead to equivalent performance. The result section provides some experimental evidence on this topic.

[1] The symbols <s> and </s> denote the begin and end of sentence marker respectively.

3 Continuous Space Language Models

The architecture of the neural network LM is shown in Figure 2. A standard fully-connected multi-layer perceptron is used. The inputs to the neural network are the indices of the $n-1$ previous words in the vocabulary h_j=w_{j-n+1}, $\ldots, w_{j-2}, w_{j-1}$ and the outputs are the posterior probabilities of *all* words of the vocabulary:

$$P(w_j = i|h_j) \qquad \forall i \in [1, N] \tag{2}$$

where N is the size of the vocabulary. The input uses the so-called 1-of-n coding, i.e., the ith word of the vocabulary is coded by setting the ith element of the vector to 1 and all the other elements to 0. The ith line of the $N \times P$ dimensional projection matrix corresponds to the continuous representation of the ith word. Let us denote c_l these projections, d_j the hidden layer activities, o_i the outputs, p_i their softmax normalization, and m_{jl}, b_j, v_{ij} and k_i the hidden and output layer weights and the corresponding biases. Using these notations, the neural network performs the following operations:

$$d_j = \tanh\left(\sum_l m_{jl}\, c_l + b_j\right) \tag{3}$$

$$o_i = \sum_j v_{ij}\, d_j + k_i \tag{4}$$

$$p_i = e^{o_i} \,/\, \sum_{r=1}^{N} e^{o_r} \tag{5}$$

The value of the output neuron p_i corresponds directly to the probability $P(w_j = i|h_j)$. Training is performed with the standard back-propagation algorithm minimizing the following error function:

$$E = \sum_{i=1}^{N} t_i \log p_i + \beta\left(\sum_{jl} m_{jl}^2 + \sum_{ij} v_{ij}^2\right) \tag{6}$$

where t_i denotes the desired output, i.e., the probability should be 1.0 for the next word in the training sentence and 0.0 for all the other ones. The first part of this equation is the cross-entropy between the output and the target probability distributions, and the second part is a regularization term that aims to prevent the neural network from overfitting the training data (weight decay). The parameter β has to be determined experimentally. Training is done using a resampling algorithm (Schwenk and Gauvain, 2005).

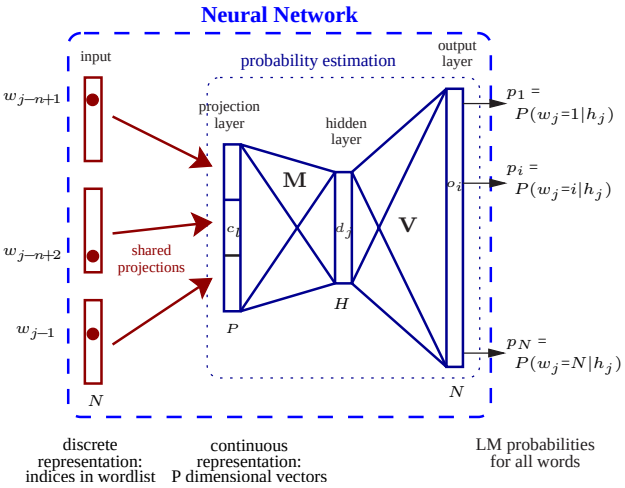


Figure 2: Architecture of the continuous space LM. h_j denotes the context $w_{j-n+1}, \ldots, w_{j-1}$. P is the size of one projection and H,N is the size of the hidden and output layer respectively. When short-lists are used the size of the output layer is much smaller then the size of the vocabulary.

It can be shown that the outputs of a neural network trained in this manner converge to the posterior probabilities. Therefore, the neural network directly minimizes the perplexity on the training data. Note also that the gradient is back-propagated through the projection-layer, which means that the neural network learns the projection of the words onto the continuous space that is best for the probability estimation task.

The complexity to calculate one probability with this basic version of the neural network LM is quite high due to the large output layer. To speed up the processing several improvements were used (Schwenk, 2004):

1. *Lattice rescoring*: the statistical machine translation decoder generates a lattice using a 3-gram back-off LM. The neural network LM is then used to rescore the lattice.

2. *Shortlists*: the neural network is only used to predict the LM probabilities of a subset of the whole vocabulary.

3. *Efficient implementation*: collection of all LM probability requests with the same context h_t in one lattice, propagation of several examples at once through the neural network and utilization of libraries with CPU optimized matrix-operations.

The idea behind short-lists is to use the neural

network only to predict the s most frequent words, s being much smaller than the size of the vocabulary. All words in the vocabulary are still considered at the input of the neural network. The LM probabilities of words in the short-list ($\hat{P}_N$) are calculated by the neural network and the LM probabilities of the remaining words ($\hat{P}_B$) are obtained from a standard 4-gram back-off LM:

$$\hat{P}(w_t|h_t) = \begin{cases} \hat{P}_N(w_t|h_t)P_S(h_t) & \text{if } w_t \in \text{short-list} \\ \hat{P}_B(w_t|h_t) & \text{else} \end{cases} \quad (7)$$

$$P_S(h_t) = \sum_{w \in short-list(h_t)} \hat{P}_B(w|h_t) \quad (8)$$

It can be considered that the neural network redistributes the probability mass of all the words in the short-list. This probability mass is precalculated and stored in the data structures of the back-off LM. A back-off technique is used if the probability mass for a input context is not directly available.

It was not envisaged to use the neural network LM directly during decoding. First, this would probably lead to slow translation times due to the higher complexity of the proposed LM. Second, it is quite difficult to incorporate n-gram language models into decoding, for n>3. Finally, we believe that the lattice framework can give the same performances than direct decoding, under the condition that the alternative hypotheses in the lattices are rich enough. Estimates of the lattice oracle BLEU score are given in the result section.

4 Experimental Evaluation

The experimental results provided here were obtained in the framework of an international evaluation organized by the European TC-STAR project[2] in February 2006. This project is envisaged as a long-term effort to advance research in all core technologies for speech-to-speech translation.

The main goal of this evaluation is to translate public European Parliament Plenary Sessions (EPPS). The training material consists of the minutes edited by the European Parliament in several languages, also known as the Final Text Editions (Gollan et al., 2005). These texts were aligned at the sentence level and they are used to train the statistical translation models (see Table 1 for some statistics). In addition, about 100h of Parliament plenary sessions were recorded and transcribed. This data is mainly used to train the speech recognizers, but the transcriptions were also used for the target LM of the translation system (about 740k words).

	Spanish	*English*
Sentence Pairs	1.2M	
Total # Words	37.7M	33.8M
Vocabulary size	129k	74k

Table 1: Statistics of the parallel texts used to train the statistical machine translation system.

Three different conditions are considered in the TC-STAR evaluation: translation of the Final Text Edition (*text*), translation of the transcriptions of the acoustic development data (*verbatim*) and translation of speech recognizer output (*ASR*). Here we only consider the *verbatim* condition, translating from Spanish to English. For this task, the development data consists of 792 sentences (25k words) and the evaluation data of 1597 sentences (61k words). Parts of the test data origins from the Spanish parliament which results in a (small) mismatch between the development and test data. Two reference translations are provided. The scoring is case sensitive and includes punctuation symbols.

The translation model was trained on 1.2M sentences of parallel text using the Giza++ tool. All back-off LMs were built using modified Kneser-Ney smoothing and the SRI LM-toolkit (Stolcke, 2002). Separate LMs were first trained on the English EPPS texts (33.8M words) and the transcriptions of the acoustic training material (740k words) respectively. These two LMs were then interpolated together. Interpolation usually results in lower perplexities than training directly one LM on the pooled data, in particular if the corpora come from different sources. An EM procedure was used to find the interpolation coefficients that minimize the perplexity on the development data. The optimal coefficients are 0.78 for the Final Text edition and 0.22 for the transcriptions.

4.1 Performance of the sentence splitting algorithm

In this section, we first analyze the performance of the sentence split algorithm. Table 2 compares the results for different ways to translate the individual chunks (using a standard 3-gram LM versus an LM trained on texts with sentence breaks inserted), and to extracted the global solution (con-

[2] http://www.tc-star.org/

LM used during decoding	*Concatenate 1-best*	*Lattice join*
Without sentence breaks	40.20	41.63
With sentence breaks	41.45	42.35

Table 2: BLEU scores for different ways to translate sentence chunks and to extract the global solution (see text for details).

catenating the 1-best solutions versus joining the lattices followed by LM rescoring). It can be clearly seen that joining the lattices and recalculating the LM probabilities gives better results than just concatenating the 1-best solutions of the individual chunks (first line: BLEU score of 41.63 compared to 40.20). Using a LM trained on texts with sentence breaks during decoding gives an additional improvement of about 0.7 points BLEU (42.35 compared to 41.63).

In our current implementation, the selection of the sentence splits is based on punctuation marks in the source text, but our procedure is compatible with other methods. We just need to apply the sentence splitting algorithm on the training data used to build the LM during decoding.

4.2 Using the continuous space language model

The continuous space language model was trained on exactly the same data than the back-off reference language model, using the resampling algorithm described in (Schwenk and Gauvain, 2005). In this work, we use only 4-gram LMs, but the complexity of the neural network LM increases only slightly with the order of the LM. For each experiment, the parameters of the log-linear combination were optimized on the development data.

Perplexity on the development data set is a popular and easy to calculate measure to evaluate the quality of a language model. However, it is not clear if perplexity is a good criterion to predict the improvements when the language model will be used in a SMT system. For information, and comparison with the back-off LM, Figure 3 shows the perplexities for different configurations of the continuous space LM. The perplexity clearly decreases with increasing size of the short-list and a value of 8192 was used. In this case, 99% of the requested LM probabilities are calculated by the neural network when rescoring a lattice.

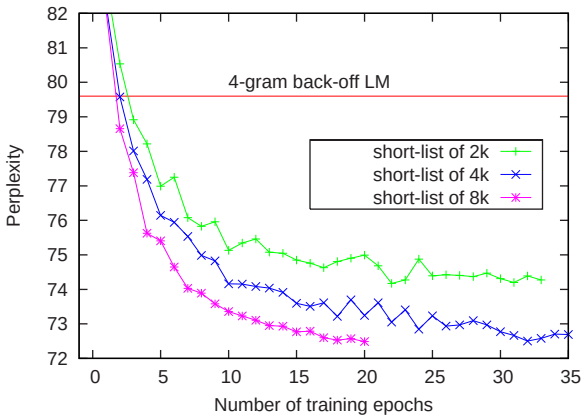


Figure 3: Perplexity of different configurations of the continuous space LM.

Although the neural network LM could be used alone, better results are obtained when interpolating it with the 4-gram back-off LM. It has even turned out that it was advantageous to train several neural network LMs with different context sizes[3] and to interpolate them altogether. In that way, a perplexity decrease from 79.6 to 65.0 was obtained. For the sake of simplicity we will still call this interpolation the neural network LM.

	Back-off LM		Neural LM
	3-gram	4-gram	4-gram
Perplexity	85.5	79.6	65.0
Dev data:			
BLEU	42.35	43.36	**44.42**
WER	45.9%	45.1%	44.4%
PER	31.8%	31.3%	30.8%
Eval data:			
BLEU	39.77	40.62	**41.45**
WER	48.2%	47.4%	46.7%
PER	33.6%	33.1%	32.8%

Table 3: Result comparison for the different LMs. BLEU uses 2 reference translations. WER=word error rate, PER=position independent WER.

Table 3 summarizes the results on the development and evaluation data. The coefficients of the feature functions are always those optimized on the development data. The joined translation lattices were rescored with a 4-gram back-off and the neural network LM. Using a 4-gram back-off LM gives an improvement of 1 point BLEU

[3]The values are in the range 150. . .400. The other parameters are: H=500, β=0.00003 and the initial learning rate was 0.005 with an exponential decay. The networks were trained for 20 epochs through the training data.

Spanish:	*es el nico premio Sajarov que no ha podido recibir su premio despus de ms de tres mil quinientos das de cautiverio .*
Backoff LM:	*it is only the Sakharov Prize has not been able to receive the prize after three thousand , five days of detention .*
CSLM :	*it is the only Sakharov Prize has not been able to receive the prize after three thousand five days of detention .*
Reference 1:	*she is the only Sakharov laureate who has not been able to receive her prize after more than three thousand five hundred days in captivity .*
Reference 2:	*she is the only Sacharov prizewinner who couldn't yet pick up her prize after more than three thousand five hundred days of imprisonment .*

Figure 4: Example translation using the back-off and the continuous space language model (CSLM).

on the Dev data (+0.8 on Test set) compared to the 3-gram back-off LM. The neural network LM achieves an additional improvement of 1 point BLEU (+0.8 on Test data), on top of the 4-gram back-off LM. Small improvements of the word error rate (WER) and the position independent word error rate (PER) were also observed.

As usually observed in SMT, the improvements on the test data are smaller than those on the development data which was used to tune the parameters. As a rule of thumb, the gain on the test data is often half as large as on the Dev-data. The 4-gram back-off and neural network LM show both a good generalization behavior.

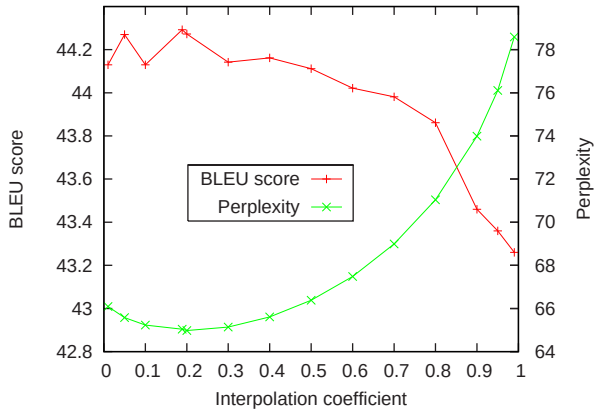


Figure 5: BLEU score and perplexity in function of the interpolation coefficient of the back-off 4-gram LM.

Figure 5 shows the perplexity and the BLEU score for different interpolation coefficients of the 4-gram back-off LM. For a value of 1.0 the back-off LM is used alone, while only the neural network LMs are used for a value of 0.0. Using an EM procedure to minimize perplexity of the interpolated model gives a value of 0.189. This value also seems to correspond to the best BLEU score.

This is a surprising result, and has the advantage that we do not need to tune the interpolation coefficient in the framework of the log-linear feature function combination. The weights of the other feature functions were optimized separately for each experiment. We noticed a tendency to a slightly higher weight for the continuous space LM and a lower sentence length penalty.

In a contrastive experiment, the LM training data was substantially increased by adding 352M words of commercial Broadcast News data and 232M words of CNN news collected on the Internet. Although the perplexity of the 4-gram back-off LM decreased by 5 points to 74.1, we observed no change in the BLEU score. In order to estimate the oracle BLEU score of the lattices we build a 4-gram back-off LM on the development data. Lattice rescoring achieved a BLEU score of 59.10.

There are many discussions about the BLEU score being or not a meaningful measure to assess the quality of an automatic translation system. It would be interesting to verify if the continuous space LM has an impact when human judgments of the translation quality are used, in particular with respect to fluency. Unfortunately, this is not planed in the TC-STAR evaluation campaign, and we give instead an example translation (see Figure 4). In this case, two errors were corrected (insertion of the word "the" and deletion of the comma).

5 Conclusion and Future Work

Some SMT decoders have an execution complexity that increases rapidly with the length of the sentences to be translated, which are usually split

into smaller chunks and translated separately. This can lead to translation errors and bad modeling of the LM probabilities of the words at both ends of the chunks. We have presented a lattice joining and rescoring approach that obtained significant improvements in the BLEU score compared to simply concatenating the 1-best solutions of the individual chunks. The task considered is the translation of European Parliament Speeches in the framework of the TC-STAR project.

We have also presented a neural network LM that performs probability estimation in a continuous space. Since the resulting probability functions are smooth functions of the word representation, better generalization to unknown n-grams can be expected. This is particularly interesting for tasks where only limited amounts of appropriate LM training material are available, but the proposed LM can be also trained on several hundred millions words. The continuous space LM is used to rescore the translation lattices. We obtained an improvement of 0.8 points BLEU on the test data compared to a 4-gram back-off LM, which itself had already achieved the same improvement in comparison to a 3-gram LM.

The results reported in this paper have been obtained with a word based SMT system, but the continuous space LM can also be used with a phrase-based system. One could expect that the target language model plays a different role in a phrase-based system since the phrases induce some local coherency on the target sentence. This will be studied in the future. Another promising direction that we have not yet explored, is to build long-span LM, i.e. with n much greater than 4. The complexity of our approach increases only slightly with n. Long-span LM could possibly improve the word-ordering of the generated sentence if the translation lattices include the correct paths.

References

Yoshua Bengio, Rejean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3(2):1137–1155.

A. Berger, S. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22:39–71.

Frank Vanden Berghen and Hugues Bersini. 2005. CONDOR, a new parallel, constrained extension of powell's UOBYQA algorithm: Experimental results and comparison with the DFO algorithm. *Journal of Computational and Applied Mathematics*, 181:157–175.

P. Brown, S. Della Pietra, Vincent J. Della Pietra, and R. Mercer. 1993. The mathematics of statistical machine translation. *Computational Linguistics*, 19(2):263–311.

E. Charniak, K. Knight, and K. Yamada. 2003. Syntax-based language models for machine translation. In *Machine Translation Summit*.

Daniel Déchelotte, Holger Schwenk, and Jean-Luc Gauvain. 2006. The 2006 LIMSI statistical machine translation system for TC-STAR. In TC-STAR *Speech to Speech Translation Workshop, Barcelona*.

Ahmad Emami and Frederick Jelinek. 2005. Random clusterings for language modeling. In *ICASSP*, pages I:581–584.

C. Gollan, M. Bisani, S. Kanthak, R. Schlueter, and H. Ney. 2005. Cross domain automatic transcription on the TC-STAR EPPS corpus. In *ICASSP*.

Sasa Hasan, Olivier Bender, and Hermann Ney. 2006. Reranking translation hypothesis using structural properties. In *LREC*.

Katrin Kirchhoff and Mei Yang. 2005. Improved language modeling for statistical machine translation. In *ACL'05 workshop on Building and Using Parallel Text*, pages 125–128.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *ACL*, pages 295–302, University of Pennsylvania.

F.-J. Och, D. Gildea, S. Khudanpur, A. Sarkar, K. Yamada, A. Fraser, S. Kumar, L. Shen, D. Smith, K. Eng, V. Jain, Z. Jin, and D. Radev. 2004. A smorgasbord of features for statistical machine translation. In *NAACL*, pages 161–168.

Franz-Joseph Och. 2005. The Google statistical machine translation system for the 2005 Nist MT evaluation, Oral presentation at the 2005 Nist MT Evaluation workshop, June 20.

Holger Schwenk and Jean-Luc Gauvain. 2005. Training neural network language models on very large corpora. In *EMNLP*, pages 201–208.

Holger Schwenk. 2004. Efficient training of large neural networks for language modeling. In *IJCNN*, pages 3059–3062.

Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *ICSLP*, pages II: 901–904.

Peng Xu and Frederick Jelinek. 2004. Random forest in language modeling. In *EMNLP*, pages 325–332.

On-Demand Information Extraction

Satoshi Sekine
Computer Science Department
New York University
715 Broadway, 7th floor
New York, NY 10003 USA
sekine@cs.nyu.edu

Abstract

At present, adapting an Information Extraction system to new topics is an expensive and slow process, requiring some knowledge engineering for each new topic. We propose a new paradigm of Information Extraction which operates 'on demand' in response to a user's query. On-demand Information Extraction (ODIE) aims to completely eliminate the customization effort. Given a user's query, the system will automatically create patterns to extract salient relations in the text of the topic, and build tables from the extracted information using paraphrase discovery technology. It relies on recent advances in pattern discovery, paraphrase discovery, and extended named entity tagging. We report on experimental results in which the system created useful tables for many topics, demonstrating the feasibility of this approach.

1 Introduction

Most of the world's information is recorded, passed down, and transmitted between people in text form. Implicit in most types of text are regularities of information structure - events which are reported many times, about different individuals, in different forms, such as layoffs or mergers and acquisitions in news articles. The goal of information extraction (IE) is to extract such information: to make these regular structures explicit, in forms such as tabular databases. Once the information structures are explicit, they can be processed in many ways: to mine information, to search for specific information, to generate graphical displays and other summaries.

However, at present, a great deal of knowledge for automatic Information Extraction must be coded by hand to move a system to a new topic. For example, at the later MUC evaluations, system developers spent one month for the knowledge engineering to customize the system to the given test topic. Research over the last decade has shown how some of this knowledge can be obtained from annotated corpora, but this still requires a large amount of annotation in preparation for a new task. Improving portability - being able to adapt to a new topic with minimal effort – is necessary to make Information Extraction technology useful for real users and, we believe, lead to a breakthrough for the application of the technology.

We propose 'On-demand information extraction (ODIE)': a system which *automatically identifies the most salient structures and extracts the information on the topic the user demands*. This new IE paradigm becomes feasible due to recent developments in machine learning for NLP, in particular unsupervised learning methods, and it is created on top of a range of basic language analysis tools, including POS taggers, dependency analyzers, and extended Named Entity taggers.

2 Overview

The basic functionality of the system is the following. The user types a query / topic description in keywords (for example, "merge" or "merger"). Then tables will be created automatically in several minutes, rather than in a month of human labor. These tables are expected to show information about the salient relations for the topic.

Figure 1 describes the components and how this system works. There are six major components in the system. We will briefly describe each component and how the data is processed; then, in the next section, four important components will be described in more detail.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 731–738,
Sydney, July 2006.

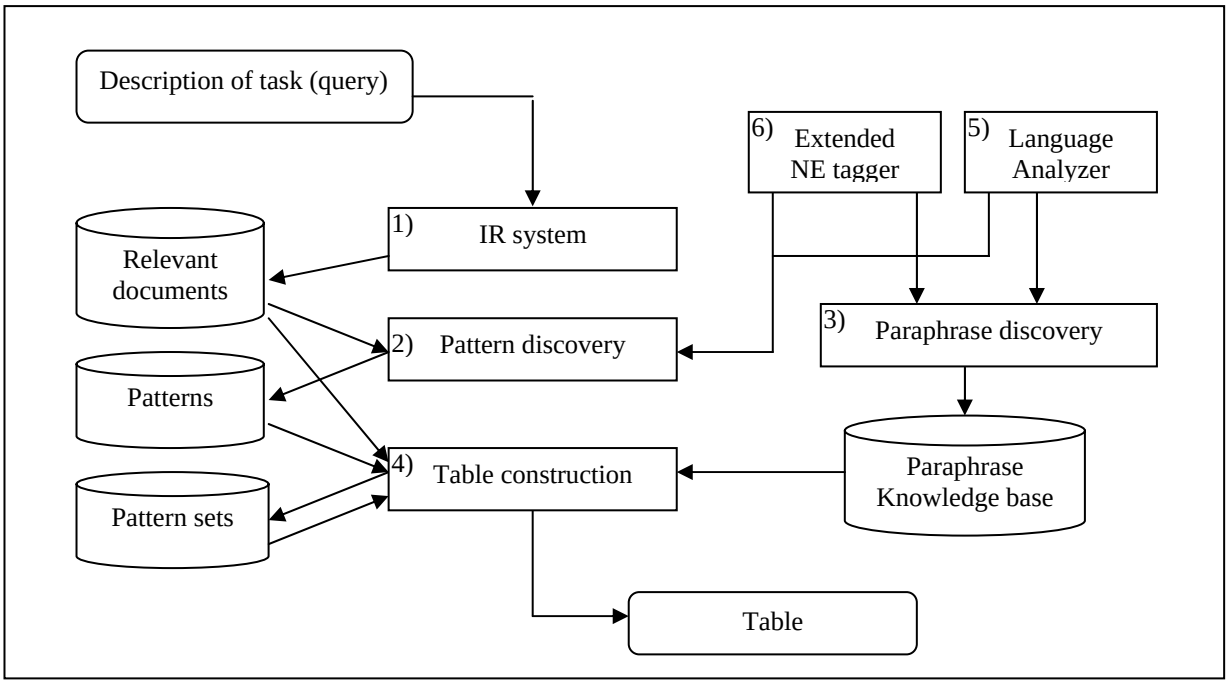


Figure 1. System overview

1) IR system: Based on the query given by the user, it retrieves relevant documents from the document database. We used a simple TF/IDF IR system we developed.

2) Pattern discovery: First, the texts in the retrieved documents are analyzed using a POS tagger, a dependency analyzer and an Extended NE (Named Entity) tagger, which will be described later. Then this component extracts sub-trees of dependency trees which are relatively frequent in the retrieved documents compared to the entire corpus. It counts the frequencies in the retrieved texts of all sub-trees with more than a certain number of nodes and uses TF/IDF methods to score them. The top-ranking sub-trees which contain NEs will be called *patterns*, which are expected to indicate salient relationships of the topic and will be used in the later components.

3) Paraphrase discovery: In order to find semantic relationships between patterns, i.e. to find patterns which should be used to build the same table, we use paraphrase discovery techniques. The paraphrase discovery was conducted off-line and created a paraphrase knowledge base.

4) Table construction: In this component, the patterns created in (2) are linked based on the paraphrase knowledge base created by (3), producing sets of patterns which are semantically equivalent. Once the sets of patterns are created, these patterns are applied to the documents retrieved by the IR system (1). The matched patterns pull out the entity instances and these entities are aligned to build the final tables.

5) Language analyzers: We use a POS tagger and a dependency analyzer to analyze the text. The analyzed texts are used in pattern discovery and paraphrase discovery.

6) Extended NE tagger: Most of the participants in events are likely to be Named Entities. However, the traditional NE categories are not sufficient to cover most participants of various events. For example, the standard MUC's 7 NE categories (i.e. person, location, organization, percent, money, time and date) miss product names (e.g. Windows XP, Boeing 747), event names (Olympics, World War II), numerical expressions other than monetary expressions, etc. We used the Extended NE categories with 140 categories and a tagger based on the categories.

3 Details of Components

In this section, four important components will be described in detail. Prior work related to each component is explained and the techniques used in our system are presented.

3.1 Pattern Discovery

The pattern discovery component is responsible for discovering salient patterns for the topic. The patterns will be extracted from the documents relevant to the topic which are gathered by an IR system.

Several unsupervised pattern discovery techniques have been proposed, e.g. (Riloff 96), (Agichtein and Gravano 00) and (Yangarber et al. 00). Most recently we (Sudo et al. 03) proposed a method which is triggered by a user query to discover important patterns fully automatically. In this work, three different representation models for IE patterns were compared, and the sub-tree model was found more effective compared to the predicate-argument model and the chain model. In the sub-tree model, any connected part of a dependency tree for a sentence can be considered as a pattern. As it counts all possible sub-trees from all sentences in the retrieved documents, the computation is very expensive. This problem was solved by requiring that the sub-trees contain a predicate (verb) and restricting the number of nodes. It was implemented using the sub-tree counting algorithm proposed by (Abe et al. 02). The patterns are scored based on the relative frequency of the pattern in the retrieved documents (f_r) and in the entire corpus (f_{all}). The formula uses the TF/IDF idea (Formula 1). The system ignores very frequent patterns, as those patterns are so common that they are not likely to be important to any particular topic, and also very rare patterns, as most of those patterns are noise.

$$score(t : subtree) = \frac{f_r(t)}{\log(f_{all}(t) + c)} \qquad (1)$$

The scoring function sorts all patterns which contain at least one extended NE and the top 100 patterns are selected for later processing. Figure 2 shows examples of the discovered patterns for the "merger and acquisition" topic. Chunks are shown in brackets and extended NEs are shown in upper case words. (COM means "company" and MNY means "money")

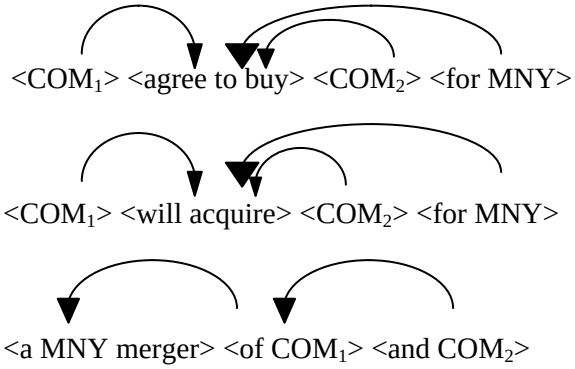


Figure 2. Pattern examples

3.2 Paraphrase Discovery

The role of the paraphrase discovery component is to link the patterns which mean the same thing for the task. Recently there has been a growing amount of research on automatic paraphrase discovery. For example, (Barzilay 01) proposed a method to extract paraphrases from parallel translations derived from one original document. We proposed to find paraphrases from multiple newspapers reporting the same event, using shared Named Entities to align the phrases (Shinyama et al. 02). We also proposed a method to find paraphrases in the context of two Named Entity instances in a large un-annotated corpus (Sekine 05). The phrases connecting two NEs are grouped based on two types of evidence. One is the identity of the NE instance pairs, as multiple instances of the same NE pair (e.g. Yahoo! and Overture) are likely to refer to the same relationship (e.g. acquisition). The other type of evidence is the keywords in the phrase. If we gather a lot of phrases connecting NE's of the same two NE types (e.g. company and company), we can cluster these phrases and find some typical expressions (e.g. merge, acquisition, buy). The phrases are clustered based on these two types of evidence and sets of paraphrases are created.

Basically, we used the paraphrases found by the approach mentioned above. For example, the expressions in Figure 2 are identified as paraphrases by this method; so these three patterns will be placed in the same pattern set.

Note that there is an alternative method of paraphrase discovery, using a hand crafted synonym dictionary like WordNet (WordNet Home page). However, we found that the coverage of WordNet for a particular topic is not sufficient. For example, no synset covers any combinations of the main words in Figure 2, namely "buy", "acquire" and "merger". Furthermore, even if these words are found as synonyms, there is the additional task of linking expressions. For example, if one of the expressions is "reject the merger", it shouldn't be a paraphrase of "acquire".

3.3 Extended NE tagging

Named Entities (NE) were first introduced by the MUC evaluations (Grishman and Sundheim 96). As the MUCs concentrated on business and military topics, the important entity types were limited to a few classes of names and numerical expressions. However, along with the development of Information Extraction and Question Answering technologies, people realized that there should be more and finer categories for NE. We proposed one of those extended NE sets (Sekine 02). It includes 140 hierarchical categories. For example, the categories include Company, Company group, Military, Government, Political party, and International Organization as subcategories of Organization. Also, new categories are introduced such as Vehicle, Food, Award, Religion, Language, Offense, Art and so on as subcategories of Product, as well as Event, Natural Object, Vocation, Unit, Weight, Temperature, Number of people and so on. We used a rule-based tagger developed to tag the 140 categories for this experiment.

Note that, in the proposed method, the slots of the final table will be filled in only with instances of these extended Named Entities. Most common nouns, verbs or sentences can't be entries in the table. This is obviously a limitation of the proposed method; however, as the categories are designed to provide good coverage for a factoid type QA system, most interesting types of entities are covered by the categories.

3.4 Table Construction

Basically the table construction is done by applying the discovered patterns to the original corpus. The discovered patterns are grouped into pattern set using discovered paraphrase knowledge. Once the pattern sets are built, a table is created for each pattern set. We gather all NE instances matched by one of the patterns in the set. These instances are put in the same column of the table for the pattern set. When creating tables, we impose some restrictions in order to reduce the number of meaningless tables and to gather the same relations in one table. We require columns to have at least three filled instances and delete tables with fewer than three rows. These thresholds are empirically determined using training data.

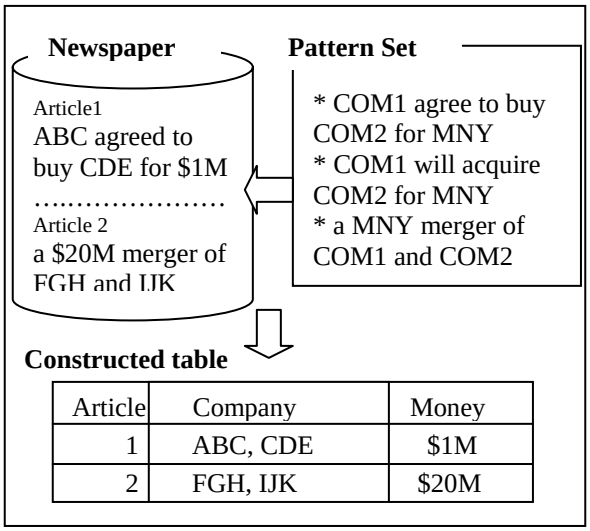


Figure 3. Table Construction

4 Experiments

4.1 Data and Processing

We conducted the experiments using the 1995 New York Times as the corpus. The queries used for system development and threshold tuning were created by the authors, while queries based on the set of event types in the ACE extraction evaluations were used for testing. A total of 31 test queries were used; we discarded several queries which were ambiguous or uncertain. The test queries were derived from the example sentences for each event type in the ACE guidelines. Examples of queries are shown in the Appendix.

At the moment, the whole process takes about 15 minutes on average for each query on a Pentium 2.80GHz processor running Linux. The corpus was analyzed in advance by a POS tagger, NE tagger and dependency analyzer. The processing

and counting of sub-trees takes the majority (more than 90%) of the time. We believe we can easily make it faster by programming techniques, for example, using distributed computing.

4.2 Result and Evaluation

Out of 31 queries, the system is unable to build any tables for 11 queries. The major reason is that the IR component can't find enough newspaper articles on the topic. It retrieved only a few articles for topics like "born", "divorce" or "injure" from The New York Times. For the moment, we will focus on the 20 queries for which tables were built. The Appendix shows some examples of queries and the generated tables. In total, 127 tables are created for the 20 topics, with one to thirteen tables for each topic. The number of columns in a table ranges from 2 to 10, including the document ID column, and the average number of columns is 3.0. The number of rows in a table range from 3 to 125, and the average number of rows is 16.9. The created tables are usually not fully filled; the average rate is 20.0%.

In order to measure the potential and the usefulness of the proposed method, we evaluate the result based on three measures: usefulness, argument role coverage, and correctness. For the usefulness evaluation, we manually reviewed the tables to determine whether a useful table is included or not. This is inevitably subjective, as the user does not specify in advance what table rows and columns are expected. We asked a subject to judge usefulness in three grades; A) very useful – for the query, many people might want to use this table for the further investigation of the topic, B) useful – at least, for some purpose, some people might want to use this table for further investigation and C) not useful – no one will be interested in using this table for further investigation. The argument role coverage measures the percentage of the roles specified for each ACE event type which appeared as a column in one or more of the created tables for that event type. The correctness was measured based on whether a row of a table reflects the correct information. As it is impossible to evaluate all the data, the evaluation data are selected randomly.

Table 1 shows the usefulness evaluation result. Out of 20 topics, two topics are judged very useful and twelve are judged useful. The very useful topics are "fine" (Q4 in the appendix) and "acquit" (not shown in the appendix). Compared to the results in the 'useful' category, the tables for these two topics have more slots filled and the NE types of the fillers have fewer mistakes. The topics in the "not useful" category are "appeal", "execute", "fired", "pardon", "release" and "trial". These are again topics with very few relevant articles. By increasing the corpus size or improving the IR component, we may be able to improve the performance for these topics. The majority category, "useful", has 12 topics. Five of them can be found in the appendix (all those besides Q4). For these topics, the number of relevant articles in the corpus is relatively high and interesting relations are found. The examples in the appendix are selected from larger tables with many columns. Although there are columns that cannot be filled for every event instance, we found that the more columns that are filled in, the more useful and interesting the information is.

Table 1. Usefulness evaluation result

Evaluation	Number of topics
Very useful	2
Useful	12
Not useful	6

For the 14 "very useful" and "useful" topics, the role coverage was measured. Some of the roles in the ACE task can be filled by different types of Named Entities, for example, the "defendant" of a "sentence" event can be a Person, Organization or GPE. However, the system creates tables based on NE types; e.g. for the "sentence" event, a Person column is created, in which most of the fillers are defendants. In such cases, we regard the column as covering the role. Out of 63 roles for the 14 event types, 38 are found in the created tables, for a role coverage of 60.3%. Note that, by lowering the thresholds, the coverage can be increased to as much as 90% (some roles can't be found because of Extended NE limitations or the rare appearance of roles) but with some sacrifice of precision.

Table 2 shows the correctness evaluation results. We randomly select 100 table rows among the topics which were judged "very useful" or "useful", and determine the correctness of the information by reading the newspaper articles the information was extracted from. Out of 100 rows, 84 rows have correct information in all slots. 4

rows have some incorrect information in some of the columns, and 12 contain wrong information. Most errors are due to NE tagging errors (11 NE errors out of 16 errors). These errors include instances of people which are tagged as other categories, and so on. Also, by looking at the actual articles, we found that co-reference resolution could help to fill in more information. Because the important information is repeatedly mentioned in newspaper articles, referential expressions are often used. For example, in a sentence "In 1968 he was elected mayor of Indianapolis.", we could not extract "he" at the moment. We plan to add coreference resolution in the near future. Other sources of error include:

- The role of the entity is confused, i.e. victim and murderer
- Different kinds of events are found in one table, e.g., the victory of Jack Nicklaus was found in the political election query (as both of them use terms like "win")
- An unrelated but often collocate entity was included. For example, Year period expressions are found in "fine" events, as there are many expressions like "He was sentenced 3 years and fined $1,000".

Table 2. Correctness evaluation result

Evaluation	Number of rows
Correct	84
Partially correct	4
Incorrect	12

5 Related Work

As far as the authors know, there is no system similar to ODIE. Several methods have been proposed to produce IE patterns automatically to facilitate IE knowledge creation, as is described in Section 3.1. But those are not targeting the fully automatic creation of a complete IE system for a new topic.

There exists another strategy to extend the range of IE systems. It involves trying to cover a wide variety of topics with a large inventory of relations and events. It is not certain if there are only a limited number of topics in the world, but there are a limited number of high-interest topics, so this may be a reasonable solution from an engineering point of view. This line of research was first proposed by (Aone and Ramos-Santacruz 00) and the ACE evaluations of event detection follow this line (ACE Home Page).

An unsupervised learning method has been applied to a more restricted IE task, Relation Discovery. (Hasegawa et al. 2004) used large corpora and an Extended Named Entity tagger to find novel relations and their participants. However, the results are limited to a pair of participants and because of the nature of the procedure, the discovered relations are static relations like a country and its presidents rather than events.

Topic-oriented summarization, currently pursued by the DUC evaluations (DUC Home Page), is also closely related. The systems are trying to create summaries based on the specified topic for a manually prepared set of documents. In this case, if the result is suitable to present in table format, it can be handled by ODIE. Our previous study (Sekine and Nobata 03) found that about one third of randomly constructed similar newspaper article clusters are well-suited to be presented in table format, and another one third of the clusters can be acceptably expressed in table format. This suggests there is a big potential where an ODIE-type system can be beneficial.

6 Future Work

We demonstrated a new paradigm of Information Extraction technology and showed the potential of this method. However, there are problems to be solved to advance the technology. One of them is the coverage of the extracted information. Although we have created useful tables for some topics, there are event instances which are not found. This problem is mostly due to the inadequate performance of the language analyzers (information retrieval component, dependency analyzer or Extended NE tagger) and the lack of a coreference analyzer. Even though there are possible applications with limited coverage, it will be essential to enhance these components and add coreference in order to increase coverage. Also, there are basic domain limitations. We made the system "on-demand" for any topic, but currently only within regular news domains. As configured, the system would not work on other domains such as a medical, legal, or patent domain, mainly due to the design of the extended NE hierarchy. While specific hierarchies could be incorporated

for new domains, it will also be desirable to integrate bootstrapping techniques for rapid incremental additions to the hierarchy. Also at the moment, table column labels are simply Extended

NE categories, and do not indicate the role. We would like to investigate this problem in the future.

7 Conclusion

In this paper, we proposed "On-demand Information Extraction (ODIE)". It is a system which automatically identifies the most salient structures and extracts the information on whatever topic the user demands. It relies on recent advances in NLP technologies; unsupervised learning and several advanced NLP analyzers. Although it is at a preliminary stage, we developed a prototype system which has created useful tables for many topics and demonstrates the feasibility of this approach.

8 Acknowledgements

This research was supported in part by the Defense Advanced Research Projects Agency under Contract HR0011-06-C-0023 and by the National Science Foundation under Grant IIS-0325657. This paper does not necessarily reflect the position of the U.S. Government.

We would like to thank Prof. Ralph Grishman, Dr. Kiyoshi Sudo, Dr. Chikashi Nobata, Mr. Takaaki Hasegawa, Mr. Koji Murakami and Mr. Yusuke Shinyama for useful comments, discussion.

References

ACE Home Page: http://www.ldc.upenn.edu/Projects/ace

DUC Home Page: http://duc.nist.gov

WordNet Home Page: http://wordnet.princeton.edu/

Kenji Abe, Shinji Kawasone, Tatsuya Asai, Hiroki Arimura and Setsuo Arikawa. 2002. "Optimized Substructure Discovery for Semi-structured Data". In Proceedings of the 6th European Conference on Principles and Practice of Knowledge in Database (PKDD-02)

Chinatsu Aone; Mila Ramos-Santacruz. 2000. "REES: A Large-Scale Relation and Event Extraction System" In Proceedings of the 6th Applied Natural Language Processing Conference (ANLP-00)

Eugene Agichtein and L. Gravano. 2000. "Snowball: Extracting Relations from Large Plaintext Collectionss". In Proceedings of the 5th ACM International Conference on Digital Libraries (DL-00)

Regina Barzilay and Kathleen McKeown. 2001. "Extracting Paraphrases from a Parallel Corpus. In Proceedings of the Annual Meeting of Association of Computational Linguistics/ and European Chapter of Association of Computational Linguistics (ACL/EACL-01)

Ralph Grishman and Beth Sundheim.1996. "Message Understanding Conference - 6: A Brief History", in Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)

Takaaki Hasegawa, Satoshi Sekine and Ralph Grishman 2004. "Discovering Relations among Named Entities from Large Corpora", In Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL-04)

Ellen Riloff. 1996. "Automatically Generating Extraction Patterns from Untagged Text". In Proceedings of Thirteen National Conference on Artificial Intelligence (AAAI-96)

Satoshi Sekine, Kiyoshi Sudo and Chikashi Nobata. 2002 "Extended Named Entity Hierarchy" In Proceefings of the third International Conference on Language Resources and Evaluation (LREC-02)

Satoshi Sekine and Chikashi Nobata. 2003. "A survey for Multi-Document Summarization" In the proceedings of Text Summarization Workshop.

Satoshi Sekine. 2005. "Automatic Paraphrase Discovery based on Context and Keywords between NE Pairs". In Proceedings of International Workshop on Paraphrase (IWP-05)

Yusuke Shinyama, Satoshi Sekine and Kiyoshi Sudo. 2002. "Automatic Paraphrase Acquisition from News Articles". In Proceedings of the Human Language Technology Conference (HLT-02)

Kiyoshi Sudo, Satsohi Sekine and Ralph Grishman. 2003. "An Improved Extraction Pattern Representation Model for Automatic IE Pattern Acquisition". In Proceedings of the Annual Meeting of Association of Computational Linguistics (ACL-03)

Roman Yangarber, Ralph Grishman, Pasi Tapanainen and Silja Huttunen. 2000. "Unsupervised Discovery of Scenario-Level Patterns for Information Extraction". In Proceedings of 18th International Conference on Computational Linguistics (COLING-00)

Appendix: Sample queries and tables
(Note that this is only a part of created tables)

Q1: acquire, acquisition, merge, merger, buy purchase

docid	MONEY	COMPANY	DATE
nyt950714.0324	About $3 billion	PNC Bank Corp., Midlantic Corp.	
nyt950831.0485	$900 million	Ceridian Corp., Comdata Holdings Corp.	Last week
nyt950909.0449	About $1.6 billion	Bank South Corp	
nyt951010.0389	$3.1 billion	CoreStates Financial Corp.	
nyt951113.0483	$286 million	Potash Corp.	Last month
nyt951113.0483	$400 million	Chemicals Inc.	Last year

Q2: convict, guilty

docid	PERSON	DATE	AGE
nyt950207.0001	Fleiss	Dec. 2	28
nyt950327.0402	Gerald_Amirault	1986	41
nyt950720.0145	Hedayat_Eslaminia	1988	
nyt950731.0138	James McNally, James Johnson Bey, Jose Prieto, Patterson	1993, 1991, this year, 1984	
nyt951229.0525	Kane	Last year	

Q3: elect

Docid	POSITION TITLE	PERSON	DATE
nyt950404.0197	president	Havel	Dec. 29, 1989
nyt950916.0222	president	Ronald Reagan	1980
nyt951120.0355	president	Aleksander Kwasniewski	

Q4: fine

Docid	PERSON	MONEY	DATE
nyt950420.0056	Van Halen	$1,000	
nyt950525.0024	Derek Meredith	$300	
nyt950704.0016	Tarango	At least $15,500	
nyt951025.0501	Hamilton	$12,000	This week
nyt951209.0115	Wheatley	Approximately $2,000	

Q5: arrest jail incarcerate imprison

Docid	PERSON	YEAR PERIOD
nyt950817.0544	Nguyen Tan Tri	Four years
nyt951018.0762	Wolf	Six years
nyt951218.0091	Carlos Mendoza-Lugo	One year

Q6: sentence

Docid	PERSON	YEAR PERIOD
nyt950412.0448	Mitchell Antar	Four years
nyt950421.0509	MacDonald	14 years
nyt950622.0512	Aramony	Three years
nyt950814.0106	Obasanjo	25 years

Using comparable corpora to solve problems difficult for human translators

Serge Sharoff, Bogdan Babych, Anthony Hartley
Centre for Translation Studies
University of Leeds, LS2 9JT UK
{s.sharoff,b.babych,a.hartley}@leeds.ac.uk

Abstract

In this paper we present a tool that uses comparable corpora to find appropriate translation equivalents for expressions that are considered by translators as difficult. For a phrase in the source language the tool identifies a range of possible expressions used in similar contexts in target language corpora and presents them to the translator as a list of suggestions. In the paper we discuss the method and present results of human evaluation of the performance of the tool, which highlight its usefulness when dictionary solutions are lacking.

1 Introduction

There is no doubt that both professional and trainee translators need access to authentic data provided by corpora. With respect to polysemous lexical items, bilingual dictionaries list several translation equivalents for a headword, but words taken in their contexts can be translated in many more ways than indicated in dictionaries. For instance, the Oxford Russian Dictionary (ORD) lacks a translation for the Russian expression исчерпывающий ответ ('comprehensive answer'), while the Multitran Russian-English dictionary suggests that it can be translated as *irrefragable answer*. Yet this expression is extremely rare in English; on the Internet it occurs mostly in pages produced by Russian speakers.

On the other hand, translations for polysemous words are too numerous to be listed for all possible contexts. For example, the entry for *strong* in ORD already has 57 subentries and yet it fails to mention many word combinations frequent in the British National Corpus (BNC), such as *strong {feeling, field, opposition, sense, voice}*. *Strong voice* is also not listed in the Oxford French, German or Spanish Dictionaries.

There has been surprisingly little research on computational methods for finding translation equivalents of words from the general lexicon. Practically all previous studies have concerned detection of terminological equivalence. For instance, project Termight at AT&T aimed to develop a tool for semi-automatic acquisition of termbanks in the computer science domain (Dagan and Church, 1997). There was also a study concerning the use of multilingual webpages to develop bilingual lexicons and termbanks (Grefenstette, 2002). However, neither of them concerned translations of words from the general lexicon. At the same time, translators often experience more difficulty in dealing with such general expressions because of their polysemy, which is reflected differently in the target language, thus causing the dependency of their translation on the corresponding context. Such variation is often not captured by dictionaries.

Because of their importance, words from the general lexicon are studied by translation researchers, and comparable corpora are increasingly used in translation practice and training (Varantola, 2003). However, such studies are mostly confined to lexicographic exercises, which compare the contexts and functions of potential translation equivalents once they are known, for instance, *absolutely* vs. *assolutamente* in Italian (Partington, 1998). Such studies do not provide a computational model for *finding* appropriate translation equivalents for expressions that are not listed or are inadequate in dictionaries.

Parallel corpora, conisting of original texts and

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 739–746,
Sydney, July 2006.

their exact translations, provide a useful supplement to decontextualised translation equivalents listed in dictionaries. However, parallel corpora are not representative. Many of them are in the range of a few million words, which is simply too small to account for variations in translation of moderately frequent words. Those that are a bit larger, such as the Europarl corpus, are restricted in their domain. For instance, all of the 14 instances of *strong voice* in the English section of Europarl are used in the sense of 'the opinion of a political institution'. At the same time the BNC contains 46 instances of *strong voice* covering several different meanings.

In this paper we propose a computational method for using comparable corpora to find translation equivalents for source language expressions that are considered as difficult by trainee or professional translators. The model is based on detecting frequent multi-word expressions (MWEs) in the source and target languages and finding a mapping between them in comparable monolingual corpora, which are designed in a similar way in the two languages.

The described methodology is implemented in ASSIST, a tool that helps translators to find solutions for difficult translation problems. The tool presents the results as lists of translation suggestions (usually 50 to 100 items) ordered alphabetically or by their frequency in target language corpora. Translators can skim through these lists and identify an example which is most appropriate in a given context.

In the following sections we outline our approach, evaluate the output of the prototype of ASSIST and discuss future work.

2 Finding translations in comparable corpora

The proposed model finds potential translation equivalents in four steps, which include

1. expansion of words in the original expression using related words;
2. translation of the resultant set using existing bilingual dictionaries;
3. further expansion of the set using related words in the target language;
4. filtering of the set according to expressions frequent in the target language corpus.

In this study we use several comparable corpora for English and Russian, including large reference corpora (the BNC and the Russian Reference Corpus) and corpora of major British and Russian newspapers. All corpora used in the study are quite large, i.e. the size of each corpus is in the range of 100-200 million words (MW), so that they provide enough evidence to detect such collocations as *strong voice* and *clear defiance*.

Although the current study is restricted to the English-Russian pair, the methodology does not rely on any particular language. It can be extended to other languages for which large comparable corpora, POS-tagging and lemmatisation tools, and bilingual dictionaries are available. For example, we conducted a small study for translation between English and German using the Oxford German Dictionary and a 200 MW German corpus derived from the Internet (Sharoff, 2006).

2.1 Query expansion

The problem with using comparable corpora to find translation equivalents is that there is no obvious bridge between the two languages. Unlike aligned parallel corpora, comparable corpora provide a model for each individual language, while dictionaries, which can serve as a bridge, are inadequate for the task in question, because the problem we want to address involves precisely translation equivalents that are not listed there.

Therefore, a specific query needs first to be generalised in order to then retrieve a suitable candidate from a set of candidates. One way to generalise the query is by using *similarity classes*, i.e. groups of words with lexically similar behaviour. In his work on distributional similarity (Lin, 1998) designed a parser to identify grammatical relationships between words. However, broad-coverage parsers suitable for processing BNC-like corpora are not available for many languages. Another, resource-light approach treats the context as a bag of words (BoW) and detects the similarity of contexts on the basis of collocations in a window of a certain size, typically 3-4 words, e.g. (Rapp, 2004). Even if using a parser can increase precision in identification of contexts in the case of long-distance dependencies (e.g. *to cook Alice a whole meal*), we can find a reasonable set of relevant terms returned using the BoW approach, cf. the results of human evaluation for English and German by (Rapp, 2004).

For each source word s_0 we produce a list of similar words: $\Theta(s_0) = s_1, \ldots, s_N$ (in our tool we use $N = 20$ as the cutoff). Since lists of distributionally words can contain words irrelevant to the source word, we filter them to produce a more reliable similarity class $S(s_0)$ using the assumption that the similarity classes of similar words have common members:

$\forall w \in S(s_0), w \in \Theta(s_0) \& w \in \bigcup \Theta(s_i)$

This yields for *experience* the following similarity class: *knowledge, opportunity, life, encounter, skill, feeling, reality, sensation, dream, vision, learning, perception, learn.*[1] Even if there is no requirement in the BoW approach that words in the similarity class are of the same part of speech, it happens quite frequently that most words have the same part of speech because of the similarity of contexts.

2.2 Query translation and further expansion

In the next step we produce a translation class by translating all words from the similarity class into the target language using a bilingual dictionary ($T(w)$ for the translation of w). Then for Step 3 we have two options: a full translation class (TF) and a reduced one (TR).

TF consists of similarity classes produced for all translations: $S(T(S(s_0)))$. However, this causes a combinatorial explosion. If a similarity class contains N words (the average figure is 16) and a dictionary lists on average M equivalents for a source word (the average figure is 11), this procedure outputs on average $M \times N^2$ words in the full translation class. For instance, the complete translation class for *experience* contains 998 words. What is worse, some words from the full translation class do not refer to the domain implied in the original expression because of the ambiguity of the translation operation. For instance, the word *dream* belongs to the similarity class of *experience*. Since it can be translated into Russian as сказка ('fairy-tale'), the latter Russian word will be expanded in the full translation class with words referring to legends and stories. In the later stages of the project, word sense disambiguation in corpora could improve precision of translation classes. However at the present stage we attempt to trade the recall of the tool for greater precision by translating words in the source similarity class, and generating the similarity classes of translations only for the source word:

$TR(s_0) = S(T(s_0)) \cup T(S(s_0))$.

This reduces the class of *experience* to 128 words.

This step crucially relies on a wide-coverage machine readable dictionary. The bilingual dictionary resources we use are derived from the source file for the Oxford Russian Dictionary, provided by OUP.

2.3 Filtering equivalence classes

In the final step we check all possible combinations of words from the translation classes for their frequency in target language corpora.

The number of elements in the set of theoretically possible combinations is usually very large: $\prod T_i$, where T_i is the number of words in the translation class of each word of the original MWE. This number is much larger than the set of word combinations which is found in the target language corpora. For instance, *daunting experience* has 202,594 combinations for the full translation class of *daunting experience* and 6,144 for the reduced one. However, in the target language corpora we can find only 2,256 collocations with frequency > 2 for the full translation class and 92 for the reduced one.

Each theoretically possible combination is generated and looked up in a database of MWEs (which is much faster than querying corpora for frequencies of potential collocations). The MWE database was pre-compiled from corpora using a method of filtering, similar to part-of-speech filtering suggested in (Justeson and Katz, 1995): in corpora each N-gram of length 2, 3 and 4 tokens was checked against a set of filters.

However, instead of pre-defined patterns for entire expressions our filtering method uses sets of *negative* constraints, which are usually applied to the edges of expressions. This change boosts recall of retrieved MWEs and allows us to use the same set of patterns for MWEs of different length. The filter uses constraints for both lexical and part-of-speech features, which makes configuration specifications more flexible.

The idea of applying a negative feature filter rather than a set of positive patterns is based on the observation that it is easier to describe undesirable features than to enumerate complete lists of patterns. For example, MWEs of any length ending with a preposition are undesirable (particles in

[1] Ordered according to the score produced by the Singular Value Decomposition method as implemented by Rapp.

	British news	Russian news
no of words	217,394,039	77,625,002
REs in filter	25	18
2-grams	6,361,596	5,457,848
3-grams	14,306,653	11,092,908
4-grams	19,668,956	11,514,626

Table 1: MWEs in News Corpora

phrasal verbs, which are desirable, are tagged differently by the Tree Tagger, so there is no problem with ambiguity here). Our filter captures this fact by having a negative condition for the right edge of the pattern (regular expression /_IN$/), rather than enumerating all possible configurations which do not contain a preposition at the end. In this sense the filter is permissive: everything that is not explicitly forbidden is allowed, which makes the description more economical.

The same MWE database is used for checking frequencies of multiword collocates for corpus queries. For this task, candidate N-grams in the vicinity of searched patterns are filtered using the same regular expression grammar of MWE constraints, and then their corpus frequency is checked in the database. Thus scores for multiword collocates can be computed from contingency tables similarly to single-word collocates.

In addition, only MWEs with a frequency higher than 1 are stored in the database. This filters out most expressions that co-occur by chance. Table 1 gives an overview of the number of MWEs from the news corpus which pass the filter. Other corpora used in ASSIST (BNC and RRC) yield similar results. MWE frequencies for each corpus can be checked individually or joined together.

3 Evaluation

There are several attributes of our system which can be evaluated, and many of them are crucial for its efficient use in the workflow of professional translators, including: usability, quality of final solutions, trade-off between adequacy and fluency across usable examples, precision and recall of potentially relevant suggestions, as well as real-text evaluation, i.e. “What is the coverage of difficult translation problems typically found in a text that can be successfully tackled?”

In this paper we focus on evaluating the quality of potentially relevant translation solutions, which is the central point for developing and calibrating our methodology. The evaluation experiment discussed below was specifically designed to assess the usefulness of translation suggestions generated by our tool – in cases where translators have doubts about the usefulness of dictionary solutions. In this paper we do not evaluate other equally important aspects of the system’s functionality, which will be the matter of future research.

3.1 Set-up of the experiment

For each translation direction we collected ten examples of possibly recalcitrant translation problems – words or phrases whose translation is not straightforward in a given context. Some of these examples were sent to us by translators in response to our request for difficult cases. For each example, which we included in the evaluation kit, the word or phrase either does not have a translation in ORD (which is a kind of a baseline standard reference for Russian translators), or its translation has significantly lower frequency in a target language corpus in comparison to the frequency of the source expression. If an MWE is not listed in available dictionaries, we produced compositional (word-for-word) translations using ORD. In order to remove a possible anti-dictionary bias from our experiment, we also checked translations in Multitran, an on-line translation dictionary, which was often quoted as one of the best resources for translation from and into Russian.

For each translation problem five solutions were presented to translators for evaluation. One or two of these solutions were taken from a dictionary (usually from Multitran, and if available and different, from ORD). The other suggestions were manually selected from lists of possible solutions returned by ASSIST. Again, the criteria for selection were intuitive: we included those suggestions which made best sense in the given context. Dictionary suggestions and the output of ASSIST were indistinguishable in the questionnaires to the evaluators. The segments were presented in sentence context and translators had an option of providing their own solutions and comments. Table 2 shows one of the questions sent to evaluators. The problem example is четкая программа (‘precise programme’), which is presented in the context of a Russian sentence with the following (non-literal) translation *This team should be put together by responsible politicians, who have a*

Problem example	
четкая программа, as in Собрать эту команду должны ответственные люди, имеющие четкую программу выхода из кризиса.	
Translation suggestions	**Score**
clear plan clear policy clear programme clear strategy concrete plan	
Your suggestion ? (optional)	

Table 2: Example of an entry in questionnaire

clear strategy for resolving the current crisis. The third translation equivalent (*clear programme*) in the table is found in the Multitran dictionary (ORD offers no translation for четкая программа). The example was included because *clear programme* is much less frequent in English (2 examples in the BNC) in comparison to четкая программа in Russian (70). Other translation equivalents in Table 2 are generated by ASSIST.

We then asked professional translators affiliated to a translator's association (identity witheld at this stage) to rate these five potential equivalents using a five-point scale:

5 = The suggestion is an appropriate translation as it is.

4 = The suggestion can be used with some minor amendment (e.g. by turning a verb into a participle).

3 = The suggestion is useful as a hint for another, appropriate translation (e.g. suggestion *elated* cannot be used, but its close synonym *exhilarated* can).

2 = The suggestion is not useful, even though it is still in the same domain (e.g. *fear* is proposed for a problem referring to *hatred*).

1 = The suggestion is totally irrelevant.

We received responses from eight translators. Some translators did not score all solutions, but there were at least four independent judgements for each of the 100 translation variants. An example of the combined answer sheet for all responses to the question from Table 2 is given in Table 3 (t1, t2,... denote translators; the dictionary translation is *clear programme*).

Translation	t1	t2	t3	t4	t5	σ
clear plan	5	5	3	4	4	0.84
clear policy	5	5	3	4	4	0.84
clear programme	5	5	3	4	4	0.84
clear strategy	5	5	5	5	5	0.00
concrete plan	1	5	3	3	5	1.67
Best Dict	5	5	3	4	4	0.84
Best Syst	5	5	5	5	5	0.00

Table 3: Scores to translation equivalents

3.2 Interpretation of the results

The results were surprising in so far as for the majority of problems translators preferred very different translation solutions and did not agree in their scores for the same solutions. For instance, *concrete plan* in Table 3 received the score 1 from translator t1 and 5 from t2.

In general, the translators very often picked up on different opportunities presented by the suggestions from the lists, and most suggestions were equally legitimate ways of conveying the intended content, cf. the study of legitimate translation variation with respect to the BLEU score in (Babych and Hartley, 2004). In this respect it may be unfair to compute average scores for each potential solution, since for most interesting cases the scores do not fit into the normal distribution model. So averaging scores would mask the potential usability of really inventive solutions.

In this case it is more reasonable to evaluate two *sets* of solutions – the one generated by ASSIST and the other found in dictionaries – but not each solution individually. In order to do that for each translation problem the best scores given by each translator in each of these two sets were selected. This way of generalising data characterises the general quality of suggestion sets, and exactly meets the needs of translators, who collectively get ideas from the presented sets rather than from individual examples. This also allows us to measure inter-evaluator agreement on the *dictionary* set and the *ASSIST* set, for instance, via computing the standard deviation σ of absolute scores across evaluators (Table 3). This appeared to be a very informative measure for dictionary solutions.

In particular, standard deviation scores for the *dictionary* set (threshold $\sigma = 0.5$) clearly split

Agreement: σ for dictionary $\leq$ 0.5				
Example	**Dict**		**ASSIST**	
	Ave	σ	Ave	σ
political upheaval	4.83	0.41	4.67	0.82
Disagreement: σ for dictionary >0.5				
Example	**Dict**		**ASSIST**	
	Ave	σ	Ave	σ
clear defiance	4.14	0.90	4.60	0.55

Table 4: Examples for the two groups

Agreement: σ for dictionary $\leq$ 0.5				
Sub-group	**Dict**		**ASSIST**	
	Ave	σ	Ave	σ
Agreement E→R	4.73	0.46	4.47	0.80
Agreement R→E	4.90	0.23	4.52	0.60
Agreement–All	**4.81**	**0.34**	**4.49**	**0.70**
Disagreement: σ for dictionary >0.5				
Sub-group	**Dict**		**ASSIST**	
	Ave	σ	Ave	σ
Disagreement E→R	3.63	1.08	3.98	0.85
Disagreement R→E	3.90	1.02	3.96	0.73
Disagreement–All	**3.77**	**1.05**	**3.97**	**0.79**

Table 5: Averages for the two groups

our 20 problems into two distinct groups: the first group below the threshold contains 8 examples, for which translators typically agree on the quality of dictionary solutions; and the second group above the threshold contains 12 examples, for which there is less agreement. Table 4 shows some examples from both groups and Table 5 presents average evaluation scores and standard deviation figures for both groups.

Overall performance on all 20 examples is the same for the dictionary responses and for the system's responses: average of the mean top scores is about 4.2 and average standard deviation of the scores is 0.8 in both cases (for set-best responses). This shows that ASSIST can reach the level of performance of a combination of two authoritative dictionaries for MWEs, while for its own translation step it uses just a subset of one-word translation equivalents from ORD. However, there is another side to the evaluation experiment. In fact, we are less interested in the system's performance on all of these examples than on those examples for which there is greater disagreement among translators, i.e. where there is some degree of dissatisfaction with dictionary suggestions.

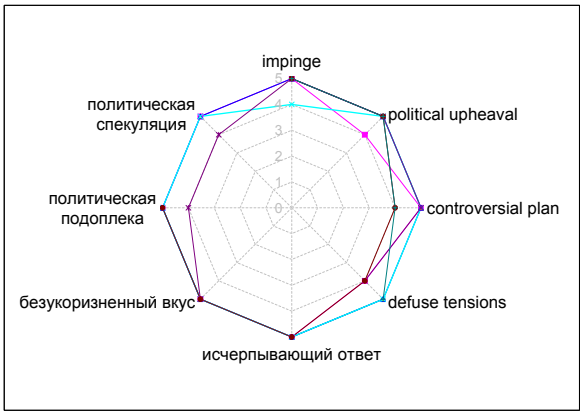


Figure 1: Agreement scores: dictionary

Interestingly, dictionary scores for the agreement group are always higher than 4, which means that whenever translators agreed on the dictionary scores they were usually satisfied with the dictionary solution. But they never agreed on the inappropriateness of the dictionary: inappropriateness revealed itself in the form of low scores from *some* translators.

This agreement/disagreement threshold can be said to characterise two types of translation problems: those for which there exist generally accepted dictionary solutions, and those for which translators doubt whether the solution is appropriate. Best-set scores for these two groups of dictionary solutions – the agreement and disagreement group – are plotted on the radar charts in Figures 1 and 2 respectively. The identifiers on the charts are problematic source language expressions as used in the questionnaire (not translation solutions to these problems, because a problem may have several solutions preferred by different judges). Scores for both translation directions are presented on the same chart, since both follow the same pattern and receive the same interpretation.

Figure 1 shows that whenever there is little doubt about the quality of dictionary solutions, the radar chart approaches a circle shape near the edge of the chart. In Figure 2 the picture is different: the circle is disturbed, and some scores frequently approach the centre. Therefore the disagreement group contains those translation problems where dictionaries provide little help.

The central problem in our evaluation experiment is whether ASSIST is helpful for problems in the second group, where translators doubt the quality of dictionary solutions.

Firstly, it can be seen from the charts that judge-

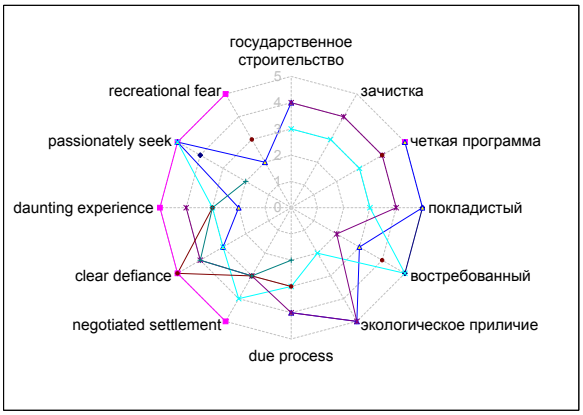


Figure 2: Disagreement scores: dictionary

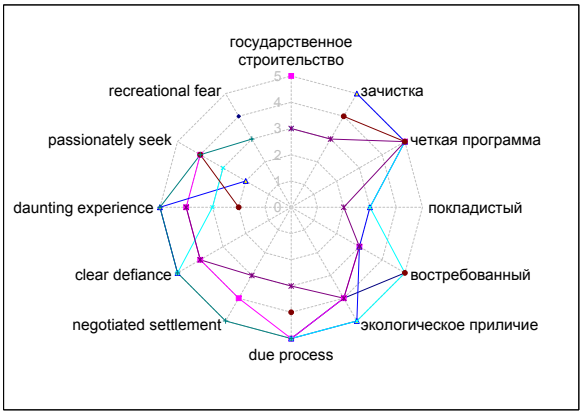


Figure 3: Disagreement scores: ASSIST

ments on the quality of the system output are more consistent: score lines for system output are closer to the circle shape in Figure 1 than those for dictionary solutions in Figure 2 (formally: the standard deviation of evaluation scores, presented in Table 4, is lower).

Secondly, as shown in Table 4, in this group average evaluation scores are slightly higher for ASSIST output than for dictionary solutions (3.97 vs 3.77) – in the eyes of human evaluators ASSIST outperforms good dictionaries. For good dictionary solutions ASSIST performance is slightly lower: (4.49 vs 4.81), but the standard deviation is about the same.

Having said this, solutions from our system are really not in competition with dictionary solutions: they provide less literal translations, which often emerge in later stages of the translation task, when translators correct and improve an initial draft, where they have usually put more literal equivalents (Shveitser, 1988). It is a known fact in translation studies that non-literal solutions are harder to see and translators often find them only upon longer reflection. Yet another fact is that non-literal translations often require re-writing other segments of the sentence, which may not be obvious at first glance.

4 Conclusions and future work

The results of evaluation show that the tool is successful in finding translation equivalents for a range of examples. What is more, in cases where the problem is genuinely difficult, ASSIST consistently provides scores around 4 – "minor adaptations needed". The precision of the tool is low, it suggests 50-100 examples with only 2-4 useful for the current context. However, recall of the output is more relevant than precision, because translators typically need just one solution for their problem, and often have to look through reasonably large lists of dictionary translations and examples to find something suitable for a problematic expression. Even if no immediately suitable translation can be found in the list of suggestions, it frequently contains a hint for solving the problem in the absence of adequate dictionary information.

The current implementation of the model is restricted in several respects. First, the majority of target language constructions mirror the syntactic structure of the source language example. Even if the procedure for producing similarity classes does not impose restrictions on POS properties, nevertheless words in the similarity class tend to follow the POS of the original word, because of the similarity of their contexts of use. Furthermore, dictionaries also tend to translate words using the same POS. This means that the existing method finds mostly NPs for NPs, verb-object pairs for verb-object pairs, etc, even if the most natural translation uses a different syntactic structure, e.g. *I like doing X* instead of *I do X gladly* (when translating from German *ich mache X gerne*).

Second, suggestions are generated for the query expression independently from the context it is used in. For instance, the words *judicial, military* and *religious* are in the similarity class of *political*, just as *reform* is in the simclass of *upheaval*. So the following example

The plan will protect EC-based investors in Russia from political upheavals damaging their business.

creates a list of "possible translations" evoking various reforms and transformations.

These issues can be addressed by introducing a model of the semantic context of situation, e.g. 'changes in business practice' as in the example above, or 'unpleasant situation' as in the case of *daunting experience*. This will allow less restrictive identification of possible translation equivalents, as well as reduction of suggestions irrelevant for the context of the current example.

Currently we are working on an option to identify semantic contexts by means of 'semantic signatures' obtained from a broad-coverage semantic parser, such as USAS (Rayson et al., 2004). The semantic tagset used by USAS is a language-independent multi-tier structure with 21 major discourse fields, subdivided into 232 sub-categories (such as `I1.1-` = Money: lack; `A5.1-` = Evaluation: bad), which can be used to detect the semantic context. Identification of semantically similar situations can be also improved by the use of segment-matching algorithms as employed in Example-Based MT (EBMT) and translation memories (Planas and Furuse, 2000; Carl and Way, 2003).

The proposed model looks similar to some implementations of statistical machine translation (SMT), which typically uses a parallel corpus for its translation model, and then finds the best possible recombination that fits into the target language model (Och and Ney, 2003). Just like an MT system, our tool can find translation equivalents for queries which are not explicitly coded as entries in system dictionaries. However, from the user perspective it resembles a dynamic dictionary or thesaurus: it translates difficult words and phrases, not entire sentences. The main thrust of our system is its ability to find translation equivalents for difficult contexts where dictionary solutions do not exist, are questionable or inappropriate.

Acknowledgements

This research is supported by EPSRC grant EP/C005902.

References

Bogdan Babych and Anthony Hartley. 2004. Extending the BLEU MT evaluation method with frequency weightings. In *Proceedings of the 42^d Annual Meeting of the Association for Computational Linguistics*, Barcelona.

Michael Carl and Andy Way, editors. 2003. *Recent advances in example-based machine translation*. Kluwer, Dordrecht.

Ido Dagan and Kenneth Church. 1997. Termight: Coordinating humans and machines in bilingual terminology acquisition. *Machine Translation*, 12(1/2):89–107.

Gregory Grefenstette. 2002. Multilingual corpus-based extraction and the very large lexicon. In Lars Borin, editor, *Language and Computers, Parallel corpora, parallel worlds*, pages 137–149. Rodopi.

John S. Justeson and Slava M. Katz. 1995. Techninal terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, 1(1):9–27.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Joint COLING-ACL-98*, pages 768–774, Montreal.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Alan Partington. 1998. *Patterns and meanings: using corpora for English language research and teaching*. John Benjamins, Amsterdam.

Emmanuel Planas and Osamu Furuse. 2000. Multi-level similar segment matching algorithm for translation memories and example-based machine translation. In *COLING, 18th International Conference on Computational Linguistics*, pages 621–627.

Reinhard Rapp. 2004. A freely available automatically generated thesaurus of related words. In *Proceedings of the Forth Language Resources and Evaluation Conference, LREC 2004*, pages 395–398, Lisbon.

Paul Rayson, Dawn Archer, Scott Piao, and Tony McEnery. 2004. The UCREL semantic analysis system. In *Proc. Beyond Named Entity Recognition Workshop in association with LREC 2004*, pages 7–12, Lisbon.

Serge Sharoff. 2006. Creating general-purpose corpora using automated search engine queries. In Marco Baroni and Silvia Bernardini, editors, *WaCky! Working papers on the Web as Corpus*. Gedit, Bologna.

A.D. Shveitser. 1988. Теория перевода: Статус, проблемы, аспекты. Nauka, Moskow. (In Russian: Theory of Translation: Status, Problems, Aspects).

Krista Varantola. 2003. Translators and disposable corpora. In Federico Zanettin, Silvia Bernardini, and Dominic Stewart, editors, *Corpora in Translator Education*, pages 55–70. St Jerome, Manchester.

Adding Syntax to Dynamic Programming for Aligning Comparable Texts for the Generation of Paraphrases

Siwei Shen[1], **Dragomir R. Radev**[1,2], **Agam Patel**[1], **Güneş Erkan**[1]
Department of Electrical Engineering and Computer Science
School of Information
University of Michigan
Ann Arbor, MI 48109
{shens, radev, agamrp, gerkan}@umich.edu

Abstract

Multiple sequence alignment techniques have recently gained popularity in the Natural Language community, especially for tasks such as machine translation, text generation, and paraphrase identification. Prior work falls into two categories, depending on the type of input used: (a) parallel corpora (e.g., multiple translations of the same text) or (b) comparable texts (non-parallel but on the same topic). So far, only techniques based on parallel texts have successfully used syntactic information to guide alignments. In this paper, we describe an algorithm for incorporating syntactic features in the alignment process for non-parallel texts with the goal of generating novel paraphrases of existing texts. Our method uses dynamic programming with alignment decision based on the local syntactic similarity between two sentences. Our results show that syntactic alignment outrivals syntax-free methods by 20% in both grammaticality and fidelity when computed over the novel sentences generated by alignment-induced finite state automata.

1 Introduction

In real life, we often encounter comparable texts such as news on the same events reported by different sources and papers on the same topic authored by different people. It is useful to recognize if one text cites another in cases like news sharing among media agencies or citations in academic work. Applications of such recognition include machine translation, text generation, paraphrase identification, and question answering, all of which have recently drawn the attention of a number of researchers in natural language processing community.

Multiple sequence alignment (MSA) is the basis for accomplishing these tasks. Previous work aligns a group of sentences into a compact word lattice (Barzilay and Lee, 2003), a finite state automaton representation that can be used to identify commonality or variability among comparable texts and generate paraphrases. Nevertheless, this approach has a drawback of over-generating ungrammatical sentences due to its "almost-free" alignment. Pang et al. provide a remedy to this problem by performing alignment on the Charniak parse trees of the clustered sentences (Pang et al., 2003). Although it is so far the most similar work to ours, Pang's solution assumes the input sentences to be semantically equivalent. Two other important references for string-based alignments algorithms, mostly with applications in Biology, are (Gusfield, 1997) and (Durbin et al., 1998).

In our approach, we work on comparable texts (not necessarily equivalent in their semantic meanings) as Barzilay and Lee did. However, we use local syntactic similarity (as opposed to lexical similarity) in doing the alignment on the raw sentences instead of on their parse trees. Because of the semantic discrepancies among the inputs, applying syntactic features in the alignment has a larger impact on the grammaticality and fidelity of the generated unseen sentences. While previous work positions the primary focus on the quality of paraphrases and/or translations, we are more interested in the relation between the use of syntactic features and the correctness of the sentences being generated, including those that are not paraphrases of the original input. Figure 1 illustrates the difference between alignment based solely on lexical similarity and alignment with consideration of syntactic features.

Ignoring syntax, the word "Milan" in both sentences is aligned. But it would unfortunately generate an ungrammatical sentence "I went to Milan is beautiful". Aligning according to syntac-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 747–754,
Sydney, July 2006.

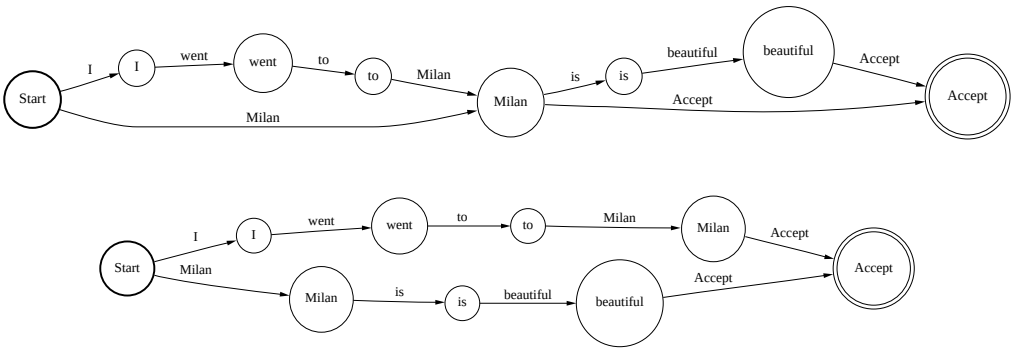


Figure 1: Alignment on lexical similarity and alignment with syntactic features of the sentences "Milan is beautiful" and "I went to Milan".

tic features, on the other hand, would avoid this improper alignment by detecting that the syntactic feature values of the two "Milan" differ too much. We shall explain syntactic features and their usages later. In this small example, our syntax-based alignment will align nothing (the bottom FSA in Figure 1) since "Milan" is the only lexically common word in both sentences. For much larger clusters in our experiments, we are able to produce a significant number of novel sentences from our alignment with such tightened syntactic conditions. Figure 2 shows one of the actual clusters used in our work that has 18 unique sentences. Two of the many automatically generated grammatical sentences are also shown.

Another piece of related work, (Quirk et al., 2004), starts off with parallel inputs and uses monolingual Statistical Machine Translation techniques to align them and generate novel sentences. In our work, the input text does not need to be nearly as parallel.

The main contribution of this paper is a syntax-based alignment technique for generating novel paraphrases of sentences that describe a particular fact. Such techniques can be potentially useful in multi-document summarizers such as Newsblaster (`http://newsblaster.cs.columbia.edu`) and NewsInEssence (`http://www.newsinessence.com`). Such systems are notorious for mostly reusing text from existing news stories. We believe that allowing them to use novel formulations of known facts will make these systems much more successful.

2 Related work

Our work is closest in spirit to the two papers that inspired us (Barzilay and Lee, 2003) and (Pang et al., 2003). Both of these papers describe how multiple sequence alignment can be used for extracting paraphrases from clustered texts. Pang et al. use as their input the multiple human English translations of Chinese documents provided by the LDC as part of the NIST machine translation evaluation. Their approach is to merge multiple parse trees into a single finite state automaton in which identical input subconstituents are merged while alternatives are converted to parallel paths in the output FSA. Barzilay and Lee, on the other hand, make use of classic techniques in biological sequence analysis to identify paraphrases from comparable texts (news from different sources on the same event).

In summary, Pang et al. use syntactic alignment of parallel texts while Barzilay and Lee use comparable (not parallel) input but ignore syntax. Our work differs from the two in that we apply syntactic information on aligning comparable texts and that the syntactic clues we use are drawn from Chunklink `ilk.uvt.nl/~sabine/homepage/software.html` output, which is further analysis from the syntactic parse trees.

Another related paper using multiple sequence alignment for text generation was (Barzilay and Lee, 2002). In that work, the authors were able to automatically acquire different lexicalizations of the same concept from "multiple-parallel corpora". We also draw some ideas from the Fitch-Margoliash method for building evolutionary trees

1. A police official said it was a Piper tourist plane and that the crash had set the top floors on fire.
2. According to ABCNEWS aviation expert John Nance, Piper planes have no history of mechanical troubles or other problems that would lead a pilot to lose control.
3. April 18, 2002 8212; A small Piper aircraft crashes into the 417-foot-tall Pirelli skyscraper in Milan, setting the top floors of the 32-story building on fire.
4. Authorities said the pilot of a small Piper plane called in a problem with the landing gear to the Milan's Linate airport at 5:54 p.m., the smaller airport that has a landing strip for private planes.
5. Initial reports described the plane as a Piper, but did not note the specific model.
6. Italian rescue officials reported that at least two people were killed after the Piper aircraft struck the 32-story Pirelli building, which is in the heart of the city s financial district.
7. MILAN, Italy AP A small piper plane with only the pilot on board crashed Thursday into a 30-story landmark skyscraper, killing at least two people and injuring at least 30.
8. Police officer Celerissimo De Simone said the pilot of the Piper Air Commander plane had sent out a distress call at 5:50 p.m. just before the crash near Milan's main train station.
9. Police officer Celerissimo De Simone said the pilot of the Piper aircraft had sent out a distress call at 5:50 p.m. 11:50 a.m.
10. Police officer Celerissimo De Simone said the pilot of the Piper aircraft had sent out a distress call at 5:50 p.m. just before the crash near Milan's main train station.
11. Police officer Celerissimo De Simone said the pilot of the Piper aircraft sent out a distress call at 5:50 p.m. just before the crash near Milan's main train station.
12. Police officer Celerissimo De Simone told The AP the pilot of the Piper aircraft had sent out a distress call at 5:50 p.m. just before crashing.
13. Police say the aircraft was a Piper tourism plane with only the pilot on board.
14. Police say the plane was an Air Commando 8212; a small plane similar to a Piper.
15. Rescue officials said that at least three people were killed, including the pilot, while dozens were injured after the Piper aircraft struck the Pirelli high-rise in the heart of the city s financial district.
16. The crash by the Piper tourist plane into the 26th floor occurred at 5:50 p.m. 1450 GMT on Thursday, said journalist Desideria Cavina.
17. The pilot of the Piper aircraft, en route from Switzerland, sent out a distress call at 5:54 p.m. just before the crash, said police officer Celerissimo De Simone.
18. There were conflicting reports as to whether it was a terrorist attack or an accident after the pilot of the Piper tourist plane reported that he had lost control.

1. Police officer Celerissimo De Simone said the pilot of the Piper aircraft, en route from Switzerland, sent out a distress call at 5:54 p.m. just before the crash near Milan's main train station.
2. Italian rescue officials reported that at least three people were killed, including the pilot, while dozens were injured after the Piper aircraft struck the 32-story Pirelli building, which is in the heart of the city s financial district.

Figure 2: A comparable cluster of size 18 and 2 novel sentences produced by syntax-based alignment.

described in (Fitch and Margoliash, 1967). That method and related techniques in Bioinformatics such as (Felsenstein, 1995) also make use of a similarity matrix for aligning a number of sequences.

3 Alignment Algorithms

Our alignment algorithm can be described as modifying Levenshtein Edit Distance by assigning different scores to lexically matched words according to their syntactic similarity. And the decision of whether to align a pair of words is based on such syntax scores.

3.1 Modified Levenshtein Edit Distance

The Levenshtein Edit Distance (LED) is a measure of similarity between two strings named after the Russian scientist Vladimir Levenshtein, who devised the algorithm in 1965. It is the number of substitutions, deletions or insertions (hence "edits") needed to transform one string into the other. We extend LED to sentence level by counting the substitutions, deletions and insertions of words necessary to transform a sentence into the other. We abbreviate this sentence-level edit distance as MLED. Similar to LED, MLED computation produces an M+1 by N+1 distance matrix, D, given two input sentences of length M and N respectively. This matrix is constructed through dynamic programming as shown in Figure 3.

$$D[i][j] = \begin{cases} 0 & \text{if } j = 0 \\ 0 & \text{if } i = 0 \\ \max \begin{pmatrix} D[i-1][j-1] + \text{match}, \\ D[i-1][j] + \text{gap}, \\ D[i][j-1] + \text{gap} \end{pmatrix} & \text{otherwise} \end{cases}$$

Figure 3: Dynamic programming in computing MLED of two sentences of length M and N.

"match" is 2 if the i^{th} word in Sentence 1 and the j^{th} word in Sentence 2 syntactically match, and is -1 otherwise. "gap" represents the score for inserting a gap rather than aligning, and is set to -1. The matching conditions of two words are far more complicated than lexical equality. Rather, we judge whether two lexically equal words match based on a predefined set of syntactic features.

The output matrix is used to guide the alignment. Starting from the bottom right entry of the matrix, we go to the matrix entry from which the value of the current cell is derived in the recursion of the dynamic programming. Call the current entry $D[i][j]$. If it gets its value from $D[i-1][j-1]$, the i^{th} word in Sentence 1 and the j^{th} word in Sentence 2 are either aligned or both aligned to a gap depending on whether they syntactically match; if the value of $D[i][j]$ is derived from $D[i][j-1]$ +

"gap", the i^{th} word in Sentence 1 is aligned to a gap inserted into Sentence 2 (the j^{th} word in Sentence 2 is not consumed); otherwise, the j^{th} word in Sentence 2 is aligned to a gap inserted into Sentence 1.

Now that we know how to align two sentences, aligning a cluster of sentences is done progressively. We start with the overall most similar pair and then respect the initial ordering of the cluster, aligning remaining sentences sequentially. Each sentence is aligned against its best match in the pool of already-aligned ones. This approach is a hybrid of the Feng-Doolittle's Algorithm (Feng and Doolittle, 1987) and a variant described in (Fitch and Margoliash, 1967).

3.2 Syntax-based Alignment

As remarked earlier, our alignment scheme judges whether two words match according to their syntactic similarity on top of lexical equality. The syntactic features are obtained from running Chunklink (Buchholz, 2000) on the Charniak parses of the clustered sentences.

3.2.1 Syntactic Features

Among all the information Chunklink provides, we use in particular the part-of-speech tags, the Chunk tags, and the syntactic dependence traces. The Chunk tag shows the constituent of a word and its relative position in that constituent. It can take one of the three values,

- "O" meaning that the word is outside of any chunk;
- "I-XP" meaning that this word is inside an XP chunk where X = N, V, P, ADV, ...;
- "B-XP" meaning that the word is at the beginning of an XP chunk.

From now on, we shall refer to the Chunk tag of a word as its IOB value (IOB was named by Tjong Kim Sang and Jorn Veeenstra (Tjong Kim Sang and Veenstra, 1999) after Ratnaparkhi (Ratnaparkhi, 1998)). For example, in the sentence "I visited Milan Theater", the IOB value for "I" is B-NP since it marks the beginning of a noun-phrase (NP). On the other hand, "Theater" has an IOB value of I-NP because it is inside a noun-phrase (Milan Theater) and is not at the beginning of that constituent. Finally, the syntactic dependence trace of a word is the path of IOB values from the root of the tree to the word itself. The last element in the trace is hence the IOB of the word itself.

3.2.2 The Algorithm

Lexically matched words but with different POS are considered not syntactically matched (e.g., race VB vs. race NN). Hence, our focus is really on pairs of lexically matched words with the same POS. We first compare their IOB values. Two IOB values are exactly matched only if they are identical (same constituent and same position); they are partially matched if they share a common constituent but have different position (e.g., B-PP vs. I-PP); and they are unmatched otherwise. For a pair of words with exactly matched IOB values, we assign 1 as their **IOB-score**; for those with partially matched IOB values, 0; and -1 for those with unmatched IOB values. The numeric values of the score are from experimental experience.

The next step is to compare syntactic dependence traces of the two words. We start with the second last element in the traces and go backward because the last one is already taken care of by the previous step. We also discard the front element of both traces since it is "I-S" for all words. The corresponding elements in the two traces are checked by the IOB-comparison described above and the scores accumulated. The process terminates as soon as one of the two traces is exhausted. Last, we adjust down the cumulative score by the length difference between the two traces. Such final score is named the **trace-score** of the two words.

We declare "unmatched" if the sum of the IOB-score and the trace-score falls below 0. Otherwise, we perform one last measurement – the relative position of the two words in their respective sentences. The relative position is defined to be the word's absolute position divided by the length of the sentence it appears in (e.g. the 4th word of a 20-word sentence has a relative position of 0.2). If the difference between two relative positions is larger than 0.4 (empirically chosen before running the experiments), we consider the two words "unmatched". Otherwise, they are syntactically matched.

The pseudo-code of checking syntactic match is shown in Figure 4.

Algorithm Check Syntactic Match of Two Words
For a pair of words W_1, W_2
if $W_1 \neq W_2$ or $pos(W_1) \neq pos(W_2)$ **then**
return "unmatched"
endif
$score := 0$
$iob_1 := iob(W_1)$
$iob_2 := iob(W_2)$
$score \mathrel{+}= compare\_iobs(iob_1, iob_2)$
$trace_1 := trace(W_1)$
$trace_2 := trace(W_2)$
$score \mathrel{+}= compare\_traces(trace_1, trace_2)$
if score < 0 **then**
return "unmatched"
endif
$relpos_1 := pos(W_1)/lengthOf(S_1)$
$relpos_2 := pos(W_2)/lengthOf(S_2)$
if $|relpos_1 - relpos_2| \geq 0.4$ **then**
return "unmatched"
endif
return "matched"
Function $compare\_iobs(iob_1, iob_2)$
if $iob_1 = iob_2$ **then**
return 1
endif
if $substring(iob_1, 1) = substring(iob_2, 1)$ **then**
return 0
endif
return -1
Function $compare\_traces(trace_1, trace_2)$
Remove first and last elements from both traces
$score := 0$
$i := lengthOf(trace_1) - 1$
$j := lengthOf(trace_2) - 1$
while $i \geq 0$ and $j \geq 0$ **do**
$next := compare\_iobs(trace_1[i], trace_2[j])$
$score \mathrel{+}= next * 0.5$
$i--$
$j--$
endwhile
$score -= |lengthOf(trace_1) - lengthOf(trace_2)| * 0.5$
return $score$

Figure 4: Algorithm for checking the syntactic match between two words.

4 Evaluation

4.1 Experimental Setup

4.1.1 Data

The data we use in our experiment come from a number of sentence clusters on a variety of topics, but all related to the Milan plane crash event. This cluster was collected manually from the Web of five different news agencies (ABC, CNN, Fox, MSNBC, and USAToday). It concerns the April 2002 crash of a small plane into a building in Milan, Italy and contains a total of 56 documents published over a period of 1.5 days. To divide this corpus into representative smaller clusters, we had a colleague thoroughly read all 56 documents in the cluster and then create a list of important facts surrounding the story. We then picked key terms related to these facts, such as names (Fasulo - the pilot) and locations (Locarno - the city from which the plane had departed). Finally, we automatically clustered sentences based on the presence of these key terms, resulting in 21 clusters of topically related (comparable) sentences. The 21 clusters are grouped into three categories: 7 in training set, 3 in dev-testing set, and the remaining 11 in testing set. Table 1 shows the name and size of each cluster.

Cluster	Number of Sentences
Training clusters	
ambulance	10
belie	14
built	6
malpensa	4
piper	18
president	17
route	11
Dev-test clusters	
hospital	17
rescue	12
witness	6
Test clusters	
accident	30
cause	18
fasulo	33
floor	79
government	22
injur	43
linate	21
rockwell	9
spokes	18
suicide	22
terror	62

Table 1: Experimental clusters.

4.1.2 Different Versions of Alignment

To test the usefulness of our work, we ran 5 different alignments on the clusters. The first three represent different levels of baseline performance (without syntax consideration) whereas the last two fully employ the syntactic features but treat stop words differently. Table 2 describes the 5 versions of alignment.

Run	Description
V1	Lexical alignment on everything possible
V2	Lexical alignment on everything but commas
V3	Lexical alignment on everything but commas and stop words
V4	Syntactic alignment on everything but commas and stop words
V5	Syntactic alignment on everything but commas

Table 2: Alignment techniques used in the experiments.

Alignment	Grammaticality	Fidelity
V1	2.89	2.98
V2	3.00	2.95
V3	3.15	3.22
V4	3.68	3.59
V5	3.47	3.30

Table 3: Evaluation results on training and dev-testing clusters. For the results on the test clusters, see Table 6

The motivation of trying such variations is as follows. Stop words often cause invalid alignment because of their high frequencies, and so do punctuations. Aligning on commas, in particular, is likely to produce long sentences that contain multiple sentence segments ungrammatically patched together.

4.1.3 Training and Testing

In order to get the best possible performance of the syntactic alignment versions, we use clusters in the training and dev-test sets to tune up the parameter values in our algorithm for checking syntactic match. The parameters in our algorithm are not independent. We pay special attention to the threshold of relative position difference, the discount factor of the trace length difference penalty, and the scores for exactly matched and partially matched IOB values. We try different parameter settings on the training clusters, and apply the top ranking combinations (according to human judgments described later) on clusters in the dev-testing set. The values presented in this paper are the manually selected ones that yield the best performance on the training and dev-testing sets.

Experimenting on the testing data, we have two hypotheses to verify: 1) the 2 syntactic versions outperform the 3 baseline versions by both grammaticality and fidelity (discussed later) of the novel sentences produced by alignment; and 2) disallowing alignment on stop words and commas enhances the performance.

4.2 Experimental Results

For each cluster, we ran the 5 alignment versions and produce 5 FSA's. From each FSA (corresponding to a cluster A and alignment version i), 100 sentences are randomly generated. We removed those that appear in the original cluster. The remaining ones are hence novel sentences, among which we randomly chose 10 to test the performance of alignment version i on cluster A.

In the human evaluation, each sentence received two scores – grammaticality and fidelity. These two properties are independent since a sentence could possibly score high on fidelity even if it is not fully grammatical. Four different scores are possible for both criteria: (4) perfect (fully grammatical or faithful); (3) good (occasional errors or quite faithful); (2) bad (many grammar errors or unfaithful pieces); and (1) nonsense.

4.2.1 Results from the Training Phase

Four judges help our evaluation in the training phase. They are provided with the original clusters during the evaluation process, yet they are given the sentences in shuffled order so that they have no knowledge about from which alignment version each sentence is generated. Table 3 shows the averages of their evaluation on the 10 clusters in training and dev-testing set. Each cell corresponds to 400 data points as we presented 10 sentences per cluster per alignment version to each of the 4 judges (10 x 10 x 4 = 400).

4.2.2 Results from the Testing Phase

After we have optimized the parameter configuration for our syntactic alignment in the training phase, we ask another 6 human judges to evaluate our work on the testing data. These 6 judges come from diverse background including Information, Computer Science, Linguistics, and Bioinformatics. We distribute the 11 testing clusters among them so that each cluster gets evaluated by at least 3 judges. The workload for each judge is 6 clusters x 5 versions/cluster x 10 sentences/cluster-version = 300 sentences. Similar to the training phase, they receive the sentences in shuffled order without knowing the correspondence between

sentences and alignment versions. Detailed average statistics are shown in Table 4 and Table 5 for grammaticality and fidelity, respectively. Each cell is the average over 30 - 40 data points, and notice the last row is not the mean of the other rows since the number of sentences evaluated for each cluster varies.

Cluster	V1	V2	V3	V4	V5
rockwell	2.27	2.93	3.00	3.60	3.03
cause	2.77	2.83	3.07	3.10	2.93
spokes	2.87	3.07	3.57	3.83	3.50
linate	2.93	3.14	3.26	3.64	3.77
government	2.75	2.83	3.27	3.80	3.20
suicide	2.19	2.51	3.29	3.57	3.11
accident	2.92	3.27	3.54	3.72	3.56
fasulo	2.52	2.52	3.15	3.54	3.32
injur	2.29	2.92	3.03	3.62	3.29
terror	3.04	3.11	3.61	3.23	3.63
floor	2.47	2.77	3.40	3.47	3.27
Overall	2.74	2.75	3.12	3.74	3.29

Table 4: Average grammaticality scores on testing clusters.

Cluster	V1	V2	V3	V4	V5
rockwell	2.25	2.75	3.20	3.80	2.70
cause	2.42	3.04	2.92	3.48	3.17
spokes	2.65	2.50	3.20	3.00	3.05
linate	3.15	3.27	3.15	3.36	3.42
government	2.85	3.24	3.14	3.81	3.20
suicide	2.38	2.69	2.93	3.68	3.23
accident	3.14	3.42	3.56	3.91	3.57
fasulo	2.30	2.48	3.14	3.50	3.48
injur	2.56	2.28	2.29	3.18	3.22
terror	2.65	2.48	3.68	3.47	3.20
floor	2.80	2.90	3.10	3.70	3.30
Overall	2.67	2.69	3.07	3.77	3.23

Table 5: Average fidelity scores on testing clusters.

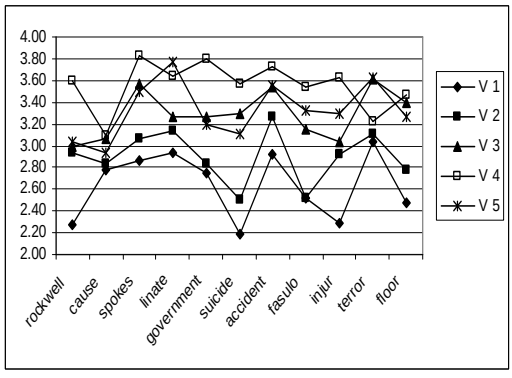


Figure 5: Performance of 5 alignment versions by grammaticality.

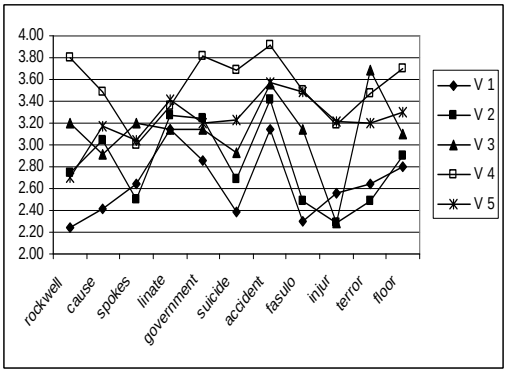


Figure 6: Performance of 5 alignment versions by fidelity.

4.3 Result Analysis

The results support both our hypotheses. For Hypothesis I, we see that the performance of the two syntactic alignments was higher than the non-syntactic versions. In particular, Version 4 outperforms the the best baseline version by 19.9% on grammaticality and by 22.8% on fidelity. Our second hypothesis is also verified – disallowing alignment on stop words and commas yields better results. This is reflected by the fact that Version 4 beats Version 5, and Version 3 wins over the other two baseline versions by both criteria.

At the level of individual clusters, the syntactic versions are also found to outrival the syntax-blind baselines. Applying a t-test on the score sets for the 5 versions, we can reject the null hypothesis with 99.5% confidence to ensure that the syntactic alignment performs better. Similarly, for hypothesis II, the same is true for the versions with and without stop word alignment. Figures 5 and 6 provide a graphical view of how each alignment version performs on the testing clusters. The clusters along the x-axis are listed in the order of increasing size.

We have also done an analysis on interjudge agreement in the evaluation. The judges are instructed about the evaluation scheme individually, and do their work independently. We do not enforce them to be mutually consistent, as long as they are self-consistent. However, Table 6 shows the mean and standard deviation of human judgments (grammaticality and fidelity) on each version. The small deviation values indicate a fairly high agreement.

Finally, because human evaluation is expensive, we additionally tried to use a language-model ap-

Alignment	Gr. Mean	Gr. StdDev	Fi. Mean	Fi. StdDev
V1	2.74	0.11	2.67	0.43
V2	2.75	0.08	2.69	0.30
V3	3.12	0.07	3.07	0.27
V4	3.74	0.08	3.77	0.16
V5	3.29	0.16	3.23	0.33

Table 6: Mean and standard deviation of human judgments.

proach in the training phase for automatic evaluation of grammaticality. We have used BLEU scores(Papineni et al., 2001), but have observed that they are not consistent with those of human judges. In particular, BLEU assigns too high scores to segmented sentences that are otherwise grammatical. It has been noted in the literature that metrics like BLEU that are solely based on N-grams might not be suitable for checking grammaticality.

5 Conclusion

In this paper, we presented a paraphrase generation method based on multiple sequence alignment which combines traditional dynamic programming techniques with linguistically motivated syntactic information. We apply our work on comparable texts for which syntax has not been successfully explored in alignment by previous work. We showed that using syntactic features improves the quality of the alignment-induced finite state automaton when it is used for generating novel sentences. The strongest syntax guided alignment significantly outperformed all other versions in both grammaticality and fidelity of the novel sentences. In this paper we showed the effectiveness of using syntax in the alignment of structurally diverse comparable texts as needed for text generation.

References

Regina Barzilay and Lillian Lee. 2002. Bootstrapping Lexical Choice via Multiple-Sequence Alignment. In *Proceedings of EMNLP 2002*, Philadelphia.

Regina Barzilay and Lillian Lee. 2003. Learning to Paraphrase: An Unsupervised Approach Using Multiple-Sequence Alignment. In *Proceedings of NAACL-HLT03*, Edmonton.

Sabine Buchholz. 2000. Readme for perl script chunklink.pl. http://ilk.uvt.nl/ sabine/chunklink/README.html.

Richard Durbin, Sean R. Eddy, Anders Krogh, and Graeme Mitchison. 1998. *Biological Sequence Analysis. Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press.

Joseph Felsenstein. 1995. PHYLIP: Phylogeny Inference Package. http://evolution.genetics.washington.edu/phylip.html.

DF. Feng and Russell F. Doolittle. 1987. Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *Journal of Molecular Evolution*, 25(4).

Walter M. Fitch and Emanuel Margoliash. 1967. Construction of Phylogenetic Trees. *Science*, 155(3760):279–284, January.

Dan Gusfield, 1997. *Algorithms On Strings: A Dual View from Computer Science and Computational Molecular Biology*. Cambridge University Press.

Bo Pang, Kevin Knight, and Daniel Marcu. 2003. Syntax-based Alignment of Multiple Translations: Extracting Paraphrases and Generating New Sentences. In *Proceedings of HLT/NAACL 2003*, Edmonton, Canada.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. BLEU: A Method for Automatic Evaluation of Machine Translation. Research Report RC22176, IBM.

Chris Quirk, Chris Brockett, and William Dolan. 2004. Monolingual machine translation for paraphrase generation. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 142–149, Barcelona, Spain, July. Association for Computational Linguistics.

A Ratnaparkhi. 1998. *Maximum Entropy Models for Natural Language Ambiguity Resolution*. Phd. Thesis, University of Pennsylvania.

Erik F. Tjong Kim Sang and Jorn Veenstra. 1999. Representing text chunks. In *EACL*, pages 173–179.

Unsupervised Topic Identification by Integrating Linguistic and Visual Information Based on Hidden Markov Models

Tomohide Shibata
Graduate School of Information Science and Technology, University of Tokyo
7-3-1 Hongo, Bunkyo-ku,
Tokyo, 113-8656, Japan
shibata@kc.t.u-tokyo.ac.jp

Sadao Kurohashi
Graduate School of Informatics,
Kyoto University
Yoshida-honmachi, Sakyo-ku,
Kyoto, 606-8501, Japan
kuro@i.kyoto-u.ac.jp

Abstract

This paper presents an unsupervised topic identification method integrating linguistic and visual information based on Hidden Markov Models (HMMs). We employ HMMs for topic identification, wherein a state corresponds to a topic and various features including linguistic, visual and audio information are observed. Our experiments on two kinds of cooking TV programs show the effectiveness of our proposed method.

1 Introduction

Recent years have seen the rapid increase of multimedia contents with the continuing advance of information technology. To make the best use of multimedia contents, it is necessary to segment them into meaningful segments and annotate them. Because manual annotation is extremely expensive and time consuming, automatic annotation technique is required.

In the field of video analysis, there have been a number of studies on shot analysis for video retrieval or summarization (highlight extraction) using Hidden Markov Models (HMMs) (e.g., (Chang et al., 2002; Nguyen et al., 2005; Q.Phung et al., 2005)). These studies first segmented videos into shots, within which the camera motion is continuous, and extracted features such as color histograms and motion vectors. Then, they classified the shots based on HMMs into several classes (for baseball sports video, for example, pitch view, running overview or audience view). In these studies, to achieve high accuracy, they relied on handmade domain-specific knowledge or trained HMMs with manually labeled data. Therefore, they cannot be easily extended to new domains on a large scale. In addition, although linguistic information, such as narration, speech of characters, and commentary, is intuitively useful for shot analysis, it is not utilized by many of the previous studies. Although some studies attempted to utilize linguistic information (Jasinschi et al., 2001; Babaguchi and Nitta, 2003), it was just keywords.

In the field of Natural Language Processing, Barzilay and Lee have recently proposed a probabilistic content model for representing topics and topic shifts (Barzilay and Lee, 2004). This content model is based on HMMs wherein a state corresponds to a topic and generates sentences relevant to that topic according to a state-specific language model, which are learned from raw texts via analysis of word distribution patterns.

In this paper, we describe an unsupervised topic identification method integrating linguistic and visual information using HMMs. Among several types of videos, in which instruction videos (*how-to* videos) about sports, cooking, D.I.Y., and others are the most valuable, we focus on cooking TV programs. In an example shown in Figure 1, *preparation*, *sauteing*, and *dishing up* are automatically labeled in sequence. Identified topics lead to video segmentation and can be utilized for video summarization.

Inspired by Barzilay's work, we employ HMMs for topic identification, wherein a state corresponds to a topic, like *preparation* and *frying*, and various features, which include visual and audio information as well as linguistic information (instructor's utterances), are observed. This study considers a clause as an unit of analysis and the following eight topics as a set of states: *preparation*, *sauteing*, *frying*, *baking*, *simmering*, *boiling*, *dishing up*, *steaming*.

In Barzilay's model, although domain-specific

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 755–762,
Sydney, July 2006.

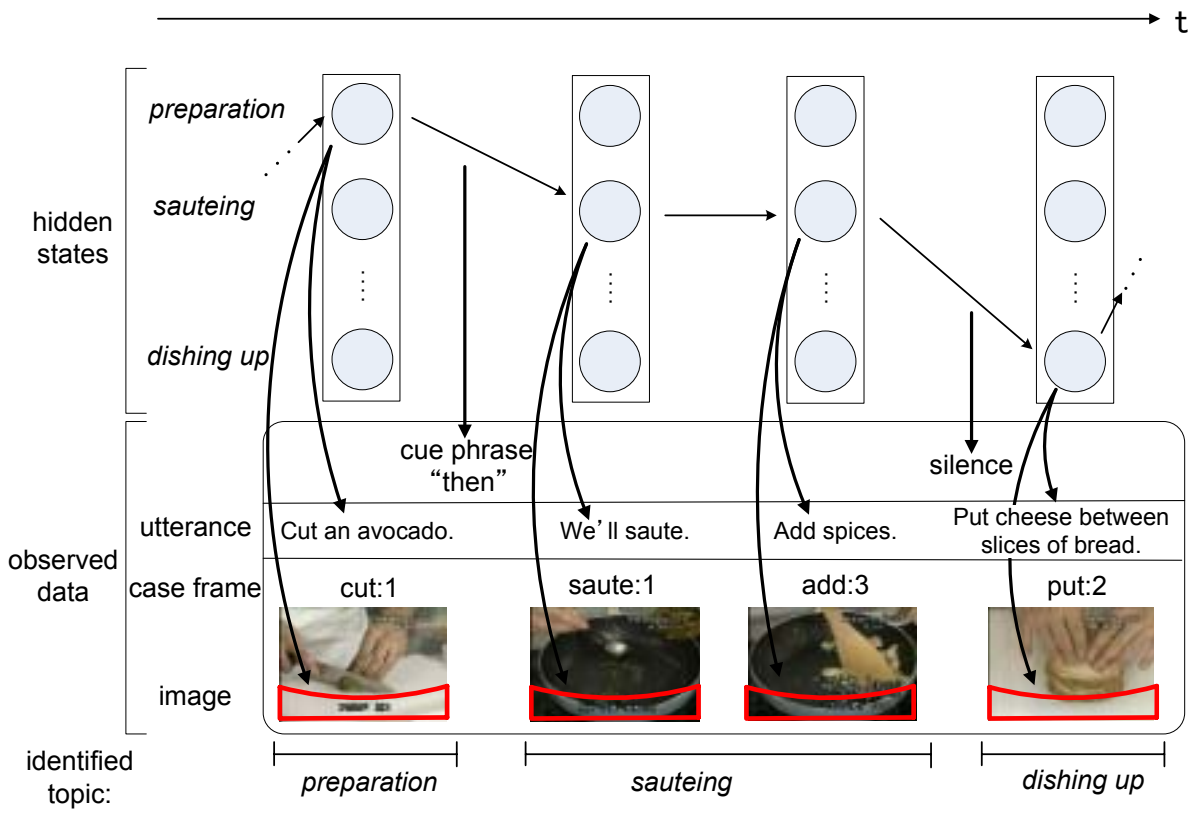


Figure 1: Topic identification with Hidden Markov Models.

word distribution can be learned from raw texts, their model cannot utilize discourse features, such as cue phrases and lexical chains. We incorporate domain-independent discourse features such as cue phrases, noun/verb chaining, which indicate topic change/persistence, into the domain-specific word distribution.

Our main claim is that we utilize visual and audio information to achieve robust topic identification. As for visual information, we can utilize background color distribution of the image. For example, *frying* and *boiling* are usually performed on a gas range and *preparation* and *dishing up* are usually performed on a cutting board. This information can be an aid to topic identification. As for audio information, silence can be utilized as a clue to a topic shift.

2 Related Work

In Natural Language Processing, text segmentation tasks have been actively studied for information retrieval and summarization. Hearst proposed a technique called TextTiling for subdividing texts into sub-topics (Hearst.M, 1997). This method is based on lexical co-occurrence. Galley et al. presented a domain-independent topic segmentation algorithm for multi-party speech (Galley et al., 2003). This segmentation algorithm uses automatically induced decision rules to combine linguistic features (lexical cohesion and cue phrases) and speech features (silences, overlaps and speaker change). These studies aim just at segmenting a given text, not at identifying topics of segmented texts.

Marcu performed rhetorical parsing in the framework of Rhetorical Structure Theory (RST) based on a discourse-annotated corpus (Marcu, 2000). Although this model is suitable for analyzing local modification in a text, it is difficult for this model to capture the structure of topic transition in the whole text.

In contrast, Barzilay and Lee modeled a content structure of texts within specific domains, such as earthquake and finance (Barzilay and Lee, 2004). They used HMMs wherein each state corresponds to a distinct topic (e.g., in earthquake domain, earthquake magnitude or previous earthquake occurrences) and generates sentences relevant to that topic according to a state-specific language model. Their method first create clusters via complete-link clustering, measuring sentence similarity by the cosine metric using word bigrams as features. They calculate initial probabilities: state s_i specific language model $p_{s_i}(w'|w)$

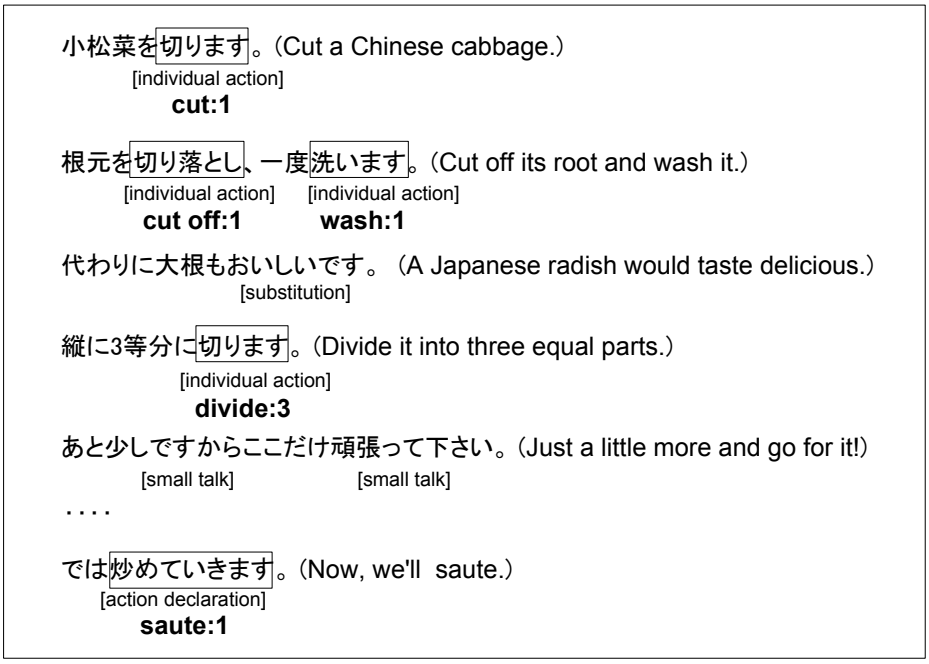


Figure 2: An example of closed captions. (The phrase sandwiched by a square bracket means an utterance type and the word surrounded by a rectangle means an extracted utterance referring to an action. The bold word means a case frame assigned to the verb.)

and state-transition probability $p(s_j|s_i)$ from state s_i to state s_j. Then, they continue to estimate HMM parameters with the Viterbi algorithm until the clustering stabilizes. They applied the constructed content model to two tasks: information ordering and summarization. We differ from this study in that we utilize multimodal features and domain-independent discourse features to achieve robust topic identification.

In the field of video analysis, there have been a number of studies on shot analysis with HMMs. Chang et al. described a method for classifying shots into several classes for highlight extraction in baseball games (Chang et al., 2002). Nguyen et al. proposed a robust statistical framework to extract highlights from a baseball video (Nguyen et al., 2005). They applied multi-stream HMMs to control the weight among different features, such as principal component features capturing color information and frame-difference features for moving objects. Phung et al. proposed a probabilistic framework to exploit hierarchy structure for topic transition detection in educational videos (Q.Phung et al., 2005).

Some studies attempted to utilize linguistic information in shot analysis (Jasinschi et al., 2001; Babaguchi and Nitta, 2003). For example, Babaguchi and Nitta segmented closed caption text into meaningful units and linked them to video streams in sports video. However, linguistic information they utilized was just keywords.

3 Features for Topic Identification

First, we'll describe the features that we use for topic identification, which are listed in Table 1. They consist of three modalities: linguistic, visual and audio modality.

We utilize as linguistic information the instructor's utterances in video, which can be divided into various types such as actions, tips, and even small talk. Among them, actions, such as cut, peel and grease a pan, are dominant and supposed to be useful for topic identification and others can be noise.

In the case of analyzing utterances in video, it is natural to utilize visual information as well as linguistic information for robust analysis. We utilize background image as visual information. For example, *frying* and *boiling* are usually performed on a gas range and *preparation* and *dishing up* are usually performed on a cutting board.

Furthermore, we utilize cue phrases and silence as a clue to a topic shift, and noun/verb chaining as a clue to a topic persistence.

We describe these features in detail in the following sections.

3.1 Linguistic Features

Closed captions of Japanese cooking TV programs are used as a source for extracting linguistic fea-

Table 1: Features for topic identification.

Modality	Feature	Domain dependent	Domain independent
linguistic	case frame	utterance generalization	
	cue phrases		topic change
	noun chaining		topic persistence
	verb chaining		topic persistence
visual	background image	bottom of image	
audio	silence		topic change

Table 2: Utterance-type classification. (An underlined phrase means a pattern for recognizing utterance type.)

[action declaration]
ex. さ，では，ステーキにかかります． (Then, we 'll cook a steak)
じゃあ炒めていきましょう． (OK, we'll fry.)
[individual action]
ex. なすはヘタを取ります。 (Cut off a step of this eggplant.)
お鍋にお水を入れます． (Pour water into a pan.)
[food state]
ex. ニンジンの水分がなくなりました． (There is no water in the carrot.)
[note]
ex. 芯は切らないで下さい． (Don't cut this core off.)
[substitution]
ex. 青ねぎでも結構です． (You may use a leek.)
[food/tool presentation]
ex. 今日はこのハンドミキサーを使います． Today, we use this handy mixer.)
[small talk]
ex. こんにちは． (Hello.)

tures. An example of closed captions is shown in Figure 2. We first process them with the Japanese morphological analyzer, JUMAN (Kurohashi et al., 1994), and make syntactic/case analysis and anaphora resolution with the Japanese analyzer, KNP (Kurohashi and Nagao, 1994). Then, we perform the following process to extract linguistic features.

3.1.1 Extracting Utterances Referring to Actions

Considering a clause as a basic unit, utterances referring to an action are extracted in the form of case frame, which is assigned by case analysis. This procedure is performed for generalization and word sense disambiguation. For example, "塩を入れる (add salt)" and "砂糖を鍋に入れる (add sugar into a pan)" are assigned to case frame ireru:1 (add) and "包丁を入れる (carve with a knife)" is assigned to case frame ireru:2 (carve). We describe this procedure in detail below.

Utterance-type recognition

To extract utterances referring to actions, we classify utterances into several types listed in Table 2[1]. Note that actions are supposed to have two levels: [action declaration] means a declaration of beginning a series of actions and [individual action] means an action that is the finest one.

[1]In this paper, [ ] means an utterance type.

Input sentences are first segmented into clauses and their utterance type is recognized. Among several utterance types, [individual action], [food/tool presentation], [substitution], [note], and [small talk] can be recognized by clause-end patterns. We prepare approximately 500 patterns for recognizing the utterance type. As for [individual action] and [food state], considering the portability of our system, we use general rules regarding intransitive verbs or adjective + "なる (become)" as [food state], and others as [individual action].

Action extraction

We extract utterances whose utterance type is recognized as action ([action declaration] or [individual action]). For example, "むく (peel)" and "切る (cut)" are extracted from the following sentence.

(1) にんじんは皮をむき [individual action] 半分の長さに切ります [individual action]。(We peel this carrot and cut it in half.)

We make two exceptions to reduce noises. One is that clauses are not extracted from the sentence in which sentence-end clause's utterance-type is not recognized as an action. In the following example, "煮る (simmer)" and "切る (cut)" are not extracted because the utterance type of

Table 3: An example of the automatically constructed case frame.

Verb	Case marker	Examples
kiru:1 (cut)	*ga*	<agent>
	wo	pork, carrot, vegetable, · · ·
	ni	rectangle, diamonds, · · ·
kiru:2 (drain)	*ga*	<agent>
	wo	damp · · ·
	no	eggplant, bean curd, · · ·
ireru:1 (add)	*ga*	<agent>
	wo	salt, oil, vegetable, · · ·
	ni	pan, bowl, · · ·
ireru:2 (carve)	*ga*	<agent>
	wo	knife · · ·
	ni	fish · · ·

the sentence-end clause is recognized as [substitution].

(2) 煮てから [individual action] 切っても [individual action] 構いません [substitution]。(It doesn't matter if you cut it after simmering.)

The other is that conditional/causal clauses are not extracted because they sometimes refer to the previous/next topic.

(3) 切りましたら 炒めていきます。(After we finish cutting it, we'll fry.)

(4) プチトマトは油で 揚げるので、切り込みを入れます。(We cut in this cherry tomato, because we'll fry it in oil.)

Note that relations between clauses are recognized by clause-end patterns.

Verb sense disambiguation by assigning to a case frame

In general, a verb has multiple meanings/usages. For example, "入れる" has multiple usages, "塩を入れる (add salt)" and "包丁を入れる (carve with a knife)" , which appear in different topics. We do not extract a surface form of verb but a case frame, which is assigned by case analysis. Case frames are automatically constructed from Web cooking texts (12 million sentences) by clustering similar verb usages (Kawahara and Kurohashi, 2002). An example of the automatically constructed case frame is shown in Table 3. For example, "塩を入れる (add salt)" is assigned to ireru:1 (add) and "包丁を入れる (carve with a knife)" is assigned to case frame ireru:2 (carve).

3.1.2 Cue phrases

As Grosz and Sidner (Grosz and Sidner, 1986) pointed out, cue phrases such as *now* and *well* serve to indicate a topic change. We use approximately 20 domain-independent cue phrases, such as "では (then)", "次は (next)" and "そうしましたら (then)".

3.1.3 Noun Chaining

In text segmentation algorithms such as TextTiling (Hearst.M, 1997), lexical chains are widely utilized for detecting a topic shift. We utilize such a feature as a clue to topic persistence.

When two continuous actions are performed to the same ingredient, their topics are often identical. For example, because "おろす (grate)" and "上げる (raise)" are performed to the same ingredient "かぶら (turnip)" , the topics (in this instance, *preparation*) in the two utterances are identical.

(5) a. かぶらをおろし金でおろしていきます。(We'll grate a turnip.)
b. おろしたかぶらをざるに上げます。(Raise this turnip on this basket.)

However, in the case of spoken language, because there exist many omissions, it is often the case that noun chaining cannot be detected with surface word matching. Therefore, we detect noun chaining by using the anaphora resolution result[2] of verbs (ex.(6)) and nouns (ex.(7)). The verb, noun anaphora resolution is conducted by the method proposed by (Kawahara and Kurohashi, 2004), (Sasano et al., 2004), respectively.

(6) a. キャベツを切ります。 (Cut a cabbage.)
b. 一度 [キャベツを] 洗います。 (Wash it once.)

(7) a. にんじんを大体４ｃｍくらい切ります。(Slice a carrot into 4-cm pieces.)
b. [にんじんの] 皮をぐるっとむきます。(Peel its skin.)

3.1.4 Verb Chaining

When a verb of a clause is identical with that of the previous clause, they are likely to have the same topic. We utilize the fact that the adjoining two clauses contain an identical verbs or not as an observed feature.

(8) a. とうがらしを入れて下さい。(Add some red peppers.)

[2] [ ] indicates an element complemented with anaphora resolution.

b. 鶏手羽を入れます。 (Add chicken wings.)

3.2 Image Features

It is difficult for the current image processing technique to extract what object appears or what action is performing in video unless a detailed object/action model for a specific domain is constructed by hand. Therefore, referring to (Hamada et al., 2000), we focus our attention on color distribution at the bottom of the image, which is comparatively easy to exploit. As shown in Figure 1, we utilize the mass point of RGB in the bottom of the image at each clause.

3.3 Audio Features

A cooking video contains various types of audio information, such as instructor's speech, cutting sounds and frizzling sound. If cutting sound or frizzling sound could be distinguished from other sounds, they could be an aid to topic identification, but it is difficult to recognize them.

As Galley et al. (Galley et al., 2003) pointed out, a longer silence often appears when topic changes, and we can utilize it as a clue to topic change. In this study, silence is automatically extracted by finding duration below a certain amplitude level which lasts more than one second.

4 Topic Identification based on HMMs

We employ HMMs for topic identification, where a hidden state corresponds to a topic and various features described in Section 3 are observed. In our model, considering the case frame as a basic unit, the case frame and background image are observed from the state, and discourse features indicating to topic shift/persistence (cue phrases, noun/verb chaining and silence) are observed when the state transits.

4.1 Parameters

HMM parameters are as follows:

- initial state distribution π_i : the probability that state s_i is a start state.
- state transition probability a_{ij} : the probability that state s_i transits to state s_j.
- observation probability $b_{ij}(o_t)$: the probability that symbol o_t is emitted when state s_i transits to state s_j. This probability is given by the following equation:

$$b_{ij}(o_t) = b_j(cf_k) \cdot b_j(R,G,B) \cdot b_{ij}(discourse\ features) \quad (1)$$

- **case frame** $b_j(cf_k)$: the probability that case frame cf_k is emitted by state s_j.

- **background image** $b_j(R,G,B)$: the probability that background image $b_j(R,G,B)$ is emitted by state s_j. The emission probability is modeled by a single Gaussian distribution with mean (R_j,G_j,B_j) and variance σ_j.

- **discourse features** : the probability that discourse features are emitted when state s_i transits to state s_j. This probability is defined as multiplication of the observation probability of each feature (cue phrase, noun chaining, verb chaining, silence). The observation probability of each feature does not depend on state s_i and s_j, but on whether s_i and s_j are the same or different. For example, in the case of cue phrase (c), the probability is given by the following equation:

$$b_{ij}(c) = \begin{cases} p_{same}(c)(i=j) \\ p_{diff}(c)(i \neq j) \end{cases} \quad (2)$$

4.2 Parameters Estimation

We apply the Baum-Welch algorithm for estimating these parameters. To achieve high accuracy with the Baum-Welch algorithm, which is an unsupervised learning method, some labeled data have been required or proper initial parameters have been set depending on domain-specific knowledge. These requirements, however, make it difficult to extend to other domains. We automatically extract "pseudo-labeled" data focusing on the following linguistic expressions: if a clause has the utterance-type [action declaration] and an original form of its verb corresponds to a topic, its topic is set to that topic. Remind that [action declaration] is a kind of declaration of starting a series of actions. For example, in Figure 1, the topic of the clause "We'll saute." is set to *sauteing* because its utterance-type is recognized as [action declaration] and the original form of its verb is topic *sauteing*.

By using a small amounts of "pseudo-labeled" data as well as unlabeled data, we train the HMM parameters. Once the HMM parameters are trained, the topic identification is performed using the standard Viterbi algorithm.

5 Experiments and Discussion

5.1 Data

To demonstrate the effectiveness of our proposed method, we made experiments on two kinds of cooking TV programs: NHK "Today's Cooking"

Table 5: Experimental result of topic identification.

Features				Accuracy	
case frame	background image	discourse features	silence	"Today's Cooking"	"Kewpie 3-Min Cooking"
√				61.7%	66.4%
	√			56.8%	72.9%
√	√			69.9%	77.1%
√	√	√		**70.5%**	**82.9%**
√	√	√	√	70.5%	82.9%

Table 4: Characteristics of the two cooking programs we used for our experiments.

Program	Today's Cooking	Kewpie 3-Min Cooking
Videos	200	70
Duration	25min	10min
# of utterances per video	249.4	183.4

and NTV "Kewpie 3-Min Cooking". Table 4 presents the characteristics of the two programs. Note that time stamps of closed captions synchronize themselves with the video stream. Extracted "pseudo-labeled" data by the expression mentioned in Section 4.2 are 525 clauses out of 13564 (3.87%) in "Today's Cooking", and 107 clauses out of 1865 (5.74%) in "Kewpie 3-Min Cooking".

5.2 Experiments and Discussion

We conducted the experiment of the topic identification. We first trained HMM parameters for each program, and then applied the trained model to five videos each, in which, we manually assigned appropriate topics to clauses. Table 5 gives the evaluation results. The unit of evaluation was a clause. The accuracy was improved by integrating linguistic and visual information compared to using linguistic / visual information alone. (Note that "visual information" uses pseudo-labeled data.) In addition, the accuracy was improved by using various discourse features. The reason why silence did not contribute to accuracy improvement is supposed to be that closed captions and video streams were not synchronized precisely due to time lagging of closed captions. To deal with this problem, an automatic closed caption alignment technique (Huang et al., 2003) will be applied or automatic speech recognition will be used as texts instead of closed captions with the advance of speech recognition technology.

Figure 3 illustrates an improved example by adding visual information. In the case of using only linguistic information, this topic was recognized as *sauteing*, but this topic was actually *preparation*, which referred to the next topic. By using the visual information that background color was white, this topic was correctly recognized as *preparation*.

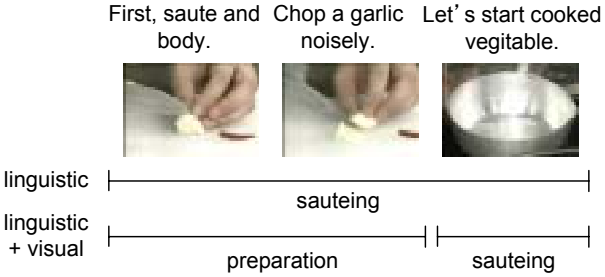


Figure 3: An improved example by adding visual information.

We conducted another experiment to demonstrate the validity of several linguistic processes, such as utterance-type recognition and word sense disambiguation with case frames, for extracting linguistic information from closed captions described in Section 3.1.1. We compared our method to three methods: a method that does not perform word sense disambiguation with case frames (**w/o cf**), a method that does not perform utterance-type recognition for extracting actions (uses all utterance-type texts) (**w/o utype**), a method, in which a sentence is emitted according to a state-specific language model (bigram) as Barzilay and Lee adopted (**bigram**). Figure 6 gives the experimental result, which demonstrates our method is appropriate.

One cause of errors in topic identification is that some case frames are incorrectly constructed. For example, *kiru*:1 (cut) contains "野菜を切る (cut a vegetable)" and "油を切る (drain oil)". This leads to incorrect parameter training. Other cause is that some verbs are assigned to an inaccurate case frame by the failure of case analysis.

6 Conclusions

This paper has described an unsupervised topic identification method integrating linguistic and visual information based on Hidden Markov Mod-

Table 6: Results of the experiment that compares our method to three methods.

Method	Accuracy	
	"Today's Cooking"	"Kewpie 3-Min Cooking"
proposed method	**61.7%**	**66.4%**
w/o cf	57.1%	60.0%
w/o utype	61.7%	62.1%
bigram	54.7%	59.3%

els. Our experiments on the two kinds of cooking TV programs showed the effectiveness of integration of linguistic and visual information and incorporation of domain-independent discourse features to domain-dependent features (case frame and background image).

We are planning to perform object recognition using the automatically-constructed object model and utilize the object recognition results as a feature for HMM-based topic identification.

References

Noboru Babaguchi and Naoko Nitta. 2003. Intermodal collaboration: A strategy for semantic content analysis for broadcasted sports video. In *Proceedings of IEEE International Conference on Image Processing(ICIP2003)*, pages 13–16.

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *Proceedings of the NAACL/HLT*, pages 113–120.

Peng Chang, Mei Han, and Yihong Gong. 2002. Extract highlights from baseball game video with hidden markov models. In *Proceedings of the International Conference on Image Processing 2002(ICIP2002)*, pages 609–612.

Michel Galley, Kathleen McKeown, Eric Fosler-Lussier, and Hongyan Jing. 2003. Discourse segmentation of multi-party conversation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 562–569, 7.

Barbara J. Grosz and Candace L. Sidner. 1986. Attention, intentions, and the structure of discourse. *Computational Linguistic*, 12:175–204.

Reiko Hamada, Ichiro Ide, Shuichi Sakai, and Hidehiko Tanaka. 2000. Associating cooking video with related textbook. In *Proceedings of ACM Multimedia 2000 workshops*, pages 237–241.

Hearst.M. 1997. TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64, March.

Chih-Wei Huang, Winston Hsu, and Shin-Fu Chang. 2003. Automatic closed caption alignment based on speech recognition transcripts. Technical report, Columbia ADVENT.

Radu Jasinschi, Nevenka Dimitrova, Thomas McGee, Lalitha Agnihotri, John Zimmerman, and Dongge. 2001. Integrated multimedia processing for topic segmentation and classification. In *Proceedings of IEEE International Conference on Image Processing(ICIP2003)*, pages 366–369.

Daisuke Kawahara and Sadao Kurohashi. 2002. Fertilization of case frame dictionary for robust japanese case analysis. In *Proceedings of 19th COLING (COLING02)*, pages 425–431.

Daisuke Kawahara and Sadao Kurohashi. 2004. Zero pronoun resolution based on automatically constructed case frames and structural preference of antecedents. In *Proceedings of The 1st International Joint Conference on Natural Language Processing*, pages 334–341.

Sadao Kurohashi and Makoto Nagao. 1994. A syntactic analysis method of long japanese sentences based on the detection of conjunctive structures. *Computational Linguistics*, 20(4).

Sadao Kurohashi, Toshihisa Nakamura, Yuji Matsumoto, and Makoto Nagao. 1994. Improvements of Japanese morphological analyzer JUMAN. In *Proceedings of the International Workshop on Sharable Natural Language*, pages 22–28.

Daniel Marcu. 2000. The rhetorical parsing of unrestricted texts: A surface-based approach. *Computational Linguistics*, 26(3):395–448.

Huu Bach Nguyen, Koichi Shinoda, and Sadaoki Furui. 2005. Robust highlight extraction using multi-stream hidden markov models for baseball video. In *Proceedings of the International Conference on Image Processing 2005(ICIP2005)*, pages 173–176.

Dinh Q.Phung, Thi V.T Duong, Hung H.Bui, and S.Venkatesh. 2005. Topic transition detection using hierarchical hidden markov and semi-markov models. In *Proceedings of ACM International Conference on Multimedia(ACM-MM05)*, pages 6–11.

Ryohei Sasano, Daisuke Kawahara, and Sadao Kurohashi. 2004. Automatic construction of nominal case frames and its application to indirect anaphora resolution. In *Proceedings of the 20th International Conference on Computational Linguistics*, number 1201–1207, 8.

Exact Decoding for Jointly Labeling and Chunking Sequences

Nobuyuki Shimizu
Department of Computer Science
State University of New York at Albany
Albany, NY 12222, USA
nobuyuki@shimizu.name

Andrew Haas
Department of Computer Science
State University of New York at Albany
Albany, NY 12222 USA
haas@cs.albany.edu

Abstract

There are two decoding algorithms essential to the area of natural language processing. One is the Viterbi algorithm for linear-chain models, such as HMMs or CRFs. The other is the CKY algorithm for probabilistic context free grammars. However, tasks such as noun phrase chunking and relation extraction seem to fall between the two, neither of them being the best fit. Ideally we would like to model entities and relations, with two layers of labels. We present a tractable algorithm for exact inference over two layers of labels and chunks with time complexity $O(n^2)$, and provide empirical results comparing our model with linear-chain models.

1 Introduction

The Viterbi algorithm and the CKY algorithms are two decoding algorithms essential to the area of natural language processing. The former models a linear chain of labels such as part of speech tags, and the latter models a parse tree. Both are used to extract the best prediction from the model (Manning and Schutze, 1999).

However, some tasks seem to fall between the two, having more than one layer but flatter than the trees created by parsers. For example, in relation extraction, we have entities in one layer and relations between entities as another layer. Another task is shallow parsing. We may want to model part-of-speech tags and noun/verb chunks at the same time, since performing simultaneous labeling may result in increased joint accuracy by sharing information between the two layers of labels.

To apply the Viterbi decoder to such tasks, we need two models, one for each layer. We must feed the output of one layer to the next layer. In such an approach, errors in earlier processing nearly always accumulate and produce erroneous results at the end. If we use CKY, we usually end up flattening the output tree to obtain the desired output. This seems like a round-about way of modeling two layers.

There are previous attempts at modeling two layer labeling. Dynamic Conditional Random Fields (DCRFs) by (McCallum et al, 2003; Sutton et al, 2004) is one such attempt, however, exact inference is in general intractable for these models and the authors were forced to settle for approximate inference.

Our contribution is a novel model for two layer labeling, for which exact decoding is tractable. Our experiments show that our use of label-chunk structures results in significantly better performance over cascaded CRFs, and that the model is a promising alternative to DCRFs.

The paper is organaized a follows: In Section 2 and 3, we describe the model and present the decoding algorithm. Section 4 describes the learning methods applicable to our model and the baseline models. In Section 5 and 6, we describe the experiments and the results.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 763–770,
Sydney, July 2006.

Token	POS	NP
U.K.	JADJ	B
base	NOUN	I
rates	NOUN	I
are	VERB	O
at	OTHER	O
their	OTHER	B
highest	JADJ	I
level	NOUN	I
in	OTHER	O
eight	OTHER	B
years	NOUN	I
.	OTHER	O

Table 1: Example with POS and NP tags

Token	First Layer (POS)	Second Layer (NP)
U.K.	$\langle I, \textit{JADJ}, 0\rangle$	
	$\langle I, \textit{JADJ}, \textit{NOUN}, 0, 1\rangle$	
base	$\langle I, \textit{NOUN}, 1\rangle$	
	$\langle I, \textit{NOUN}, \textit{NOUN}, 1, 2\rangle$	
rates	$\langle I, \textit{NOUN}, 2\rangle$	$\langle I, 0, 2\rangle$
		$\langle I, O, 2, 3\rangle$
are	$\langle O, \textit{VERB}, 3\rangle$	
	$\langle O, \textit{VERB}, \textit{OTHER}, 3, 4\rangle$	
at	$\langle O, \textit{OTHER}, 4\rangle$	$\langle O, 3, 4\rangle$
		$\langle O, I, 4, 5\rangle$
their	$\langle I, \textit{OTHER}, 5\rangle$	
	$\langle I, \textit{OTHER}, \textit{JADJ}, 5, 6\rangle$	
highest	$\langle I, \textit{JADJ}, 6\rangle$	
	$\langle I, \textit{JADJ}, \textit{NOUN}, 6, 7\rangle$	
level	$\langle I, \textit{NOUN}, 7\rangle$	$\langle I, 5, 7\rangle$
		$\langle I, O, 7, 8\rangle$
in	$\langle O, \textit{OTHER}, 8\rangle$	$\langle O, 8, 8\rangle$
		$\langle O, I, 8, 9\rangle$
eight	$\langle I, \textit{OTHER}, 9\rangle$	
	$\langle I, \textit{OTHER}, \textit{NOUN}, 9, 10\rangle$	
years	$\langle I, \textit{NOUN}, 10\rangle$	$\langle I, 9, 10\rangle$
		$\langle I, O, 10, 11\rangle$
.	$\langle O, \textit{OTHER}, 11\rangle$	$\langle O, 11, 11\rangle$

Table 2: Example Parts

2 Model for Joint Labeling and Chunking

Consider the task of finding *noun chunks*. The noun chunk extends from the beginning of a noun phrase to the head noun, excluding postmodifiers (which are difficult to attach correctly). Table 1 shows a sentence labeled with POS tags and segmented into noun chunks. B marks the first word of a noun chunk, I the other words in a noun chunk, and O the words that are not in a noun chunk. Note that we collapsed the 45 different POS labels into 5 labels, following (McCallum et al, 2003). All different types of adjectives are labeled as JADJ.

Each word carries two tags. Given the first layer, our aim is to present a model that can predict the second and third layers of tags at the same time. Assume we have n training samples, $\{(x^i, y^i)\}_{i=1}^n$, where x^i is a sequence of input tokens and y^i is a label-chunk structure for x^i. In this example, the first column contains the tokens x^i and the second and third columns together represent the label-chunk structures y^i. We will present an efficient exact decoding for this structure.

The label-chunk structure, shown in Table 2, is a representation of the two layers of tags. The tuples in Table 2 are called *parts*. If the token at index r carries a POS tag P and a chunk tag C, the first layer includes part $\langle C, P, r\rangle$. This part is called a *node*. If the tokens at index $r-1$ and r are in the same chunk, and C is the label of that chunk, the first layer also includes part $\langle C, P0, P, r-1, r\rangle$ (where $P0$ and P are the POS tags of the tokens at $r-1$ and r respectively). This part is called a *transition*. If a chunk tagged C extends from the token at q to the token at r inclusive, the second layer includes part $\langle C, q, r\rangle$. This part is a *chunk node*. And if the token at $q-1$ is the last token in a chunk tagged $C0$, while the token at q is the first token of a chunk tagged C, the second layer includes part $\langle C0, C, q-1, q\rangle$. This part is a *chunk transition*.

In this paper we use the common method of factoring the score of the label-chunk structure as the sum of the scores of all the parts. Each part in a label-chunk structure can be lexicalized, and gives rise to several features. For each feature, we have a corresponding weight. If we sum up the weights for these features, we have the score for the part, and if we sum up the scores of the parts, we have the score for the label-chunk structure.

Suppose we would like to score a pair (x^i, y^i) in the training set, and it happens to be the one shown in Table 2. To begin, let's say we would like to find the features for the part $\langle I, \textit{NOUN}, 7\rangle$ of POS node type (1st Layer). This is the NOUN tag on the seventh token "level" in Table 2. By default, the POS node type generates the following binary feature.

- Is there a token labeled with "NOUN" in a chunk labeled with "I"?

Now, to have more features, we can lexicalize POS node type. Suppose we use x_r to lexicalize POS node $\langle C, P, r\rangle$, then we have the following binary feature, as it is $\langle I, \mathit{NOUN}, 7\rangle$ and x_7^i = "level".

- Is there a token "level" labeled with "NOUN" in a chunk labeled with "I"?

We can also use x_{r-1} and x_r to lexicalize the parts of POS node type.

- Is there a token "level" labeled with "NOUN" in a chunk labeled with "I" that's preceded by "highest"?

This way, we have a complete specification of the feature set given the part type, lexicalization for each part type and the training set. Let us define $\mathbf{f}$ a boolean feature vector function such that each dimension of $\mathbf{f}(x^i, y^i)$ contains 1 if the pair (x^i, y^i) has the feature, 0 otherwise. Now define a real-valued weight vector $\mathbf{w}$ with the same dimension as $\mathbf{f}$. To represent the score of the pair (x^i, y^i), we write $s(x^i, y^i) = \mathbf{w}^\top \mathbf{f}(x^i, y^i)$ We could also have $\mathbf{w}^\top \mathbf{f}(x^i, \{p\})$ where p just a single part, in which case we just write $s(p)$.

Assuming an appropriate feature representation as well as a weight vector $\mathbf{w}$, we would like to find the highest scoring label-chunk structure $y = argmax_{y'}(\mathbf{w}^\top \mathbf{f}(x, y'))$ given an input sentence x.

In the upcoming section, we present a decoding algorithm for the label-chunk structures, and later we give a method for learning the weight vector used in the decoding.

3 Decoding

The decoding algorithm is shown in Figure 1. The idea is to use two tables for dynamic programming: label_table and chunk_table.

Suppose we are examining the current position r, and would like to consider extending the chunk $[q, r-1]$ to $[q, r]$. We need to know the chunk tag C for $[q, r-1]$ and the last POS tag $P0$ at index $r-1$. The array entry label_table$[q][r-1]$ keeps track of this information.

Then we examine how the current chunk is connected with the previous chunk. The array entry chunk_table$[q][C0]$ keeps track of the score of the best label-chunk structure from 0 up to the index q that has the ending chunk tag $C0$. Now checking the chunk transition from $C0$ to C at the index q is simple, and we can record the score of this chunk to chunk_table$[r][C]$, so that the next chunk starting at r can use this information.

In short, we are executing two Viterbi algorithms on the first and second layer at the same time. One extends $[q, r-1]$ to $[q, r]$, considering the node indexed by r (first layer). The other extends $[0, q]$ to $[0, r]$, considering the node indexed by $[q, r]$ (second layer). The dynamic programming table for the first layer is kept in the label_table ($r-1$ and $P0$ are used in the Viterbi algorithm for this layer) and that for the second layer in the chunk_table (q and $C0$ used). The algorithm returns the best score of the label-chunk structure.

To recover the structure, we simply need to maintain back pointers to the items that gave rise to the each item in the dynamic programming table. This is just like maintaining back pointers in the Viterbi algorithm for sequences, or the CKY algorithm for parsing.

The pseudo-code shows that the run-time complexity of the decoding algorithm is O(n^2) unlike that of CFG parsing, O(n^3). Thus the algorithm performs better on long sentences. On the other hand, the constant is c^2p^2 where c is the number of chunk tags and p is the number of POS tags.

4 Learning

4.1 Voted Perceptron

In the CKY and Viterbi decoders, we use the forward-backward or inside-outside algorithm to find the marginal probabilities. Since we don't yet have the inference algorithm to find the marginal probabilities of the parts of a label-chunk structure, we use an online learning algorithm to train the model. Despite this restriction, the voted perceptron is known for its performance (Sha and Pereira, 2003).

The voted perceptron we use is the adaptation of (Freund and Schapire, 1999) to the structured setting. Algorithm 4.1 shows the pseudo code for the training, and the function $update(\mathbf{w}_k, x^i, y^i, y')$ returns $\mathbf{w}_k - \mathbf{f}(x^i, y') + \mathbf{f}(x^i, y^i)$.

Given a training set $\{(x^i y^i)\}_{i=1}^n$ and the epoch number T, Algorithm 4.1 will return a list of

Algorithm 3.1: DECODE(the scoring function $s(p)$)

$score := 0$;
for $q := index\_start$ **to** $index\_end$
  for $length := 1$ **to** $index\_end - q$
    $r := q + length$;
    for each Chunk Tag C
      for each Chunk Tag $C0$
        for each POS Tag P
          for each POS Tag $P0$
            $score := 0$;
            if $(length > 1)$
              #Add the score of the transition from r-2 to r-1. (1st Layer, POS)
              $score := score + s(\langle C, P0, P, r-2, r-1\rangle) + label\_table[q][r-1][C][P0]$;
            #Add the score of the node at r-1. (1st Layer, POS)
            $score := score + s(\langle C, P, r-1\rangle)$;
            if $(score >= label\_table[q][r][C][P])$
              $label\_table[q][r][C][P] := score$;
            #Add the score of the chunk node at [q,r-1]. (2nd Layer, NP)
            $score := score + s(\langle C, q, r-1\rangle)$;
            if $(index\_start < q)$
              #Add the score of the chunk transition from q-1 to q. (2nd Layer, NP)
              $score := score + s(\langle C0, C, q-1, q\rangle) + chunk\_table[q][C0]$;
            if $(score >= chunk\_table[r][C])$
              $chunk\_table[r][C] := score$;
          end for
        end for
      end for
    end for
  end for
end for
$score := 0$;
for each C **in** $chunk\_tags$
  if $(chunk\_table[index\_end][C] >= score)$
    $score := chunk\_table[index\_end][C]$;
    $last\_symbol := C$;
end for
return $(score)$

Note: Since the scoring function $s(p)$ is defined as $\mathbf{w}^\top \mathbf{f}(\mathbf{x}^i, \{p\})$, the input sequence $\mathbf{x}^i$ and the weight vector $\mathbf{w}$ are also the inputs to the algorithm.

Figure 1: Decoding Algorithm

weighted perceptrons $\{(\mathbf{w}_1, c_1), ..(\mathbf{w}_k, c_k)\}$. The final model V uses the weight vector

$$\mathbf{w} = \frac{\sum_{j=1}^{k}(c_j \mathbf{w}_j)}{Tn}$$

(Collins, 2002).

Algorithm 4.1: TRAIN($T, \{(x^i, y^i)\}_{i=1}^n$)

$k := 0$;
$\mathbf{w}_1 := \mathbf{0}$;
$c_1 := 0$;
for $t := 1$ **to** T
  for $i := 1$ **to** n
    $y' := argmax_y(\mathbf{w}_k^\top \mathbf{f}(y, x^i))$
    if $(y' = y^i)$
      $c_k := c_k + 1$;
    else
      $\mathbf{w}_{k+1} := update(\mathbf{w}_k, x^i, y^i, y')$;
      $c_{k+1} := 1$;
      $k := k + 1$;
      $c_k := c_k + 1$;
  end for
end for
return $(\{(\mathbf{w}_1, c_1), ..(\mathbf{w}_k, c_k)\})$

Algorithm 4.2: UPDATE1($\mathbf{w}_k, x^i, y^i, y'$)

return $(\mathbf{w}_k - \mathbf{f}(x^i, y') + \mathbf{f}(x^i, y^i))$

Algorithm 4.3: UPDATE2($\mathbf{w}_k, x^i, y^i, y'$)

$\delta = max(0, min(\frac{l_i(y') - s(x^i, y^i) + s(x^i, y')}{\|\mathbf{f}_i(y^i) - \mathbf{f}_i(y')\|^2}, 1))$;
return $(\mathbf{w}_k - \delta\mathbf{f}(x^i, y') + \delta\mathbf{f}(x^i, y^i))$

4.2 Max Margin

4.2.1 Sequential Minimum Optimization

A max margin method minimizes the regularized empirical risk function with the hard (penalized) margin

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 - \sum_i (s(x^i, y^i) - \max_y (s(x^i, y) - l_i(y)))$$

l_i finds the loss for y with respect to y^i, and it is assumed that the function is decomposable just as y is decomposable to the parts. This equation is equivalent to

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i$$
$$\forall i, y,\ s(x^i, y^i) + \xi_i \geq s(x^i, y) - l_i(y)$$

After taking the Lagrange dual formation, we have

$$\max_{\alpha \geq 0} -\frac{1}{2}\| \sum_{i,y} \alpha_i(y)(\mathbf{f}(x^i, y^i) - \mathbf{f}(x^i, y))\|^2 + \sum_{i,y} \alpha_i(y) l_i(y)$$

$$\text{such that } \sum_y \alpha_i(y) = C$$

and

$$\mathbf{w} = \sum_{i,y} \alpha_i(y)(\mathbf{f}(x^i, y^i) - \mathbf{f}(x^i, y)) \quad (1)$$

This quadratic program can be optimized by bi-coordinate descent, known as Sequential Minimum Optimization. Given an example i and two label-chunk structures y' and y'',

$$d = \frac{l_i(y') - l_i(y'') - (s(x^i, y'') - s(x^i, y'))}{\|\mathbf{f}_i(y'') - \mathbf{f}_i(y')\|^2} \quad (2)$$

$$\delta = max(-\alpha_i(y'), min(d, \alpha_i(y'')))$$

The updated values are : $\alpha_i(y') := \alpha_i(y') + \delta$ and $\alpha_i(y'') := \alpha_i(y'') - \delta$.

Using the equation (1), any increase in α can be translated to $\mathbf{w}$. For a naive SMO, this update is executed for each training sample i, for all pairs of possible parses y' and y'' for x^i. See (Taskar and Klein, 2005; Zhang, 2001; Jaakkola et al, 2000).

Here is where we differ from (Taskar et al, 2004). We choose y'' to be the correct parse y^i, and y' to be the best runner-up. After setting the initial weights using y^i, we also set $\alpha_i(y^i) = 1$ and $\alpha_i(y') = 0$. Although these alphas are not correct, as optimization nears the end, the margin is wider; $\alpha_i(y^i)$ and $\alpha_i(y')$ gets closer to 1 and 0 respectively. Given this approximation, we can compute δ. Then, the function $update(\mathbf{w}_k, x^i, y^i, y')$ will return $\mathbf{w}_k - \delta\mathbf{f}(x^i, y') + \delta\mathbf{f}(x^i, y^i)$ and we have reduced the SMO to the perceptron weight update.

4.2.2 Margin Infused Relaxed Algorithm

We can think of maximizing the margin in terms of extending the Margin Infused Relaxed Algorithm (MIRA) (Crammer and Singer, 2003; Crammer et al, 2003) to learning with structured outputs. (McDonald et al, 2005) presents this approach for dependency parsing.

In particuler, Single-best MIRA (McDonald et al, 2005) uses only the single margin constraint for the runner up y' with the highest score. The resulting online update would be $\mathbf{w}_{k+1}$ with the following

condition: $min\|\mathbf{w}_{k+1} - \mathbf{w}_k\|$ such that $s(x^i, y^i) - s(x^i, y') \geq l_i(y')$ where $y' = argmax_y s(x^i, y)$.

Incidentally, the equation (2) for d above when $\alpha_i(y^i) = 1$ and $\alpha_i(y') = 0$ solves this minimization problem as well, and the weight update is the same as the SMO case.

4.2.3 Conditional Random Fields

Instead of minimizing the regularized empirical risk function with the hard (penalized) margin, conditional random fields try to minimize the same with the negative log loss:

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 - \sum_i (s(x^i, y^i) - log(\sum_y s(x^i, y)))$$

Usually, CRFs use marginal probabilities of parts to do the optimization. Since we have not yet come up with the algorithm to compute marginals for a label-chunk structure, the training methods for CRFs is not applicable to our purpose. However, on sequence labeling tasks CRFs have shown very good performance (Lafferty et al, 2001; Sha and Pereira, 2003), and we will use them for the baseline comparison.

5 Experiments

5.1 Task: Base Noun Phrase Chunking

The data for the training and evaluation comes from the CoNLL 2000 shared task (Tjong Kim Sang and Buchholz, 2000), which is a portion of the Wall Street Journal.

We consider each sentence to be a training instance x^i, with single words as tokens.

The shared task data have a standard training set of 8936 sentences and a test set of 2012 sentences. For the training, we used the first 447 sentences from the standard training set, and our evaluation was done on the standard test set of the 2012 sentences. Let us define the set D to be the first 447 samples from the standard training set .

There are 45 different POS labels, and the three NP labels: begin-phrase, inside-phrase, and other. (Ramshaw and Marcus, 1995) To reduce the inference time, following (McCallum et al, 2003), we collapsed the 45 different POS labels contained in the original data. The rules for collapsing the POS labels are listed in the Table 3.

Original	Collapsed
all different types of nouns	NOUN
all different types of verbs	VERB
all different types of adjectives	JADJ
all different types of adverbs	RBP
the remaining POS labels	OTHER

Table 3: Rules for collapsing POS tags

Token	POS	Collapsed	Chunk	NP
U.K.	JJ	JADJ	B-NP	B
base	NN	NOUN	I-NP	I
rates	NNS	NOUN	I-NP	I
are	VBP	VERB	B-VP	O
at	IN	OTHER	B-PP	O
their	PRP$	OTHER	B-NP	B
highest	JJS	JADJ	I-NP	I
level	NN	NOUN	I-NP	I
in	IN	OTHER	B-PP	O
eight	CD	OTHER	B-NP	B
years	NNS	NOUN	I-NP	I
.	.	OTHER	O	O

Table 4: Example with POS and NP labels, before and after collapsing the labels.

We present two experiments: one comparing our label-chunk model with a cascaded linear-chain model and a simple linear-chain model, and one comparing different learning algorithms.

The cascaded linear-chain model uses one linear-chain model to predict POS tags, and another linear-chain model to predict NP labels, using the POS tags predicted by the first model as a feature.

More specifically, we trained a POS-tagger using the training set D. We then used the learned model and replaced the POS labels of the test set with the labels predicted by the learned model. The linear-chain NP chunker was again trained on D and evaluated on this new test set with POS supplied by the earlier processing. Note that the new test set has exactly the same word tokens and noun chunks as the original test set.

5.2 Systems

5.2.1 POS Tagger and NP Chunker

There are three versions of POS taggers and NP chunkers: CRF, VP, MMVP. For CRF, L-BFGS, a quasi-Newton optimization method was used for the training, and the implementation we used is CRF++ (Kudo, 2005). VP uses voted perceptron, and MMVP uses max margin update for the voted perceptron. For the voted perceptron, we used aver-

if x_q matches	then t_q is
[A-Z][a-z]+	CAPITAL
[A-Z]	CAP_ONE
[A-Z]+	CAP_ALL
[A-Z]+[a-z]+[A-Z]+[a-z]	CAP_MIX
.*[0-9].*	NUMBER

Table 5: Rules to create t_q for each token x_q

First Layer (POS)

Node $\langle C, P, r\rangle$	Trans. $\langle C, P0, P, r-1, r\rangle$
x_{r-1}	x_{r-1}
x_r	x_r
x_{r+1}	
t_r	

Second Layer (NP)

Node $\langle C, q, r\rangle$	Trans. $\langle C0, C, q-1, q\rangle$
x_q	x_{q-1}
x_{q-1}	x_q
x_r	
x_{r+1}	

Table 6: Lexicalized Features for Joint Models

aging of the weights suggested by (Collins, 2002). The features are exactly the same for all three systems.

5.2.2 Cascaded Models

For each CRF, VP, MMVP, the output of a POS tagger was used as a feature for the NP chunker. The feeds always consist of a POS tagger and NP chunker of the same kind, thus we have CRF+CRF, VP+VP, and MMVP+MMVP.

5.2.3 Joint Models

Since CRF requires the computation of marginals for each part, we were not able to use the learning method. VP and MMVP were used to train the label-chunk structures with the features explained in the following section.

5.3 Features

First, as a preprocessing step, for each word token x_q, feature t_q was created with the rule in Table 5, and included in the input files. This feature is included in x along with the word tokens. The feature tells us whether the token is capitalized, and whether digits occur in the token. No outside resources such as a list of names or a gazetteer were used.

Table 6 shows the lexicalized features for the joint labeling and chunking. For the first iteration of training, the weights for the lexicalized features were not updated. The intention is to have more weights on the unlexicalized features, so that when lexical feature is not found, unlexicalized features could provide useful information and avoid overfitting, much as back-off probabilities do.

POS tagging	POS	NP	F1
CRF	91.56%	N/A	N/A
VP	90.55%	N/A	N/A
MMVP	90.02%	N/A	N/A
NP chunking	POS	NP	F1
CRF	given	94.44%	87.52%
VP	given	94.28%	86.96%
MMVP	given	94.17%	86.79%
Both POS & NP	POS	NP	F1
CRF + CRF	above	90.16%	79.08%
VP + VP	above	89.21%	76.26%
MMVP + MMVP	above	88.95%	75.28%
VP Joint	88.42%	90.60%	79.69%
MMVP Joint	88.69%	90.84%	80.34%

Table 7: Performance

6 Result

We evaluated the performance of the systems using three measures: POS accuracy, NP accuracy, and F1 measure on NP. These figures show how errors accumulate as the systems are chained together. For the statistical significance testing, we have used pair-samples t test, and for the joint labeling and chunking task, everything was found to be statistically significant except for CRF + CRF vs VP Joint.

One can see that the systems with joint labeling and chunking models perform much better than the cascaded models. Surprisingly, the perceptron update motivated by the max margin principle performed significantly worse than the simple perceptron update for linear-chain models but performed better on joint labeling and chunking.

Although joint labeling and chunking model takes longer time per sample because of the time complexity of decoding, the number of iteration needed to achieve the best result is very low compared to other systems. The CPU time required to run 10 iterations of MMVP is 112 minutes.

7 Conclusion

We have presented the decoding algorithm for label-chunk structure and showed its effectiveness in finding two layers of information, POS tags and NP chunks. This algorithm has a place between the

POS tagging	Iterations
VP	30
MMVP	40
CRF	126
NP chunking	Iterations
VP	70
MMVP	50
CRF	101
Both POS & NP	Iterations
VP	10
MMVP	10

Table 8: Iterations needed for the result

Viterbi algorithm for linear-chain models and the CKY algorithm for parsing, and the time complexity is O(n^2). The use of our label-chunk structure significantly boosted the performance over cascaded CRFs despite the online learning algorithms used to train the system, and shows itself as a promising alternative to cascaded models, and possibly dynamic conditional random fields for modeling two layers of tags. Further work includes applying the algorithm to relation extraction, and devising an effective algorithm to find the marginal probabilities of parts.

References

M. Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proc. of Empirical Methods in Natural Language Processing (EMNLP)*

K. Crammer and Y. Singer. 2003. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*

K. Crammer, O. Dekel, S. Shalev-Shwartz, and Y. Singer. 2003. Online passive aggressive algorithms. In *Advances in Neural Information Processing Systems 15*

K. Crammer, R. McDonald, and F. Pereira. 2004. New large margin algorithms for structured prediction. In *Learning with Structured Outputs Workshop (NIPS)*

Y. Freund and R. Schapire 1999. Large Margin Classification using the Perceptron Algorithm. In *Machine Learning*, 37(3):277-296.

T.S. Jaakkola, M. Diekhans, and D. Haussler. 2000. A discriminative framework for detecting remote protein homologies. *Journal of Computational Biology*

T. Kudo 2005. CRF++: Yet Another CRF toolkit. Available at *http://chasen.org/˜taku/software/CRF++/*

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proc. of the 18th International Conference on Machine Learning (ICML)*

F. Peng and A. McCallum. 2004. Accurate Information Extraction from Research Papers using Conditional Random Fields. In *Proc. of the Human Language Technology Conf. (HLT)*

F. Sha and F. Pereira. 2003. Shallow parsing with conditional random fields. In *Proc. of the Human Language Technology Conf. (HLT)*

C. Manning and H. Schutze. 1999. *Foundations of Statistical Natural Language Processing* MIT Press.

A. McCallum, K. Rohanimanesh and C. Sutton. 2003. Dynamic Conditional Random Fields for Jointly Labeling Multiple Sequences. In *Proc. of Workshop on Syntax, Semantics, Statistics. (NIPS)*

R. McDonald, K. Crammer, and F. Pereira. 2005. Online large-margin training of dependency parsers. In *Proc. of the 43rd Annual Meeting of the ACL*

L. Ramshaw and M. Marcus. 1995. Text chunking using transformation-based learning. In *Proc. of Third Workshop on Very Large Corpora. ACL*

C. Sutton, K. Rohanimanesh and A. McCallum. 2004. Dynamic Conditional Random Fields: Factorized Probabilistic Models for Labeling and Segmenting Sequence Data. In *Proc. of the 21st International Conference on Machine Learning (ICML)*

B. Taskar, D. Klein, M. Collins, D. Koller, and C. Manning 2004. Max Margin Parsing. In *Proc. of Empirical Methods in Natural Language Processing (EMNLP)*

B. Taskar and D. Klein. 2005. Max-Margin Methods for NLP: Estimation, Structure, and Applications Available at *http://www.cs.berkeley.edu/˜taskar/pubs/max-margin-acl05-tutorial.pdf*

E. F. Tjong Kim Sang and S. Buchholz. 2000. Introduction to the CoNLL-2000 shared task: Chunking. In *Proc. of the 4th Conf. on Computational Natural Language Learning (CoNLL)*

T. Zhang. 2001. Regularized winnow methods. In *Advances in Neural Information Processing Systems 13*

Compiling a Lexicon of Cooking Actions for Animation Generation

Kiyoaki Shirai **Hiroshi Ookawa**
Japan Advanced Institute of Science and Technology
1-1, Asahidai, Nomi, 923-1292, Ishikawa, Japan
{kshirai,h-ookawa}@jaist.ac.jp

Abstract

This paper describes a system which generates animations for cooking actions in recipes, to help people understand recipes written in Japanese. The major goal of this research is to increase the scalability of the system, i.e., to develop a system which can handle various kinds of cooking actions. We designed and compiled the lexicon of cooking actions required for the animation generation system. The lexicon includes the action plan used for animation generation, and the information about ingredients upon which the cooking action is taken. Preliminary evaluation shows that our lexicon contains most of the cooking actions that appear in Japanese recipes. We also discuss how to handle linguistic expressions in recipes, which are not included in the lexicon, in order to generate animations for them.

1 Introduction

The ability to visualize procedures or instructions is important for understanding documents that guide or instruct us, such as computer manuals or cooking recipes. We can understand such documents more easily by seeing corresponding figures or animations. Several researchers have studied the visualization of documents (Coyne and Sproat, 2001), including the generation of animation (Andre and Rist, 1996; Towns et al., 1998). Such animation systems help people to understand instructions in documents. Among the various types of documents, this research focuses on the visualization of cooking recipes.

Many studies related to the analysis or generation of cooking recipes have been done (Adachi, 1997; Webber and Eugenio, 1990; Hayashi et al., 2003; Shibata et al., 2003). Especially, several researchers have proposed animation generation systems in the cooking domain. Karlin, for example, developed SEAFACT (Semantic Analysis For the Animation of Cooking Tasks), which analyzed verbal modifiers to determine several features of an action, such as the aspectual category of an event, the number of repetitions, duration, speed, and so on (Karlin, 1988). Uematsu developed "Captain Cook," which generated animations from cooking recipes written in Japanese (Uematsu et al., 2001). However, these previous works did not mention the scalability of the systems. There are many linguistic expressions in the cooking domain, but it is uncertain to what extent these systems can convert them to animations.

This paper also aims at developing a system to generate animations from cooking recipes written in Japanese. We especially focused on increasing the variety of recipes that could be accepted. After presenting an overview of our proposed system in Subsections 2.1 and 2.2, the more concrete goals of this paper will be described in Subsection 2.3.

2 Proposed System

2.1 Overview

The overview of our animation generation system is as follows. The system displays a cooking recipe in a browser. As in a typical recipe, cooking instructions are displayed step by step, and sentences or phrases representing a cooking action in the recipe are highlighted. When a user does not understand a certain cooking action, he/she can click the highlighted sentence/phrase. Then the system will show the corresponding animation to help the user understand the cooking instruction.

Note that the system does not show all procedures in a recipe like a movie, but generates an animation of a single action on demand. Furthermore, we do not aim at the reproduction of recipe sentences in detail. Especially, we will not prepare object data for many different kinds of ingredients. For example, suppose that the system has object data for a mackerel, but not for a sardine. When a user clicks the sentence "fillet a sardine" to see the animation, the system will show how to fillet a "mackerel" instead of "sardine", with a note indicating that the ingredient is different. We believe

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 771–778,
Sydney, July 2006.

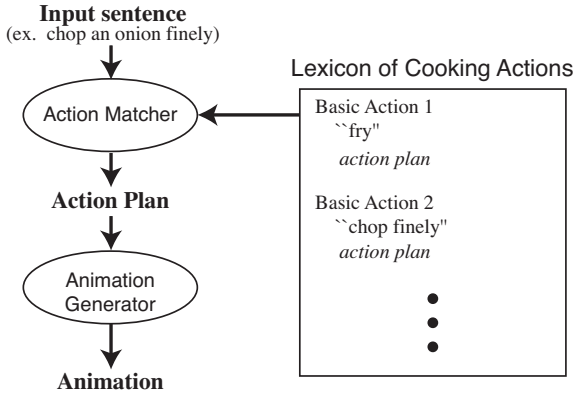


Figure 1: System Architecture

that the user will be more interested in "how to fillet" than in the specific ingredient to be filleted. In other words, the animation of the action will be equally helpful as long as the ingredients are similar. Thus we will not make a great effort to prepare animations for many kinds of ingredients. Instead, we will focus on producing the various kinds of cooking actions, to support users in understanding cooking instructions in recipes.

2.2 System Architecture

Figure 1 illustrates the architecture of the proposed system. First, we prepare the lexicon of cooking actions. This is the collection of cooking actions such as "fry", "chop finely", etc. The lexicon has enough knowledge to generate an animation for each cooking action. Figure 2 shows an example of an entry in the lexicon. In the figure, "*expression*" is a linguistic expression for the action; "*action plan*" is a sequence of action primitives, which are the minimum action units for animation generation. Roughly speaking, the action plan in Figure 2 represents a series of primitive actions, such as cutting and rotating an ingredient, for the basic action "chop finely". The system will generate an animation according to the action plan in the lexicon. Other features, "*ingredient examples*" and "*ingredient requirement*", will be explained later.

The process of generating an animation is as follows. First, as shown in Figure 1, the system compares an input sentence and *expression* of the entries in the lexicon of cooking actions, and finds the appropriate cooking action. This is done by the module "*Action Matcher*". Then, the system extracts an action plan from the lexicon and passes it to the "*Animation Generator*" module. Finally *Animation Generator* interprets the action plan and produces the animation.

2.3 Goal

The major goals of this paper are summarized as follows:

G1. Construct a large-scale lexicon of cooking actions

In order to generate animations for various kinds of cooking actions, we must prepare a lexicon containing many basic actions.

G2. Handle a variety of linguistic expressions

Various linguistic expressions for cooking actions may occur in recipes. It is not realistic to include all possible expressions in the lexicon. Therefore, when a linguistic expression in an input sentence is not included in the lexicon, the system should calculate the similarity between it and the basic action in the lexicon, and find an equivalent or almost similar action.

G3. Include information about acceptable ingredients in the lexicon

Even though linguistic expressions are the same, cooking actions may be different according to the ingredient upon which the action is taken. For example, "cut into fine strips" may stand for several different cooking actions. That is, the action of "cut **cucumber** into fine strips" may be different than "cut **cabbage** into fine strips", because the shapes of cucumber and cabbage are rather different. Therefore, each entry in the lexicon should include information about what kinds of ingredients are acceptable for a certain cooking action.

As mentioned earlier, the main goal of this research is to increase the scalability of the system, i.e., to develop an animation generation system that can handle various cooking actions. We hope that this can be accomplished through goals G1 and G2.

In the rest of this paper, Section 3 describes how to define the set of actions to be compiled into the lexicon of cooking actions. This concerns goal G1. Section 4 explains two major features in the lexicon, "*action plan*" and "*ingredient requirement*". The feature *ingredient requirement* is

Basic Action 2	
expression	みじん切りにする (chop finely)
action plan	cut(*ingredient*,*utensil*,*location*, 2)
	rotate(*ingredient*,*location*, *x*, 90)
	cut(*ingredient*,*utensil*,*location*,20)
	rotate(*ingredient*,*location*, *z*, 90)
	cut2(*ingredient*,*utensil*,*location*, 10)
	cut(*ingredient*,*utensil*,*location*, 20)
ingredient examples	おくら (okra), しいたけ (shiitake mushroom)
ingredient requirement	kind=vegetable|mushroom

Figure 2: Example of an Entry in the Lexicon of Cooking Actions

related to goal G3. Section 5 reports a preliminary survey to construct the module *Action Matcher* in Figure 1, which is related to goal G2. Finally, Section 6 concludes the paper.

3 Defining the Set of Basic Actions

In this and the following sections, we will explain how to construct the lexicon of cooking actions. The first step in constructing the lexicon is to define the set of basic actions. As mentioned earlier (goal G1 in Subsection 2.3), a large-scale lexicon is required for our system. Therefore, the set of basic actions should include various kinds of cooking actions.

3.1 Procedure

We referred to three cooking textbooks or manuals (Atsuta, 2004; Fujino, 2003; Takashiro and Kenmizaki, 2004) in Japanese to define the set of basic actions. These books explain the fundamental cooking operations with pictures, e.g., how to cut, roast, or remove skins/seeds for various kinds of ingredients. We extracted the cooking operations explained in these three textbooks, and defined them as the basic actions for the lexicon. In other words, we defined the basic actions according to the cooking textbooks. The reasons why we used the cooking manuals as the standard for the basic actions are summarized as follows:

1. The aim of cooking manuals used here is to comprehensively explain basic cooking operations. Therefore, we expect that we can collect an exhaustive set of basic actions in the cooking domain.

2. Cooking manuals are for beginners. The aim of animation generation system is to help people, especially novices, to understand cooking actions in recipes. The lexicon of cooking actions based on the cooking textbooks includes many cooking operations that novices may not know well.

3. The definition of basic actions does not depend on the module *Animation Generator*.

   One of the standards for the definition of basic actions is animations generated by the system. That is, we can define basic cooking actions so that each cooking action corresponds to an unique animation. This approach seems to be reasonable for an animation generation system; however, it depends on the module *Animation Generator* in Figure 1. Many kinds of rendering engines are now available to generate animations. Therefore, *Animation Generator* can be implemented in various ways. When changing the rendering engine used in *Animation Generator*, the lexicon of cooking actions must also be changed. So we decided that it would not be desirable to define the set of basic actions according to their corresponding animations.

   In our framework, the definition of basic actions in the lexicon does not depend on *Animation Generator*. This enables us to use any kind of rendering engine to produce an animation. For example, when we use a poor engine and want to design the system so that it generates the same animation for two or more basic actions, we just describe the same *action plan* for these actions.

We manually excerpted 267 basic actions from three cooking textbooks. Although it is just a collection of basic actions, we refer it as the initial

Table 1: Examples of Basic Actions

expression	ingredient examples
三枚におろす (fillet)	あじ (mackerel)
炊き込む (boil)	
炊く (boil)	
くし形切りにする (cut into a comb shape)	トマト (tomato), じゃがいも (potato)
くし形切りにする (cut into a comb shape)	かぼちゃ (pumpkin)
くし形切りにする (cut into a comb shape)	カブ (turnip)

lexicon of cooking actions. Table 1 illustrates several examples of basic actions in the initial lexicon. In the cooking manuals, every cooking operation is illustrated with pictures. "*Ingredient examples*" indicates ingredients in pictures used to explain cooking actions.

3.2 Preliminary Evaluation

A preliminary experiment was conducted to evaluate the scalability of our initial lexicon of basic actions. The aim of this experiment was to check how many cooking actions appearing in real recipes are included in the initial lexicon.

First, we collected 200 recipes which are available on web pages [1]. We refer to this recipe corpus as R_a hereafter. Next, we analyzed the sentences in R_a and automatically extracted verbal phrases representing cooking actions. We used JUMAN [2] for word segmentation and part-of-speech tagging, and KNP [3] for syntactic analysis. Finally, we manually checked whether each extracted verbal phrase could be matched to one of the basic actions in the initial lexicon.

Table 2 (A) shows the result of our survey. The number of basic actions was 267 (a). Among these actions, 145 (54.3%) actions occurred in R_a (a1). About half of the actions in the initial lexicon did not occur in the recipe corpus. We guessed that this was because the size of the recipe corpus was not very large.

The number of verbal phrases in R_a was 3977 (b). We classified them into the following five cases: (b1) the verbal phrase corresponded with one of the basic actions in the initial lexicon, and its linguistic expression was the same as one in the lexicon; (b2) the verbal phrase corresponded with a basic action, but its linguistic expression differed from one in the lexicon; (b3) no corresponding basic action was found in the initial lexicon, (b4) the extracted phrase was not a verbal phrase, caused by error in analysis, (b5) the verbal phrase did not stand for a cooking action. Note that the cases in which verbal phrases should be converted to animations were (b1), (b2) and (b3). The numbers in parentheses (...) indicate the ratio of each case to the total number of verbal phrases, while numbers in square brackets [...] indicate a ratio of each case to the total number of (b1), (b2) and (b3).

We expected that the verbal phrases in (b1) and (b2) could be handled by our animation generation system because the initial lexicon contained the corresponding basic actions. On the other hand, our system cannot generate animations for verbal phrases in (b3), which was 42.3% of the verbal phrases our system should handle. Thus the applicability of the initial lexicon was poor.

3.3 Adding Basic Actions from Recipe Corpus

We have examined what kinds of verbal phrases were in (b3). We found that there were many general verbs, such as "加える (add)", "入れる (put in)", "熱する (heat)", "付ける (attach)", "のせる (put on)", etc. Such general actions were not included in the initial lexicon, because we constructed it by extracting basic actions from cooking textbooks, and such general actions are not explained in these books.

In order to increase the scalability of the lexicon of cooking actions, we selected verbs satisfying the following conditions: (1) no corresponding basic action was found in the lexicon for a verb; (2) a verb occurred more than 10 times in R_a. In all, 31 verbs were found and added to the lexicon as new basic actions. It is undesirable to define basic actions in this way, because the lexicon may then depend on a particular recipe corpus. However, we believe that the new basic actions are very general, and can be regarded as almost independent of with the corpus from which they were extracted.

In order to evaluate the new lexicon, we prepared another 50 cooking recipes (R_b hereafter). Then we classified the verbal phrases in R_b in the same way as in Subsection 3.2. The results are shown in Table 2 (B). Notice that the ratio

[1] http://www.bob-an.com/

[2] http://www.kc.t.u-tokyo.ac.jp/nl-resource/juman.html

[3] http://www.kc.t.u-tokyo.ac.jp/nl-resource/knp.html

Table 2: Result of Preliminary Evaluation

(A) Survey on R_a

(a)	# of basic actions	267
(a1)	basic actions occurred in R_a	145 (54.3%)
(b)	# of verbal phrases	3977
(b1)	basic action(same)	974 (24.5%) [28.0%]
(b2)	basic action(dif.)	1031 (25.9%) [29.7%]
(b3)	not basic action	1469 (36.9%) [42.3%]
(b4)	analysis error	180 (4.5%)
(b5)	not cooking action	323 (8.1%)

(B) Survey on R_b

(a)	298
(a1)	106 (35.6%)
(b)	959
(b1)	521 (54.3%) [62.2%]
(b2)	262 (27.3%) [31.3%]
(b3)	55 (5.7%) [6.6%]
(b4)	45 (4.7%)
(b5)	76 (7.9%)

of the number of verbal phrases contained in the lexicon to the total number of target verb phrases was 94.5% ((b1)62.2% + (b2)31.3%). This is much greater than the ratio in Table 2 (A) (57.7%). Therefore, although the size of test corpus is small, we hope that the scalability of our lexicon is large enough to generate animations for most of the verbal phrases in cooking recipes.

4 Compilation of the Lexicon of Basic Actions

After defining the set of basic actions for the lexicon, the information of each basic action must be described. As shown in Figure 2, the main features in our lexicon are *expression*, *action plan*, *ingredient examples* and *ingredient requirement*. The term *expression* stands for linguistic expressions of basic actions, while *ingredient examples* stands for examples of ingredients described in the cooking manuals we referred to when defining the set of basic actions. As shown in Table 1, these two features have already been included in the initial lexicon created by the procedure in Section 3. This section describes the compilation of the rest of the features: *action plan* in Subsection 4.1 and *ingredient requirement* in Subsection 4.2.

4.1 Action Plan

For each basic action in the lexicon, the action plan to generate the corresponding animation is described. *Action plan* is the sequence of action primitives as shown in Figure 2. Of the 298 basic actions in the lexicon, we have currently described action plans for only 80 actions. Most of them are actions to cut something.

We have also started to develop *Animation Generator* (see Figure 1), which is the module that interprets action plans and generates animations. We used VRML for animation generation. Figure 3 is a snapshot of the animation for the basic action "みじん切りにする (chop finely)" generated by our system.

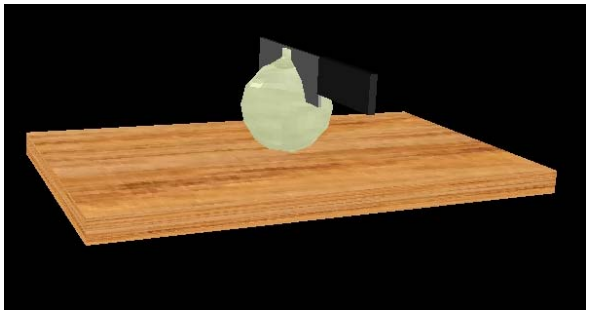


Figure 3: Snapshot of Generated Animation

Our current focus has been on the design and development of the lexicon of cooking actions, rather than on animation generation. Implementation of the complete *Animation Generator* as well as a description of the action plans for all basic actions in the lexicon are important future works.

4.2 Ingredient Requirement

Several basic actions have the same expression in our lexicon. For instance, in Figure 1, there are three basic actions represented by the same linguistic expression "くし形切りにする (cut into a comb shape)". These three actions stand for different cooking actions. The first one stands for the action used to cut something like a "tomato" or "potato" into a comb shape. The second stands for the following sequence of actions: first cut something in half, remove its core or seeds, and cut it into a comb shape. This action is taken on pumpkin, for instance. The third action represents the cooking action for "turnip": remove the leaves of the turnip and cut it into a comb shape. In other words, there are different ways to cut different in-

gredients into a comb shape. Differences among these actions depend on what kinds of ingredients are to be cut.

As described in Section 2.2, the module *Action Matcher* accepts a sentence or phrase for which a user wants to see the animation, then finds a corresponding basic action from the lexicon. In order to find an appropriate basic action for a recipe sentence, the lexicon of cooking actions should include information about what kinds of ingredients are acceptable for each basic action. Note that the judgment as to whether an ingredient is suitable or not highly depends on its features such as kind, shape, and components (seed, peel etc.) of the ingredient. Therefore, the lexicon should include information about what features of the ingredients must be operated upon by the basic actions.

For the above reason, *ingredient requirement* was introduced in the lexicon of cooking actions. In this field, we manually describe the required features of ingredients for each basic action. Figure 4 illustrates the three basic actions of くし形切りにする (chop into a comb shape) in the lexicon [4]. The basic action *a1*, "kind=vegetable, shape=sphere" in *ingredient requirement*, means that only a vegetable whose shape is spherical is acceptable as an ingredient for this cooking action. On the other hand, for the basic action *a2*, only a vegetable whose shape is spherical and containing seeds is acceptable. For *a3*, "instance=カブ (turnip)" means that only a turnip is suitable for this action. In our lexicon, such specific cooking actions are also included when the reference cookbooks illustrate special cooking actions for certain ingredients. In this case, a cookbook illustrates cutting a *turnip* into a comb shape in a different way than for other ingredients.

4.2.1 Feature Set of Ingredient Requirement

Here are all the attributes and possible values prepared for the *ingredient requirement* field:

- kind

  This attribute specifies kinds of ingredients. The possible values are:

  vegetable, mushroom, fruit, meat, fish, shellfish, seafood, condiment

  "Seafood" means seafood other than fish or shellfish, such as イカ (squid), タラコ (cod roe) and so on.

- veg

  This attribute specifies subtypes of vegetables. Possible values for this attribute are "green", "root" and "layer". "Green" stands for green vegetables such as ほうれん草 (spinach) and 白菜 (Chinese cabbage). "Root" stands for root vegetables such as じゃがいも (potato) and ごぼう (burdock). "Layer" stands for vegetables consisting of layers of edible leaves such as レタス (lettuce) and キャベツ (cabbage).

- shape

  This attribute specifies shapes of ingredients. The possible values are:

  sphere, stick, cube, oval, plate, filiform

- peel, seed, core

  These attributes specify various components of ingredients. Values are always 1. For example, "peel=1" stands for ingredients with peel.

- instance

  This specifies a certain ingredient, as shown in basic action *a3* in Figure 4.

The information about ingredient requirements was added for 186 basic actions out of the 298 actions in the lexicon. No requirement was needed for the other actions, i.e., these actions accept any kind of ingredients.

4.2.2 Lexicon of Ingredients

In addition to the lexicon of cooking actions, the lexicon of ingredients is also required for our system. It includes ingredients and their features such as kind, shape and components. We believe that this is domain-specific knowledge for the cooking domain. Thesauri or other general-purpose language resources would not provide such information. Therefore, we newly compiled the lexicon of ingredients, which consists of only those ingredients appearing in the *ingredients example* in the lexicon of cooking actions. The number of ingredients included in the lexicon is 93. For each entry, features of the ingredient are described. The feature set used for this lexicon is the same as that for the *ingredient requirement* described in 4.2.1, except for the "instance" attribute.

[4] *action plan* is omitted in Figure 4.

Basic Action *a1*	
expression	くし形切りにする (cut into a comb shape)
ingredient examples	トマト (tomato), じゃがいも (potato)
ingredient requirement	kind=vegetable, shape=sphere
Basic Action *a2*	
expression	くし形切りにする (cut into a comb shape)
ingredient examples	かぼちゃ (pumpkin)
ingredient requirement	kind=vegetable, shape=sphere, seed=1
Basic Action *a3*	
expression	くし形切りにする (cut into a comb shape)
ingredient examples	カブ (turnip)
ingredient requirement	instance=カブ (turnip)

Figure 4: Three Basic Actions of "くし形切りにする (cut into a comb shape)"

The current lexicon of ingredients is too small. Only 93 ingredients are included. A larger lexicon is required to handle various recipe sentences. In order to enlarge the lexicon of ingredients, we will investigate a method for the automatically acquisition of new ingredients with their features from a collection of recipe documents.

5 Matching between Actions in a Recipe and the Lexicon

Action Matcher in Figure 1 is the module which accepts a recipe sentence and finds a basic action corresponding to it from the lexicon. One of the biggest difficulties in developing this module is that linguistic expressions in a recipe may differ from those in the lexicon. So we have to consider a flexible matching algorithm between them.

To construct *Action Matcher*, we refer to the verbal phrases classified in (b2) in Table 2. Note that the linguistic expressions of these verbal phrases are inconsistent with the expressions in the lexicon. We examined the major causes of inconsistency for these verbal phrases. In this paper, we will report the result of our analysis, and suggest some possible ways to find the equivalent action even when the linguistic expressions in a recipe and the lexicon are different. The realization of *Action Matcher* still remains as future work.

Figure 5 shows some examples of observed inconsistency in linguistic expressions. In Figure 5, the left hand side represents verbal phrases in recipes, while the right hand side represents expressions in the lexicon of cooking actions. A slash indicates word segmentation. Causes of inconsistency in linguistic expressions are classified as follows:

- Inconsistency in word segmentation

  Word segmentation of verbal phrases in recipes, as automatically given by a morphological analyzer, is different from one of the basic actions in the lexicon, as shown in Figure 5 (a).

  In order to succeed in matching, we need an operation to concatenate two or more morphemes in a phrase or to divide a morpheme into to two or more, then try to check the equivalence of both expressions.

- Inconsistency in case fillers

  Verbs in a recipe and the lexicon agree, but their case fillers are different. For instance, in Figure 5 (b), the verb "ふる (sprinkle)" is the same, but the accusative case fillers "唐辛子 (chili)" and "塩 (salt)" are different. In this case, we can regard both as representing the same action: to sprinkle a kind of condiment.

  In this case, the lexicon of ingredients (see 4.2.2) would be helpful for matching. That is, if both 唐辛子 (chili) and 塩 (salt) have the same feature "kind=condiment" in the lexicon of ingredients, we can judge that the phrase "唐辛子/を/ふる (sprinkle chili)" corresponds to the basic action "塩/を/ふる (sprinkle salt)".

- Inconsistency in verbs

  Disagreement between verbs in a recipe and the lexicon is one of the major causes of inconsistency. See Figure 5 (c), for instance.

	Expressions in Recipes	Expressions in Lexicon
(a)	割り / ほぐす ···break (egg) (divide) (loosen)	割りほぐす ···break (egg) (break)
(b)	唐辛子 / を / ふる ···sprinkle chili (chili) (ACC) (sprinkle)	塩 / を / ふる ···sprinkle salt (salt) (ACC) (sprinkle)
(c)	砂出し / を /する ···make (shellfish) spew out sand (Spewing sand) (ACC) (do)	塩水 / に /ひたす ···dip it into salt water (salt water) (LOC) (dip)

Figure 5: Inconsistency in Linguistic Expressions

These two phrases represent the same action [5], but the linguistic expressions are totally different.

In this case, the matching between them is rather difficult. One solution would be to describe all equivalent expressions for each action in the lexicon. Since it is not realistic to list equivalent expressions exhaustively, however, we want to automatically collect pairs of equivalent expressions from a large recipe corpus.

6 Conclusion

In this paper, we have described the basic idea for a system to generate animations for cooking actions in recipes. Although the system is not yet complete and much work still remains to be done, the main contribution of this paper is to show the direction for improving the scalability of the system. First, we designed a lexicon of cooking actions including information about action plans and ingredient requirements, which are needed to generate the appropriate cooking animations. We also showed that our lexicon covers most of the cooking actions appearing in recipes. Furthermore, we analyzed the recipe corpus and investigated how to match actions in a recipe to the corresponding basic action in the lexicon, even when they have different linguistic expressions. Such a flexible matching method would also increase the scalability of the system.

References

Hisahiro Adachi. 1997. GCD: A generation method of cooking definitions based on similarity between a couple of recipes. In *Proceedings of the Natural Language Processing Pacific Rim Symposium*, pages 135–140.

Elisabeth Andre and Thomas Rist. 1996. Coping with temporal constraints in multimedia presentation planning. In *Proceedings of the National Conference on Artificial Intelligence*, pages 142–147.

Yoko Atsuta. 2004. *How to cut vegetables (in Japanese)*. Syûeisha.

Bob Coyne and Richard Sproat. 2001. WordsEye: An automatic text-to-scene conversion system. In *Proceedings of the SIGGRAPH*, pages 487–496.

Yoshiko Fujino. 2003. *New Fundamental Cooking (in Japanese)*. SS Communications.

Eri Hayashi, Suguru Yoshioka, and Satoshi Tojo. 2003. Automatic generation of event structure for Japanese cooking recipes (in Japanese). *Journal of Natural Language Processing*, 10(2):3–17.

Robin F. Karlin. 1988. Defining the semantics of verbal modifiers in the domain of cooking tasks. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 61–67.

Tomohide Shibata, Daisuke Kawahara, Masashi Okamoto, Sadao Kurohashi, and Toyoaki Nishida. 2003. Structural analysis of instruction utterances. In *Proceedings of the Seventh International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES2003)*, pages 1054–1061.

Junko Takashiro and Satomi Kenmizaki. 2004. *Standard Cooking: Fundamentals of Cooking (in Japanese)*. Shôgakukan.

Stuart G. Towns, Charles B. Callaway, and James C. Lester. 1998. Generating coordinated natural language and 3D animations for complex spatial explanations. In *Proceedings of the National Conference on Artificial Intelligence*, pages 112–119.

Hideki Uematsu, Akira Shimazu, and Manabu Okumura. 2001. Generation of 3D CG animations from recipe sentences. In *Proceedings of the Natural Language Processing Pacific Rim Symposium*, pages 461–466.

Bonnie Lynn Webber and Barbara Di Eugenio. 1990. Free adjuncts in natural language instructions. In *Proceedings of the International Conference on Computational Linguistics*, pages 395–400.

[5] Note that it is required to dip shellfish into salt water in order to make it spew out sand.

Morphological Richness Offsets Resource Demand- Experiences in Constructing a POS Tagger for Hindi

Smriti Singh **Kuhoo Gupta** **Manish Shrivastava** **Pushpak Bhattacharyya**
Department of Computer Science and Engineering
Indian Institute of Technology, Bombay
Powai, Mumbai
400076 Maharashtra, India
{smriti,kuhoo,manshri,pb}@cse.iitb.ac.in

Abstract

In this paper we report our work on building a POS tagger for a morphologically rich language- Hindi. The theme of the research is to vindicate the stand that- if morphology is strong and harnessable, then lack of training corpora is not debilitating. We establish a methodology of POS tagging which the resource disadvantaged (lacking annotated corpora) languages can make use of. The methodology makes use of locally annotated modestly-sized corpora (15,562 words), exhaustive morpohological analysis backed by high-coverage lexicon and a decision tree based learning algorithm (CN2). The evaluation of the system was done with 4-fold cross validation of the corpora in the news domain (www.bbc.co.uk/hindi). The current accuracy of POS tagging is 93.45% and can be further improved.

1 Motivation and Problem Definition

Part-Of-Speech (POS) tagging is a complex task fraught with challenges like *ambiguity of parts of speech* and *handling of "lexical absence" (proper nouns, foreign words, derivationally morphed words, spelling variations* and *other unknown words)* (Manning and Schutze, 2002). For English there are many POS taggers, employing machine learning techniques like transformation-based error-driven learning (Brill, 1995), decision trees (Black et al., 1992), markov model (Cutting et al. 1992), maximum entropy methods (Ratnaparkhi, 1996) *etc*. There are also taggers which are hybrid using both stochastic and rule-based approaches, such as CLAWS (Garside and Smith, 1997). The accuracy of these taggers ranges from 93-98% approximately. English has annotated corpora in abundance, enabling usage of powerful data driven machine learning methods. But, very few languages in the world have the resource advantage that English enjoys.

In this scenario, POS tagging of highly inflectional languages presents an interesting case study. Morphologically rich languages are characterized by a large number of morphemes in a single word, where morpheme boundaries are difficult to detect because they are fused together. They are typically free-word ordered, which causes fixed-context systems to be hardly adequate for statistical approaches (Samuelsson and Voutilainen, 1997). Morphology-based POS tagging of some languages like Turkish (Oflazer and Kuruoz, 1994), Arabic (Guiassa, 2006), Czech (Hajic et al., 2001), Modern Greek (Orphanos et al., 1999) and Hungarian (Megyesi, 1999) has been tried out using a combination of hand-crafted rules and statistical learning. These systems use large amount of corpora along with morphological analysis to POS tag the texts. It may be noted that a purely rule-based or a purely stochastic approach will not be effective for such

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 779–786,
Sydney, July 2006.

languages, since the former demands subtle linguistic expertise and the latter variously permuted corpora.

1.1 Previous Work on Hindi POS Tagging

There is some amount of work done on morphology-based disambiguation in Hindi POS tagging. Bharati *et al.* (1995) in their work on computational Paninian parser, describe a technique where POS tagging is implicit and is merged with the parsing phase. Ray *et al.* (2003) proposed an algorithm that identifies Hindi word groups on the basis of the lexical tags of the individual words. Their partial POS tagger (as they call it) reduces the number of possible tags for a given sentence by imposing some constraints on the sequence of lexical categories that are possible in a Hindi sentence. UPENN also has an online Hindi morphological tagger[1] but there exists no literature discussing the performance of the tagger.

1.2 Our Approach

We present in this paper a POS tagger for Hindi- the national language of India, spoken by 500 million people and ranking 4th in the world. We establish a methodology of POS tagging which **the resource disadvantaged (lacking annotated corpora) languages can make use of**. This methodology uses locally annotated modestly sized corpora (15,562 words), exhaustive morphological analysis backed by high-coverage lexicon and a decision tree based learning algorithm- CN2 (Clark and Niblett, 1989). To the best of our knowledge, such an approach has never been tried out for Hindi. The heart of the system is the detailed linguistic analysis of morphosyntactic phenomena, adroit handling of suffixes, accurate verb group identification and learning of disambiguation rules.

The approach can be used for other inflectional languages by providing the language specific resources in the form of suffix replacement rules (SRRs), lexicon, group identification and morpheme analysis rules *etc.* and keeping the processes the same as shown in Figure 1. The similar kind of work exploiting morphological information to assign POS tags is under progress for Marathi which is also an Indian language.

In what follows, we discuss in section 2 the challenges in Hindi POS tagging followed by a section on morphological structure of Hindi. Section 4 presents the design of Hindi POS tagger. The experimental setup and results are given in sections 5 and 6. Section 7 concludes the paper.

2 Challenges of POS Tagging in Hindi

The inter-POS ambiguity surfaces when a word or a morpheme displays an ambiguity across POS categories. Such a word has multiple entries in the lexicon (one for each category). After stemming, the word would be assigned all possible POS tags based on the number of entries it has in the lexicon. The complexity of the task can be understood looking at the following English sentence where the word 'back' falls into three different POS categories-

"I get back to the back seat to give rest to my back."

The complexity further increases when it comes to tagging a free-word order language like Hindi where almost all the permutations of words in a clause are possible (Shrivastava et al., 2005). This phenomenon in the language, makes the task of a stochastic tagger difficult.

Intra-POS ambiguity arises when a word has one POS with different feature values, *e.g.*, the word 'लड़के' {laDke} *(boys/boy)* in Hindi is a noun but can be analyzed in two ways in terms of its feature values:

1. **POS**: *Noun*, **Number**: *Sg*, **Case**: *Oblique*
   मैंने लड़के को एक आम दिया.
   maine laDke ko ek aam diyaa.
   I-erg boy to one mango gave.
   I gave a mango to the boy.

2. **POS**: *Noun*, **Number**: *Pl*, **Case**: *Direct*
   लड़के आम खाते हैं.
   laDke aam khaate hain.
   Boys mangoes eat.
   Boys eat mangoes.

[1] http://ccat.sas.upenn.edu/plc/tamilweb/hindi.html

One of the difficult tasks here is to choose the appropriate tag based on the morphology of the word and the context used. Also, new words appear all the time in the texts. Thus, a method for determining the tag of a new word is needed when it is not present in the lexicon. This is done using context information and the information coded in the affixes, as affixes in Hindi (especially in nouns and verbs) are strong indicators of a word's POS category. For example, it is possible to determine that the word 'जाएगा' {jaaegaa} *(will go)* is a verb, based on the environment in which it appears and the knowledge that it carries the inflectional suffix -एगा {egaa} that attaches to the base verb 'जा' {jaa}.

2.1 Ambiguity Schemes

The criterion to decide whether the tag of a word is a *Noun* or a *Verb* is entirely different from that of whether a word is an *Adjective* or an *Adverb*. For example, the word 'पर' can occur as *conjunction*, *post-position* or a *noun* (as shown previously), hence it falls in an *Ambiguity Scheme* '*Conjunction-Noun-Postposition*'. We grouped all the ambiguous words into sets according to the *Ambiguity Schemes* that are possible in Hindi, e.g., *Adjective-Noun*, *Adjective-Adverb*, *Noun-Verb*, *etc*. This idea was first proposed by Orphanos *et al.* (1999) for Modern Greek POS tagging.

3 Morphological Structure Of Hindi

In Hindi, *Nouns* inflect for number and case. To capture their morphological variations, they can be categorized into various *paradigms*[2] (Narayana, 1994) based on their *vowel ending, gender, number* and *case information*. We have a list of around 29,000 Hindi nouns that are categorized into such paradigms[3]. Looking at the morphological patterns of the words in a paradigm, suffix-replacement rules have been developed. These rules help in separating out a valid suffix from an inflected word to output the correct stem and consequently, get the correct root.

Hindi *Adjectives* may be inflected or uninflected, *e.g.*, 'चमकीला' {chamkiilaa} *(shiny)*, 'अच्छा' {acchaa} *(nice)*, 'लंबा' {lambaa} *(long)* inflect based on the number and case values of their head nouns while 'सुंदर' {sundar} *(beautiful)*, 'भारी' {bhaarii} *(heavy) etc*. do not inflect.

Hindi *Verbs* inflect for the following grammatical properties (GNPTAM):

1. Gender: Masculine, Feminine, Non-specific
2. Number: Singular, Plural, Non-specific
3. Person: 1st, 2nd and 3rd
4. Tense: Past, Present, Future
5. Aspect: Perfective, Completive, Frequentative, Habitual, Durative, Inceptive, Stative
6. Modality: Imperative, Probabilitive, Subjunctive, Conditional, Deontic, Abilitive, Permissive

The morphemes attached to a verb along with their corresponding analyses help identify values for GNPTAM features for a given verb form.

Division of Information Load in Hindi Verb Groups

A *Verb Group* (VG) primarily comprises main verb and auxiliaries. Constituents like particles, negation markers, conjunction, *etc*. can also occur within a VG. It is important to know how much of GNPTAM feature information is stored in VG constituents individually and what is the load division in the absence or presence of auxiliaries. In a Hindi VG, when there is no auxiliary present, the complete information load falls on the main verb which carries information for GNPTAM features. In presence of auxiliaries, the load gets shared between the main verb and auxiliaries, and is represented in the form of different morphemes (inflected or uninflected), *e.g.*, in the sentence -

[2] A paradigm systematically arranges and identifies the uninflected forms of the words that share similar inflectional patterns.

[3] Anusaaraka system developed at IIT Kanpur (INDIA) uses similar noun sets in the form of paradigms

मैं बोल पा रहा हूँ
main bol paa rahaa hoon
I am able to speak

1. Main verb 'बोल' {bol} is uninflected and does not carry any information for any of the GNPTAM features.

2. 'पा' {paa} is uninflected and gives modality information, *i.e.*, Abilitive.

3. 'रहा' {rahaa} gives Number (Sg), Gender (Masculine), Aspect (Durative)

4. 'हूँ' {hoon} gives Number (Sg), Person (1st), Tense (Present)

Gerund Identification
In Hindi, the attachment of verbal suffixes like 'ना' {naa} and 'ने' {ne} to a verb root results either in a gerund like 'तैरना' {tairnaa} *(swimming)* or in an infinitival verb form like 'तैरना' {tairnaa} *(to swim)*. We observed that it is easy to detect a gerund if it is followed by a case-marker or by any other infinitival verb form.

4 Design of Hindi POS Tagger

4.1 Morphology Driven Tagger

Morphology driven tagger makes use of the affix information stored in a word and assigns a POS tag using no contextual information. Though, it does take into account the previous and the next word in a VG to correctly identify the main verb and the auxiliaries, other POS categories are identified through lexicon lookup of the root form. The current lexicon[4] has around 42,000 entries belonging to the major categories as mentioned in Figure 3. The format of each entry is ⟨word⟩,⟨paradigm⟩,⟨category⟩.

The process does not involve learning or disambiguation of any sort and is completely driven by hand-crafted morphology rules. The architecture of the tagger is shown in Figure 1. The work progresses at two levels:

[4]The lexicon was developed using the wordlist from Hindi Wordnet (http://www.cfilt.iitb.ac.in/wordnet/webhwn/) and partial noun list from Anusaraka. It is being enhanced by adding new words from the corpus and removing the inconsistencies.

1. **At Word Level**: A *stemmer* is used in conjunction with lexicon and Suffix Replacement Rules (SRRs) to output all possible root-suffix pairs along with POS category label for a word. There is a possibility that the input word is not found in the lexicon and does not carry any inflectional suffix. In such a case, *derivational morphology rules* are applied.

2. **At Group Level**: At this level a *Morphological Analyzer* (MA) uses the information encoded in the extracted suffix to add morphological information to the word. For nouns, the information provided by the suffixes is restricted only to '*Number*'. '*Case*' can be inferred later by looking at the neighbouring words.

   For verbs, GNP values are found at the word level, while TAM values are identified during the VG Identification phase, described later. The analysis of the suffix is done in a discrete manner, *i.e.*, each component of the suffix is analyzed separately. A morpheme analysis table comprising individual morphemes with their paradigm information and analyses is used for this purpose. MA's output for the word खाऊंगी {khaaoongii} *(will eat)* looks like -
   Stem: खा *(eat)*
   Suffix: ऊंगी Category: *Verb*
   Morpheme 1: ऊं Analysis: *1 Per, Sg*
   Morpheme 2: ग Analysis: *Future*
   Morpheme 3: ई Analysis: *Feminine*

4.1.1 Verb Group Identification

The structure of a Hindi VG is relatively rigid and can be captured well using simple syntactic rules. In Hindi, certain auxiliaries like 'रह' {rah}, 'पा' {paa}, 'सक', {sak} or 'पड़' {paD} can also occur as main verbs in some contexts. VG identification deals with identifying the main verb and the auxiliaries of a VG while discounting for particles, conjunctions and negation markers. The VG identification goes left to right by marking the first constituent as the main verb or copula verb and making every other verb con-

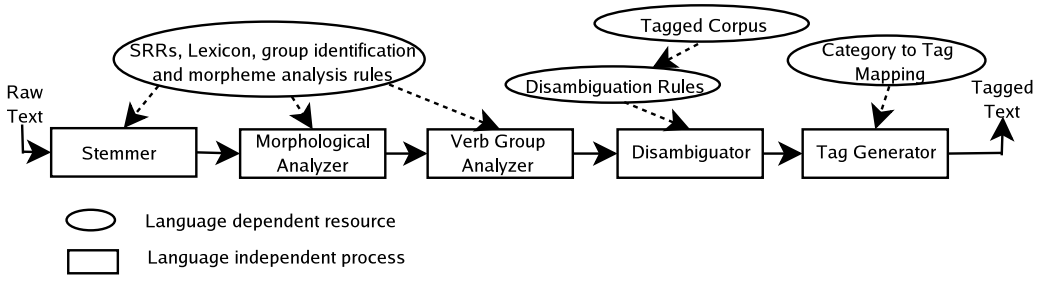


Figure 1: Overall Architecture of the Tagger

Table 1: Average Accuracy(%) Comparison of Various Approaches

LLB	LLBD	MD	BL	LB
61.19	86.77	73.62	82.63	93.45

struct an auxiliary till a non-VG constituent is encountered. Main verb and copula verb can take the head position of a VG and can occur with or without auxiliary verbs. Auxiliary verbs, on the other hand, always come along with a main verb or a copula verb. This results in a very high accuracy of 99.5% for verb auxiliaries. Ambiguity between a main verb and a copula verb remains unresolved at this level and asks for disambiguation rules.

4.2 Need for Disambiguation

The accuracy obtained by simple lexicon lookup based approach (LLB) comes out to be 61.19%. The morphology-driven tagger, on the other hand, performs better than just lexicon lookup but still results in considerable ambiguity. These results are significant as they present a strong case in favor of using detailed morphological analysis. Similar observation has been presented by Uchimoto *et al.* (2001) for Japanese language. According to the tagging performed by SRRs and the lexicon, a word receives *n* tags if it belongs to *n* POSs. If we consider multiple tags for a word as an error of the tagger (even when the options contain the correct tag for a word), then the accuracy of the tagger comes to be 73.62% (as shown in Table 1). The goal is to keep the contextually appropriate tag and eliminate others which can be achieved by devising a disambiguation technique. The disambiguation task can be naively addressed by choosing the most frequent tag for a word. This approach is also known as baseline (BL) tagging. The baseline accuracy turns out to be 82.63% which is still higher than that of the morphology-driven tagger[5]. The drawback with baseline tagging is that its accuracy cannot be further improved. On the other hand, there is enough room for improving upon the accuracy of morphology-driven (MD) tagger. It is quite evident that though the MD tagger works well for VG and many close categories, around 30% of the words are either ambiguous or unknown. Hence, a disambiguation stage is needed to shoot up the accuracy.

The common choice for disambiguation rule learning in POS tagging task is usually machine learning techniques mainly focussing on decision tree based algorithms (Orphanos and Christodoulalds, 1999), neural networks (Schmid, 1994), *etc*. Among the various decision tree based algorithms like ID3, AQR, ASSISTANT and CN2, CN2 is known to perform better than the rest (Clark and Niblett, 1989). Since no such machine learning technique has been used for Hindi language, we thought of choosing CN2 as it performs well on noisy data[6].

[5]These numbers may change if we experiment on a different dataset

[6]The training annotated corpora becomes noisy by virtue of intuitions of different annotators (trained native Hindi speakers)

4.2.1 Training Corpora

We set up a corpus, collecting sentences from BBC news site[7] and let the morphology-driven tagger assign morphosyntactic tags to all the words. For an ambiguous word, the contextually appropriate POS tag is manually chosen. Unknown words are assigned a correct tag based on their context and usage.

4.2.2 Learning

Out of the completely manually corrected corpora of 15,562 tokens, we created training instances for each *Ambiguity Scheme* and for *Unknown* words. These training instances take into account the POS categories of the neighbouring words and not the feature values[8]. The experiments were carried out for different context window sizes ranging from 2 to 20 to find the best configuration.

4.2.3 Rule Generation

The rules are generated from the training corpora by extracting the ambiguity scheme (AS) of each word. If the word is not present in the lexicon then its AS is set as 'unknown'. Once the AS is identified, a training instance is formed. This training instance contains the neighbouring correct POS categories as attributes. The number of neighbours included in the training instance is the window size for CN2. After all the ambiguous words are processed and training instances for all seen ASs are created, the CN2 algorithm is applied over the training instances to generate actual rule-sets for each AS. The CN2 algorithm gives one set of *If-Then* rules (either ordered or unordered) for each AS including 'unknown'[9]. The AS of every ambiguous word is formed while tagging. A corresponding rule-set for that AS is then identified and traversed to get the contextually appropriate rule. The resultant category outputted by this rule is then assigned to the ambiguous word. The traversal rule differs for ordered and unordered implementation. The POS of an unknown word is guessed by traversing the rule-set for unknown words[10] and assigning it the resultant tag.

[7] http://www.bbc.co.uk/hindi/

[8] Considering that a tag encodes 0 to 6 morphosyntactic features and each feature takes either one or a disjunction of 2 to 7 values, the total number of different tags can count up to several hundreds

[9] We used the CN2 algorithm implementation (1990) by Robin Boswell. The software is available at *ftp://ftp.cs.utexas.edu/pub/pclark/cn2.tar.Z*

5 Experimental Setup

The experimentation involved, first, identifying the best parameter values for the CN2 algorithm and second, evaluating the performance of the disambiguation rules generated by CN2 for the POS tagging task.

5.1 CN2 Parameters

The various parameters in CN2 algorithm are: rule type (ordered or unordered), star size, significance threshold and size of the training instances (window size). The best results are empirically achieved with ordered rules, star size as 1, significance threshold as 10 and window size 4, *i.e.*, two neighbours on either side are used to generate the training instances.

5.2 Evaluation

The tests are performed on contiguous partitions of the corpora (15,562 words) that are 75% training set and 25% testing set.

$$Accuracy = \frac{no.\ of\ single\ correct\ tags}{total\ no.\ of\ tokens}$$

The results are obtained by performing a 4-fold cross validation over the corpora. Figure 2 gives the learning curve of the disambiguation module for varying corpora sizes starting from 1000 to the complete training corpora size. The accuracy for known and unknown words is also measured separately.

6 Results and Discussion

The average accuracy of the learning based (LB) tagger after 4-fold cross validation is 93.45%. To

[10] Most of the unknown words are proper nouns, which cannot be stored in the lexicon extensively. So, it also helps in *named-entity detection*.

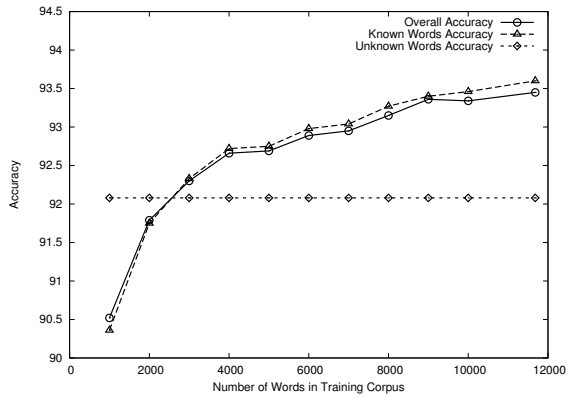


Figure 2: POS Learning Curve

the best of our knowledge no comparable results have been reported so far for Hindi.

From Table 1, we can see that the disambiguation module brings up the accuracy of simple lexicon lookup based approach by around 25% (LLBD). The overall average accuracy is also brought up by around 20% by augmenting the morphology-driven (MD) tagger by a disambiguation module; hence justifying our belief that a disambiguation module over a morphology driven approach yields better results.

One interesting observation is the performance of the tagger on individual POS categories. Figure 3 shows clearly that the per POS accuracies of the LB tagger highly exceeds those of the MD and BL tagger for most categories. This means that the disambiguation module correctly disambiguates and correctly identifies the unknown words too. The accuracy on unknown words, as earlier shown in Figure 2, is very high at 92.08%. The percentage of unknown words in the test corpora is 0.013. It seems independent of the size of training corpus because the corpora is unbalanced having most of the unknowns as proper nouns. The rules are formed on this bias, and hence the application of these rules assigns PPN tag to an unknown which is mostly the case.

From Figure 3 again we see that the accuracy on some categories remains very low even after disambiguation. This calls for some detailed failure analysis. By looking at the categories having low accuracy, such as pronoun, intensifier, demonstratives and verb copula, we find that all of them are highly ambiguous and, almost invariably, very rare in the corpus. Also, most of them are hard to disambiguate without any semantic information.

7 Conclusions & Future Work

We have described in this paper a POS tagger for Hindi which can overcome the handicap of annotated corpora scarcity by exploiting the rich morphology of the language and the relatively rigid word-order within a VG. The whole work was driven by hunting down the factors that lower the accuracy of *Verbs* and weeding them out. A detailed study of accuracy distribution across the POS tags pointed out the cases calling for elaborate disambiguation rules. A major strength of the work is the learning of disambiguation rules, which otherwise would have been hand-coded, thus demanding exhaustive analysis of language phenomena. Attaining an accuracy of close to 94%, from a corpora of just about 15,562 words lends credence to the belief that "*morphological richness can offset resource scarcity*". The work could lead such efforts of POS tag building for all those languages which have rich morphology, but cannot afford to invest a lot in creating large annotated corpora.

Several interesting future directions suggest themselves. It will be worthwhile to investigate a statistical approach like Conditional Random Fields in which the feature functions would be constructed from morphology. The next logical step from the POS tagger is a chunker for Hindi. In fact a start on this has already been made through VG identification.

References

A. Ratnaparakhi. 1996. *A Maximum Entropy Part-Of-Speech Tagger.* EMNLP 1996

A. Bharati, V. Chaitanya, R. Sangal 1995. *Natural Language Processing : A Paninian Perspective* . Prentice Hall India.

A. Kuba, A. Hcza, J. Csirik 2004. *POS Tagging of Hungarian with Combined Statistical and Rule-Based Methods.* TSD 2004

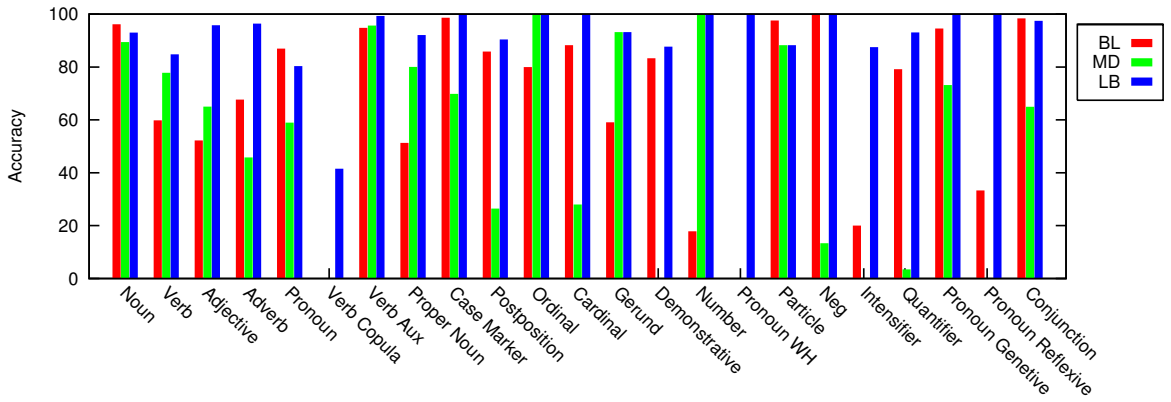


Figure 3: Per-POS Accuracy Distribution

B. Megyesi. 1999. *Improving Brill's POS tagger for an agglutinative language.* Joint Sigdat Conference EMNLP/VLC 1999.

C. D. Manning and H. Schutze. 2002. *Foundations of Statistical Natural Language Processing,* MIT Press 2002.

D. Cutting et al. 1992. *A practical part-of-speech tagger.* In Proc. of the Third Conf. on Applied Natural Language Processing. ACL 1992.

E. Brill. 1995. *Transformation-Based Error Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging* . Computational Linguistics 21(94): 543-566. 1995.

E. Black et al. 1992. *Decision tree models applied to the labeling of text with parts-of-speech.* In Darpa Workshop on Speech and Natural Language 1992.

G. Leech, R. Garside and M. Bryant. 1992. *Automatic POS-Tagging of the corpus*. BNC2 POS-tagging Manual.

G. Orphanos, D. Kalles, A. Papagelis, D. Christodoulakis. 1999 *Decision trees and NLP: A Case Study in POS Tagging* . In proceedings of ACAI 1999.

H. Schmid 1994 *Part-of-Speech Tagging with Neural Networks* . In proceedings of COLING 1994.

J. Hajic, P. Krbec, P. Kveton, K. Oliva, V. Petkevic 2001 *A Case Study in Czech Tagging.* In Proceedings of the 39th Annual Meeting of the ACL 2001

K. Uchimoto, S. Sekine, H. Isahara. 2001. *The Unknown Word Problem: a Morphological Analysis of Japanese Using Maximum Entropy Aided by a Dictionary.* In Proceedings of the Conference on EMNLP 2001

K. Oflazer and I. Kuruoz. 1994. *Tagging and morphological disambiguation of Turkish text.* In Proceedings of the 4 ACL Conference on Applied Natural Language Processing Conference 1994

M. Shrivastava, N. Agrawal, S. Singh, P. Bhattacharya. 2005. *Harnessing Morphological Analysis in POS Tagging Task.* In Proceedings of the ICON 2005

P. R. Ray , V. Harish, A. Basu and S. Sarkar 2003. *Part of Speech Tagging and Local Word Grouping Techniques for Natural Language Parsing in Hindi*. In Proceedings of ICON 2003

P. Clark and T. Niblett 1989. *The CN2 Induction Algorithm.* Journal of Machine Learning, vol(3), pages 261-283, 1989

R. Garside, N. Smith 1997. *A hybrid grammatical tagger: CLAWS4* . In R. Garside, G. Leech, A. McEnery (eds.) Corpus annotation: Linguistic information from computer text corpora. Longman. Pp. 102-121.

C. Samuelsson and A. Voutilainen 1997. *Comparing a Linguistic and a Stochastic Tagger.* In Procs. Joint 35th Annual Meeting of the ACL and 8th Conference of the European Chapter of the ACL 1997.

Y. Tlili-Guiassa 2006. *Hybrid Method for Tagging Arabic Text*. Journal of Computer Science 2 (3): 245-248, 2006

Minimum Risk Annealing for Training Log-Linear Models*

David A. Smith and **Jason Eisner**
Department of Computer Science
Center for Language and Speech Processing
Johns Hopkins University
Baltimore, MD 21218, USA
{dasmith,eisner}@jhu.edu

Abstract

When training the parameters for a natural language system, one would prefer to minimize 1-best *loss* (error) on an evaluation set. Since the error surface for many natural language problems is piecewise constant and riddled with local minima, many systems instead optimize log-likelihood, which is conveniently differentiable and convex. We propose training instead to minimize the *expected loss*, or *risk*. We define this expectation using a probability distribution over hypotheses that we gradually sharpen (anneal) to focus on the 1-best hypothesis. Besides the *linear* loss functions used in previous work, we also describe techniques for optimizing *nonlinear* functions such as precision or the BLEU metric. We present experiments training log-linear combinations of models for dependency parsing and for machine translation. In machine translation, annealed minimum risk training achieves significant improvements in BLEU over standard minimum error training. We also show improvements in labeled dependency parsing.

1 Direct Minimization of Error

Researchers in empirical natural language processing have expended substantial ink and effort in developing metrics to evaluate systems automatically against gold-standard corpora. The ongoing evaluation literature is perhaps most obvious in the machine translation community's efforts to better BLEU (Papineni et al., 2002).

Despite this research, parsing or machine translation systems are often trained using the much simpler and harsher metric of maximum likelihood. One reason is that in supervised training, the log-likelihood objective function is generally convex, meaning that it has a single global maximum that can be easily found (indeed, for supervised generative models, the parameters at this maximum may even have a closed-form solution). In contrast to the likelihood surface, the *error* surface for discrete structured prediction is not only riddled with local minima, but piecewise constant and not everywhere differentiable with respect to the model parameters (Figure 1). Despite these difficulties, some work has shown it worthwhile to minimize error directly (Och, 2003; Bahl et al., 1988).

We show improvements over previous work on error minimization by minimizing the **risk** or **expected error**—a continuous function that can be derived by combining the likelihood with any evaluation metric (§2). Seeking to avoid local minima, deterministic annealing (Rose, 1998) gradually changes the objective function from a convex entropy surface to the more complex risk surface (§3). We also discuss regularizing the objective function to prevent overfitting (§4). We explain how to compute expected loss under some evaluation metrics common in natural language tasks (§5). We then apply this machinery to training log-linear combinations of models for dependency parsing and for machine translation (§6). Finally, we note the connections of minimum risk *training* to max-margin training and minimum Bayes risk *decoding* (§7), and recapitulate our results (§8).

2 Training Log-Linear Models

In this work, we focus on rescoring with log-linear models. In particular, our experiments consider log-linear combinations of a relatively small number of features over entire complex structures, such as trees or translations, known in some previous work as *products of experts* (Hinton, 1999) or *logarithmic opinion pools* (Smith et al., 2005). A feature in the combined model might thus be a log probability from an entire submodel. Giving this feature a small or negative weight can discount a submodel that is foolishly structured, badly trained, or redundant with the other features.

For each sentence x_i in our training corpus S, we are given K_i possible analyses $y_{i,1}, \ldots y_{i,K_i}$. (These may be all of the possible translations or parse trees; or only the K_i most probable under

*This work was supported by an NSF graduate research fellowship for the first author and by NSF ITR grant IIS-0313193 and ONR grant N00014-01-1-0685. The views expressed are not necessarily endorsed by the sponsors. We thank Sanjeev Khudanpur, Noah Smith, Markus Dreyer, and the reviewers for helpful discussions and comments.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 787–794,
Sydney, July 2006.

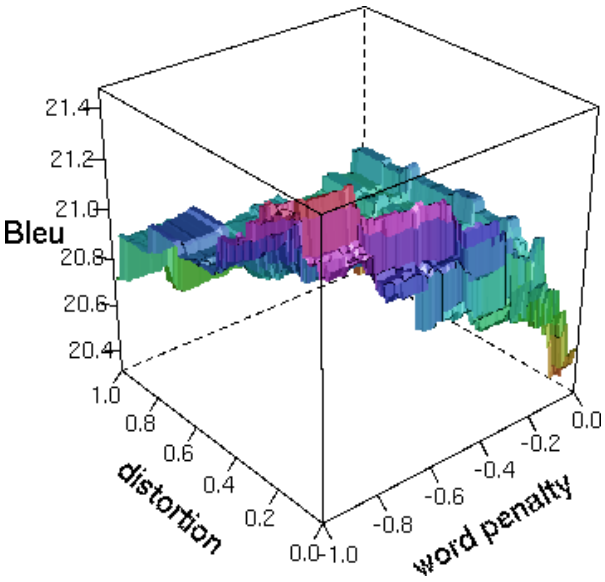


Figure 1: The loss surface for a machine translation system: while other parameters are held constant, we vary the weights on the distortion and word penalty features. Note the piecewise constant regions with several local maxima.

some other model; or only a random sample of size K_i.) Each analysis has a vector of real-valued features (i.e., factors, or experts) denoted $f_{i,k}$. The score of the analysis $y_{i,k}$ is $\theta \cdot f_{i,k}$, the dot product of its features with a parameter vector θ. For each sentence, we obtain a normalized probability distribution over the K_i analyses as

$$p_\theta(y_{i,k} \mid x_i) = \frac{\exp \theta \cdot f_{i,k}}{\sum_{k'=1}^{K_i} \exp \theta \cdot f_{i,k'}} \quad (1)$$

We wish to adjust this model's parameters θ to minimize the severity of the errors we make when using it to choose among analyses. A **loss function** $L_{y^*}(y)$ assesses a penalty for choosing y when y^* is correct. We will usually write this simply as $L(y)$ since y^* is fixed and clear from context. For clearer exposition, we assume below that the total loss over some test corpus is the sum of the losses on individual sentences, although we will revisit that assumption in §5.

2.1 Minimizing Loss or Expected Loss

One training criterion directly mimics test conditions. It looks at the loss incurred if we choose the best analysis of each x_i according to the model:

$$\min_\theta \sum_i L(\operatorname*{argmax}_{y_i} p_\theta(y_i \mid x_i)) \quad (2)$$

Since small changes in θ either do not change the best analysis or else push a different analysis to the top, this objective function is piecewise constant, hence not amenable to gradient descent. Och (2003) observed, however, that the piecewise-constant property could be exploited to characterize the function exhaustively along any line in parameter space, and hence to minimize it globally along that line. By calling this global line minimization as a subroutine of multidimensional optimization, he was able to minimize (2) well enough to improve over likelihood maximization for training factored machine translation systems.

Instead of considering only the best hypothesis for any θ, we can minimize **risk**, i.e., the **expected loss** under p_θ across all analyses y_i:

$$\min_\theta \mathbf{E}_{p_\theta} L(y_{i,k}) \stackrel{\text{def}}{=} \min_\theta \sum_i \sum_k L(y_{i,k}) p_\theta(y_{i,k} \mid x_i) \quad (3)$$

This "smoothed" objective is now continuous and differentiable. However, it no longer exactly mimics test conditions, and it typically remains non-convex, so that gradient descent is still not guaranteed to find a global minimum. Och (2003) found that such smoothing during training "gives almost identical results" on translation metrics.

The simplest possible loss function is 0/1 loss, where $L(y)$ is 0 if y is the true analysis y_i^* and 1 otherwise. This loss function does not attempt to give partial credit. Even in this simple case, assuming P $\neq$ NP, there exists no general polynomial-time algorithm for even approximating (2) to within any constant factor, even for $K_i = 2$ (Hoffgen et al., 1995, from Theorem 4.10.4).[1] The same is true for for (3), since for $K_i = 2$ it can be easily shown that the min 0/1 risk is between 50% and 100% of the min 0/1 loss.

2.2 Maximizing Likelihood

Rather than minimizing a loss function suited to the task, many systems (especially for language modeling) choose simply to maximize the probability of the gold standard. The log of this **likelihood** is a convex function of the parameters θ:

$$\max_\theta \sum_i \log p_\theta(y_i^* \mid x_i) \quad (4)$$

where y_i^* is the true analysis of sentence x_i. The only wrinkle is that $p_\theta(y_i^* \mid x_i)$ may be left undefined by equation (1) if y_i^* is not in our set of K_i hypotheses. When maximizing likelihood, therefore, we will replace y_i^* with the min-loss analysis in the hypothesis set; if multiple analyses tie

[1] Known algorithms are exponential but only in the dimensionality of the feature space (Johnson and Preparata, 1978).

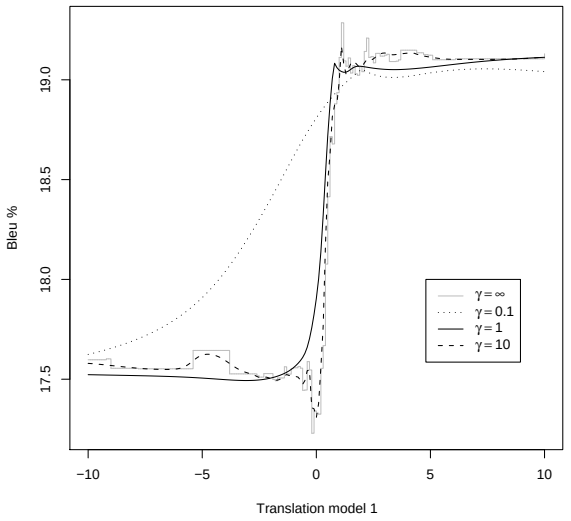


Figure 2: Loss and expected loss as one translation model's weight varies: the gray line ($\gamma = \infty$) shows true BLEU (to be optimized in equation (2)). The black lines show the expected BLEU as γ in equation (5) increases from 0.1 toward ∞.

for this honor, we follow Charniak and Johnson (2005) in summing their probabilities.[2]

Maximizing (4) is equivalent to minimizing an upper bound on the expected 0/1 loss $\sum_i (1 - p_\theta(y_i^* \mid x_i))$. Though the log makes it tractable, this remains a 0/1 objective that does not give partial credit to wrong answers, such as imperfect but useful translations. Most systems should be evaluated and preferably trained on less harsh metrics.

3 Deterministic Annealing

To balance the advantages of direct loss minimization, continuous risk minimization, and convex optimization, **deterministic annealing** attempts the solution of increasingly difficult optimization problems (Rose, 1998). Adding a scale hyperparameter γ to equation (1), we have the following family of distributions:

$$p_{\gamma,\theta}(y_{i,k} \mid x_i) = \frac{(\exp \theta \cdot f_{i,k})^\gamma}{\sum_{k'=1}^{K_i} \left(\exp \theta \cdot f_{i,k'}\right)^\gamma} \quad (5)$$

When $\gamma = 0$, all $y_{i,k}$ are equally likely, giving the uniform distribution; when $\gamma = 1$, we recover the model in equation (1); and as $\gamma \to \infty$, we approach the winner-take-all Viterbi function that assigns probability 1 to the top-scoring analysis.

For a fixed γ, deterministic annealing solves

$$\min_\theta \mathbf{E}_{p_{\gamma,\theta}}[L(y_{i,k})] \quad (6)$$

We then increase γ according to some *schedule* and optimize θ again. When γ is low, the smooth objective might allow us to pass over local minima that could open up at higher γ. Figure 3 shows how the smoothing is gradually weakened to reach the risk objective (3) as $\gamma \to 1$ and approach the true error objective (2) as $\gamma \to \infty$.

Our risk minimization most resembles the work of Rao and Rose (2001), who trained an isolated-word speech recognition system for expected word-error rate. Deterministic annealing has also been used to tackle non-convex likelihood surfaces in unsupervised learning with EM (Ueda and Nakano, 1998; Smith and Eisner, 2004). Other work on "generalized probabilistic descent" minimizes a similar objective function but with γ held constant (Katagiri et al., 1998).

Although the entropy is generally higher at lower values of γ, it varies as the optimization changes θ. In particular, a pure unregularized log-linear model such as (5) is really a function of $\gamma \cdot \theta$, so the optimizer could exactly compensate for increased γ by decreasing the θ vector proportionately![3] Most deterministic annealing procedures, therefore, express a direct preference on the entropy H, and choose γ and θ accordingly:

$$\min_{\gamma,\theta} \mathbf{E}_{p_{\gamma,\theta}}[L(y_{i,k})] - T \cdot H(p_{\gamma,\theta}) \quad (7)$$

In place of a schedule for raising γ, we now use a **cooling** schedule to lower T from ∞ to $-\infty$, thereby weakening the preference for high entropy. The Lagrange multiplier T on entropy is called "temperature" due to a satisfying connection to statistical mechanics. Once T is quite cool, it is common in practice to switch to raising γ directly and rapidly (**quenching**) until some convergence criterion is met (Rao and Rose, 2001).

4 Regularization

Informally, high temperature or $\gamma < 1$ smooths our model during training toward higher-entropy conditional distributions that are not so peaked at the desired analyses y_i^*. Another reason for such smoothing is simply to prevent overfitting to these training examples.

A typical way to control overfitting is to use a quadratic regularizing term, $||\theta||^2$ or more generally $\sum_d \theta_d^2 / 2\sigma_d^2$. Keeping this small keeps weights

[2] An alternative would be to artificially add y_i^* (e.g., the reference translation(s)) to the hypothesis set during training.

[3] For such models, γ merely aids the nonlinear optimizer in its search, by making it easier to scale all of θ at once.

low and entropy high. We may add this regularizer to equation (6) or (7). In the maximum likelihood framework, we may subtract it from equation (4), which is equivalent to maximum *a posteriori* estimation with a diagonal Gaussian prior (Chen and Rosenfeld, 1999). The variance σ_d^2 may reflect a prior belief about the potential usefulness of feature d, or may be tuned on heldout data.

Another simple regularization method is to stop cooling before T reaches 0 (cf. Elidan and Friedman (2005)). If loss on heldout data begins to increase, we may be starting to overfit. This technique can be used along with annealing or quadratic regularization and can achieve additional accuracy gains, which we report elsewhere (Dreyer et al., 2006).

5 Computing Expected Loss

At each temperature setting of deterministic annealing, we need to minimize the expected loss on the training corpus. We now discuss how this expectation is computed. When rescoring, we assume that we simply wish to combine, in some way, statistics of whole sentences[4] to arrive at the overall loss for the corpus. We consider evaluation metrics for natural language tasks from two broadly applicable classes: **linear** and **nonlinear**.

A **linear** metric is a sum (or other linear combination) of the loss or gain on individual sentences. Accuracy—in dependency parsing, part-of-speech tagging, and other labeling tasks—falls into this class, as do recall, word error rate in ASR, and the crossing-brackets metric in parsing. Thanks to the linearity of expectation, we can easily compute our *expected* loss in equation (6) by adding up the expected loss on each sentence.

Some other metrics involve **nonlinear** combinations over the sentences of the corpus. One common example is precision, $P \stackrel{\text{def}}{=} \sum_i c_i / \sum_i a_i$, where c_i is the number of correctly posited elements, and a_i is the total number of posited elements, in the decoding of sentence i. (Depending on the task, the elements may be words, bigrams, labeled constituents, etc.) Our goal is to maximize P, so during a step of deterministic annealing, we need to maximize the *expectation* of P when the sentences are decoded randomly according to equation (5). Although this expectation is continuous and differentiable as a function of θ, unfortunately it seems hard to compute for any given θ. We observe however that an equivalent goal is to minimize $-\log P$. Taking that as our loss function instead, equation (6) now needs to minimize the expectation of $-\log P$,[5] which decomposes somewhat more nicely:

$$\begin{aligned} \mathbf{E}[-\log P] &= \mathbf{E}[\log \sum_i a_i - \log \sum_i c_i] \\ &= \mathbf{E}[\log A] - \mathbf{E}[\log C] \end{aligned} \quad (8)$$

where the integer random variables $A = \sum_i a_i$ and $C = \sum_i c_i$ count the number of posited and correctly posited elements over the whole corpus.

To approximate $\mathbf{E}[g(A)]$, where g is any twice-differentiable function (here $g = \log$), we can approximate g locally by a quadratic, given by the Taylor expansion of g about A's mean $\mu_A = \mathbf{E}[A]$:

$$\begin{aligned} \mathbf{E}[g(A)] &\approx \mathbf{E}[g(\mu_A) + (A - \mu_A)g'(\mu_A) \\ &\quad + \frac{1}{2}(A - \mu_A)^2 g''(\mu_A)] \\ &= g(\mu_A) + \mathbf{E}[A - \mu_A]g'(\mu_A) \\ &\quad + \frac{1}{2}\mathbf{E}[(A - \mu_A)^2]g''(\mu_A) \\ &= g(\mu_A) + \frac{1}{2}\sigma_A^2 g''(\mu_A). \end{aligned}$$

Here $\mu_A = \sum_i \mu_{a_i}$ and $\sigma_A^2 = \sum_i \sigma_{a_i}^2$, since A is a sum of *independent* random variables a_i (i.e., given the current model parameters θ, our randomized decoder decodes each sentence independently). In other words, given our quadratic approximation to g, $\mathbf{E}[g(A)]$ depends on the (true) distribution of A only through the single-sentence means μ_{a_i} and variances $\sigma_{a_i}^2$, which can be found by enumerating the K_i decodings of sentence i. The approximation becomes arbitrarily good as we anneal $\gamma \to \infty$, since then $\sigma_A^2 \to 0$ and $\mathbf{E}[g(A)]$ focuses on g near μ_A. For equation (8),

$$\mathbf{E}[g(A)] = \mathbf{E}[\log A] \approx \log(\mu_A) - \frac{\sigma_A^2}{2\mu_A^2}$$

and $\mathbf{E}[\log C]$ is found similarly.

Similar techniques can be used to compute the expected logarithms of some other non-linear metrics, such as F-measure (the harmonic mean of precision and recall)[6] and Papineni et al. (2002)'s

[4] Computing sentence x_i's statistics usually involves iterating over hypotheses $y_{i,1}, \ldots y_{i,K_i}$. If these share substructure in a hypothesis lattice, dynamic programming may help.

[5] This changes the trajectory that DA takes through parameter space, but ultimately the objective is the same: as $\gamma \to \infty$ over the course of DA, minimizing $\mathbf{E}[-\log P]$ becomes indistinguishable from maximizing $\mathbf{E}[P]$.

[6] $R \stackrel{\text{def}}{=} C/B$; the count B of correct elements is *known*. So $\log F \stackrel{\text{def}}{=} \log 2PR/(P+R) = \log 2R/(1+R/P) = \log 2C/B - \log(1 + A/B)$. Consider $g(x) = \log 1 + x/B$.

BLEU translation metric (the geometric mean of several precisions). In particular, the expectation of $\log$ BLEU distributes over its $N+1$ summands:

$$\log \text{BLEU} = \min(1 - \frac{r}{A_1}, 0) + \sum_{n=1}^{N} w_n \log P_n$$

where P_n is the precision of the n-gram elements in the decoding.[7] As is standard in MT research, we take $w_n = 1/N$ and $N = 4$. The first term in the BLEU score is the log *brevity penalty*, a continuous function of A_1 (the total number of unigram tokens in the decoded corpus) that fires only if $A_1 < r$ (the average word count of the reference corpus). We again use a Taylor series to approximate the expected log brevity penalty.

We mention an alternative way to compute (say) the expected precision C/A: integrate numerically over the *joint* density of C and A. How can we obtain this density? As $(C, A) = \sum_i (c_i, a_i)$ is a sum of *independent* random length-2 vectors, its mean vector and 2×2 covariance matrix can be respectively found by summing the means and covariance matrices of the (c_i, a_i), each exactly computed from the distribution (5) over K_i hypotheses. We can easily approximate (C, A) by the (continuous) bivariate normal with that mean and covariance matrix[8]—or else accumulate an exact representation of its (discrete) probability mass function by a sequence of numerical convolutions.

6 Experiments

We tested the above training methods on two different tasks: dependency parsing and phrase-based machine translation. Since the basic setup was the same for both, we outline it here before describing the tasks in detail.

In both cases, we start with 8 to 10 models (the "experts") already trained on separate training data. To find the optimal coefficients θ for a log-linear combination of these experts, we use separate development data, using the following procedure due to Och (2003):

1. **Initialization:** Initialize θ to the 0 vector. For each development sentence x_i, set its K_i-best list to $\emptyset$ (thus $K_i = 0$).
2. **Decoding:** For each development sentence x_i, use the current θ to extract the 200 analyses $y_{i,k}$ with the greatest scores $\exp \theta \cdot f_{i,k}$. Calcuate each analysis's loss statistics (e.g., c_i and a_i), and add it to the K_i-best list if it is not already there.
3. **Convergence:** If K_i has not increased for any development sentence, or if we have reached our limit of 20 iterations, stop: the search has converged.
4. **Optimization:** Adjust θ to improve our objective function over the whole development corpus. Return to step 2.

Our experiments simply compare three procedures at step 4. We may either

- **maximize log-likelihood (4)**, a convex function, at a given level of quadratic regularization, by BFGS gradient descent;
- **minimize error (2)** by Och's line search method, which *globally* optimizes each component of θ while holding the others constant;[9] or
- minimize the same error (2) more effectively, by raising $\gamma \to \infty$ while **minimizing the annealed risk (6)**, that is, cooling $T \to -\infty$ (or $\gamma \to \infty$) and at each value, locally minimizing equation (7) using BFGS.

Since these different optimization procedures will usually find different θ at step 4, their K-best lists will diverge after the first iteration.

For final testing, we selected among several variants of each procedure using a separate small heldout set. Final results are reported for a larger, disjoint test set.

6.1 Machine Translation

For our machine translation experiments, we trained phrase-based alignment template models of Finnish-English, French-English, and German-English, as follows. For each language pair, we aligned 100,000 sentence pairs from European Parliament transcripts using GIZA++. We then used Philip Koehn's phrase extraction software to merge the GIZA++ alignments and to extract

[7] BLEU is careful when measuring c_i on a particular decoding $y_{i,k}$. It only counts the first two copies of *the* (e.g.) as correct if *the* occurs at most twice in any reference translation of x_i. This "clipping" does not affect the rest of our method.

[8] Reasonable for a large corpus, by Lyapunov's central limit theorem (allows non-identically distributed summands).

[9] The component whose optimization achieved the lowest loss is then updated. The process iterates until no lower loss can be found. In contrast, Papineni (1999) proposed a linear programming method that may search along diagonal lines.

and score the alignment template model's phrases (Koehn et al., 2003).

The Pharaoh phrase-based decoder uses precisely the setup of this paper. It scores a candidate translation (including its phrasal alignment to the original text) as $\theta \cdot f$, where f is a vector of the following 8 features:

1. the probability of the source phrase given the target phrase
2. the probability of the target phrase given the source phrase
3. the weighted lexical probability of the source words given the target words
4. the weighted lexical probability of the target words given the source words
5. a phrase penalty that fires for each template in the translation
6. a distortion penalty that fires when phrases translate out of order
7. a word penalty that fires for each English word in the output
8. a trigram language model estimated on the English side of the bitext

Our goal was to train the weights θ of these 8 features. We used the method described above, employing the Pharaoh decoder at step 2 to generate the 200-best translations according to the current θ. As explained above, we compared three procedures at step 4: maximum log-likelihood by gradient ascent; minimum error using Och's line-search method; and annealed minimum risk. As our development data for training θ, we used 200 sentence pairs for each language pair.

Since our methods can be tuned with hyperparameters, we used performance on a separate 200-sentence held-out set to choose the best hyperparameter values. The hyperparameter levels for each method were

- **maximum likelihood:** a Gaussian prior with all σ_d^2 at 0.25, 0.5, 1, or ∞
- **minimum error:** 1, 5, or 10 different random starting points, drawn from a uniform distribution on $[-1, 1] \times [-1, 1] \times \cdots$, when optimizing θ at an iteration of step 4.[10]
- **annealed minimum risk:** with explicit entropy constraints, starting temperature $T \in \{100, 200, 1000\}$; stopping temperature $T \in \{0.01, 0.001\}$. The temperature was cooled by half at each step; then we quenched by doubling γ at each step. (We also ran experiments with quadratic regularization with all σ_d^2 at 0.5, 1, or 2 (§4) in addition to the entropy constraint. Also, instead of the entropy constraint, we simply annealed on γ while adding a quadratic regularization term. None of these regularized models beat the best setting of standard deterministic annealing on heldout or test data.)

Optimization Procedure	Finnish-English	French-English	German-English
Max. like.	5.02	5.31	7.43
Min. error	10.27	26.16	20.94
Ann. min. risk	**16.43**	**27.31**	**21.30**

Table 1: BLEU 4n1 percentage on translating 2000-sentence test corpora, after training the 8 experts on 100,000 sentence pairs and fitting their weights θ on 200 more, using settings tuned on a further 200. The current minimum risk annealing method achieved significant improvements over minimum error and maximum likelihood at or below the 0.001 level, using a permutation test with 1000 replications.

Final results on a separate 2000-sentence test set are shown in table 1. We evaluated translation using BLEU with one reference translation and n-grams up to 4. The minimum risk annealing procedure significantly outperformed maximum likelihood and minimum error training in all three language pairs ($p < 0.001$, paired-sample permutation test with 1000 replications).

Minimum risk annealing generally outperformed minimum error training on the held-out set, regardless of the starting temperature T. However, higher starting temperatures do give better performance and a more monotonic learning curve (Figure 3), a pattern that held up on test data. (In the same way, for minimum error training,

[10] That is, we run step 4 from several starting points, finishing at several different points; we pick the finishing point with lowest development error (2). This reduces the sensitivity of this method to the starting value of θ. Maximum likelihood is not sensitive to the starting value of θ because it has only a global optimum; annealed minimum risk is not sensitive to it either, because initially $\gamma \approx 0$, making equation (6) flat.

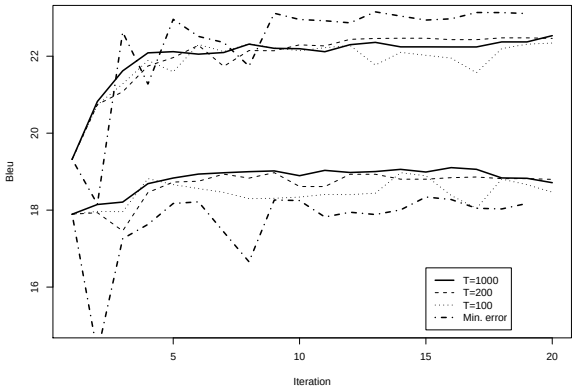


Figure 3: Iterative change in BLEU on German-English development (upper) and held-out (lower), under annealed minimum risk training with different *starting* temperatures, versus minimum error training with 10 random restarts.

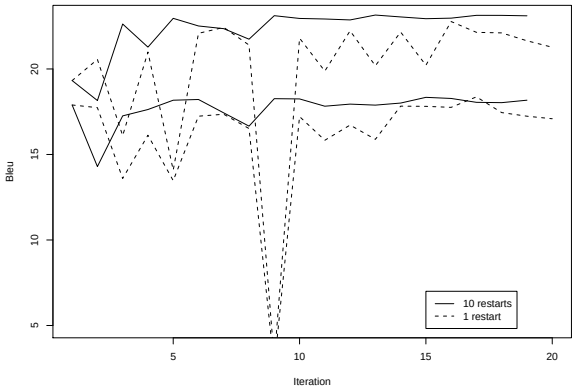


Figure 4: Iterative change in BLEU on German-English development (upper) and held-out (lower), using 10 random restarts vs. only 1.

more random restarts give better performance and a more monotonic learning curve—see Figure 4.)

Minimum risk annealing did *not* always win on the *training* set, suggesting that its advantage is not superior minimization but rather superior generalization: under the risk criterion, *multiple* low-loss hypotheses per sentence can help guide the learner to the right part of parameter space.

Although the components of the translation and language models interact in complex ways, the improvement on Finnish-English may be due in part to the higher weight that minimum risk annealing found for the word penalty. That system is therefore more likely to produce shorter output like *i have taken note of your remarks and i also agree with that .* than like this longer output from the minimum-error-trained system: *i have taken note of your remarks and i shall also agree with all that the union .*

We annealed using our novel expected-BLEU approximation from §5. We found this to perform significantly better on BLEU evaluation than if we trained with a "linearized" BLEU that summed per-sentence BLEU scores (as used in minimum Bayes risk decoding by Kumar and Byrne (2004)).

6.2 Dependency Parsing

We trained dependency parsers for three different languages: Bulgarian, Dutch, and Slovenian.[11] Input sentences to the parser were already tagged for parts of speech. Each parser employed 10 experts, each parameterized as a globally normalized log-linear model (Lafferty et al., 2001). For example, the 9th component of the feature vector $f_{i,k}$ (which described the k^{th} parse of the i^{th} sentence) was the log of that parse's normalized probability according to the 9th expert.

Each expert was trained separately to maximize the conditional probability of the correct parse given the sentence. We used 10 iterations of gradient ascent. To speed training, for each of the first 9 iterations, the gradient was estimated on a (different) sample of only 1000 training sentences.

We then trained the vector θ, used to combine the experts, to minimize the number of labeled dependency attachment errors on a 200-sentence development set. Optimization proceeded over lists of the 200-best parses of each sentence produced by a joint decoder using the 10 experts.

Evaluating on labeled dependency accuracy on 200 test sentences for each language, we see that minimum error and annealed minimum risk training are much closer than for MT. For Bulgarian and Dutch, they are statistically indistinguishable using a paired-sample permutations test with 1000 replications. Indeed, on Dutch, all three optimization procedures produce indistinguishable results. On Slovenian, annealed minimum risk training does show a significant improvement over the other two methods. Overall, however, the results for this task are mediocre. We are still working on improving the underlying experts.

7 Related Work

We have seen that annealed minimum risk training provides a useful alternative to maximum likelihood and minimum error training. In our experiments, it never performed significantly worse

[11] For information on these corpora, see the CoNLL-X shared task on multilingual dependency parsing: `http://nextens.uvt.nl/~conll/`.

Optimization Procedure	labeled dependency acc. [%] Slovenian	Bulgarian	Dutch
Max. like.	27.78	47.23	**36.78**
Min. error	22.52	**54.72**	**36.78**
Ann. min. risk	**31.16**	*54.66*	*36.71*

Table 2: Labeled dependency accuracy on parsing 200-sentence test corpora, after training 10 experts on 1000 sentences and fitting their weights θ on 200 more. For Slovenian, minimum risk annealing is significantly better than the other training methods, while minimum error is significantly worse. For Bulgarian, both minimum error and annealed minimum risk training achieve significant gains over maximum likelihood, but are indistinguishable from each other. For Dutch, the three methods are indistinguishable.

than either and in some cases significantly helped. Note, however, that annealed minimum risk training results in a deterministic classifier just as these other training procedures do. The orthogonal technique of **minimum Bayes risk decoding** has achieved gains on parsing (Goodman, 1996) and machine translation (Kumar and Byrne, 2004). In speech recognition, researchers have improved decoding by smoothing probability estimates numerically on heldout data in a manner reminiscent of annealing (Goel and Byrne, 2000). We are interested in applying our techniques for approximating nonlinear loss functions to MBR by performing the risk minimization inside the dynamic programming or other decoder.

Another training approach that incorporates arbitrary loss functions is found in the *structured prediction* literature in the margin-based-learning community (Taskar et al., 2004; Crammer et al., 2004). Like other max-margin techniques, these attempt to make the best hypothesis far away from the inferior ones. The distinction is in using a loss function to calculate the required margins.

8 Conclusions

Despite the challenging shape of the error surface, we have seen that it is practical to optimize task-specific error measures rather than optimizing likelihood—it produces lower-error systems. Different methods can be used to attempt this global, non-convex optimization. We showed that for MT, and sometimes for dependency parsing, an annealed minimum risk approach to optimization performs significantly better than a previous line-search method that does not smooth the error surface. It never does significantly worse. With such improved methods for minimizing error, we can hope to make better use of task-specific training criteria in NLP.

References

L. R. Bahl, P. F. Brown, P. V. de Souza, and R. L. Mercer. 1988. A new algorithm for the estimation of hidden Markov model parameters. In *ICASSP*, pages 493–496.

E. Charniak and M. Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *ACL*, pages 173–180.

S. F. Chen and R. Rosenfeld. 1999. A gaussian prior for smoothing maximum entropy models. Technical report, CS Dept., Carnegie Mellon University.

K. Crammer, R. McDonald, and F. Pereira. 2004. New large margin algorithms for structured prediction. In *Learning with Structured Outputs (NIPS)*.

M. Dreyer, D. A. Smith, and N. A. Smith. 2006. Vine parsing and minimum risk reranking for speed and precision. In *CoNLL*.

G. Elidan and N. Friedman. 2005. Learning hidden variable networks: The information bottleneck approach. *JMLR*, 6:81–127.

V. Goel and W. J. Byrne. 2000. Minimum Bayes-Risk automatic speech recognition. *Computer Speech and Language*, 14(2):115–135.

J. T. Goodman. 1996. Parsing algorithms and metrics. In *ACL*, pages 177–183.

G. Hinton. 1999. Products of experts. In *Proc. of ICANN*, volume 1, pages 1–6.

K.-U. Hoffgen, H.-U. Simon, and K. S. Van Horn. 1995. Robust trainability of single neurons. *J. of Computer and System Sciences*, 50(1):114–125.

D. S. Johnson and F. P. Preparata. 1978. The densest hemisphere problem. *Theoretical Comp. Sci.*, 6(93–107).

S. Katagiri, B.-H. Juang, and C.-H. Lee. 1998. Pattern recognition using a family of design algorithms based upon the generalized probabilistic descent method. *Proc. IEEE*, 86(11):2345–2373, November.

P. Koehn, F. J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *HLT-NAACL*, pages 48–54.

S. Kumar and W. Byrne. 2004. Minimum bayes-risk decoding for statistical machine translation. In *HLT-NAACL*.

J. Lafferty, A. McCallum, and F. C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*.

F. J. Och. 2003. Minimum error rate training in statistical machine translation. In *ACL*, pages 160–167.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *ACL*, pages 311–318.

K. A. Papineni. 1999. Discriminative training via linear programming. In *ICASSP*.

A. Rao and K. Rose. 2001. Deterministically annealed design of Hidden Markov Model speech recognizers. *IEEE Trans. on Speech and Audio Processing*, 9(2):111–126.

K. Rose. 1998. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. *Proc. IEEE*, 86(11):2210–2239.

N. A. Smith and J. Eisner. 2004. Annealing techniques for unsupervised statistical language learning. In *ACL*, pages 486–493.

A. Smith, T. Cohn, and M. Osborne. 2005. Logarithmic opinion pools for conditional random fields. In *ACL*, pages 18–25.

B. Taskar, D. Klein, M. Collins, D. Koller, and C. Manning. 2004. Max-margin parsing. In *EMNLP*, pages 1–8.

N. Ueda and R. Nakano. 1998. Deterministic annealing EM algorithm. *Neural Networks*, 11(2):271–282.

Unsupervised Induction of Modern Standard Arabic Verb Classes Using Syntactic Frames and LSA

Neal Snider
Linguistics Department
Stanford University
Stanford, CA 94305
snider@stanford.edu

Mona Diab
Center for Computational Learning Systems
Columbia University
New York, NY 10115
mdiab@cs.columbia.edu

Abstract

We exploit the resources in the Arabic Treebank (ATB) and Arabic Gigaword (AG) to determine the best features for the novel task of automatically creating lexical semantic verb classes for Modern Standard Arabic (MSA). The verbs are classified into groups that share semantic elements of meaning as they exhibit similar syntactic behavior. The results of the clustering experiments are compared with a gold standard set of classes, which is approximated by using the noisy English translations provided in the ATB to create Levin-like classes for MSA. The quality of the clusters is found to be sensitive to the inclusion of syntactic frames, LSA vectors, morphological pattern, and subject animacy. The best set of parameters yields an $F_{\beta=1}$ score of 0.456, compared to a random baseline of an $F_{\beta=1}$ score of 0.205.

1 Introduction

The creation of the Arabic Treebank (ATB) and Arabic Gigaword (AG) facilitates corpus based studies of many interesting linguistic phenomena in Modern Standard Arabic (MSA).[1] The ATB comprises manually annotated morphological and syntactic analyses of newswire text from different Arabic sources, while the AG is simply a huge collection of raw Arabic newswire text. In our ongoing project, we exploit the ATB and AG to determine the best features for the novel task of automatically creating lexical semantic verb classes for MSA. We are interested in the problem of classifying verbs in MSA into groups that share semantic elements of meaning as they exhibit similar syntactic behavior. This manner of classifying verbs in a language is mainly advocated by Levin (1993). The Levin Hypothesis (LH) contends that verbs that exhibit similar syntactic behavior share element(s) of meaning. There exists a relatively extensive classification of English verbs according to different syntactic alternations. Numerous linguistic studies of other languages illustrate that LH holds cross linguistically, in spite of variations in the verb class assignment. For example, in a wide cross-linguistic study, Guerssel et al (1985) found that the Conative Alternation exists in the Austronesian language Warlpiri. As in English, the alternation is found with *hit-* and *cut*-type verbs, but not with *touch-* and *break*-type verbs.

A strong version of the LH claims that comparable syntactic alternations hold cross-linguistically. Evidence against this strong version of LH is presented by Jones et al (1994). For the purposes of this paper, we maintain that although the syntactic alternations will differ across languages, the semantic similarities that they signal will hold cross linguistically. For Arabic, a significant test of LH has been the work of Fareh and Hamdan (2000), who argue the existence of the Locative Alternation in Jordanian Arabic. However, to date no general study of MSA verbs and alternations exists. We address this problem by automatically inducing such classes, exploiting explicit syntactic and morphological information in the ATB using unsupervised clustering techniques.

This paper is an extension of our previous work in Snider and Diab (2006), which found a preliminary effect of syntactic frames on the precision of MSA verb clustering. In this work, we find

[1] http://www.ldc.upenn.edu/

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 795–802,
Sydney, July 2006.

effects of three more features, and report results using both precision and recall. This project is inspired by previous approaches to automatically induce lexical semantic classes for English verbs, which have met with success (Merlo and Stevenson, 2001; Schulte im Walde, 2000) , comparing their results with manually created Levin verb classes. However, Arabic morphology has well known correlations with the kind of event structure that forms the basis of the Levin classification. (Fassi-Fehri, 2003). This characteristic of the language makes this a particularly interesting task to perform in MSA. Thus, the scientific goal of this project is to determine the features that best aid verb clustering, particularly the language-specific features that are unique to MSA and related languages.

Inducing such classes automatically allows for a large-scale study of different linguistic phenomena within the MSA verb system, as well as cross-linguistic comparison with their English counterparts. Moreover, drawing on generalizations yielded by such a classification could potentially be useful in several NLP problems such as Information Extraction, Event Detection, Information Retrieval and Word Sense Disambiguation, not to mention the facilitation of lexical resource creation such as MSA WordNets and ontologies.

Unfortunately, a gold standard resource comparable to Levin's English classification for evaluation does not exist in MSA. Therefore, in this paper, as before, we evaluate the quality of the automatically induced MSA verb classes both qualitatively and quantitatively against a noisy MSA translation of Levin classes in an attempt to create such classes for MSA verbs.

The paper is organized as follows: Section 2 describes Levin classes for English; Section 3 describes some relevant previous work; In Section 4 we discuss relevant phenomena of MSA morphology and syntax; In Section 5, we briefly describe the clustering algorithm; Section 6 gives a detailed account of the features we use to induce the verb clusters; Then, Section 7, describes our evaluation data, metric, gold standard and results; In Section 8, we discuss the results and draw on some quantitative and qualitative observations of the data; Finally, we conclude this paper in Section 9 with concluding remarks and a look into future directions.

2 Levin Classes

The idea that verbs form lexical semantic clusters based on their syntactic frames and argument selection preferences is inspired by the work of Levin, who defined classes of verbs based on their syntactic alternation behavior. For example, the class `Vehicle Names` (*e.g. bicycle, canoe, skate, ski*) is defined by the following syntactic alternations (among others):

1. INTRANSITIVE USE, optionally followed by a path
   They skated (along the river bank).
2. INDUCED ACTION (some verbs)
   Pat skated (Kim) around the rink.

Levin lists 184 manually created classes for English, which is not intended as an exhaustive classification. Many verbs are in multiple classes both due to the inherent polysemy of the verbs as well as other aspectual variations such as argument structure preferences. As an example of the latter, a verb such as *eat* occurs in two different classes; one defined by the *Unspecified Object Alternation* where it can appear both with and without an explicit direct object, and another defined by the *Connative Alternation* where its second argument appears either as a direct object or the object of the preposition *at*. It is important to note that the Levin classes aim to group verbs based on their event structure, reflecting aspectual and manner similarities rather than similarity due to their describing the same or similar events. Therefore, the semantic class similarity in Levin classes is coarser grained than what one would expect resulting from a semantic classification based on distributional similarity such as Latent Semantic Analysis (LSA) algorithms. For illustration, one would expect an LSA algorithm to group *skate, rollerblade* in one class and *bicycle, motorbike, scooter* in another; yet Levin puts them all in the same class based on their syntactic behavior, which reflects their common event structure: an activity with a possible causative participant. One of the purposes of this work is to test this hypothesis by examining the relative contributions of LSA and syntactic frames to verb clustering.

3 Related Work

Based on the Levin classes, many researchers attempt to induce such classes automatically. No-

tably the work of Merlo and Stevenson (2001) attempts to induce three main English verb classes on a large scale from parsed corpora, the class of Ergative, Unaccusative, and Object-drop verbs. They report results of 69.8% accuracy on a task whose baseline is 34%, and whose expert-based upper bound is 86.5%. In a task similar to ours except for its use of English, Schulte im Walde clusters English verbs semantically by using their alternation behavior, using frames from a statistical parser combined with WordNet classes. She evaluates against the published Levin classes, and reports that 61% of all verbs are clustered into correct classes, with a baseline of 5%.

4 Arabic Linguistic Phenomena

In this paper, the language of interest is MSA. Arabic verbal morphology provides an interesting piece of explicit lexical semantic information in the lexical form of the verb. Arabic verbs have two basic parts, the root and pattern/template, which combine to form the basic derivational form of a verb. Typically a root consists of three or four consonants, referred to as radicals. A pattern, on the other hand, is a distribution of vowel and consonant affixes on the root resulting in Arabic derivational lexical morphology. As an example, the root *k t b*,[2] if interspersed with the pattern *1a2a3* – the numbers correspond to the positions of the first, second and third radicals in the root, respectively – yields *katab* meaning *write*. However, if the pattern were *ma1A2i3*, resulting in the word *makAtib*, it would mean *offices/desks* or *correspondences*. There are fifteen pattern forms for MSA verbs, of which ten are commonly used. Not all verbs occur with all ten patterns. These root-pattern combinations tend to indicate a particular lexical semantic event structure in the verb.

5 Clustering

Taking the linguistic phenomena of MSA as features, we apply clustering techniques to the problem of inducing verb classes. We showed in Snider & Diab (2006) that soft clustering performs best on this task compared to hard clustering, therefore we employ soft clustering techniques to induce the verb classes here. Clustering algorithms partition a set of data into groups, or clusters based on a similarity metric. Soft clustering allows elements to be members of multiple clusters simultaneously, and have degrees of membership in all clusters. This membership is sometimes represented in a probabilistic framework by a distribution $P(x_i, c)$, which characterizes the probability that a verb x_i is a member of cluster c.

6 Features

Syntactic frames The syntactic frames are defined as the sister constituents of the verb in a Verb Phrase (VP) constituent, namely, Noun Phrases (NP), Prepositional Phrases (PP), and Sentential Complements (SBARs and Ss). Not all of these constituents are necessarily arguments of the verb, so we take advantage of functional tag annotations in the ATB. Hence, we only include NPs with function annotation: subjects (NP-SBJ), topicalized subjects (NP-TPC),[3] objects (NP-OBJ), and second objects in dative constructions (NP-DTV). The PPs deemed relevant to the particular sense of the verb are tagged by the ATB annotators as PP-CLR. We assume that these are argument PPs, and include them in our frames. Finally, we include sentential complements (SBAR and S). While some of these will no doubt be adjuncts (i.e. purpose clauses and the like), we assume that those that are arguments will occur in greater numbers with particular verbs, while adjuncts will be randomly distributed with all verbs.

Given Arabic's somewhat free constituent order, frames are counted as the same when they contain the same constituents, regardless of order. Also, for each constituent that is headed by a function word (PPs and SBARs) such as prepositions and complementizers, the headword is extracted to include syntactic alternations that are sensitive to preposition or complementizer type. It is worth noting that this corresponds to the FRAME1 configuration described in our previous study.(Snider and Diab, 2006) Finally, only active verbs are included in this study, rather than attempt to reconstruct the argument structure of passives.

Verb pattern The ATB includes morphological analyses for each verb resulting from the Buckwalter Analyzer (BAMA).[4] For each verb, one of the analyses resulting from BAMA is chosen manually by the treebankers. The analyses are

[2] All Arabic in the paper is rendered in the Buckwalter transliteration scheme http::://www.ldc.upenn.edu.

[3] These are displaced NP-SBJ marked differently in the ATB to indicate SVO order rather than the canonical VSO order in MSA. NP-TPC occurs in 35% of the ATB.

[4] http://www.ldc.upenn.edu

matched with the root and pattern information derived manually in a study by Nizar Habash (personal communication).This feature is of particular scientific interest because it is unique to Semitic languages, and, as mentioned above, has an interesting potential correlation with argument structure.

Subject animacy In an attempt to allow the clustering algorithm to use information closer to actual argument structure than mere syntactic frames, we add a feature that indicates whether a verb requires an animate subject. Merlo and Stevenson (2001) found that this feature improved their English verb clusters, but in Snider & Diab (2006), we found this feature to not contribute significantly to Arabic verb clustering quality. However, upon further inspection of the data, we discovered we could improve the quality of this feature extraction in this study. Automatically determining animacy is difficult because it requires extensive manual annotation or access to an external resource such as WordNet, neither of which currently exist for Arabic. Instead we rely on an approximation that takes advantage of two generalizations from linguistics: the animacy hierarchy and zero-anaphora. According to the animacy hierarchy, as described in Silverstein (1976), pronouns tend to describe animate entities. Following a technique suggested by Merlo and Stevenson(2001), we take advantage of this tendency by adding a feature that is the number of times each verb occurs with a pronominal subject. We also take advantage of the phenomenon of zero-anaphora, or pro-drop, in Arabic as an additional indicator subject animacy. Pro-drop is a common phenomenon in Romance languages, as well as Semitic languages, where the subject is implicit and the only indicator of a subject is incorporated in the conjugation of the verb. According to work on information structure in discourse (Vallduví, 1992), pro-drop tends to occur with more given and animate subjects. To capture this generalization, we add a feature for the frequency with which a given verb occurs without an explicit subject. We further hypothesize that proper names are more likely to describe animates (humans, or organizations which metonymically often behave like animates), adding a feature for the frequency with which a given verb occurs with a proper name. With these three features, we provide the clustering algorithm with subject animacy indicators.

LSA semantic vector This feature is the semantic vector for each verb, as derived by Latent Semantic Analysis (LSA) of the AG. LSA is a dimensionality reduction technique that relies on Singular Value Decomposition (SVD) (Landauer and Dumais, 1997). The main strength in applying LSA to large quantities of text is that it discovers the latent similarities between concepts. It may be viewed as a form of clustering in conceptual space.

7 Evaluation

7.1 Data Preparation

The four sets of features are cast as the column dimensions of a matrix, with the MSA lemmatized verbs constituting the row entries. The data used for the syntactic frames is obtained from the ATB corresponding to ATB1v3, ATB2v2 and ATB3v2. The ATB is a collection of 1800 stories of newswire text from three different press agencies, comprising a total of $800,000$ Arabic tokens after clitic segmentation. The domain of the corpus covers mostly politics, economics and sports journalism. To extract data sets for the frames, the treebank is first lemmatized by looking up lemma information for each word in its manually chosen (information provided in the Treebank files) corresponding output of BAMA. Next, each active verb is extracted along with its sister constituents under the VP in addition to NP-TPC. As mentioned above, the only constituents kept as the frame are those labeled NP-TPC, NP-SBJ, NP-OBJ, NP-DTV, PP-CLR, and SBAR. For PP-CLRs and SBARs, the head preposition or complementizer which is assumed to be the left-most daughter of the phrase, is extracted. The verbs and frames are put into a matrix where the row entries are the verbs and the column entries are the frames. The elements of the matrix are the frequency of the row verb occurring in a given frame column entry. There are 2401 verb types and 320 frame types, corresponding to 52167 total verb frame tokens.

For the LSA feature, we apply LSA to the AG corpus. AG (GIGAWORD 2) comprises 481 million words of newswire text. The AG corpus is morphologically disambiguated using MADA.[5] MADA is an SVM based system that disambiguates among different morphological analyses produced by BAMA.(Habash and Rambow, 2005) We extract the lemma forms of all the words in AG

[5] http://www.ccls.columbia.edu/cadim/resources

and use them for the LSA algorithm. To extract the LSA vectors, first the lemmatized AG data is split into 100 sentence long pseudo-documents. Next, an LSA model is trained using the Infomap software [6] on half of the AG (due to size limitations of Infomap). Infomap constructs a word similarity matrix in document space, then reduces the dimensionality of the data using SVD. LSA reduces AG to 44 dimensions. The 44-dimensional vector is extracted for each verb, which forms the LSA data set for clustering.

Subject animacy information is represented as three feature columns in our matrix. One column entry represents the frequency a verb co-occurs with an empty subject (represented as an NP-SBJ dominating the NONE tag, 21586 tokens). Another column has the frequency the NP-SBJ/NP-TPC dominates a pronoun (represented in the corpus as the tag PRON 3715 tokens). Finally, the last subject animacy column entry represents the frequency an NP-SBJ/NP-TPC dominates a proper name (tagged NOUN_PROP, 4221 tokens).

The morphological pattern associated with each verb is extracted by looking up the lemma in the output of BAMA. The pattern information is added as a feature column to our matrix of verbs by features.

7.2 Gold Standard Data

The gold standard data is created automatically by taking the English translations corresponding to the MSA verb entries provided with the ATB distributions. We use these English translations to locate the lemmatized MSA verbs in the Levin English classes represented in the Levin Verb Index (Levin, 1993), thereby creating an approximated MSA set of verb classes corresponding to the English Levin classes. Admittedly, this is a crude manner to create a gold standard set. Given lack of a pre-existing classification for MSA verbs, and the novelty of the task, we consider it a first approximation step towards the creation of a real gold standard classification set in the near future. Since the translations are assigned manually to the verb entries in the ATB, we assume that they are a faithful representation of the MSA language used. Moreover, we contend that lexical semantic meanings, if they hold cross linguistically, would be defined by distributions of syntactic alternations. Unfortunately, this gold standard set is more noisy than expected due to several factors: each MSA morphological analysis in the ATB has several associated translations, which include both polysemy and homonymy. Moreover, some of these translations are adjectives and nouns as well as phrasal expressions. Such divergences occur naturally but they are rampant in this data set. Hence, the resulting Arabic classes are at a finer level of granularity than their English counterparts because of missing verbs in each cluster. There are also many gaps – unclassified verbs – when the translation is not a verb, or a verb that is not in the Levin classification. Of the 480 most frequent verb types used in this study, 74 are not in the translated Levin classification.

[6] http://infomap.stanford.edu/

7.3 Clustering Algorithms

We use the clustering algorithms implemented in the library *cluster* (Kaufman and Rousseeuw, 1990) in the *R* statistical computing language. The soft clustering algorithm, called FANNY, is a type of fuzzy clustering, where each observation is "spread out" over various clusters. Thus, the output is a membership function $P(x_i, c)$, the membership of element x_i to cluster c. The memberships are nonnegative and sum to 1 for each fixed observation. The algorithm takes k, the number of clusters, as a parameter and uses a Euclidean distance measure. We determine k empirically, as explained below.

7.4 Evaluation Metric

The evaluation metric used here is a variation on an F-score derived for hard clustering (Chklovski and Mihalcea, 2003). The result is an F_β measure, where β is the coefficient of the relative strengths of precision and recall. $\beta = 1$ for all results we report. The score measures the maximum overlap between a hypothesized cluster (HYP) and a corresponding gold standard cluster (GOLD), and computes a weighted average across all the GOLD clusters:

$$F_\beta = \sum_{C \in \mathcal{C}} \frac{\|C\|}{V_{tot}} \max_{A \in \mathcal{A}} \frac{(\beta^2 + 1)\|A \cap C\|}{\beta^2 \|C\| + \|A\|}$$

$\mathcal{A}$ is the set of HYP clusters, $\mathcal{C}$ is the set of GOLD clusters, and $V_{tot} = \sum_{C \in \mathcal{C}} \|C\|$ is the total number of verbs to be clustered. This is the measure that we report, which weights precision and recall equally.

7.5 Results

To determine the features that yield the best clustering of the extracted verbs, we run tests comparing seven different factors of the model, in a $2x2x2x2x3x3x5$ design, with the first four parameters being the substantive informational factors, and the last three being parameters of the clustering algorithm. For the feature selection experiments, the informational factors all have two conditions, which encode the presence or absence of the information associated with them. The first factor represents the syntactic frame vectors, the second the LSA semantic vectors, the third the subject animacy, and the fourth the morphological pattern of the verb.

The fifth through seventh factors are parameters of the clustering algorithm: The fifth factor is three different numbers of verbs clustered: the 115, 268, and 406 most frequent verb types, respectively. The sixth factor represents numbers of clusters (*k*). These values are dependent on the number of verbs tested at a time. Therefore, this factor is represented as a fraction of the number of verbs. Hence, the chosen values are $\frac{1}{6}$, $\frac{1}{3}$, and $\frac{1}{2}$ of the number of verbs. The seventh and last factor is a threshold probability used to derive discrete members for each cluster from the cluster probability distribution as rendered by the soft clustering algorithm. In order to get a good range of the variation in the effect of the threshold, we empirically choose five different threshold values: 0.03, 0.06, 0.09, 0.16, and 0.21. The purpose of the last three factors is to control for the amount of variation introduced by the parameters of the clustering algorithm, in order to determine the effect of the informational factors. Evaluation scores are obtained for all combinations of all seven factors (minus the no information condition - the algorithm must have some input!), resulting in 704 conditions.

We compare our best results to a random baseline. In the baseline, verbs are randomly assigned to clusters where a random cluster size is on average the same size as each other and as GOLD.[7] The highest overall scored $F_{\beta=1}$ is 0.456 and it results from using syntactic frames, LSA vectors, subject animacy, 406 verbs, 202 clusters, and a threshold of 0.16. The average cluster size is 3, because this is a soft clustering. The random baseline achieves an overall $F_{\beta=1}$ of 0.205 with comparable settings of 406 verbs randomly assigned to 202 clusters of approximately equal size.

To determine which features contribute significantly to clustering quality, a statistical analysis of the clustering experiments is undertaken in the next section.

8 Discussion

For further quantitative error analysis of the data and feature selection, we perform an ANOVA to test the significance of the differences among information factors and the various parameter settings of the clustering algorithm. This error analysis uses the error metric from Snider & Diab (2006) that allows us to test just the HYP verbs that match the GOLD set. The emphasis on precision in the feature selection serves the purpose of countering the large underestimation of recall that is due to a noisy gold standard. We believe that the features that are found to be significant by this metric stand the best chance of being useful once a better gold standard is available.

The ANOVA analyzes the effects of syntactic frame, LSA vectors, subject animacy, verb pattern, verb number, cluster number, and threshold. Syntactic frame information contributes positively to clustering quality ($p < .03$), as does LSA ($p < .001$). Contrary to the result in Snider & Diab (2006), subject animacy has a significant positive contribution ($p < .002$). Interestingly, the morphological pattern contributes negatively to clustering quality ($p < .001$). As expected, the control parameters all have a significant effect: number of verbs ($p < .001$), number of clusters ($p < .001$), and threshold ($p < .001$).

As evident from the results of the statistical analysis, the various informational factors have an interesting effect on the quality of the clusters. Both syntactic frames and LSA vectors contribute independently to clustering quality. This indicates that successfully clustering verbs requires information at the relatively coarse level of event structure, as well as the finer grained semantics provided by word co-occurrence techniques such as LSA.

Subject animacy is found to improve clustering, which is consistent with the results for English found by Merlo and Stevenson. This is definite improvement over our previous study, and indicates

[7] It is worth noting that this gives an added advantage to the random baseline, since a comparable to GOLD size implicitly contibutes to a higher overlap score.

that the extraction of the feature has been much improved.

Most interesting from a linguistic perspective is the finding that morphological pattern information about the verb actually worsens clustering quality. This could be explained by the fact that the morphological patterns are productive, so that two different verb lemmas actually describe the same event structure. This would worsen the clustering because these morphological alternations that are represented by the different patterns actually change the lemma form of the verb, unlike syntactic alternations. If only syntactic alternation features are taken into account, the different pattern forms of the same root could still be clustered together; however, our design of the pattern feature does not allow for variation in the lemma form, therefore, we are in effect preventing the useful exploitation of the pattern information. Further evidence comes from the positive effect of the LSA feature, which effectively clusters together these productive patterns hence yielding the significant impact on the clustering.

Overall, the scores that we report use the evaluation metric that equally weights precision and recall. This metric disfavors clusters that are too large or too small. Models perform better when the average size of HYP is the same as that of GOLD. It is worth noting that comparing our current results to those obtained in Snider & Diab (2006), we show a significant improvement given the same precision oriented metric. In the same condition settings, our previous results are an F_β score of 0.51 and in this study, a precision oriented metric yields a significant improvement of 17 absolute points, at an F_β score of 0.68. Even though we do not report this number as the main result of our study, we tend to have more confidence in it due to the noise associated with the GOLD set.

The score of the best parameter settings with respect to the baseline is considerable given the novelty of the task and lack of good quality resources for evaluation. Moreover, there is no reason to expect that there would be perfect alignment between the Arabic clusters and the corresponding translated Levin clusters, primarily because of the quality of the translation, but also because there is unlikely to be an isomorphism between English and Arabic lexical semantics, as assumed here as a means of approximating the problem. In fact, it would be quite noteworthy if we did find a high level of agreement.

In an attempt at a qualitative analysis of the resulting clusters, we manually examine four HYP clusters.

- The first cluster includes the verbs *>aloqaY [meet], $ahid [view], >ajoraY [run an interview], {isotaqobal [receive a guest], Eaqad [hold a conference], >aSodar [issue]*. We note that they all share the concept of convening, or formal meetings. The verbs are clearly related in terms of their event structure (they are all activities, without an associated change of state) yet are not semantically similar. Therefore, our clustering approach yields a classification that is on par with the Levin classes in the coarseness of the cluster membership granularity.

- The second consists of *∗akar [mention], >afAd [report]* which is evaluated against the GOLD cluster class comprising the verbs *>aEolan [announce], >a$Ar [indicate], ∗akar [mention], >afAd [report], Sar∼aH [report/confirm], $ahid [relay/witness], ka$af [uncover]* corresponding to the `Say Verb` Levin class. The HYP cluster, though correct, loses significantly on recall. This is due to the low frequency of some of the verbs in the GOLD set, which in turn affects the overall score of this HYP cluster.

- Finally, the HYP cluster comprising *Eamil [work continuously on], jA' [occur], {isotamar [continue], zAl [remain], baqiy [remain], jaraY [occur]* corresponds to the `Occurrence Verb` Levin class. The corresponding GOLD class comprises *jA' [occur], HaSal [happen], jaraY [occur]*. The HYP cluster contains most of the relevant verbs and adds others that would fall in that same class such as *{isotamar [continue], zAl [remain], baqiy [remain]*, since they have similar syntactic diagnostics where they do not appear in the transitive uses and with locative inversions. However they are not found in the Levin English class since it is not a comprehensive listing of all English verbs.

In summary, we observe very interesting clusters of verbs which indeed require more in depth lexical semantic study as MSA verbs in their own right.

9 Conclusions

We found new features that help us successfully perform the novel task of applying clustering techniques to verbs acquired from the ATB and AG to induce lexical semantic classes for MSA verbs. In doing this, we find that the quality of the clusters is sensitive to the inclusion of information about the syntactic frames, word co-occurence (LSA), and animacy of the subject, as well as parameters of the clustering algorithm such as the number of clusters, and number of verbs clustered. Our classification performs well with respect to a gold standard clusters produced by noisy translations of English verbs in the Levin classes. Our best clustering condition when we use all frame information and the most frequent verbs in the ATB and a high number of clusters outperforms a random baseline by $F_{\beta=1}$ difference of 0.251. This analysis leads us to conclude that the clusters are induced from the structure in the data

Our results are reported with a caveat on the gold standard data. We are in the process of manually cleaning the English translations corresponding to the MSA verbs. Moreover, we are exploring the possibility of improving the gold standard clusters by examining the lexical semantic attributes of the MSA verbs. We also plan to add semantic word co-occurrence information via other sources besides LSA, to determine if having semantic components in addition to the argument structure component improves the quality of the clusters. Further semantic information will be acquired from a WordNet similarity of the cleaned translated English verbs. In the long term, we envision a series of psycholinguistic experiments with native speakers to determine which Arabic verbs group together based on their argument structure.

Acknowledgements We would like to thank three anonymous reviewers for their helpful comments. We would like to acknowledge Nizar Habash for supplying us with a pattern and root list for MSA verb lemmas. The second author was supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-06-C-0023.

References

T. Chklovski and R. Mihalcea. 2003. Exploiting agreement and disagreement of human annotators for word sense disambiguation. In *Proceedings of Recent Advances In NLP (RANLP 2003)*.

Shehdeh Fareh and Jihad Hamdan. 2000. Locative alternation in english and jordanian spoken arabic. In *Poznan Studies in Contemporary Linguistics*, volume 36, pages 71–93. School of English, Adam Mickiewicz University, Poznan, Poland.

Abdelkader Fassi-Fehri. 2003. Verbal plurality, transitivity, and causativity. In *Research in Afroasiatic Grammar*, volume 11, pages 151–185. John Benjamins, Amsterdam.

M. Guerssel, K. Hale, M. Laughren, B. Levin, and J. White Eagle. 1985. A cross linguistic study of transitivity alternations. In *Papers from the Parasession on Causatives and Agentivity*, volume 21:2, pages 48–63. CLS, Chicago.

Nizar Habash and Owen Rambow. 2005. Arabic tokenization, morphological analysis, and part-of-speech tagging in one fell swoop. In *Proceedings of the Conference of American Association for Computational Linguistics (ACL05)*.

D. Jones. 1994. Working papers and projects on verb class alternations in Bangla, German, English, and Korean. AI Memo 1517, MIT.

L. Kaufman and P.J. Rousseeuw. 1990. *Finding Groups in Data*. John Wiley and Sons, New York.

Thomas K. Landauer and Susan T. Dumais. 1997. A solution to platos problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review*, 104:2:211–240.

Beth Levin. 1993. *English Verb Classes and Alternations: A Preliminary Investigation*. University of Chicago Press, Chicago.

Paola Merlo and Suzanne Stevenson. 2001. Automatic verb classification based on statistical distributions of argument structure. *Computational Linguistics*, 27(4).

Sabine Schulte im Walde. 2000. Clustering verbs semantically according to their alternation behaviour. In *18th International Conference on Computational Linguistics (COLING 2000)*, Saarbrucken, Germany.

Michael Silverstein. 1976. Hierarchy of features and ergativity. In Robert Dixon, editor, *Grammatical Categories in Australian Languages*. Australian Institute of Aboriginal Studies, Canberra.

N. Snider and M. Diab. 2006. Unsupervized induction of modern standard arabic verb classes. In *Proceedings of HLT-NAACL*.

Enric Vallduví. 1992. *The Informational Component*. Garland, New York.

Discourse Generation Using Utility-Trained Coherence Models

Radu Soricut
Information Sciences Institute
University of Southern California
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292
radu@isi.edu

Daniel Marcu
Information Sciences Institute
University of Southern California
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292
marcu@isi.edu

Abstract

We describe a generic framework for integrating various stochastic models of discourse coherence in a manner that takes advantage of their individual strengths. An integral part of this framework are algorithms for searching and training these stochastic coherence models. We evaluate the performance of our models and algorithms and show empirically that utility-trained log-linear coherence models outperform each of the individual coherence models considered.

1 Introduction

Various theories of discourse coherence (Mann and Thompson, 1988; Grosz et al., 1995) have been applied successfully in discourse analysis (Marcu, 2000; Forbes et al., 2001) and discourse generation (Scott and de Souza, 1990; Kibble and Power, 2004). Most of these efforts, however, have limited applicability. Those that use manually written rules model only the most visible discourse constraints (e.g., the discourse connective "although" marks a CONCESSION relation), while being oblivious to fine-grained lexical indicators. And the methods that utilize manually annotated corpora (Carlson et al., 2003; Karamanis et al., 2004) and supervised learning algorithms have high costs associated with the annotation procedure, and cannot be easily adapted to different domains and genres.

In contrast, more recent research has focused on stochastic approaches that model discourse coherence at the local lexical (Lapata, 2003) and global levels (Barzilay and Lee, 2004), while preserving regularities recognized by classic discourse theories (Barzilay and Lapata, 2005). These stochastic coherence models use simple, non-hierarchical representations of discourse, and can be trained with minimal human intervention, using large collections of existing human-authored documents. These models are attractive due to their increased scalability and portability. As each of these stochastic models captures different aspects of coherence, an important question is whether we can combine them in a model capable of exploiting all coherence indicators.

A frequently used testbed for coherence models is the discourse ordering problem, which occurs often in text generation, complex question answering, and multi-document summarization: given N discourse units, what is the most coherent ordering of them (Marcu, 1996; Lapata, 2003; Barzilay and Lee, 2004; Barzilay and Lapata, 2005)? Because the problem is NP-complete (Althaus et al., 2005), it is critical how coherence model evaluation is intertwined with search: if the search for the best ordering is greedy and has many errors, one is not able to properly evaluate whether a model is better than another. If the search is exhaustive, the ordering procedure may take too long to be useful.

In this paper, we propose an A* search algorithm for the discourse ordering problem that comes with strong theoretical guarantees. For a wide range of practical problems (discourse ordering of up to 15 units), the algorithm finds an optimal solution in reasonable time (on the order of seconds). A beam search version of the algorithm enables one to find good, approximate solutions for very large reordering tasks. These algorithms enable us not only to compare head-to-head, for the first time, a set of coherence models, but also to combine these models so as to benefit from their complementary strengths. The model com-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 803–810,
Sydney, July 2006.

bination is accomplished using statistically well-founded utility training procedures which automatically optimize the contributions of the individual models on a development corpus. We empirically show that utility-based models of discourse coherence outperform each of the individual coherence models considered.

In the following section, we describe previously-proposed and new coherence models. Then, we present our search algorithms and the input representation they use. Finally, we show evaluation results and discuss their implications.

2 Stochastic Models of Discourse Coherence

2.1 Local Models of Discourse Coherence

Stochastic local models of coherence work under the assumption that well-formed discourse can be characterized in terms of specific distributions of local recurring patterns. These distributions can be defined at the lexical level or entity-based levels.

Word-Coocurrence Coherence Models. We propose a new coherence model, inspired by (Knight, 2003), that models the intuition that the usage of certain words in a discourse unit (sentence) tends to trigger the usage of other words in subsequent discourse units. (A similar intuition holds for the Machine Translation models generically known as the IBM models (Brown et al., 1993), which assume that certain words in a source language sentence tend to trigger the usage of certain words in a target language translation of that sentence.)

We train models able to recognize local recurring patterns of word usage across sentences in an unsupervised manner, by running an Expectation-Maximization (EM) procedure over pairs of consecutive sentences extracted from a large collection of training documents[1]. We expect EM to detect and assign higher probabilities to recurring word patterns compared to casually occurring word patterns.

A local coherence model based on IBM Model 1 assigns the following probability to a text T consisting of n sentences $s_1, s_2, \ldots, s_n$:

$$P_{\mathrm{IBM}_1^{\mathrm{D}}}(T) = \Pi_{i=1}^{n-1}\Pi_{j=1}^{|s_{i+1}|}\frac{\epsilon}{|s_i|+1}\Sigma_{k=0}^{|s_i|}t(s_{i+1}^j|s_i^k)$$

[1]We use for training the publicly-available GIZA++ toolkit, http://www.fjoch.com/GIZA++.html

We call the above equation the direct IBM Model 1, as this model considers the words in sentence s_{i+1} (the s_{i+1}^j events) as being generated by the words in sentence s_i (the s_i^k events, which include the special s_i^0 event called the NULL word), with probability $t(s_{i+1}^j|s_i^k)$. We also define a local coherence inverse IBM Model 1:

$$P_{\mathrm{IBM}_1^{\mathrm{I}}}(T) = \Pi_{i=1}^{n-1}\Pi_{k=1}^{|s_i|}\frac{\epsilon}{|s_{i+1}|+1}\Sigma_{j=0}^{|s_{i+1}|}t(s_i^k|s_{i+1}^j)$$

This model considers the words in sentence s_i (the s_i^k events) as being generated by the words in sentence s_{i+1} (the s_{i+1}^j events, which include the special s_{i+1}^0 event called the NULL word), with probability $t(s_i^k|s_{i+1}^j)$.

Entity-based Coherence Models. Barzilay and Lapata (2005) recently proposed an entity-based coherence model that aims to learn abstract coherence properties, similar to those stipulated by Centering Theory (Grosz et al., 1995). Their model learns distribution patterns for transitions between discourse entities that are abstracted into their syntactic roles – subject (**S**), object (**O**), other (**X**), missing (**-**). The feature values are computed using an entity-grid representation for the discourse that records the syntactic role of each entity as it appears in each sentence. Also, salient entities are differentiated from casually occurring entities, based on the widely used assumption that occurrence frequency correlates with discourse prominence (Morris and Hirst, 1991; Grosz et al., 1995). We exclude the coreference information from this model, as the discourse ordering problem cannot accommodate current coreference solutions, which assume a pre-specified order (Ng, 2005).

In the jargon of (Barzilay and Lapata, 2005), the model we implemented is called Syntax+Salience. The probability assigned to a text $T = s_1 \ldots s_n$ by this Entity-Based model (henceforth called EB) can be locally computed (i.e., at sentence transition level) using M feature functions, as follows:

$$P_{\mathrm{EB}}(T) = \Pi_{i=1}^{n-1}\Sigma_{k=1}^{M}w_k e_k(s_{i+1}|s_i)$$

Here, $e_k(s_{i+1}|s_i)$ are feature values, and w_k are weights trained to discriminate between coherent, human-authored documents and examples assumed to have lost some degree of coherence (scrambled versions of the original documents).

2.2 Global Models of Discourse Coherence

Barzilay and Lee (2004) propose a document content model that uses a Hidden Markov Model

(HMM) to capture more global aspects of coherence. Each state in their HMM corresponds to a distinct "topic". Topics are determined by an unsupervised algorithm via complete-link clustering, and are written as h_i, with $h_i \in H$.

The probability assigned to a text $T = s_1 \dots s_n$ by this Content Model (henceforth called CM) can be written as follows:

$$P_{\mathrm{CM}}(T) = \max_{h_1 \dots h_n} \Pi_{i=1}^{n} P_{\mathrm{TT}}(h_i|h_{i-1}) \times P_{\mathrm{TM}}(s_i|h_i)$$

The first term, P_{TT}, models the probability of changing from topic h_{i-1} to topic h_i. The second term, P_{TM}, models the probability of generating sentences from topic h_i.

2.3 Combining Local and Global Models of Discourse Coherence

We can model the probability $P(T)$ of a text T using a log-linear model that combines the discourse coherence models presented above. In this framework, we have a set of M feature functions $f_m(T)$, $1 \leq m \leq M$. For each feature function, there exists a model parameter λ_m, $1 \leq m \leq M$. The probability $P(T)$ can be written under the log-linear model as follows:

$$P(T) = \frac{\exp[\Sigma_{m=1}^{M} \lambda_m f_m(T)]}{\Sigma_{T'} \exp[\Sigma_{m=1}^{M} \lambda_m f_m(T')]}$$

Under this model, finding the most probable text T is equivalent with solving Equation 1, and therefore we do not need to be concerned about computing expensive normalization factors.

$$\arg\max_{T} P(T) = \arg\min_{T} -\Sigma_{m=1}^{M} \lambda_m \log f_m(T) \quad (1)$$

In this framework, we distinguish between the *modeling* problem, which amounts to finding appropriate feature functions for the discourse coherence task, and the *training* problem, which amounts to finding appropriate values for λ_m, $1 \leq m \leq M$. We address the modeling problem by using as feature functions the discourse coherence models presented in the previous sections. In Section 3, we address the training problem by performing a discriminative training procedure of the λ_m parameters, using as utility functions a metric that measures how different a training instance is from a given reference.

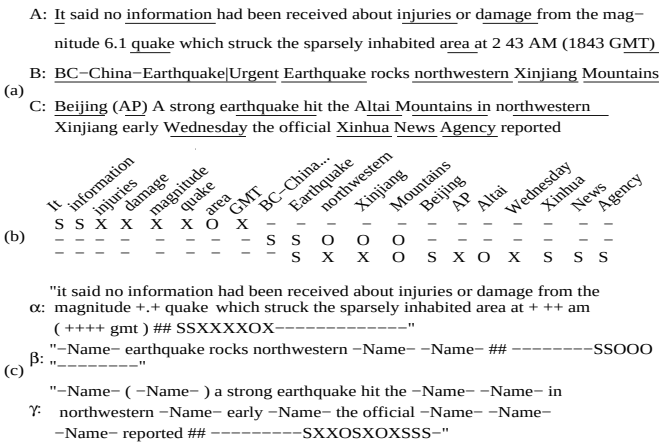


Figure 1: Example consisting of discourse units A, B, and C (a). In (b), their entities are detected (underlined) and assigned syntactic roles: **S** (subject), **O** (object), **X** (other), **-** (missing). In (c), terms α, β, and γ encode these discourse units for model scoring purposes.

3 Search Algorithms for Coherent Discourses and Utility-Based Training

The algorithms we propose use as input representation the IDL-expressions formalism (Nederhof and Satta, 2004; Soricut and Marcu, 2005). We use here the IDL formalism (which stands for Interleave, Disjunction, Lock, after the names of its operators) to define finite sets of possible discourses over given discourse units. Without losing generality, we will consider sentences as discourse units in our examples and experiments.

3.1 Input Representation

Consider the discourse units A-C presented in Figure 1(a). Each of these units undergoes various processing stages in order to provide the information needed by our coherence models. The entity-based model (EB) (Section 2), for instance, makes use of a syntactic parser to determine the syntactic role played by each detected entity (Figure 1(b)). For example, the string SSXXXXOX---------- (first row of the grid in Figure 1(b), corresponding to discourse unit A) encodes that It and information have subject (S) role, injuries, etc. have other (X) roles, area has object (O) role, and the rest of the entities do not appear (-) in this unit.

In order to be able to solve Equation 1, the input representation needs to provide the necessary information to compute all f_m terms, that is, all individual model scores. Textual units A, B,

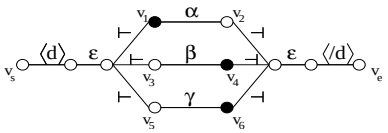


Figure 2: The IDL-graph corresponding to the IDL-expression $\langle d \rangle \cdot \|(\alpha, \beta, \gamma) \cdot \langle /d \rangle$.

and C in our example are therefore represented as terms α, β, and γ, respectively[2] (Figure 1(c)). These terms act like building blocks for IDL-expressions, as in the following example:

$$E = \langle d \rangle \cdot \|(\alpha, \beta, \gamma) \cdot \langle /d \rangle$$

E uses the $\|$ (Interleave) operator to create a bag-of-units representation. That is, E stands for the set of all possible order permutations of α, β, and γ, with the additional information that any of these orders are to appear between the beginning $\langle d \rangle$ and end of document $\langle /d \rangle$. An equivalent representation, called IDL-graphs, captures the same information using vertices and edges, which stand in a direct correspondence with the operators and atomic symbols of IDL-expressions. For instance, each $\vdash$ and $\dashv$–labeled edge k-pair, and their source and target vertices, respectively, correspond to a k-argument $\|$ operator. In Figure 2, we show the IDL-graph corresponding to IDL-expression E.

3.2 Search Algorithms

Algorithms that operate on IDL-graphs have been recently proposed by Soricut and Marcu (2005). We extend these algorithms to take as input IDL-graphs over non-atomic symbols (such that the coherence models can operate inside terms like α, β, and γ from Figure 1), and also to work under models with hidden variables such as CM (Section 2.2).

These algorithm, called IDL-CH-A* (A* search for IDL-expressions under Coherence models) and IDL-CH-HB^b (Histogram-Based beam search for IDL-expressions under Coherence models, with histogram beam b), assume an alphabet Σ of non-atomic (visible) variables (over which the input IDL-expressions are defined), and an alphabet H of hidden variables. They *unfold* an input IDL-graph on-the-fly, as follows: starting from the initial vertex v_s, the input graph is traversed in an IDL-specific manner, by creating states which keep track of k positions in any subgraph corresponding to a k-argument $\|$ operator, as well as the last edge traversed and the last hidden variable considered. For instance, state $\mathrm{S} = (\mathrm{v_1 v_4 v_6}, \gamma, h_i)$ (see the blackened vertices in Figure 2) records that expressions β and γ have already been considered (while α is still in the future of state S), and γ was the last one considered, evaluated under the hidden variable h_i. The information recorded in each state allows for the computation of a current coherence cost under any of the models described in Section 2. In what follows, we assume this model to be the model from Equation 1, since each of the individual models can be obtained by setting the other λs to 0.

We also define an *admissible heuristic* function (Russell and Norvig, 1995), which is used to compute an admissible future cost a for state S, using the following equation:

$$a(S) = -\sum_{e \in F} \sum_{m=1}^{M} \lambda_m \min_{\substack{h_i \in H \\ \langle ce, h_j \rangle \in C \times H}} \log f_m(\langle e, h_i \rangle | \langle ce, h_j \rangle)$$

F is the set of future (visible) *events* for state S, which can be computed directly from an input IDL-graph, as the set of all Σ–edge-labels between the vertices of state S and final vertex v_e. For example, for state $\mathrm{S} = (\mathrm{v_1 v_4 v_6}, \gamma, h_i)$, we have $F = \{\alpha, \langle /d \rangle\}$. C is the set of future (visible) *conditions* for state S, which can be obtained from F (any non-final future event may become a future conditioning event), by eliminating $\langle /d \rangle$ and adding the current conditioning event of S. For the considered example state S, we have $C = \{\alpha, \gamma\}$. The value $a(S)$ is admissible because, for each future event $\langle e, h_i \rangle$, with $e \in F$ and $h_i \in H$, its cost is computed using the most inexpensive conditioning event $\langle ce, h_j \rangle \in C \times H$.

The IDL-CH-A* algorithm uses a priority queue Q (sorted according to total cost, computed as current + admissible) to control the unfolding of an input IDL-graph, by processing, at each unfolding step, the most inexpensive state (extracted from the top of Q). The admissibility of the future costs and the monotonicity property enforced by the priority queue guarantees that IDL-CH-A* finds an *optimal* solution to Equation 1 (Russell and Norvig, 1995).

The IDL-CH-HB^b algorithm uses a histogram beam b to control the unfolding of an input IDL-graph, by processing, at each unfolding step, the

[2] Following Barzilay and Lee (2004), proper names, dates, and numbers are replaced with generic tokens.

top b most inexpensive states (according to total cost). This algorithm can be tuned (via b) to achieve good trade-off between speed and accuracy. We refer the reader to (Soricut, 2006) for additional details regarding the optimality and the theoretical run-time behavior of these algorithms.

3.3 Utility-based Training

In addition to the modeling problem, we must also address the training problem, which amounts to finding appropriate values for the λ_m parameters from Equation 1.

The solution we employ here is the discriminative training procedure of Och (2003). This procedure learns an optimal setting of the λ_m parameters using as optimality criterion the utility of the proposed solution. There are two necessary ingredients to implement Och's (2003) training procedure. First, it needs a search algorithm that is able to produce ranked k-best lists of the most promising candidates in a reasonably fast manner (Huang and Chiang, 2005). We accommodate k-best computation within the IDL-CH-HB^{100} algorithm, which decodes bag-of-units IDL-expressions at an average speed of 75.4 sec./exp. on a 3.0 GHz CPU Linux machine, for an average input of 11.5 units per expression.

Second, it needs a criterion which can automatically assess the quality of the proposed candidates. To this end, we employ two different metrics, such that we can measure the impact of using different utility functions on performance.

TAU (Kendall's τ). One of the most frequently used metrics for the automatic evaluation of document coherence is Kendall's τ (Lapata, 2003; Barzilay and Lee, 2004). TAU measures the minimum number of adjacent transpositions needed to transform a proposed order into a reference order. The range of the TAU metric is between -1 (the worst) to 1 (the best).

BLEU. One of the most successful metrics for judging machine-generated text is BLEU (Papineni et al., 2002). It counts the number of unigram, bigram, trigram, and four-gram matches between hypothesis and reference, and combines them using geometric mean. For the discourse ordering problem, we represent hypotheses and references by index sequences (e.g., "4 2 3 1" is a hypothesis order over four discourse units, in which the first and last units have been swapped with respect to the reference order). The range of BLEU scores is between 0 (the worst) and 1 (the best).

We run different discriminative training sessions using TAU and BLEU, and train two different sets of the λ_m parameters for Equation 1. The log-linear models thus obtained are called Log-linear$_{\mathsf{maxTAU}}$ and Log-linear$_{\mathsf{maxBLEU}}$, respectively.

4 Experiments

We evaluate empirically two different aspects of our work. First, we measure the performance of our search algorithms across different models. Second, we compare the performance of each individual coherence model, and also the performance of the discriminatively trained log-linear models. We also compare the overall performance (model & decoding strategy) obtained in our framework with previously reported results.

4.1 Evaluation setting

The task on which we conduct our evaluation is information ordering (Lapata, 2003; Barzilay and Lee, 2004; Barzilay and Lapata, 2005). In this task, a pre-selected set of information-bearing document units (in our case, sentences) needs to be arranged in a sequence which maximizes some specific information quality (in our case, document coherence). We use the information-ordering task as a means to measure the performance of our algorithms and models in a well-controlled setting. As described in Section 3, our framework can be used in applications such as multi-document summarization. In fact, Barzilay et al. (2002) formulate the multi-document summarization problem as an information ordering problem, and show that naive ordering algorithms such as majority ordering (select most frequent orders across input documents) and chronological ordering (order facts according to publication date) do not always yield coherent summaries.

Data. For training and testing, we use documents from two different genres: newspaper articles and accident reports written by government officials (Barzilay and Lapata, 2005). The first collection (henceforth called EARTHQUAKES) consists of Associated Press articles from the North American News Corpus on the topic of natural disasters. The second collection (henceforth called ACCIDENTS) consists of aviation accident reports from the National Transportation Safety

Search Algorithm	IBM_1^D			IBM_1^I			CM			EB		
	ESE	TAU	BLEU	ESE	TAU	BLEU	ESE	TAU	BLEU	ESE	TAU	BLEU
	EARTHQUAKES											
IDL-CH-A*	0%	.39	.12	0%	.33	.13	0%	.39	.12	0%	.19	.05
IDL-CH-HB^{100}	0%	.38	.12	0%	.32	.13	0%	.39	.12	0%	.19	.06
IDL-CH-HB^{1}	4%	.37	.13	13%	.34	.14	36%	.32	.11	16%	.18	.05
Lapata, 2003	90%	.01	.04	58%	.02	.06	97%	.05	.04	46%	-.05	.00
	ACCIDENTS											
IDL-CH-A*	0%	.41	.21	0%	.40	.21	0%	.37	.15	0%	.13	.10
IDL-CH-HB^{100}	0%	.41	.20	0%	.40	.21	2%	.36	.15	0%	.12	.10
IDL-CH-HB^{1}	0%	.38	.19	12%	.32	.20	13%	.34	.13	33%	-.04	.06
Lapata, 2003	86%	.11	.03	67%	.12	.05	85%	.18	.00	24%	-.05	.06

Table 1: Evaluation of search algorithms for document coherence, for both EARTHQUAKES and ACCIDENTS genres, across the IBM_1^D, IBM_1^I, CM, and EB models. Performance is measured in terms of percentage of Estimated Search Errors (ESE), as well as quality of found realizations (average TAU and BLEU).

Model	TAU	BLEU	TAU	BLEU
	EARTHQUAKES		ACCIDENTS	
IBM_1^D	.38	.12	.41	.20
IBM_1^I	.32	.13	.40	.21
CM	.39	.12	.36	.15
EB	.19	.06	.12	.10
Log-$linear_{uniform}$	.34	.14	.48	.23
Log-$linear_{maxTAU}$	**.47**	.15	**.50**	.23
Log-$linear_{maxBLEU}$	.46	**.16**	.49	**.24**

Table 2: Evaluation of stochastic models for document coherence, for both EARTHQUAKES and ACCIDENTS genre, using IDL-CH-HB^{100}.

Overall performance	TAU	
	QUAKES	ACCID.
Lapata (2003)	0.48	0.07
Barzilay & Lee (2004)	**0.81**	0.44
Barzilay & Lee (reproduced)	0.39	0.36
Barzilay & Lapata (2005)	0.19	0.12
IBM_1^D, IDL-CH-HB^{100}	0.38	0.41
Log-lin_{maxTAU}, IDL-CH-HB^{100}	0.47	**0.50**

Table 3: Comparison of overall performance (affected by both model & search procedure) of our framework with previous results.

Board's database.

For both collections, we used 100 documents for training and 100 documents for testing. A fraction of 40% of the training documents was temporarily removed and used as a development set, on which we performed the discriminative training procedure.

4.2 Evaluation of Search Algorithms

We evaluated the performance of several search algorithms across four stochastic models of document coherence: the IBM_1^D and IBM_1^I coherence models, the content model of Barzilay and Lee (2004) (CM), and the entity-based model of Barzilay and Lapata (2005) (EB) (Section 2). We measure search performance using an Estimated Search Error (ESE) figure, which reports the percentage of times when the search algorithm proposes a sentence order which scores lower than the original sentence order (OSO). We also measure the quality of the proposed documents using TAU and BLEU, using as reference the OSO.

In Table 1, we report the performance of four search algorithms. The first three, IDL-CH-A*, IDL-CH-HB^{100}, and IDL-CH-HB^{1} are the IDL-based search algorithms of Section 3, implementing A* search, histogram beam search with a beam of 100, and histogram beam search with a beam of 1, respectively. We compare our algorithms against the greedy algorithm used by Lapata (2003). We note here that the comparison is rendered meaningful by the observation that this algorithm performs search identically with algorithm IDL-CH-HB^{1} (histogram beam 1), when setting the heuristic function for future costs a to constant 0.

The results in Table 1 clearly show the superiority of the IDL-CH-A* and IDL-CH-HB^{100} algo-

rithms. Across all models considered, they consistently propose documents with scores at least as good as OSO (0% Estimated Search Error). As the original documents were coherent, it follows that the proposed document realizations also exhibit coherence. In contrast, the greedy algorithm of Lapata (2003) makes grave search errors. As the comparison between IDL-CH-HB^{100} and IDL-CH-HB^{1} shows, the superiority of the IDL-CH algorithms depends more on the admissible heuristic function a than in the ability to maintain multiple hypotheses while searching.

4.3 Evaluation of Log-linear Models

For this round of experiments, we held constant the search procedure (IDL-CH-HB^{100}), and varied the λ_m parameters of Equation 1. The utility-trained log-linear models are compared here against a baseline log-linear model log-linear$_{\text{uniform}}$, for which all λ_m parameters are set to 1, and also against the individual models. The results are presented in Table 2.

If not properly weighted, the log-linear combination may yield poorer results than those of individual models (average TAU of .34 for log-linear$_{\text{uniform}}$, versus .38 for IBM$_1^{\text{D}}$ and .39 for CM, on the EARTHQUAKES domain). The highest TAU accuracy is obtained when using TAU to perform utility-based training of the λ_m parameters (.47 for EARTHQUAKES, .50 for ACCIDENTS). The highest BLEU accuracy is obtained when using BLEU to perform utility-based training of the λ_m parameters (.16 for EARTHQUAKES, .24 for the ACCIDENTS). For both genres, the differences between the highest accuracy figures (in bold) and the accuracy of the individual models are statistically significant at 95% confidence (using bootstrap resampling).

4.4 Overall Performance Evaluation

The last comparison we provide is between the performance provided by our framework and previously-reported performance results (Table 3). We are able to provide this comparison based on the TAU figures reported in (Barzilay and Lee, 2004). The training and test data for both genres is the same, and therefore the figures can be directly compared. These figures account for combined model and search performance.

We first note that, unfortunately, we failed to accurately reproduce the model of Barzilay and Lee (2004). Our reproduction has an average TAU figure of only .39 versus the original figure of .81 for EARTHQUAKES, and .36 versus .44 for ACCIDENTS. On the other hand, we reproduced successfully the model of Barzilay and Lapata (2005), and the average TAU figure is .19 for EARTHQUAKES, and .12 for ACCIDENTS[3]. The large difference on the EARTHQUAKES corpus between the performance of Barzilay and Lee (2004) and our reproduction of their model is responsible for the overall lower performance (0.47) of our log-linear$_{\text{maxTAU}}$ model and IDL-CH-HB^{100} search algorithm, which is nevertheless higher than that of its component model CM (0.39). On the other hand, we achieve the highest accuracy figure (0.50) on the ACCIDENTS corpus, outperforming the previous-highest figure (0.44) of Barzilay and Lee (2004). These result empirically show that utility-trained log-linear models of discourse coherence outperform each of the individual coherence models considered.

5 Discussion and Conclusions

We presented a generic framework that is capable of integrating various stochastic models of discourse coherence into a more powerful model that combines the strengths of the individual models. An important ingredient of this framework are the search algorithms based on IDL-expressions, which provide a flexible way of solving discourse generation problems using stochastic models. Our generation algorithms are fundamentally different from previously-proposed algorithms for discourse generation. The genetic algorithms of Mellish et al. (1998) and Karamanis and Manarung (2002), as well as the greedy algorithm of Lapata (2003), provide no theoretical guarantees on the optimality of the solutions they propose. At the other end of the spectrum, the exhaustive search of Barzilay and Lee (2004), while ensuring optimal solutions, is prohibitively expensive, and cannot be used to perform utility-based training. The linear programming algorithm of Althaus et al. (2005) is the only proposal that achieves both good speed and accuracy. Their algorithm, however, cannot handle models with hidden states, cannot compute k-best lists, and does not have the representation flexibility provided by

[3] Note that these figures cannot be compared directly with the figures reported in (Barzilay and Lapata, 2005), as they use a different type of evaluation. Our EB model achieves the same performance as the original Syntax+Salience model, in their evaluation setting.

IDL-expressions, which is crucial for coherence decoding in realistic applications such as multi-document summarization.

For each of the coherence model combinations that we have utility trained, we obtained improved results on the discourse ordering problem compared to the individual models. This is important for two reasons. Our improvements can have an immediate impact on multi-document summarization applications (Barzilay et al., 2002). Also, our framework provides a solid foundation for subsequent research on discourse coherence models and related applications.

Acknowledgments This work was partially supported under the GALE program of the Defense Advanced Research Projects Agency, Contract No. HR0011-06-C-0022.

References

Ernst Althaus, Nikiforos Karamanis, and Alexander Koller. 2005. Computing locally coherent discourse. In *Proceedings of the ACL*, pages 399–406.

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the ACL*, pages 141–148.

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *Proceedings of the HLT-NAACL*, pages 113–120.

Regina Barzilay, Noemie Elhadad, and Kathleen R. McKeown. 2002. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, 17:35–55.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

L. Carlson, D. Marcu, and M. E. Okurowski. 2003. Building a discourse-tagged corpus in the framework of Rhetorical Structure Theory. In J. van Kuppevelt and R. Smith, eds., *Current Directions in Discourse and Dialogue*. Kluwer Academic Publishers.

K. Forbes, E. Miltsakaki, R. Prasad, A. Sarkar, A. Joshi, and B. Webber. 2001. D-LTAG System: Discourse parsing with a lexicalized tree-adjoining grammar. In *Workshop on Information Structure, Discourse Structure and Discourse Semantics*.

Barbara J. Grosz, Aravind K. Joshi, and Scott Weinstein. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–226.

Liang Huang and David Chiang. 2005. Better k-best parsing. In *Proceedings of the International Workshop on Parsing Technologies (IWPT 2005)*.

Nikiforos Karamanis and Hisar M. Manurung. 2002. Stochastic text structuring using the principle of continuity. In *Proceedings of INLG*, pages 81–88.

Nikiforos Karamanis, Massimo Poesio, Chris Mellish, and Jon Oberlander. 2004. Evaluating centering-based metrics of coherence for text structuring using a reliably annotated corpus. In *Proc. of the ACL*.

Rodger Kibble and Richard Power. 2004. Optimising referential coherence in text generation. *Computational Linguistics*, 30(4):410–416.

Kevin Knight. 2003. Personal Communication.

Mirella Lapata. 2003. Probabilistic text structuring: Experiments with text ordering. In *Proceedings of the ACL*, pages 545–552.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical Structure Theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.

Daniel Marcu. 1996. In *Proceedings of the Student Conference on Computational Linguistics*, pages 136-143.

Daniel Marcu. 2000. *The Theory and Practice of Discourse Parsing and Summarization*. The MIT Press.

Chris Mellish, Alistair Knott, Jon Oberlander, and Mick O'Donnell. 1998. Experiments using stochastic search for text planning. In *Proceedings of the INLG*, pages 98–107.

Jane Morris and Graeme Hirst. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17(1):21–48.

Mark-Jan Nederhof and Giorgio Satta. 2004. IDL-expressions: a formalism for representing and parsing finite languages in natural language processing. *Journal of Artificial Intelligence Research*, pages 287–317.

Vincent Ng. 2005. Machine learning for coreference resolution: from local clasifficiation to global reranking. In *Procedings of the ACL*, pages 157–164.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the ACL*, pages 160–167.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the ACL*, pages 311–318.

Stuart Russell and Peter Norvig. 1995. *Artificial Intelligence. A Modern Approach*. Prentice Hall.

Donia R. Scott and Clarisse S. de Souza. 1990. Getting the message across in RST-based text generation. In Robert Dale, Chris Mellish, and Michael Zock, eds., *Current Research in Natural Language Generation*, pages 47–73. Academic Press.

Radu Soricut and Daniel Marcu. 2005. Towards developing generation algorithms for text-to-text applications. In *Proceedings of the ACL*, pages 66–74.

Radu Soricut. 2006. *Natural Language Generation for Text-to-Text Applications Using an Information-Slim Representation*. Ph.D. thesis, University of Southern California.

A Comparison of Alternative Parse Tree Paths for Labeling Semantic Roles

Reid Swanson and Andrew S. Gordon
Institute for Creative Technologies
University of Southern California
13274 Fiji Way, Marina del Rey, CA 90292 USA
swansonr@ict.usc.edu, gordon@ict.usc.edu

Abstract

The integration of sophisticated inference-based techniques into natural language processing applications first requires a reliable method of encoding the predicate-argument structure of the propositional content of text. Recent statistical approaches to automated predicate-argument annotation have utilized parse tree paths as predictive features, which encode the path between a verb predicate and a node in the parse tree that governs its argument. In this paper, we explore a number of alternatives for how these parse tree paths are encoded, focusing on the difference between automatically generated constituency parses and dependency parses. After describing five alternatives for encoding parse tree paths, we investigate how well each can be aligned with the argument substrings in annotated text corpora, their relative precision and recall performance, and their comparative learning curves. Results indicate that constituency parsers produce parse tree paths that can more easily be aligned to argument substrings, perform better in precision and recall, and have more favorable learning curves than those produced by a dependency parser.

1 Introduction

A persistent goal of natural language processing research has been the automated transformation of natural language texts into representations that unambiguously encode their propositional content in formal notation. Increasingly, first-order predicate calculus representations of textual meaning have been used in natural lanugage processing applications that involve automated inference. For example, Moldovan et al. (2003) demonstrate how predicate-argument formulations of questions and candidate answer sentences are unified using logical inference in a top-performing question-answering application. The importance of robust techniques for predicate-argument transformation has motivated the development of large-scale text corpora with predicate-argument annotations such as PropBank (Palmer et al., 2005) and FrameNet (Baker et al., 1998). These corpora typically take a pragmatic approach to the predicate-argument representations of sentences, where predicates correspond to single word triggers in the surface form of the sentence (typically verb lemmas), and arguments can be identified as substrings of the sentence.

Along with the development of annotated corpora, researchers have developed new techniques for automatically identifying the arguments of predications by labeling text segments in sentences with semantic roles. Both Gildea & Jurafsky (2002) and Palmer et al. (2005) describe statistical labeling algorithms that achieve high accuracy in assigning semantic role labels to appropropriate constituents in a parse tree of a sentence. Each of these efforts employed the use of *parse tree paths* as predictive features, encoding the series of up and down transitions through a parse tree to move from the node of the verb (predicate) to the governing node of the constituent (argument). Palmer et al. (2005) demonstrate that utilizing the gold-standard parse trees of the Penn treebank (Marcus et al., 1993) to encode parse tree paths yields significantly better labeling accuracy than when using an automatic syntactical parser, namely that of Collins (1999).

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 811–818,
Sydney, July 2006.

Parse tree paths (between verbs and arguments that fill semantic roles) are particularly interesting because they symbolically encode the relationship between the syntactic and semantic aspects of verbs, and are potentially generalized across other verbs within the same class (Levin, 1993). However, the encoding of individual parse tree paths for predicates is wholly dependent on the characteristics of the parse tree of a sentence, for which competing approaches could be taken.

The research effort described in this paper further explores the role of parse tree paths in identifying the argument structure of verb-based predications. We are particularly interested in exploring alternatives to the constituency parses that were used in previous research, including parsing approaches that employ dependency grammars. Specifically, our aim is to answer four important questions:

1. How can parse tree paths be encoded when employing different automated constituency parsers, i.e. Charniak (2000), Klein & Manning (2003), or a dependency parser (Lin, 1998)?

2. Given that each of these alternatives creates a different formulation of the parse tree of a sentence, which of them encodes branches that are easiest to align with substrings that have been annotated with semantic role information?

3. What is the relative precision and recall performance of parse tree paths formulated using these alternative automated parsing techniques, and do the results vary depending on argument type?

4. How many examples of parse tree paths are necessary to provide as training examples in order to achieve high labeling accuracy when employing each of these parsing alternatives?

Each of these four questions is addressed in the four subsequent sections of this paper, followed by a discussion of the implications of our findings and directions for future work.

2 Alternative Parse Tree Paths

Parse tree paths were introduced by Gildea & Jurafsky (2002) as descriptive features of the syntactic relationship between predicates and arguments in the parse tree of a sentence. Predicates are typically assumed to be specific target words (usually verbs), and arguments are assumed to be a span of words in the sentence that are governed by a single node in the parse tree. A parse tree path can be described as a sequence of transitions up and down a parse tree from the target word to the governing node, as exemplified in Figure 1.

The encoding of the parse tree path feature is dependent on the syntactic representation that is produced by the parser. This, in turn, is dependant on the training corpus used to build the parser, and the conditioning factors in its probability model. As result, encodings of parse tree paths can vary greatly depending on the parser that is used, yielding parse tree paths that vary in their ability to generalize across sentences.

In this paper we explore the characteristics of parse tree paths with respect to different approaches to automated parsing. We were particularly interested in comparing traditional constituency parsing (as exemplified in Figure 1) with dependency parsing, specifically the Minipar system built by Lin (1998). Minipar is increasingly being used in semantics-based nlp applications (e.g. Pantel & Lin, 2002). Dependency parse trees differ from constituency parses in that they represent sentence structures as a set of dependency relationships between words, typed asymmetric binary relationships between head words and modifying words. Figure 2 depicts the output of Minipar on an example sentence, where each node is a word or an empty node along with the word lemma, its part of speech, and the relationship type to its governing node.

Our motivation for exploring the use of Minipar in for the creation of parse tree paths can be seen by comparing Figure 1 and Figure 2, where

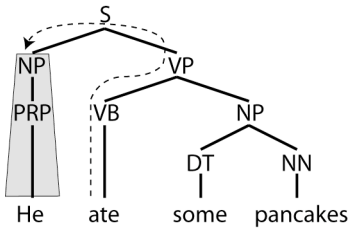


Figure 1: An example parse tree path from the predicate *ate* to the argument NP *He*, represented as VB↑VP↑S↓NP.

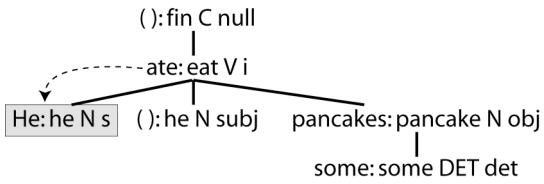


Figure 2. An example dependency parse, with a parse tree path from the predicate *ate* to the argument *He*.

the Minipar path is both shorter and simpler for the same predicate-argument relationship, and could be encoded in various ways that take advantage of the additional semantic and lexical information that is provided.

To compare traditional constituency parsing with dependency parsing, we evaluated the accuracy of argument labeling using parse tree paths generated by two leading constituency parsers and three variations of parse tree paths generated by Minipar, as follows:

Charniak: We used the Charniak parser (2000) to extract parse tree paths similar to those found in Palmer et al. (2005), with some slight modifications. In cases where the last node in the path was a non-branching pre-terminal, we added the lexical information to the path node. In addition, our paths led to the lowest governing node, rather than the highest. For example, the parse tree path for the argument in Figure 1 would be encoded as:

VB↑VP↑S↓NP↓PRP:he

Stanford: We also used the Stanford parser developed by Klein & Manning (2003), with the same path encoding as the Charniak parser.

Minipar A: We used three variations of parse tree path encodings based on Lin's dependency parser, Minipar (1998). Minipar A is the first and most restrictive path encoding, where each is annotated with the entire information output by Minpar at each node. A typical path might be:

ate:eat,V,i↓He:he,N,s

Minipar B: A second parse tree path encoding was generated from Minipar parses that relaxes some of the constraints used in Minpar A. Instead of using all the information contained at a node, in Minipar B we only encode a path with its part of speech and relational information. For example:

V,i↓N,s

Minipar C: As the converse to Minipar A we also tried one other Minipar encoding. As in Minipar A, we annotated the path with all the information output, but instead of doing a direct string comparison during our search, we considered two paths matching when there was a match between either the word, the stem, the part of speech, or the relation. For example, the following two parse tree paths would be considered a match, as both include the relation *i*.

ate:eat,V,i↓He:he,N,s
was:be,VBE,i↓He:he,N,s

We explored other combinations of dependency relation information for Minipar-derived parse tree paths, including the use of the deep relations. However, results obtained using these other combinations were not notably different from those of the three base cases listed above, and are not included in the evaluation results reported in this paper.

3 Aligning arguments to parse trees nodes in a training / testing corpus

We began our investigation by creating a training and testing corpus of 400 sentences each containing an inflection of one of four target verbs (100 each), namely *believe*, *think*, *give*, and *receive*. These sentences were selected at random from the 1994-07 section of the New York Times gigaword corpus from the Linguistic Data Consortium. These four verbs were chosen because of the synonymy among the first two, and the reflexivity of the second two, and because all four have straightforward argument structures when viewed as predicates, as follows:

predicate: **believe**
arg0: the believer
arg1: the thing that is believed

predicate: **think**
arg0: the thinker
arg1: the thing that is thought

predicate: **give**
arg0: the giver
arg1: the thing that is given
arg2: the receiver

predicate: **receive**
arg0: the receiver
arg1: the thing that is received
arg2: the giver

This corpus of sentences was then annotated with semantic role information by the authors of this paper. All annotations were made by assigning start and stop locations for each argument in the unparsed text of the sentence. After an initial pilot annotation study, the following annotation policy was adopted to overcome common disagreements: (1) When the argument is a noun and it is part of a definite description then in-

clude the entire definite description. (2) Do not include complementizers such as 'that' in 'believe that' in an argument. (3) Do include prepositions such as 'in' in 'believe in'. (4) When in doubt, assume phrases attach locally. Using this policy, an agreement of 92.8% was achieved among annotators for the set of start and stop locations for arguments. Examples of semantic role annotations in our corpus for each of the four predicates are as follows:

1. [$_{Arg0}$Those who excavated the site in 1907] believe [$_{Arg1}$ it once stood two or three stories high.]

2. Gus is in good shape and [$_{Arg0}$ I] think [$_{Arg1}$ he's happy as a bear.]

3. If successful, [$_{Arg0}$ he] will give [$_{Arg1}$ the funds] to [$_{Arg2}$ his Vietnamese family.]

4. [$_{Arg0}$ The Bosnian Serbs] have received [$_{Arg1}$ military and economic support] from [$_{Arg2}$ Serbia.]

The next step was to parse the corpus of 400 sentences using each of three automated parsing systems (Charniak, Stanford, and Minipar), and align each of the annotated arguments with its closest matching branch in the resulting parse trees. Given the differences in the parsing models used by these three systems, each yield parse tree nodes that govern different spans of text in the sentence. Often there exists no parse tree node that governs a span of text that exactly matches the span of an argument in the annotated corpus. Accordingly, it was necessary to identify the closest match possible for each of the three parsing systems in order to encode parse tree paths for each. We developed a uniform policy that would facilitate a fair comparison between parsing techniques. Our approach was to identify a single node in a given parse tree that governed a string of text with the most overlap with the text of the annotated argument. Each of the parsing methods tokenizes the input string differently, so in order to simplify the selection of the governing node with the most overlap, we made this selection based on lowest minimum edit distance (Levenshtein distance).

All three of these different parsing algorithms produced single governing nodes that overlapped well with the human-annotated corpus. However, it appeared that the two constituency parsers produced governing nodes that were more closely aligned, based on minimum edit distance. The Charniak parser aligned best with the annotated text, with an average of 2.40 characters for the lowest minimum edit distance (standard deviation = 8.64). The Stanford parser performed slightly worse (average = 2.67, standard deviation = 8.86), while distances were nearly two times larger for Minipar (average = 4.73, standard deviation = 10.44).

In each case, the most overlapping parse tree node was treated as correct for training and testing purposes.

4 Comparative Performance Evaluation

In order to evaluate the comparative performance of the parse tree paths for each of the five encodings, we divided the corpus in to equal-sized training and test sets (50 training and 50 test examples for each of the four predicates). We then constructed a system that identified the parse tree paths for each of the 10 arguments in the training sets, and applied them to the sentences in each corresponding test sets. When applying the 50 training parse tree paths to any one of the 50 test sentences for a given predicate-argument pair, a set of zero or more candidate answer nodes were returned. For the purpose of calculating precision and recall scores, credit was given when the correct answer appeared in this set. Precision scores were calculated as the number of correct answers found divided by the number of all candidate answer nodes returned. Recall scores were calculated as the number of correct answers found divided by the total number of correct answers possible. F-scores were calculated as the equally-weighted harmonic mean of precision and recall.

Our calculation of recall scores represents the best-possible performance of systems using only these types of parse-tree paths. This level of performance could be obtained if a system could always select the correct answer from the set of candidates returned. However, it is also informative to estimate the performance that could be achieved by randomly selecting among the candidate answers, representing a lower-bound on performance. Accordingly, we computed an adjusted recall score that awarded only fractional credit in cases where more than one candidate answer was returned (one divided by the set size). Adjusted recall is the sum of all of these adjusted credits divided by the total number of correct answers possible.

Figure 3 summarizes the comparative recall, precision, f-score, and adjusted recall performance for each of the five parse tree path formulations. The Charniak parser achieved the highest overall scores (precision=.49, recall=.68, f-score=.57, adjusted recall=.48), followed closely

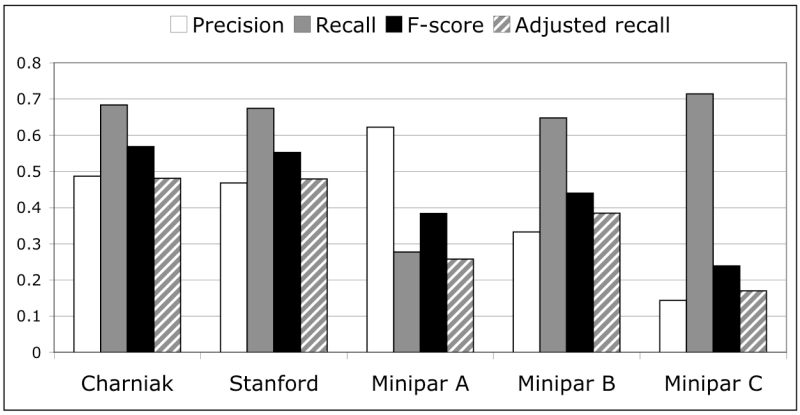


Figure 3. Precision, recall, f-scores, and adjusted recall for five parse tree path types

by the Stanford parser (precision=.47, recall=.67, f-score=.55, adjusted recall=.48).

Our expectation was that the short, semantically descriptive parse tree paths produced by Minipar would yield the highest performance. However, these results indicate the opposite; the constituency parsers produce the most accurate parse tree paths. Only Minipar C offers better recall (0.71) than the constituency parsers, but at the expense of extremely low precision. Minipar A offers excellent precision (0.62), but with extremely low recall. Minipar B provides a balance between recall and precision performance, but falls short of being competitive with the parse tree paths generated by the two constituency parsers, with an f-score of .44.

We utilized the Sign Test in order to determine the statistical significance of these differences. Rank orderings between pairs of systems were determined based on the adjusted credit that each system achieved for each test sentence. Significant differences were found between the performance of every system ($p<0.05$), with the exception of the Charniak and Stanford parsers. Interestingly, by comparing weighted values for each test example, Minipar C more frequently scores higher than Minipar A, even though the sum of these scores favors Minipar A.

In addition to overall performance, we were interested in determining whether performance varied depending on the type of the argument that is being labeled. In assigning labels to arguments in the corpus, we followed the general principles set out by Palmer et al. (2005) for labeling arguments arg0, arg1 and arg2. Across each of our four predicates, arg0 is the agent of the predication (e.g. the person that has the belief or is doing the giving), and arg1 is the thing that is acted upon by the agent (e.g. the thing that is believed or the thing that is given). Arg2 is used only for the predications based on the verbs *give* and *receive*, where it is used to indicate the other party of the action.

Our interest was in determining whether these five approaches yielded different results depending on the semantic type of the argument. Figure 4 presents the f-scores for each of these encodings across each argument type.

Results indicate that the Charniak and Stanford parsers continue to produce parse tree paths that outperform each of the Minipar-based approaches. In each approach argument 0 is the easiest to identify. Minipar A retains the general trends of Charniak and Stanford, with argument

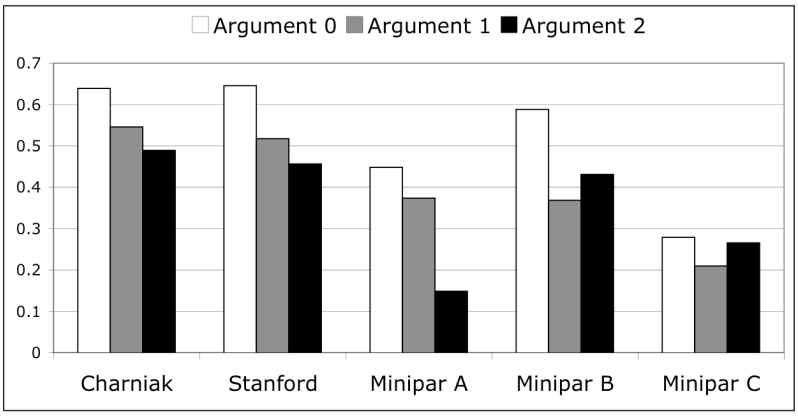


Figure 4. Comparative f-scores for arguments 0, 1, and 2 for five parse tree path types

1 easier to identify than argument 2, while Minipar B and C show the reverse. The highest f-scores for argument 0 were achieved Stanford (f=.65), while Charniak achieved the highest scores for argument 1 (f=.55) and argument 2 (f=.49).

5 Learning Curve Comparisons

The creation of large-scale text corpora with syntactic and/or semantic annotations is difficult, expensive, and time consuming. The PropBank effort has shown that producing this type of corpora is considerably easier once syntactic analysis has been done, but substantial effort and resources are still required. Better estimates of total costs could be made if it was known exactly how many annotations are necessary to achieve acceptable levels of performance. Accordingly, we investigated the learning curves of precision, recall, f-score, and adjusted recall achieved using the five different parse tree path encodings.

For each encoding approach, learning curves were created by applying successively larger subsets of the training parse tree paths to each of the items in the corresponding test set. Precision, recall, f-scores, and adjusted recall were computed as described in the previous section, and identical subsets of sentences were used across parsers, in one-sentence increments. Individual learning curves for each of the five approaches are given in Figures 5, 6, 7, 8, and 9. Figure 10 presents a comparison of the f-score learning curves for all five of the approaches.

In each approach, the precision scores slowly degrade as more training examples are provided, due to the addition of new parse tree paths that yield additional candidate answers. Conversely, the recall scores of each system show their greatest gains early, and then slowly improve with the addition of more parse tree paths. In each approach, the recall scores (estimating best-case performance) have the same general shape as the adjusted recall scores (estimating the lower-bound performance). The divergence between these two scores increases with the addition of more training examples, and is more pronounced in systems employing parse tree paths with less specific node information. The comparative f-score curves presented in Figure 10 indicate that Minipar B is competitive with Charniak and Stanford when only a small number of training examples is available. There is some evidence here that the performance of Minipar A would continue to improve with the addition of more training data, suggesting that this approach might be well-suited for applications where lots of training data is available.

6 Discussion

Annotated corpora of linguistic phenomena enable many new natural language processing applications and provide new means for tackling difficult research problems. Just as the Penn Treebank offers the possibility of developing systems capable of accurate syntactic parsing, corpora of semantic role annotations open up new possibilities for rich textual understanding and integrated inference.

In this paper, we compared five encodings of parse tree paths based on two constituency parsers and a dependency parser. Despite our expectations that the semantic richness of dependency parses would yield paths that outperformed the others, we discovered that parse tree paths from Charniak's constituency parser performed the best overall. In applications where either precision or recall is the only concern, then Minipar-derived parse tree paths would yield the best results. We also found that the performance of all of these systems varied across different argument types.

In contrast to the performance results reported by Palmer et al. (2005) and Gildea & Jurafsky (2002), our evaluation was based solely on parse tree path features. Even so, we were able to obtain reasonable levels of performance without the use of additional features or stochastic methods. Learning curves indicate that the greatest gains in performance can be garnered from the first 10 or so training examples. This result has implications for the development of large-scale corpora of semantically annotated text. Developers should distribute their effort in order to maximize the number of predicate-argument pairs with at least 10 annotations.

An automated semantic role labeling system could be constructed using only the parse tree path features described in this paper, with estimated performance between our recall scores and our adjusted recall scores. There are several ways to improve on the random selection approach used in the adjusted recall calculation. For example, one could simply select the candidate answer with the most frequent parse tree path.

The results presented in this paper help inform the design of future automated semantic role labeling systems that improve on the best-performing systems available today (Gildea &

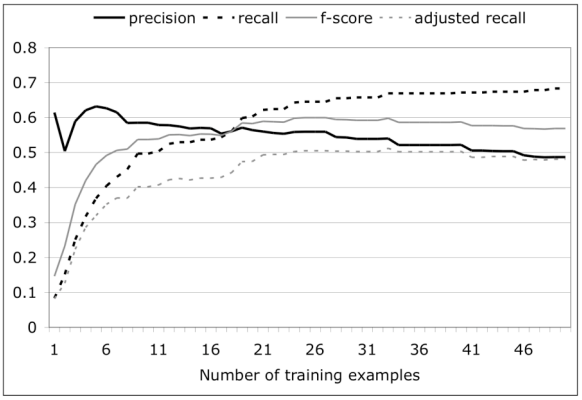


Figure 5. Charniak learning curves

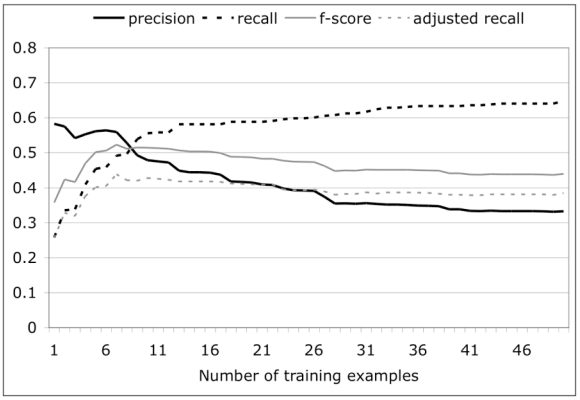


Figure 8. Minipar B learning curves

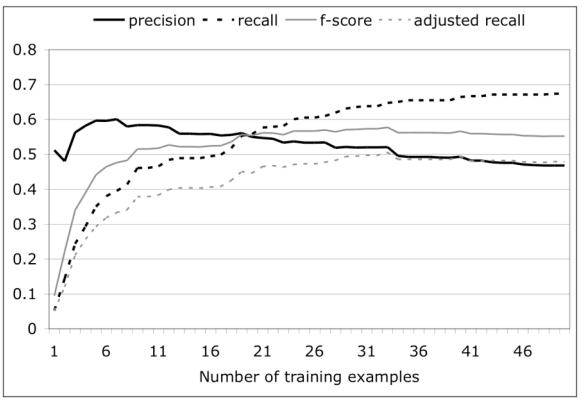


Figure 6. Stanford learning curves

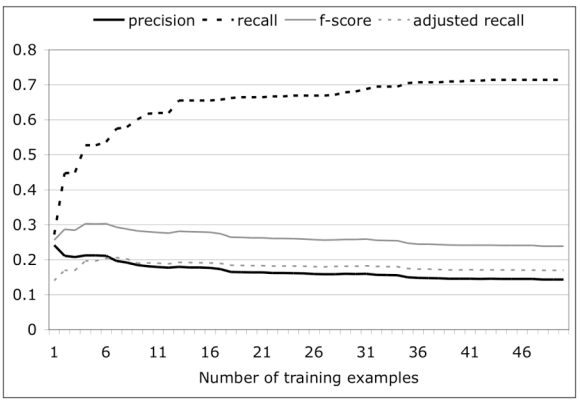


Figure 9. Minipar C learning curves

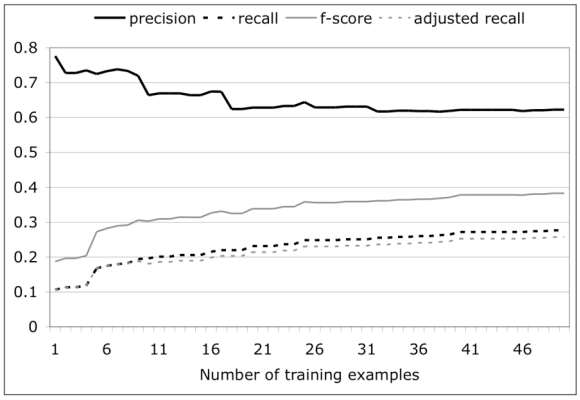


Figure 7. Minipar A learning curves

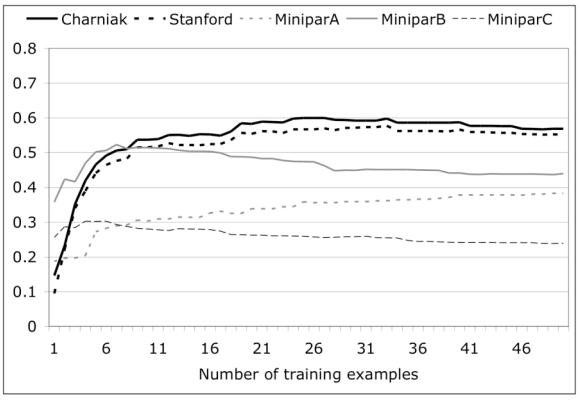


Figure 10. Comparative F-score curves

Jurafsky, 2002; Moschitti et al., 2005). We found that different parse tree paths encode different types of linguistic information, and exhibit different characteristics in the tradeoff between precision and recall. The best approaches in future systems will intelligently capitalize on these differences in the face of varying amounts of training data.

In our own future work, we are particularly interested in exploring the regularities that exist among parse tree paths for different predicates. By identifying these regularities, we believe that we will be able to significantly reduce the total number of annotations necessary to develop lexical resources that have broad coverage over natural language.

Acknowledgments

The project or effort depicted was sponsored by the U. S. Army Research, Development, and Engineering Command (RDECOM). The content or information does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred.

References

Baker, Collin, Charles J. Fillmore and John B. Lowe. 1998. The Berkeley FrameNet Project, In Proceedings of COLING-ACL, Montreal.

Charniak, Eugene. 2000. A maximum-entropy-inspired parser, In Proceedings NAACL.

Collins, Michael. 1999. Head-Driven Statistical Models for Natural Language Parsing, PhD thesis, University of Pennsylvania.

Gildea, Daniel and Daniel Jurafsky. 2002. Automatic labeling of semantic roles, Computational Linguistics, 28(3):245--288.

Klein, Dan and Christopher Manning. 2003. Accurate Unlexicalized Parsing, In Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, 423-430.

Levin, Beth. 1993. English Verb Classes and Alternations: A Preliminary Investigation. Chicago, IL: University of Chicago Press.

Lin, Dekang. 1998. Dependency-Based Evaluation of MINIPAR, In Proceedings of the Workshop on the Evaluation of Parsing Systems, First International Conference on Language Resources and Evaluation, Granada, Spain.

Marcus, Mitchell P., Beatrice Santorini and Mary Ann Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank, Computational Linguistics, 19(35):313-330

Moldovan, Dan I., Christine Clark, Sanda M. Harabagiu & Steven J. Maiorano. 2003. COGEX: A Logic Prover for Question Answering, HLT-NAACL.

Moschitti, A., Giuglea, A., Coppola, B. & Basili, R. 2005. Hierarchical Semantic Role Labeling. In Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL 2005 shared task), Ann Arbor(MI), USA.

Palmer, Martha, Dan Gildea and Paul Kingsbury. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles, Computational Linguistics.

Pantel, Patrick and Dekang Lin. 2002. Document clustering with committees, In Proceedings of SIGIRO2, Tampere, Finland.

A Logic-based Semantic Approach to Recognizing Textual Entailment

Marta Tatu and **Dan Moldovan**
Language Computer Corporation
Richardson, Texas, 75080
United States of America
marta,moldovan@languagecomputer.com

Abstract

This paper proposes a knowledge representation model and a logic proving setting with axioms on demand successfully used for recognizing textual entailments. It also details a lexical inference system which boosts the performance of the deep semantic oriented approach on the RTE data. The linear combination of two slightly different logical systems with the third lexical inference system achieves 73.75% accuracy on the RTE 2006 data.

1 Introduction

While communicating, humans use different expressions to convey the same meaning. One of the central challenges for natural language understanding systems is to determine whether different text fragments have the same meaning or, more generally, if the meaning of one text can be derived from the meaning of another. A module that recognizes the semantic entailment between two text snippets can be employed by many NLP applications. For example, Question Answering systems have to identify texts that entail expected answers. In Multi-document Summarization, the redundant information should be recognized and omitted from the summary.

Trying to boost research in textual inferences, the PASCAL Network proposed the *Recognizing Textual Entailment* (RTE) challenges (Dagan et al., 2005; Bar-Haim et al., 2006). For a pair of two text fragments, the task is to determine if the meaning of one text (the entailed hypothesis denoted by H) can be inferred from the meaning of the other text (the entailing text or T).

In this paper, we propose a model to represent the knowledge encoded in text and a logical setting suitable to a recognizing semantic entailment system. We cast the textual inference problem as a logic implication between meanings. Text T semantically entails H if its meaning logically implies the meaning of H. Thus, we, first, transform both text fragments into logic form, capture their meaning by detecting the *semantic* relations that hold between their constituents and load these rich logic representations into a natural language logic prover to decide if the entailment holds or not. Figure 1 illustrates our approach to RTE. The following sections of the paper shall detail the logic proving methodology, our logical representation of text and the various types of axioms that the prover uses.

To our knowledge, there are few logical approaches to RTE. (Bos and Markert, 2005) represents T and H into a first-order logic translation of the DRS language used in Discourse Representation Theory (Kamp and Reyle, 1993) and uses a theorem prover and a model builder with some generic, lexical and geographical background knowledge to prove the entailment between the two texts. (de Salvo Braz et al., 2005) proposes a Description Logic-based knowledge representation language used to induce the representations of T and H and uses an extended subsumption algorithm to check if any of T's representations obtained through equivalent transformations entails H.

2 Cogex - A Logic Prover for NLP

Our system uses COGEX (Moldovan et al., 2003), a natural language prover originating from OTTER (McCune, 1994). Once its set of support is loaded with T and the negated hypothesis ($\neg H$) and its usable list with the axioms needed to gener-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 819–826,
Sydney, July 2006.

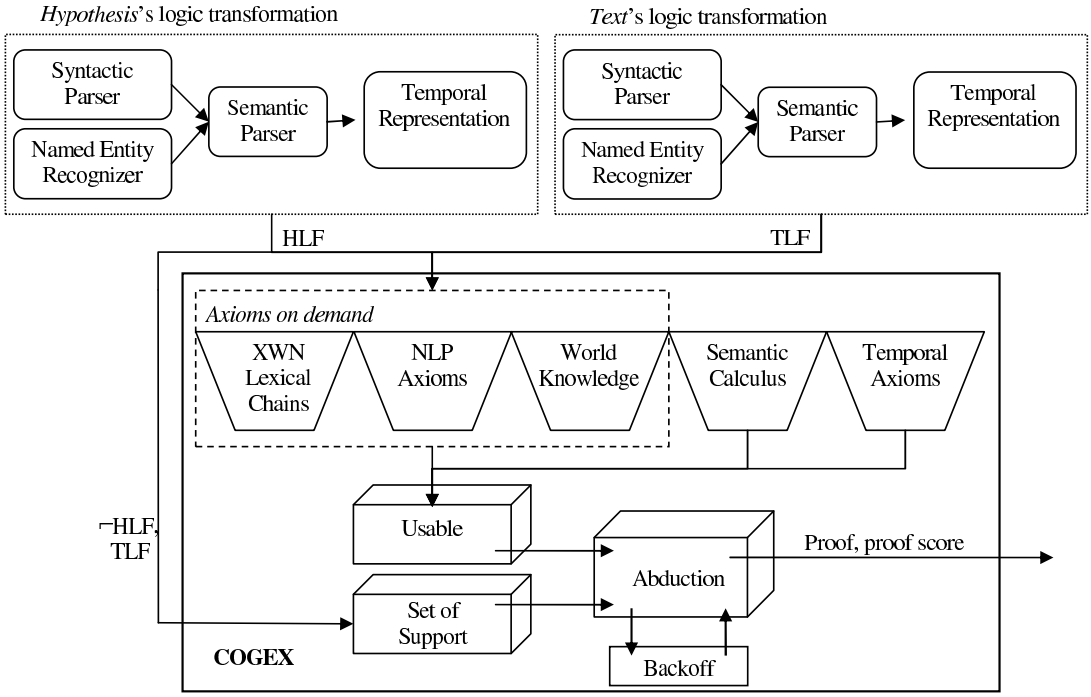


Figure 1: COGEX's Architecture

ate inferences, COGEX begins to search for proofs. To every inference, an appropriate weight is assigned depending on the axiom used for its derivation. If a refutation is found, the proof is complete; if a refutation cannot be found, then predicate arguments are relaxed. When argument relaxation fails to produce a refutation, entire predicates are dropped from the negated hypothesis until a refutation is found.

2.1 Proof scoring algorithm

Once a proof by contradiction is found, its score is computed by starting with an initial perfect score and deducting points for each axiom utilized in the proof, every relaxed argument, and dropped predicate. The computed score is a measure of the kinds of axioms used in the proof and the significance of the dropped arguments and predicates. If we assume that both text fragments are existential, then $T \vdash H$ if and only if T's entities are a subset of H's entities (*Some smart people read* $\vdash$ *Some people read*) and penalizing a pair whose H contains predicates that cannot be inferred is a correct way to ensure entailment (*Some people read* $\nvdash$ *Some smart people read*). But, if both T and H are universally quantified, then the groups mentioned in H must be a subset of the ones from T (*All people read* $\vdash$ *All smart people read* and *All smart people read* $\nvdash$ *All people read*). Thus, the scoring module adds back the points for the modifiers dropped from H and subtracts points for T's modifiers not present in H. The remaining two cases are summarized in Table 1.

Because (T, H) pairs with longer sentences can potentially drop more predicates and receive a lower score, COGEX normalizes the proof scores by dividing the assessed penalty by the maximum assessable penalty (all the predicates from H are dropped). If this final proof score is above a threshold learned on the development data, then the pair is labeled as positive entailment.

3 Knowledge Representation

For the textual entailment task, our logic prover uses a two-layered logical representation which captures the syntactic and semantic propositions encoded in a text fragment.

3.1 Logic Form Transformation

In the first stage of our representation process, COGEX converts T and H into logic forms (Moldovan and Rus, 2001). More specifically, a predicate is created for each noun, verb, adjective and adverb. The nouns that form a noun compound are gathered under a `nn_NNC` predicate. Each named entity class of a noun has a corresponding predicate which shares its argument with the noun predicate it modifies. Predicates for

$(\forall_T, \exists_H)$	$(\exists_T, \forall_H)$
All people read $\vdash$ *Some smart people read* *All smart people read* $\vdash$ *Some people read*	*Some people read* $\nvdash$ *All smart people read* *Some smart people read* $\nvdash$ *All people read*
Add the dropped points for H's modifiers	Subtract points for modifiers not present in H

Table 1: The quantification of T and H influences the proof scoring algorithm

prepositions and conjunctions are also added to link the text's constituents. This syntactic layer of the logic representation is, automatically, derived from a full parse tree and acknowledges syntax-based relationships such as: syntactic subjects, syntactic objects, prepositional attachments, complex nominals, and adjectival/adverbial adjuncts.

In order to objectively evaluate our representation, we derived it from two different sources: *constituency parse trees* (generated with our implementation of (Collins, 1997)) and *dependency parse trees* (created using Minipar (Lin, 1998))[1]. The two logic forms are slightly different. The dependency representation captures more accurately the syntactic dependencies between the concepts, but lacks the semantic information that our semantic parser extracts from the constituency parse trees. For instance, the sentence *Gilda Flores was kidnapped on the 13th of January 1990*[2] is "constituency" represented as `Gilda_NN(x1) & Flores_NN(x2) & nn_NNC(x3,x1,x2) & _human_NE(x3) & kidnap_VB(e1,x9,x3) & on_IN(e1,x8) & 13th_NN(x4) & of_NN(x5) & January_(x6) & 1990_NN(x7) & nn_NNC(x8,x4,x5,x6,x7) & _date_NE(x8)` and its "dependency" logic form is `Gilda_Flores_NN(x2) & _human_NE(x2) & kidnap_VB(e1,x4,x2) & on_IN(e1,x3) & 13th_NN(x3) & of_IN(x3,x1) & January_1990_NN(x1)`.

3.1.1 Negation

The exceptions to the one-predicate-per-open-class-word rule include the adverbs *not* and *never*. In cases similar to *further details were not released*, the system removes `not_RB(x3,e1)` and negates the verb's predicate (`-release_VB(e1,x1,x2)`). Similarly, for nouns whose determiner is *no*, for example, *No case of indigenously acquired rabies infection has been confirmed*, the verb's predicate is negated (`case_NN(x1) & -confirm_VB(e2,x15,x1)`).

3.2 Semantic Relations

The second layer of our logic representation adds the semantic relations, the underlying relationships between concepts. They provide the semantic background for the text, which allows for a denser connectivity between the concepts expressed in text. Our semantic parser takes free English text or parsed sentences and extracts a rich set of semantic relations[3] between words or concepts in each sentence. It focuses not only on the verb and its arguments, but also on semantic relations encoded in syntactic patterns such as complex nominals, genitives, adjectival phrases, and adjectival clauses. Our representation module maps each semantic relation identified by the parser to a predicate whose arguments are the events and entities that participate in the relation and it adds these semantic predicates to the logic form. For example, the previous logic form is augmented with the `THEME_SR(x3,e1) & TIME_SR(x8,e1)` relations[4] (*Gilda Flores* is the *theme* of the *kidnap* event and *13th of January 1990* shows the *time* of the *kidnapping*).

3.3 Temporal Representation

In addition to the semantic predicates, we represent every *date/time* into a normalized form `time_TMP(BeginFn(event), year, month, date, hour, minute, second) & time_TMP(EndFn(event), year, month, date, hour, minute, second)`. Furthermore, temporal reasoning

[1] The experimental results described in this paper were performed using two systems: the logic prover when it receives as input the *constituency* logic representation (COGEX$_C$) and the *dependency* representation (COGEX$_D$).

[2] All examples shown in this paper are from the entailment corpus released as part of the Second RTE challenge (`www.pascal-network.org/Challenges/RTE2`). The RTE datasets will be described in Section 7.

[3] We consider relations such as `AGENT, THEME, TIME, LOCATION, MANNER, CAUSE, INSTRUMENT, POSSESSION, PURPOSE, MEASURE, KINSHIP, ATTRIBUTE,` etc.

[4] `R(x,y)` should be read as "`x` *is* `R` *of* `y`".

predicates are derived from both the detected semantic relations as well as from a module which utilizes a learning algorithm to detect temporally ordered events ((S, E_1, E_2), where S is the temporal signal linking two events E_1 and E_2) (Moldovan et al., 2005). From each triple, temporally related SUMO predicates are generated based on hand-coded rules for the signal classes ($(S$ sequence, $E_1, E_2) \equiv$ `earlier_TMP(e1,e2)`, $(S$ contain, $E_1, E_2) \equiv$ `during_TMP(e1,e2)`, etc.). In the above example, *13th of January 1990* is normalized to the interval `time_TMP(BeginFn(e2), 1990, 1, 13, 0, 0, 0) & time_TMP(EndFn(e2), 1990, 1, 13, 23, 59, 59)` and `during_TMP(e1,e2)` is added to the logical representation to show when the *kidnapping* occurred.

4 Axioms on Demand

COGEX's usable list consists of all the axioms generated either automatically or by hand. The system generates axioms on demand for a given (T, H) pair whenever the semantic connectivity between two concepts needs to be established in a proof. The axioms on demand are lexical chains and world knowledge axioms. We are keen on the idea of axioms on demand since it is not possible to derive apriori all axioms needed in an arbitrary proof. This brings a considerable level of robustness to our entailment system.

4.1 eXtended WordNet lexical chains

For the semantic entailment task, the ability to recognize two semantically-related words is an important requirement. Therefore, we automatically construct lexical chains of WordNet relations from T's constituents to H's (Moldovan and Novischi, 2002). In order to avoid errors introduced by a Word Sense Disambiguation system, we used the first k senses for each word[5] unless the source and the target of the chain are synonyms. If a chain exists[6], the system generates, on demand, an axiom with the predicates of the source (from T) and the target (from H). For example, given the ISA relation between *murder#1* and *kill#1*, the system generates, when needed, the axiom `murder_VB(e1,x1,x2)` $\rightarrow$ `kill_VB(e1,x1,x2)`. The remaining of this section details some of the requirements for creating accurate lexical chains.

Because our extended version of WordNet has attached named entities to each noun synset, the lexical chain axioms append the entity name of the target concept, whenever it exists. For example, the logic prover uses the axiom `Nicaraguan_JJ(x1,x2)` $\rightarrow$ `Nicaragua_NN(x1) & _country_NE(x1)` when it tries to infer *electoral campaign is held in Nicaragua* from *Nicaraguan electoral campaign*.

We ensured the relevance of the lexical chains by limiting the path length to three relations and the set of WordNet relations used to create the chains by discarding the paths that contain certain relations in a particular order. For example, the automatic axiom generation module does not consider chains with an IS-A relation followed by a HYPONYMY link ($Chicago \stackrel{is-a}{\longrightarrow} city \stackrel{hyponymy}{\longrightarrow} Detroit$). Similarly, the system rejected chains with more than one HYPONYMY relations. Although these relations link semantically related concepts, the type of semantic similarity they introduce is not suited for inferences. Another restriction imposed on the lexical chains generated for entailment is not to start from or include too general concepts[7]. Therefore, we assigned to each noun and verb synset from WordNet a generality weight based on its relative position within its hierarchy and on its frequency in a large corpus. If d_i is the depth of concept c_i, D_{H_i} is the maximum depth in c_i's hierarchy H_i and $IC(c_i) = -log(p(c_i))$ is the information content of c_i measured on the British National Corpus, then

$$generalityW(c_i) = \frac{1}{\frac{d_i+1}{D_{H_i}} * IC(c_i)}.$$

In our experiments, we discarded the chains with concepts whose generality weight exceeded 0.8 such as *object_NN#1*, *act_VB#1*, *be_VB#1*, etc.

Another important change that we introduced in our extension of WordNet is the refinement of the DERIVATION relation which links verbs with their corresponding nominalized nouns. Because the relation is ambiguous regarding the role of the noun, we split

[5] Because WordNet senses are ranked based on their frequency, the correct sense is most likely among the first k. In our experiments, $k = 3$.

[6] Each lexical chain is assigned a weight based on its properties: shorter chains are better than longer ones, the relations are not equally important and their order in the chain influences its strength. If the weight of a chain is above a given threshold, the lexical chain is discarded.

[7] There are no restrictions on the target concept.

this relation in three: ACT-DERIVATION, AGENT-DERIVATION and THEME-DERIVATION. The role of the nominalization determines the argument given to the noun predicate. For instance, the axioms `act_VB(e1,x1,x2)` $\rightarrow$ `acting_NN(e1)` (ACT), `act_VB(e1,x1,x2)` $\rightarrow$ `actor_NN(x1)` (AGENT) reflect different types of derivation.

4.2 NLP Axioms

Our NLP axioms are linguistic rewriting rules that help break down complex logic structures and express syntactic equivalence. After analyzing the logic form and the parse trees of each text fragment, the system, automatically, generates axioms to break down complex nominals and coordinating conjunctions into their constituents so that other axioms can be applied, individually, to the components. These axioms are made available only to the (T, H) pair that generated them. For example, the axiom `nn_NNC(x3,x1,x2) & francisco_NN(x1) & merino_NN(x2)` $\rightarrow$ `merino_NN(x3)` breaks down the noun compound *Francisco Merino* into *Francisco* and *Merino* and helps COGEX infer *Merino's home* from *Francisco Merino's home*.

4.3 World Knowledge Axioms

Because, sometimes, the lexical or the syntactic knowledge cannot solve an entailment pair, we exploit the WordNet glosses, an abundant source of world knowledge. We used the logic forms of the glosses provided by eXtended WordNet[8] to, automatically, create our world knowledge axioms. For example, the first sense of noun *Pope* and its definition *the head of the Roman Catholic Church* introduces the axiom `Pope_NN(x1)` $\leftrightarrow$ `head_NN(x1) & of_IN(x1,x2) & Roman_Catholic_Church_NN(x2)` which is used by prover to show the entailment between T: *A place of sorrow, after Pope John Paul II died, became a place of celebration, as Roman Catholic faithful gathered in downtown Chicago to mark the installation of new Pope Benedict XVI.* and H: *Pope Benedict XVI is the new leader of the Roman Catholic Church.*

We also incorporate in our system a small common-sense knowledge base of 383 hand-coded world knowledge axioms, where 153 have been manually designed based on the entire development set data, and 230 originate from previous projects. These axioms express knowledge that could not be derived from WordNet regarding employment[9], family relations, awards, etc.

5 Semantic Calculus

The Semantic Calculus axioms combine two semantic relations identified within a text fragment and increase the semantic connectivity of the text (Tatu and Moldovan, 2005). A semantic axiom which combines two relations, R_i and R_j, is devised by observing the semantic connection between the w_1 and w_3 words for which there exists at least one other word, w_2, such that $R_i(w_1, w_2)$ ($w_1 \stackrel{R_i}{\rightarrow} w_2$) and $R_j(w_2, w_3)$ ($w_2 \stackrel{R_j}{\rightarrow} w_3$) hold true. We note that not any two semantic relations can be combined: R_i and R_j have to be compatible with respect to the part-of-speech of the common argument. Depending on their properties, there are up to 8 combinations between any two semantic relations and their inverses, not counting the combinations between a semantic relation and itself[10]. Many combinations are not semantically significant, for example, `KINSHIP_SR(x1,x2) & TEMPORAL_SR(x2,e1)` is unlikely to be found in text. Trying to solve the semantic combinations one comes upon in text corpora, we analyzed the RTE development corpora and devised rules for some of the $R_i \circ R_j$ combinations encountered. We validated these axioms by checking all the (w_1, w_3) pairs from the LA Times text collection such that $(R_i \circ R_j)(w_1, w_3)$ holds. We have identified 82 semantic axioms that show how semantic relations can be combined. These axioms enable inference of unstated meaning from the semantics detected in text. For example, if T states explicitly the KINSHIP (KIN) relations between *Nicholas Cage* and *Alice Kim Cage* and between *Alice Kim Cage* and *Kal-el Coppola Cage*, the logic prover uses the `KIN_SR(x1,x2) & KIN_SR(x2,x3)` $\rightarrow$ `KIN_SR(x1,x3)` semantic axiom (the transitivity of the blood relation) and the symmetry of this relationship (`KIN_SR(x1,x2)`

[8] `http://xwn.hlt.utdallas.edu`

[9] For example, the axiom `_country_NE(x1) & negotiator_NN(x2) & nn_NNC(x3,x1,x2)` $\rightarrow$ `work_VB(e1,x2,x4) & for_IN(e1,x1)` helps the prover infer that *Christopher Hill works for the US* from *top US negotiator, Christopher Hill*.

[10] Harabagiu and Moldovan (1998) lists the exact number of possible combinations for several WordNet relations and part-of-speech classes.

$\rightarrow$ KIN_SR(x2,x1)) to infer H's statement (KIN(*Kal-el Coppola Cage, Nicholas Cage*)). Another frequent axiom is LOCATION_SR(x1,x2) & PARTWHOLE_SR(x2,x3) $\rightarrow$ LOCATION_SR(x1,x3). Given the text *John lives in Dallas, Texas* and using the axiom, the system infers that *John lives in Texas*. The system applies the 82 axioms independent of the concepts involved in the semantic composition. There are rules that can be applied only if the concepts that participate satisfy a certain condition or if the relations are of a certain type. For example, LOCATION_SR(x1,x2) & LOCATION_SR(x2,x3) $\rightarrow$ LOCATION_SR(x1,x3) only if the LOCATION relation shows inclusion (*John is in the car in the garage* $\rightarrow$ LOCATION_SR(John,garage). *John is near the car behind the garage* $\not\rightarrow$ LOCATION_SR(John,garage)).

6 Temporal Axioms

One of the types of temporal axioms that we load in our logic prover links specific dates to more general time intervals. For example, *October 2000* entails the year *2000*. These axioms are automatically generated before the search for a proof starts. Additionally, the prover uses a SUMO knowledge base of temporal reasoning axioms that consists of axioms for a representation of time points and time intervals, Allen (Allen, 1991) primitives, and temporal functions. For example, *during* is a transitive Allen primitive: during_TMP(e1,e2) & during_TMP(e2,e3) $\rightarrow$ during_TMP(e1,e3).

7 Experiments and Results

The benchmark corpus for the RTE 2005 task consists of seven subsets with a 50%-50% split between the positive entailment examples and the negative ones. Each subgroup corresponds to a different NLP application: Information Retrival (IR), Comparable Documents (CD), Reading Comprehension (RC), Question Answering (QA), Information Extraction (IE), Machine Translation (MT), and Paraphrase Acquisition (PP). The RTE data set includes 1367 English (T, H) pairs from the news domain (political, economical, etc.). The RTE 2006 data covered only four NLP tasks (IE, IR, QA and Multi-document Summarization (SUM)) with an identical split between positive and negative examples. Table 2 presents the data statistics.

	Development set	Test set
RTE 2005	567	800
RTE 2006	800	800

Table 2: Datasets Statistics

7.1 COGEX's Results

Tables 3 and 4 summarize COGEX's performance on the RTE datasets, when it received as input the different-source logic forms[11].

On the RTE 2005 data, the overall performance on the test set is similar for both logic proving runs, COGEX_C and COGEX_D. On the development set, the semantically enhanced logic forms helped the prover distinguish better the positive entailments (COGEX_C has an overall higher precision than COGEX_D). If we analyze the performance on the test data, then COGEX_C performs slightly better on MT, CD and PP and worse on the RC, IR and QA tasks. The major differences between the two logic forms are the semantic content (incomplete for the dependency-derived logic forms) and, because the text's tokenization is different, the number of predicates in H's logic forms is different which leads to completely different proof scores.

On the RTE 2006 test data, the system which uses the dependency logic forms outperforms COGEX_C. COGEX_D performs better on almost all tasks (except SUM) and brings a significant improvement over COGEX_C on the IR task. Some of the positive examples that the systems did not label correctly require world knowledge that we do not have encoded in our axiom set. One example for which both systems returned the wrong answer is pair 353 (test 2006) where, from *China's decade-long practice of keeping its currency valued at around 8.28 yuan to the dollar*, the system should recognize the relation between the *yuan* and *China's currency* and infer that *the currency used in China is the yuan* because a *country's currency* $\vdash$ *currency used in the country*. Some of the pairs that the prover, currently, cannot handle involve numeric calculus and human-oriented estimations. Consider, for example, pair 359 (dev set, RTE 2006) labeled as positive, for which the logic prover could not determine that *15 safety violations* $\vdash$ *numerous safety violations*.

The deeper analysis of the systems' output

[11] For the RTE 2005 data, we list the *confidence-weighted score* (*cws*) (Dagan et al., 2005) and, for the RTE 2006 data, the *average precision* (*ap*) measure (Bar-Haim et al., 2006).

Task	COGEX$_C$			COGEX$_D$			LEXALIGN			COMBINATION		
	acc	*cws*	*f*	*acc*	*cws*	*f*	*acc*	*cws*	*f*	*acc*	*cws*	*f*
IE	**58.33**	60.90	60.31	57.50	57.03	51.42	56.66	53.41	59.99	62.50	67.63	57.14
IR	52.22	62.41	15.68	**53.33**	59.67	27.58	50.00	55.92	0.00	68.88	75.77	64.10
CD	**82.00**	88.90	79.69	79.33	87.15	74.38	**82.00**	88.04	80.57	84.66	91.73	82.70
QA	50.00	56.27	0.00	51.53	42.37	64.80	**53.07**	43.76	63.90	60.76	55.05	63.82
RC	53.57	56.38	38.09	**57.14**	59.32	58.33	57.85	60.26	49.57	60.00	62.89	50.00
MT	**55.83**	55.83	53.91	52.50	58.17	27.84	51.66	45.94	67.04	64.16	63.80	66.66
PP	**56.00**	63.11	26.66	54.00	58.15	30.30	50.00	47.03	0.00	68.00	75.27	63.63
TEST	59.37	63.09	48.00	59.12	57.17	54.52	59.12	55.74	59.17	**67.25**	**67.64**	**64.69**
DEV	63.66	63.44	64.48	61.19	63.63	57.52	62.08	59.94	60.83	70.37	71.89	66.66

Table 3: RTE 2005 data results (*accuracy, confidence-weighted score*, and *f-measure* for the true class)

Task	COGEX$_C$			COGEX$_D$			LEXALIGN			COMBINATION		
	acc	*ap*	*f*	*acc*	*ap*	*f*	*acc*	*ap*	*f*	*acc*	*ap*	*f*
IE	58.00	49.71	57.57	**59.00**	59.74	63.71	54.00	49.70	67.14	71.50	62.99	71.36
IR	62.50	65.91	56.14	**73.50**	72.50	73.89	64.50	69.45	65.02	74.00	74.30	72.92
QA	62.00	67.30	48.64	**64.00**	68.16	57.64	58.50	55.78	57.86	70.50	75.10	66.67
SUM	**74.50**	77.60	74.62	74.00	79.68	73.73	70.50	76.82	73.05	79.00	80.33	78.13
TEST	64.25	66.31	60.16	67.62	70.69	67.50	61.87	57.64	66.07	**73.75**	**71.33**	**72.37**
DEV	64.50	64.05	66.19	69.00	70.92	69.31	62.25	62.66	62.72	75.12	76.28	76.83

Table 4: RTE 2006 data results (*accuracy, average precision*, and *f-measure* for the true class)

showed that while WordNet lexical chains and NLP axioms are the most frequently used axioms throughout the proofs, the semantic and temporal axioms bring the highest improvement in accuracy, for the RTE data.

7.2 Lexical Alignment

Inspired by the positive examples whose H is in a high degree lexically subsumed by T, we developed a shallow system which measures their overlap by computing an edit distance between the text and the hypothesis. The cost of *deleting* a word from T ($w_T \rightarrow *$) is equal to 0, the cost of *replacing* a word from T with another from H ($w_T \rightarrow w_H$, where $w_T \neq w_H$ and w_T and w_H are not synonyms in WordNet) equal to ∞ (we do not allow replace operations) and the cost of *inserting* a word from H ($* \rightarrow w_H$) varies with the part-of-speech of the inserted word (higher values for WordNet nouns, adjectives or adverbs, lower for verbs and a minimum value for everything else). Table 5 shows a minimum cost alignment.

The performance of this lexical method (LEXALIGN) is shown in Tables 3 and 4. The alignment technique performs significantly better on the (T, H) pairs in the CD (RTE 2005) and SUM (RTE 2006) tasks. For these tasks, all three systems performed the best because the text of false pairs is not entailing the hypothesis even at the lexical level. For pair 682 (test set, RTE 2006), T and H have very few words overlapping and there are no axioms that can be used to derive knowledge that supports the hypothesis. Contrarily, for the IE task, the systems were fooled by the high word overlap between T and H. For example, pair 678's text (test set, RTE 2006) contains the entire hypothesis in its *if* clause. For this task, we had the highest number of false positives, around double when compared to the other applications. LEXALIGN works surprisingly well on the RTE data. It outperforms the semantic systems on the 2005 QA test data, but it has its limitations. The logic representations are generated from parse trees which are not always accurate (∼86% accuracy). Once syntactic and semantic parsers are perfected, the logical semantic approach shall prove its potential.

7.3 Merging three systems

Because the two logical representations and the lexical method are very different and perform better on different sets of tasks, we combined the scores returned by each system[12] to see if a mixed approach performs better than each individual method. For each NLP task, we built a classifier based on the linear combination of the three scores. Each task's classifier labels pair i as positive if $\lambda_{cogex_C} score_C(i) + \lambda_{cogex_D} score_D(i) +$

[12]Each system returns a score between 0 and 1, a number close to 0 indicating a probable negative example and a number close to 1 indicating a probable positive example. Each (T, H) pair's lexical alignment score, $score_{LexAlign}$, is the normalized average edit distance cost.

T:	The Council of Europe	has	*	45 member states.	Three countries from ...
		DEL	INS		DEL
H:	The Council of Europe	*	is made up by	45 member states.	*

Table 5: The lexical alignment for RTE 2006 pair 615 (test set)

$\lambda_{LexAlign} score_{LexAlign}(i) > 0.5$, where the optimum values of the classifier's real-valued parameters ($\lambda_{cogex_C}, \lambda_{cogex_D}, \lambda_{LexAlign}$) were determined using a grid search on each development set. Given the different nature of each application, the λ parameters vary with each task. For example, the final score given to each IE 2006 pair is highly dependent on the score given by COGEX when it received as input the logic forms created from the constituency parse trees with a small correction from the dependency parse trees logic form system[13]. For the IE task, the lexical alignment performs the worst among the three systems. On the other hand, for the IR task, the score given by LEXALIGN is taken into account[14]. Tables 3 and 4 summarize the performance of the three system combination. This hybrid approach performs better than all other systems for all measures on all tasks. It displays the same behavior as its dependents: high accuracy on the CD and SUM tasks and many false positives for the IE task.

8 Conclusion

In this paper, we present a logic form representation of knowledge which captures syntactic dependencies as well as semantic relations between concepts and includes special temporal predicates. We implemented several changes to our WordNet lexical chains module which lead to fewer unsound axioms and incorporated in our logic prover semantic and temporal axioms which decrease its dependence on world knowledge. We plan to improve our logic prover to detect false entailments even when the two texts have a high word overlap and expand our axiom set.

References

J. Allen. 1991. Time and Time Again: The Many Ways to Represent Time. *Internatinal Journal of Intelligent Systems*, 4(6):341–355.

R. Bar-Haim, I. Dagan, B. Dolan, L. Ferro, D. Giampiccolo, B. Magnini, and I. Szpektor. 2006. The Second PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the Second PASCAL Challenges Workshop*.

J. Bos and K. Markert. 2005. Recognizing Textual Entailment with Logical Inference. In *Proceedings of HLT/EMNLP 2005*, Vancouver, Canada, October.

M. Collins. 1997. Three Generative, Lexicalized Models for Statistical Parsing. In *Proceedings of the ACL-97*.

I. Dagan, O. Glickman, and B. Magnini. 2005. The PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the PASCAL Challenges Workshop*, Southampton, U.K., April.

R. de Salvo Braz, R. Girju, V. Punyakanok, D. Roth, and M. Sammons. 2005. An Inference Model for Semantic Entailment in Natural Language. In *Proceedings of AAAI-2005*.

S. Harabagiu and D. Moldovan. 1998. Knowledge Processing on Extended WordNet. In Christiane Fellbaum, editor, *WordNet: an Electronic Lexical Database and Some of its Applications*, pages 379–405. MIT Press.

H. Kamp and U. Reyle. 1993. *From Discourse to Logic: Introduction to Model-theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Kluwer Academic Publishers.

D. Lin. 1998. Dependency-based Evaluation of MINIPAR. In *Workshop on the Evaluation of Parsing Systems*, Granada, Spain, May.

William W. McCune, 1994. *OTTER 3.0 Reference Manual and Guide*.

D. Moldovan and A. Novischi. 2002. Lexical chains for Question Answering. In *Proceedings of COLING*, Taipei, Taiwan, August.

D. Moldovan and V. Rus. 2001. Logic Form Transformation of WordNet and its Applicability to Question Answering. In *Proceedings of ACL*, France.

D. Moldovan, C. Clark, S. Harabagiu, and S. Maiorano. 2003. COGEX A Logic Prover for Question Answering. In *Proceedings of the HLT/NAACL*.

D. Moldovan, C. Clark, and S. Harabagiu. 2005. Temporal Context Representation and Reasoning. In *Proceedings of IJCAI*, Edinburgh, Scotland.

M. Tatu and D. Moldovan. 2005. A Semantic Approach to Recognizing Textual Entailment. In *Proceedings of HLT/EMNLP*.

[13] $\lambda_{cogex_C} = 1.1, \lambda_{cogex_D} = 0.3, \lambda_{LexAlign} = -0.6$

[14] $\lambda_{cogex_C} = 0.3, \lambda_{cogex_D} = 0.1, \lambda_{LexAlign} = 0.6$

Infrastructure for standardization of Asian language resources

Tokunaga Takenobu
Tokyo Inst. of Tech.

Virach Sornlertlamvanich
TCL, NICT

Thatsanee Charoenporn
TCL, NICT

Nicoletta Calzolari
ILC/CNR

Monica Monachini
ILC/CNR

Claudia Soria
ILC/CNR

Chu-Ren Huang
Academia Sinica

Xia YingJu
Fujitsu R&D Center

Yu Hao
Fujitsu R&D Center

Laurent Prevot
Academia Sinica

Shirai Kiyoaki
JAIST

Abstract

As an area of great linguistic and cultural diversity, Asian language resources have received much less attention than their western counterparts. Creating a common standard for Asian language resources that is compatible with an international standard has at least three strong advantages: to increase the competitive edge of Asian countries, to bring Asian countries to closer to their western counterparts, and to bring more cohesion among Asian countries. To achieve this goal, we have launched a two year project to create a common standard for Asian language resources. The project is comprised of four research items, (1) building a description framework of lexical entries, (2) building sample lexicons, (3) building an upper-layer ontology and (4) evaluating the proposed framework through an application. This paper outlines the project in terms of its aim and approach.

1 Introduction

There is a long history of creating a standard for western language resources. The human language technology (HLT) society in Europe has been particularly zealous for the standardization, making a series of attempts such as EAGLES[1], PAROLE/SIMPLE (Lenci et al., 2000), ISLE/MILE (Calzolari et al., 2003) and LIRICS[2]. These continuous efforts has been crystallized as activities in ISO-TC37/SC4 which aims to make an international standard for language resources.

[1] http://www.ilc.cnr.it/Eagles96/home.html

[2] lirics.loria.fr/documents.html

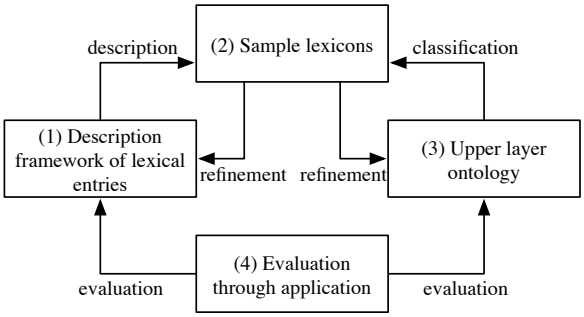


Figure 1: Relations among research items

On the other hand, since Asia has great linguistic and cultural diversity, Asian language resources have received much less attention than their western counterparts. Creating a common standard for Asian language resources that is compatible with an international standard has at least three strong advantages: to increase the competitive edge of Asian countries, to bring Asian countries to closer to their western counterparts, and to bring more cohesion among Asian countries.

To achieve this goal, we have launched a two year project to create a common standard for Asian language resources. The project is comprised of the following four research items.

(1) building a description framework of lexical entries

(2) building sample lexicons

(3) building an upper-layer ontology

(4) evaluating the proposed framework through an application

Figure 1 illustrates the relations among these research items.

Our main aim is the research item (1), building a description framework of lexical entries which

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 827–834,
Sydney, July 2006.

fits with as many Asian languages as possible, and contributing to the ISO-TC37/SC4 activities. As a starting point, we employ an existing description framework, the MILE framework (Bertagna et al., 2004a), to describe several lexical entries of several Asian languages. Through building sample lexicons (research item (2)), we will find problems of the existing framework, and extend it so as to fit with Asian languages. In this extension, we need to be careful in keeping consistency with the existing framework. We start with Chinese, Japanese and Thai as target Asian languages and plan to expand the coverage of languages. The research items (2) and (3) also comprise the similar feedback loop. Through building sample lexicons, we refine an upper-layer ontology. An application built in the research item (4) is dedicated to evaluating the proposed framework. We plan to build an information retrieval system using a lexicon built by extending the sample lexicon.

In what follows, section 2 briefly reviews the MILE framework which is a basis of our description framework. Since the MILE framework is originally designed for European languages, it does not always fit with Asian languages. We exemplify some of the problems in section 3 and suggest some directions to solve them. We expect that further problems will come into clear view through building sample lexicons. Section 4 describes a criteria to choose lexical entries in sample lexicons. Section 5 describes an approach to build an upper-layer ontology which can be sharable among languages. Section 6 describes an application through which we evaluate the proposed framework.

2 The MILE framework for interoperability of lexicons

The ISLE (International Standards for Language Engineering) Computational Lexicon Working Group has consensually defined the MILE (Multilingual ISLE Lexical Entry) as a standardized infrastructure to develop multilingual lexical resources for HLT applications, with particular attention to Machine Translation (MT) and Crosslingual Information Retrieval (CLIR) application systems.

The MILE is a general architecture devised for the encoding of multilingual lexical information, a meta-entry acting as a common representational layer for multilingual lexicons, by allowing integration and interoperability between different monolingual lexicons[3].

This formal and standardized framework to encode MILE-conformant lexical entries is provided to lexicon and application developers by the overall MILE Lexical Model (MLM). As concerns the horizontal organization, the MLM consists of two independent, but interlinked primary components, the monolingual and the multilingual modules. The monolingual component, on the vertical dimension, is organized over three different representational layers which allow to describe different dimensions of lexical entries, namely the morphological, syntactic and semantic layers. Moreover, an intermediate module allows to define mechanisms of linkage and mapping between the syntactic and semantic layers. Within each layer, a basic linguistic information unit is identified; basic units are separated but still interlinked each other across the different layers.

Within each of the MLM layers, different types of lexical object are distinguished :

- the MILE Lexical Classes (MLC) represent the main building blocks which formalize the basic lexical notions. They can be seen as a set of structural elements organized in a layered fashion: they constitute an ontology of lexical objects as an abstraction over different lexical models and architectures. These elements are the backbone of the structural model. In the MLM a definition of the classes is provided together with their attributes and the way they relate to each other. Classes represent notions like InflectionalParadigm, SyntacticFunction, SyntacticPhrase, Predicate, Argument,

- the MILE Data Categories (MDC) which constitute the attributes and values to adorn the structural classes and allow concrete entries to be instantiated. MDC can belong to a shared repository or be user-defined. “NP” and “VP” are data category instances of the class SyntacticPhrase, whereas and “subj” and “obj” are data category instances of the class SyntacticFunction.

- lexical operations, which are special lexical entities allowing the user to define multilin-

[3]MILE is based on the experience derived from existing computational lexicons (e.g. LE-PAROLE, SIMPLE, EuroWordNet, etc.).

gual conditions and perform operations on lexical entries.

Originally, in order to meet expectations placed upon lexicons as critical resources for content processing in the Semantic Web, the MILE syntactic and semantic lexical objects have been formalized in RDF(S), thus providing a web-based means to implement the MILE architecture and allowing for encoding individual lexical entries as instances of the model (Ide et al., 2003; Bertagna et al., 2004b). In the framework of our project, by situating our work in the context of W3C standards and relying on standardized technologies underlying this community, the original RDF schema for ISLE lexical entries has been made compliant to OWL. The whole data model has been formalized in OWL by using Protégé 3.2 beta and has been extended to cover the morphological component as well (see Figure 2). Protégé 3.2 beta has been also used as a tool to instantiate the lexical entries of our sample monolingual lexicons, thus ensuring adherence to the model, encoding coherence and inter- and intra-lexicon consistency.

3 Existing problems with the MILE framework for Asian languages

In this section, we will explain some problematic phenomena of Asian languages and discuss possible extensions of the MILE framework to solve them.

Inflection The MILE provides the powerful framework to describe the information about inflection. **InflectedForm** class is devoted to describe inflected forms of a word, while **InflectionalParadigm** to define general inflection rules. However, there is no inflection in several Asian languages, such as Chinese and Thai. For these languages, we do not use the Inflected Form and Inflectional Paradigm.

Classifier Many Asian languages, such as Japanese, Chinese, Thai and Korean, do not distinguish singularity and plurality of nouns, but use classifiers to denote the number of objects. The followings are examples of classifiers of Japanese.

- *inu* *ni* *hiki* $\cdots$ two dogs
  (dog) (two) (CL)
- *hon* *go* *satsu* $\cdots$ five books
  (book) (five) (CL)

"CL" stands for a classifier. They always follow cardinal numbers in Japanese. Note that different classifiers are used for different nouns. In the above examples, classifier "*hiki*" is used to count noun "*inu* (dog)", while "*satsu*" for "*hon* (book)". The classifier is determined based on the semantic type of the noun.

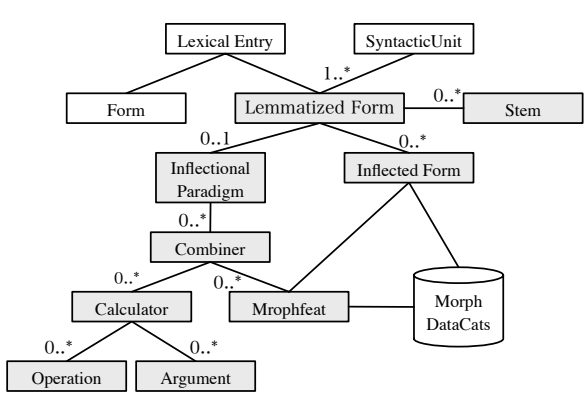
In the Thai language, classifiers are used in various situations (Sornlertlamvanich et al., 1994). The classifier plays an important role in construction with noun to express ordinal, pronoun, for instance. The classifier phrase is syntactically generated according to a specific pattern. Here are some usages of classifiers and their syntactic patterns.

- Enumeration
  (Noun/Verb)-(cardinal number)-(CL)
  e.g. *nakrian* *3* *khon* $\cdots$ three students
  (student) (CL)
- Ordinal
  (Noun)-(CL)-/thi:/-(cardinal number)
  e.g. *kaew* *bai* *thi: 4* $\cdots$ the 4th glass
  (glass) (CL) (4th)
- Determination
  (Noun)-(CL)-(Determiner)
  e.g. *kruangkhidlek* *kruang* *nii*
  (calculator) (CL) (this)
  $\cdots$ this calculator

Classifiers could be dealt as a class of the part-of-speech. However, since classifiers depend on the semantic type of nouns, we need to refer to semantic features in the morphological layer, and vice versa. Some mechanism to link between features beyond layers needs to be introduced into the current MILE framework.

Orthographic variants Many Chinese words have orthographic variants. For instance, the concept of rising can be represented by either character variants of sheng1: 升 or 昇. However, the free variants become non-free in certain compound forms. For instance, only 升 allowed for 公升 'liter', and only 昇 is allowed for 昇華 'to sublime'. The interaction of lemmas and orthographic variations is not yet represented in MILE.

Reduplication as a derivational process In some Asian languages, reduplication of words derives another word, and the derived word often has a different part-of-speech. Here are some examples of reduplication in Chinese. Man4 慢 'to be slow' is a state verb, while a reduplicated form




```
<LemmatizedForm rdf:ID="LFstar">
 <hasInflectedForm>
  <InflectedForm rdf:ID="stars">
   <hasMorphoFeat>
  <MorphoFeat rdf:ID="pl">
   <number rdf:datatype="http://www.w3c.org/
2001/ XMLSchema#string">
     plural
   </number>
  </MorphoFeat>
   </hasMorphoFeat>
  </InflectedForm>
 </hasInflectedForm>
 <hasInflectedForm>
  <InflectedForm rdf:ID="star">
   <hasMorphoFeat>
  <MorphoFeat rdf:ID="sg">
   <number rdf:datatype="http://www.w3c.org/
2001/ XMLSchema#string">
     singular
   </number>
  </MorphoFeat>
   </hasMorphoFeat>
  </InflectedForm>
 </hasInflectedForm>
</LemmatiedForm>
```

Figure 2: Formalization of the morphological layer and excerpt of a sample RDF instantiation

man4-man4 慢慢 is an adverb. Another example of reduplication involves verbal aspect. Kan4 看 'to look' is an activity verb, while the reduplicative form kan4-kan4 看看, refers to the tentative aspect, introducing either stage-like sub-division or the event or tentativeness of the action of the agent. This morphological process is not provided for in the current MILE standard.

There are also various usages of reduplication in Thai. Some words reduplicate themselves to add a specific aspect to the original meaning. The reduplication can be grouped into 3 types according to the tonal sound change of the original word.

- Word reduplication without sound change
  e.g. /dek-dek/ · · · (N) children, (ADV) childishly, (ADJ) childish
  /sa:w-sa:w/ · · · (N) women
- Word reduplication with high tone on the first word
  e.g. /dam4-dam/ · · · (ADJ) extremely black
  /bo:i4-bo:i/ · · · (ADV) really often
- Triple word reduplication with high tone on the second word
  e.g. /dern-dern4-dern/ ·· (V) intensively walk
  /norn-norn4-norn/··(V) intensively sleep

In fact, only the reduplication of the same sound is accepted in the written text, and a special symbol, namely /mai-yamok/ is attached to the original word to represent the reduplication. The reduplication occurs in many parts-of-speech, such as noun, verb, adverb, classifier, adjective, preposition. Furthermore, various aspects can be added to the original meaning of the word by reduplication, such as pluralization, emphasis, generalization, and so on. These aspects should be instantiated as features.

Change of parts-of-speech by affixes Affixes change parts-of-speech of words in Thai (Charoenporn et al., 1997). There are three prefixes changing the part-of-speech of the original word, namely /ka:n/, /khwa:m/, /ya:ng/. They are used in the following cases.

- Nominalization
  /ka:n/ is used to prefix an action verb and /khwa:m/ is used to prefix a state verb in nominalization such as /ka:n-tham-nga:n/ (working), /khwa:m-suk/ (happiness).
- Adverbialization
  An adverb can be derived by using /ya:ng/ to prefix a state verb such as /ya:ng-di:/ (well).

Note that these prefixes are also words, and form multi-word expressions with the original word. This phenomenon is similar to derivation which is not handled in the current MILE framework. Derivation is traditionally considered as a different phenomenon from inflection, and current MILE focuses on inflection. The MILE framework is already being extended to treat such linguistic phenomenon, since it is important to European languages as well. It would be handled in either the morphological layer or syntactic layer.

Function Type Function types of predicates (verbs, adjectives etc.) might be handled in a partially different way for Japanese. In the syntactic layer of the MILE framework, **FunctionType** class is prepared to denote subcategorization frames of predicates, and they have function types such as "subj" and "obj". For example, the verb "eat" has two **FunctionType** data categories of "subj" and "obj". Function types basically stand for positions of case filler nouns. In Japanese, cases are usually marked by postpositions and case filler positions themselves do not provide much information on case marking. For example, both of the following sentences mean the same, "She eats a pizza."

- *kanojo ga piza wo taberu*
  (she) (NOM) (pizza) (ACC) (eat)
- *piza wo kanojo ga taberu*
  (pizza) (ACC) (she) (NOM) (eat)

"*Ga*" and "*wo*" are postpositions which mark nominative and accusative cases respectively. Note that two case filler nouns "she" and "pizza" can be exchanged. That is, the number of slots is important, but their order is not.

For Japanese, we might use the set of postpositions as values of **FunctionType** instead of conventional function types such as "subj" and "obj". It might be an user defined data category or language dependent data category. Furthermore, it is preferable to prepare the mapping between Japanese postpositions and conventional function types. This is interesting because it seems more a terminological difference, but the model can be applied also to Japanese.

4 Building sample lexicons

4.1 Swadesh list and basic lexicon

The issue involved in defining a basic lexicon for a given language is more complicated than one may think (Zhang et al., 2004). The naive approach of simply taking the most frequent words in a language is flawed in many ways. First, all frequency counts are corpus-based and hence inherit the bias of corpus sampling. For instance, since it is easier to sample written formal texts, words used predominantly in informal contexts are usually underrepresented. Second, frequency of content words is topic-dependent and may vary from corpus to corpus. Last, and most crucially, frequency of a word does not correlate to its conceptual necessity, which should be an important, if not only, criteria for core lexicon. The definition of a cross-lingual basic lexicon is even more complicated. The first issue involves determination of cross-lingual lexical equivalencies. That is, how to determine that word *a* (and not *a'*) in language *A* really is word *b* in language *B*. The second issue involves the determination of what is a basic word in a multilingual context. In this case, not even the frequency offers an easy answer since lexical frequency may vary greatly among different languages. The third issue involves lexical gaps. That is, if there is a word that meets all criteria of being a basic word in language *A*, yet it does not exist in language *D* (though it may exist in languages *B*, and *C*). Is this word still qualified to be included in the multilingual basic lexicon?

It is clear not all the above issues can be unequivocally solved with the time frame of our project. Fortunately, there is an empirical core lexicon that we can adopt as a starting point. The Swadesh list was proposed by the historical linguist Morris Swadesh (Swadesh, 1952), and has been widely used by field and historical linguists for languages over the world. The Swadesh list was first proposed as lexico-statistical metrics. That is, these are words that can be reliably expected to occur in all historical languages and can be used as the metrics for quantifying language variations and language distance. The Swadesh list is also widely used by field linguists when they encounter a new language, since almost all of these terms can be expected to occur in any language. Note that the Swadesh list consists of terms that embody human direct experience, with culture-specific terms avoided. Swadesh started with a 215 items list, before cutting back to 200 items and then to 100 items. A standard list of 207 items is arrived at by unifying the 200 items list and the 100 items list. We take the 207 terms from the Swadesh list as the core of our basic lexicon. Inclusion of the Swadesh list also gives us the possibility of covering many Asian languages in which we do not have the resources to make a full and fully annotated lexicon. For some of these languages, a Swadesh lexicon for reference is provided by a collaborator.

4.2 Aligning multilingual lexical entries

Since our goal is to build a multilingual sample lexicon, it is required to align words in several

Asian languages. In this subsection, we propose a simple method to align words in different languages. The basic idea for multilingual alignment is an intermediary by English. That is, first we prepare word pairs between English and other languages, then combine them together to make correspondence among words in several languages. The multilingual alignment method currently we consider is as follows:

1. Preparing the set of frequent words of each language

   Suppose that $\{Jw_i\}$, $\{Cw_i\}$, $\{Tw_i\}$ is the set of frequent words of Japanese, Chinese and Thai, respectively. Now we try to construct a multilingual lexicon for these three languages, however, our multilingual alignment method can be easily extended to handle more languages.

2. Obtaining English translations

   A word Xw_i is translated into a set of English words EXw_{ij} by referring to the bilingual dictionary, where X denotes one of our languages, J, C or T. We can obtain mappings as in (1).

$$\begin{array}{ll} Jw_1: & EJw_{11}, EJw_{12}, \cdots \\ Jw_2: & EJw_{21}, EJw_{22}, \cdots \\ \vdots & \\ \hline Cw_1: & ECw_{11}, ECw_{12}, \cdots \\ Cw_2: & ECw_{21}, ECw_{22}, \cdots \\ \vdots & \\ \hline Tw_1: & ETw_{11}, ETw_{12}, \cdots \\ Tw_2: & ETw_{21}, ETw_{22}, \cdots \\ \vdots & \end{array} \quad (1)$$

   Notice that this procedure is automatically done and ambiguities would be left at this stage.

3. Generating new mapping

   From mappings in (1), a new mapping is generated by inverting the key. That is, in the new mapping, a key is an English word Ew_i and a correspondence for each key is sets of translations XEw_{ij} for 3 languages, as shown in (2):

$$\begin{array}{ll} Ew_1: & (JEw_{11}, JEw_{12}, \cdots) \\ & (CEw_{11}, CEw_{12}, \cdots) \\ & (TEw_{11}, TEw_{12}, \cdots) \\ Ew_2: & (JEw_{21}, JEw_{22}, \cdots) \\ & (CEw_{21}, CEw_{22}, \cdots) \\ & (TEw_{21}, TEw_{22}, \cdots) \\ \vdots & \end{array} \quad (2)$$

   Notice that at this stage, correspondence between different languages is very loose, since they are aligned on the basis of sharing only a single English word.

4. Refinement of alignment

   Groups of English words are constructed by referring to the WordNet synset information. For example, suppose that Ew_i and Ew_j belong to the same synset S_k. We will make a new alignment by making an intersection of $\{XEw_i\}$ and $\{XEw_j\}$ as shown in (3).

$$\begin{array}{r} Ew_i: (JEw_{i1}, \cdot\cdot)\,(CEw_{i1}, \cdot\cdot)\,(TEw_{i1}, \cdot\cdot) \\ Ew_j: (JEw_{j1}, \cdot\cdot)\,(CEw_{j1}, \cdot\cdot)\,(TEw_{j1}, \cdot\cdot) \\ \Downarrow \text{intersection} \\ S_k: \ (JEw'_{k1}, \cdot\cdot)(CEw'_{k1}, \cdot\cdot)(TEw'_{k1}, \cdot\cdot) \end{array} \quad (3)$$

   In (3), the key is a synset S_k, which is supposed to be a conjunction of Ew_i and Ew_j, and the counterpart is the intersection of set of translations for each language. This operation would reduce the number of words of each language. That means, we can expect that the correspondence among words of different languages becomes more precise. This new word alignment based on a synset is a final result.

To evaluate the performance of this method, we conducted a preliminary experiment using the Swadesh list. Given the Swadesh list of Chinese, Italian, Japanese and Thai as a gold standard, we tried to replicate these lists from the English Swadesh list and bilingual dictionaries between English and these languages. In this experiment, we did not perform the refinement step with WordNet. From 207 words in the Swadesh list, we dropped 4 words ("at", "in", "with" and "and") due to their too many ambiguities in translation.

As a result, we obtained 181 word groups aligned across 5 languages (Chinese, English, Italian, Japanese and Thai) for 203 words. An aligned word group was judged "correct" when the words of each language include only words in the Swadesh list of that language. It was judged "partially correct" when the words of a language also include the words which are not in the Swadesh list. Based on the correct instances, we obtain 0.497 for precision and 0.443 for recall. These figures go up to 0.912 for precision and 0.813 for recall when based on the partially correct instances. This is quite a promising result.

5 Upper-layer ontology

The empirical success of the Swadesh list poses an interesting question that has not been explored before. That is, does the Swadesh list instantiates a shared, fundamental human conceptual structure? And if there is such as a structure, can we discover it?

In the project these fundamental issues are associated with our quest for cross-lingual interoperability. We must make sure that the items of the basic lexicon are given the same interpretation. One measure taken to ensure this consists in constructing an upper-ontology based on the basic lexicon. Our preliminary work of mapping the Swadesh list items to SUMO (Suggested Upper Merged Ontology) (Niles and Pease, 2001) has already been completed. We are in the process of mapping the list to DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering) (Masolo et al., 2003). After the initial mapping, we carry on the work to restructure the mapped nodes to form a genuine conceptual ontology based on the language universal basic lexical items. However one important observation that we have made so far is that the success of the Swadesh list is partly due to its underspecification and to the liberty it gives to compilers of the list in a new language. If this idea of underspecification is essential for basic lexicon for human languages, then we must resolve this apparent dilemma of specifying them in a formal ontology that requires fully specified categories. For the time being, genuine ambiguities resulted in the introduction of each disambiguated sense in the ontology. We are currently investigating another solution that allows the inclusion of underspecified elements in the ontology without threatening its coherence. More specifically we introduce a underspecified relation in the structure for linking the underspecified meaning to the different specified meaning. The specified meanings are included in the taxonomic hierarchy in a traditional manner, while a hierarchy of underspecified meanings can be derived thanks to the new relation. An underspecified node only inherits from the most specific common mother of its fully specified terms. Such distinction avoids the classical misuse of the subsumption relation for representing multiple meanings. This method does not reflect a dubious collapse of the linguistic and conceptual levels but the treatment of such underspecifications as truly conceptual. Moreover we hope this proposal will provide a knowledge representation framework for the multilingual alignment method presented in the previous section.

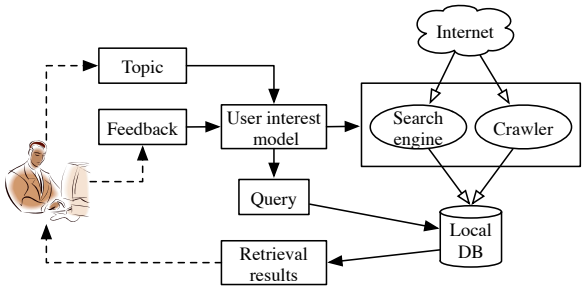


Figure 3: The system architecture

Finally, our ontology will not only play the role of a structured interlingual index. It will also serve as a common conceptual base for lexical expansion, as well as for comparative studies of the lexical differences of different languages.

6 Evaluation through an application

To evaluate the proposed framework, we are building an information retrieval system. Figure 3 shows the system architecture.

A user can input a topic to retrieve the documents related to that topic. A topic can consist of keywords, website URL's and documents which describe the topic. From the topic information, the system builds a user interest model. The system then uses a search engine and a crawler to search for information related to this topic in WWW and stores the results in the local database. Generally, the search results include many noises. To filter out these noises, we build a query from the user interest model and then use this query to retrieve documents in the local database. Those documents similar to the query are considered as more related to the topic and the user's interest, and are returned to the user. When the user obtains these retrieval results, he can evaluate these documents and give the feedback to the system, which is used for the further refinement of the user interest model.

Language resources can contribute to improving the system performance in various ways. Query expansion is a well-known technique which expands user's query terms into a set of similar and related terms by referring to ontologies. Our system is based on the vector space model (VSM) and traditional query expansion can be applicable using the ontology.

There has been less research on using lexical in-

formation for information retrieval systems. One possibility we are considering is query expansion by using predicate-argument structures of terms. Suppose a user inputs two keywords, "hockey" and "ticket" as a query. The conventional query expansion technique expands these keywords to a set of similar words based on an ontology. By referring to predicate-argument structures in the lexicon, we can derive actions and events as well which take these words as arguments. In the above example, by referring to the predicate-argument structure of "buy" or "sell", and knowing that these verbs can take "ticket" in their object role, we can add "buy" and "sell" to the user's query. This new type of expansion requires rich lexical information such as predicate argument structures, and the information retrieval system would be a good touchstone of the lexical information.

7 Concluding remarks

This paper outlined a new project for creating a common standard for Asian language resources in cooperation with other initiatives. We start with three Asian languages, Chinese, Japanese and Thai, on top of the existing framework which was designed mainly for European languages. We plan to distribute our draft to HLT societies of other Asian languages, requesting for their feedback through various networks, such as the Asian language resource committee network under Asian Federation of Natural Language Processing (AFNLP)[4], and Asian Language Resource Network project[5]. We believe our efforts contribute to international activities like ISO-TC37/SC4[6] (Francopoulo et al., 2006) and to the revision of the ISO Data Category Registry (ISO 12620), making it possible to come close to the ideal international standard of language resources.

Acknowledgment

This research was carried out through financial support provided under the NEDO International Joint Research Grant Program (NEDO Grant).

References

F. Bertagna, A. Lenci, M. Monachini, and N. Calzolari. 2004a. Content interoperability of lexical resources, open issues and "MILE" perspectives. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC2004)*, pages 131–134.

F. Bertagna, A. Lenci, M. Monachini, and N. Calzolari. 2004b. The MILE lexical classes: Data categories for content interoperability among lexicons. In *A Registry of Linguistic Data Categories within an Integrated Language Resources Repository Area – LREC2004 Satellite Workshop*, page 8.

N. Calzolari, F. Bertagna, A. Lenci, and M. Monachini. 2003. Standards and best practice for multilingual computational lexicons. MILE (the multilingual ISLE lexical entry). ISLE Deliverable D2.2&3.2.

T. Charoenporn, V. Sornlertlamvanich, and H. Isahara. 1997. Building a large Thai text corpus — part-of-speech tagged corpus: ORCHID—. In *Proceedings of the Natural Language Processing Pacific Rim Symposium*.

G. Francopoulo, G. Monte, N. Calzolari, M. Monachini, N. Bel, M. Pet, and C. Soria. 2006. Lexical markup framework (LMF). In *Proceedings of LREC2006 (forthcoming)*.

N. Ide, A. Lenci, and N. Calzolari. 2003. RDF instantiation of ISLE/MILE lexical entries. In *Proceedings of the ACL 2003 Workshop on Linguistic Annotation: Getting the Model Right*, pages 25–34.

A. Lenci, N. Bel, F. Busa, N. Calzolari, E. Gola, M. Monachini, A. Ogonowsky, I. Peters, W. Peters, N. Ruimy, M. Villegas, and A. Zampolli. 2000. SIMPLE: A general framework for the development of multilingual lexicons. *International Journal of Lexicography, Special Issue, Dictionaries, Thesauri and Lexical-Semantic Relations*, XIII(4):249–263.

C. Masolo, A. Borgo, S.; Gangemi, N. Guarino, and A. Oltramari. 2003. Wonderweb deliverable d18 –ontology library (final)–. Technical report, Laboratory for Applied Ontology, ISTC-CNR.

I. Niles and A Pease. 2001. Towards a standard upper ontology. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001)*.

V. Sornlertlamvanich, W. Pantachat, and S. Meknavin. 1994. Classifier assignment by corpus-based approach. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, pages 556–561.

M. Swadesh. 1952. Lexico-statistical dating of prehistoric ethnic contacts: With special reference to north American Indians and Eskimos. In *Proceedings of the American Philo-sophical Society*, volume 96, pages 452–463.

H. Zhang, C. Huang, and S. Yu. 2004. Distributional consistency: A general method for defining a core lexicon. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC2004)*, pages 1119–1222.

[4] http://www.afnlp.org/

[5] http://www.language-resource.net/

[6] http://www.tc37sc4.org/

Statistical phrase-based models for interactive computer-assisted translation

Jesús Tomás and **Francisco Casacuberta**
Instituto Tecnológico de Informática
Universidad Politécnica de Valencia
46071 Valencia, Spain
{jtomas,fcn}@upv.es

Abstract

Obtaining high-quality machine translations is still a long way off. A post-editing phase is required to improve the output of a machine translation system. An alternative is the so called computer-assisted translation. In this framework, a human translator interacts with the system in order to obtain high-quality translations. A statistical phrase-based approach to computer-assisted translation is described in this article. A new decoder algorithm for interactive search is also presented, that combines monotone and non-monotone search. The system has been assessed in the TransType-2 project for the translation of several printer manuals, from (to) English to (from) Spanish, German and French.

1 Introduction

Computers have become an important tool to increase the translator's productivity. In a more extended framework, a *machine translation* (MT) system can be used to obtain initial versions of the translations. Unfortunately, the state of the art in MT is far from being perfect, and a human translator must edit this output in order to achieve high-quality translations.

Another possibility is *computer-assisted translation* (CAT). In this framework, a human translator interacts with the system in order to obtain high-quality translations. This work follows the approach of *interactive CAT* initially suggested by (Foster et al., 1996) and developed in the TransType2 project (SchlumbergerSema S.A. et al., 2001; Barrachina et al., 2006). In this framework, the system suggests a possible translation of a given source sentence. The human translator can accept either the whole suggestion or accept it only up to a certain point (that is, a character prefix of this suggestion). In the latter case, he/she can type one character after the selected prefix in order to direct the system to the correct translation. The accepted prefix and the new corrected character can be used by the system to propose a new suggestion to complete the prefix. The process is repeated until the user completely accepts the suggestion proposed by the system. Figure 1 shows an example of a possible CAT system interaction.

Statistical machine translation (SMT) is an adequate framework for CAT since the MT models used can be learnt automatically from a training bilingual corpus and the search procedures developed for SMT can be adapted efficiently to this new interactive framework (Och et al., 2003). *Phrase-based* models have proved to be very adequate statistical models for MT (Tomás et al., 2005). In this work, the use of these models has been extended to interactive CAT.

The organization of the paper is as follows. The following section introduces the statistical approach to MT and section 3 introduces the statistical approach to CAT. In section 4, we review the phrase-based translation model. In section 5, we describe the decoding algorithm used in MT, and how it can be adapted to CAT. Finally, we will present some experimental results and conclusions.

2 Statistical machine translation

The goal of SMT is to translate a given source language sentence $s_1^J = s_1 ... s_J$ to a target sentence $t_1^I = t_1 ... t_I$. The methodology used (Brown et al., 1993) is based on the definition of a function $Pr(t_1^I | s_1^J)$ that returns the probability that t_1^I is a

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 835–841,
Sydney, July 2006.

source	Transferir documentos explorados a otro directorio
interaction-0	*Move documents scanned to other directory*
interaction-1	Move [s]*canned documents to other directory*
interaction-2	Move scanned documents to [a]*nother directory*
interaction-3	Move scanned documents to another [f]*older*
acceptance	Move scanned documents to another folder

Figure 1: Example of CAT system interactions to translate the Spanish source sentence into English. In interaction-0, the system suggests a translation. In interaction-1, the user accepts the first five characters "Move " and presses the key [s], then the system suggests completing the sentence with "*canned documents to other directory*". Interactions 2 and 3 are similar. In the final interaction, the user completely accepts the present suggestion.

translation of a given s_1^J. Once this function is estimated, the problem can be reduced to search a sentence $\hat{t}_1^{\hat{I}}$ that maximizes this probability for a given s_1^J.

$$\hat{t}_1^{\hat{I}} = \underset{I,t_1^I}{\operatorname{argmax}} Pr(t_1^I|s_1^J) = \underset{I,t_1^I}{\operatorname{argmax}} Pr(t_1^I)Pr(s_1^J|t_1^I) \quad (1)$$

Equation 1 summarizes the following three matters to be solved: First, an *output language model* is needed to distinguish valid sentences from invalid sentences in the target language, $Pr(t_1^I)$. Second, a *translation model*, $Pr(s_1^J|t_1^I)$. Finally, the design of an algorithm to *search* for the sentence $\hat{t}_1^I$ that maximizes this product.

3 Statistical computer-assisted translation

In a CAT scenario, the source sentence s_1^J and a given prefix of the target sentence t_1^i are given. This prefix has been validated by the user (using a previous suggestion by the system plus some corrected words). Now, we are looking for the most probable words that complete this prefix.

$$\begin{aligned} \hat{t}_{i+1}^{\hat{I}} &= \underset{I,t_{i+1}^I}{\operatorname{argmax}} Pr(t_{i+1}^I|s_1^J, t_1^i) \\ &= \underset{I,t_{i+1}^I}{\operatorname{argmax}} Pr(t_1^I)Pr(s_1^J|t_1^I) \end{aligned} \quad (2)$$

This formulation is very similar to the previous case, but in this one, the search is constrained to the set of possible suffixes t_{i+1}^I instead of the whole target sentences t_1^I. Therefore, the same techniques (translation models, decoder algorithm, etc.) which have been developed for SMT can be used in CAT.

Note that the statistical models are defined at word level. However, the CAT interface described in the first section works at character level. This is not a problem: the transformation can be performed in an easy way.

Another important issue is the computational time required by the system to produce a new suggestion. In the CAT framework, real-time is required.

4 Phrase-based models

The usual statistical translation models can be classified as single-word based alignment models. Models of this kind assume that an input word is generated by only one output word (Brown et al., 1993). This assumption does not correspond to the characteristics of natural language; in some cases, we need to know a word group in order to obtain a correct translation.

One initiative for overcoming the above-mentioned restriction of single-word models is known as the template-based approach (Och, 2002). In this approach, an entire group of adjacent words in the source sentence may be aligned with an entire group of adjacent target words. As a result, the context of words has a greater influence and the changes in word order from source to target language can be learned explicitly. A template establishes the reordering between two sequences of word classes. However, the lexical model continues to be based on word-to-word correspondence.

A simple alternative to these models has been proposed, the phrase-based (PB) approach (Tomás and Casacuberta, 2001; Marcu and Wong, 2002; Zens et al., 2002). The principal innovation of the phrase-based alignment model is that it attempts to calculate the translation probabilities of word sequences (phrases) rather than of only single words. These methods explicitly learn the probability of a

sequence of words in a source sentence ($\tilde{s}$) being translated as another sequence of words in the target sentence ($\tilde{t}$).

To define the PB model, we segment the source sentence s_1^J into K phrases ($\tilde{s}_1^K$) and the target sentence t_1^I into K phrases ($\tilde{t}_1^K$). A uniform probability distribution over all possible segmentations is assumed. If we assume a monotone alignment, that is, the target phrase in position k is produced only by the source phrase in the same position (Tomás and Casacuberta, 2001) we get:

$$Pr(s_1^J|t_1^I) \propto \sum_{K,\tilde{t}_1^K,\tilde{s}_1^K} \prod_{k=1}^{K} p(\tilde{s}_k|\tilde{t}_k) \qquad (3)$$

where the parameter $p(\tilde{s}|\tilde{t})$ estimates the probability of translating the phrase $\tilde{t}$ into the phrase $\tilde{s}$. A phrase can be comprised of a single word (but empty phrases are not allowed). Thus, the conventional word to word statistical dictionary is included.

If we permit the reordering of the target phrases, a hidden phrase level alignment variable, α_1^K, is introduced. In this case, we assume that the target phrase in position k is produced only by the source phrase in position α_k.

$$Pr(s_1^J|t_1^I) \propto \sum_{K,\tilde{t}_1^K,\tilde{s}_1^K,\alpha_1^K} \prod_{k=1}^{K} p(\alpha_k|\,\alpha_{k-1}) \cdot p(\tilde{s}_k|\tilde{t}_{\alpha_k}) \qquad (4)$$

where the distortion model $p(\alpha_k|\,\alpha_{k-1})$ (the probability of aligning the target segment k with the source segment α_k) depends only on the previous alignment α_{k-1} (first order model). For the distortion model, it is also assumed that an alignment depends only on the distance of the two phrases (Och and Ney, 2000):

$$p(\alpha_k|\alpha_{k-1}) = p_0^{|\gamma_{\alpha_k} - \gamma_{\alpha_{k-1}}|}. \qquad (5)$$

There are different approaches to the parameter estimation. The first one corresponds to a direct learning of the parameters of equations 3 or 4 from a sentence-aligned corpus using a maximum likelihood approach (Tomás and Casacuberta, 2001; Marcu and Wong, 2002). The second one is heuristic and tries to use a word-aligned corpus (Zens et al., 2002; Koehn et al., 2003). These alignments can be obtained from single-word models (Brown et al., 1993) using the available public software GIZA++ (Och and Ney, 2003). The latter approach is used in this research.

5 Decoding in interactive machine translation

The search algorithm is a crucial part of a CAT system. Its performance directly affects the quality and efficiency of translation. For CAT search we propose using the same algorithm as in MT. Thus, we first describe the search in MT.

5.1 Search for MT

The aim of the search in MT is to look for a target sentence t_1^I that maximizes the product $P(t_1^I) \cdot P(s_1^J|t_1^I)$. In practice, the search is performed to maximise a log-linear model of $Pr(t_1^I)$ and $Pr(t_1^I|s_1^J)^\lambda$ that allows a simplification of the search process and better empirical results in many translation tasks (Tomás et al., 2005). Parameter λ is introduced in order to adjust the importance of both models. In this section, we describe two search algorithms which are based on multi-stack-decoding (Berger et al., 1996) for the monotone and for the non-monotone model.

The most common statistical decoder algorithms use the concept of partial translation hypothesis to perform the search (Berger et al., 1996). In a partial hypothesis, some of the source words have been used to generate a target prefix. Each hypothesis is scored according to the translation and language model. In our implementation for the monotone model, we define a hypothesis search as the triple $(J', t_1^{I'}, g)$, where J' is the length of the source prefix we are translating (i.e. $s_1^{J'}$); the sequence of I' words, $t_1^{I'}$, is the target prefix that has been generated and g is the score of the hypothesis ($g = \Pr(t_1^{I'}) \cdot \Pr(t_1^{I'}|s_1^{J'})^\lambda$).

The translation procedure can be described as follows. The system maintains a large set of hypotheses, each of which has a corresponding translation score. This set starts with an initial empty hypothesis. Each hypothesis is stored in a different stack, according to the source words that have been considered in the hypothesis (J'). The algorithm consists of an iterative process. In each iteration, the system selects the best scored partial hypothesis to extend in each stack. The extension consists in selecting one (or more) untranslated word(s) in the source and selecting one (or more) target word(s) that are attached to the existing output prefix. The process continues several times or until there are no more hypotheses to extend. The final hypothesis with the highest score and with no untranslated source words is the out-

put of the search.

The search can be extended to allow for non-monotone translation. In this extension, several reorderings in the target sequence of phrases are scored with a corresponding probability. We define a hypothesis search as the triple $(w, t_1^{I'}, g)$, where $w = \{1..J\}$ is the coverage set that defines which positions of source words have been translated. For a better comparison of hypotheses, the store of each hypothesis in different stacks according to their value of w is proposed in (Berger et al., 1996). The number of possible stacks can be very high (2^J); thus, the stacks are created on demand. The translation procedure is similar to the previous one: In each iteration, the system selects the best scored partial hypothesis to extend in each created stack and extends it.

5.2 Search algorithms for iterative MT.

The above search algorithm can be adapted to the iterative MT introduced in the first section, i.e. given a source sentence s_1^J and a prefix of the target sentence t_1^i, the aim of the search in iterative MT is to look for a suffix of the target sentence $\hat{t}_{i+1}^{\hat{I}}$ that maximises the product $Pr(t_1^I) \cdot Pr(s_1^J | t_1^I)$ (or the log-linear model: $\Pr(t_1^{I'}) \cdot \Pr(t_1^{I'} | s_1^{J'})^{\lambda}$). A simple modification of the search algorithm is necessary. When a hypothesis is extended, if the new hypothesis is not compatible with the fixed target prefix, t_1^i, then this hypothesis is not considered. Note that this prefix is a character sequence and a hypothesis is a word sequence. Thus, the hypothesis is converted to a character sequence before the comparison.

In the CAT scenario, speed is a critical aspect. In the PB approach monotone search is more efficient than non-monotone search and obtains similar translation results for the tasks described in this article (Tomás and Casacuberta, 2004). However, the use of monotone search in the CAT scenario presents a problem: If a user introduces a prefix that cannot be obtained in a monotone way from the source, the search algorithm is not able to complete this prefix. In order to solve this problem, but without losing too much efficiency, we use the following approach: Non-monotone search is used while the target prefix is generated by the algorithm. Monotone search is used while new words are generated.

Note that searching for a prefix that we already know may seem useless. The real utility of this phase is marking the words in the target sentence that have been used in the translation of the given prefix.

A desirable feature of the iterative machine translation system is the possibility of producing a list of target suffixes, instead of only one (Civera et al., 2004). This feature can be easily obtained by keeping the N-best hypotheses in the last stack. In practice these N-best hypotheses are too similar. They differ only in one or two words at the end of the sentence. In order to solve this problem, the following procedure is performed: First, generate a hypotheses list using the N-best hypotheses of a regular search. Second, add to this list, new hypotheses formed by a single translation-word from a non-translated source word. Third, add to this list, new hypotheses formed by a single word with a high probability according to the target language model. Finally, sort the list maximising the diversity at the beginning of the suffixes and select the first N hypotheses.

6 Experimental results

6.1 Evaluation criteria

Four different measures have been used in the experiments reported in this paper. These measures are based on the comparison of the system output with a single reference.

- *Word Error Rate* (WER): Edit distance in terms of words between the target sentence provided by the system and the reference translation (Och and Ney, 2003).
- *Character Error Rate* (CER): Edit distance in terms of characters between the target sentence provided by the system and the reference translation (Civera et al., 2004).
- *Word-Stroke Ratio* (WSR): Percentage of words which, in the CAT scenario, must be changed in order to achieve the reference.
- *Key-Stroke Ratio* (KSR): Number of keystrokes that are necessary to achieve the reference translation divided by the number of running characters (Och et al., 2003) [1].

[1] In others works, an extra keystroke is added in the last iteration when the user accepts the sentence. We do not add this extra keystroke. Thus, the KSR obtained in the interaction example of Figure 1, is $3/40$.

time (ms)	WSR	KSR
10	33.9	11.2
40	30.9	9.8
100	30.0	9.3
500	27.8	8.5
13000	27.5	8.3

Table 2: Translation results obtained for several average response time in the Spanish/English "XRCE" task.

WER and CER measure the post-editing effort to achieve the reference in an MT scenario. On the other hand, WSR and KSR measure the interactive-editing effort to achieve the reference in a CAT scenario. WER and CER measures have been obtained using the first suggestion of the CAT system, when the validated prefix is void.

6.2 Task description

In order to validate the approach described in this paper a series of experiments were carried out using the XRCE corpus. They involve the translation of technical Xerox manuals from English to Spanish, French and German and from Spanish, French and German to English. In this research, we use the *raw* version of the corpus. Table 1 shows some statistics of training and test corpus.

6.3 Results

Table 2 shows the WSR and KSR obtained for several average response times, for Spanish/English translations. We can control the response time changing the number of iterations in the search algorithm. Note that real-time restrictions cause a significant degradation of the performance. However, in a real CAT scenario long iteration times can render the system useless. In order to guarantee a fast human interaction, in the remaining experiments of the paper, the mean iteration time is constrained to about 80 ms.

Table 3 shows the results using monotone search and combining monotone and non-monotone search. Using non-monotone search while the given prefix is translated improves the results significantly.

Table 4 compares the results when the system proposes only one translation (1-best) and when the system proposes five alternative translations (5-best). Results are better for 5-best. However, in this configuration the user must read five different alternatives before choosing. It is still to be shown if this extra time is compensated by the fewer key strokes needed.

	monotone		non-monotone	
	WSR	KSR	WSR	KSR
English/Spanish	36.1	11.2	28.7	8.9
Spanish/English	32.2	10.4	30.0	9.3
English/French	66.0	24.9	60.7	22.6
French/English	64.5	23.6	61.6	22.2
English/German	71.0	27.1	67.6	25.6
German/English	66.4	23.6	62.0	21.9

Table 3: Comparison of monotone and non-monotone search in "XRCE" corpora.

	1-best		5-best	
	WSR	KSR	WSR	KSR
English/Spanish	28.7	8.9	28.4	7.3
Spanish/English	30.0	9.3	29.7	7.6
English/French	60.7	22.6	59.8	18.8
French/English	61.6	22.2	60.7	17.6
English/German	67.6	25.6	67.1	20.9
German/English	62.0	21.9	61.6	16.5

Table 4: CAT results for the "XRCE" task for 1-best hypothesis and 5-best hypothesis.

Finally, in table 5 we compare the post-editing effort in an MT scenario (WER and CER) and the interactive-editing effort in a CAT scenario (WSR and KSR). These results show how the number of characters to be changed, needed to achieve the reference, is reduced by more than 50%. The reduction at word level is slight or none. Note that results from English/Spanish are much better than from English/French and English/German. This is because a large part of the English/Spanish test corpus has been obtained from the index of the technical manual, and this kind of text is easier to translate.

It is not clear how these theoretical gains translate to practical gains, when the system is used by real translators (Macklovitch, 2004).

7 Related work

Several CAT systems have been proposed in the TransType projects (SchlumbergerSema S.A. et al., 2001):

In (Foster et al., 2002) a maximum entropy version of IBM2 model is used as translation model. It is a very simple model in order to achieve rea-

		English/Spanish	English/German	English/French
Train	Sent. pairs (K)	56	49	53
	Run. words (M)	0.6/0.7	0.6/0.5	0.6/0.7
	Vocabulary (K)	26/30	25/27	25/37
Test	Sent. pairs (K)	1.1	1.0	1.0
	Run. words (K)	8/9	9/10	11/10
	Perplexity	107/60	93/169	193/135

Table 1: Statistics of the "XRCE" corpora English to/from Spanish, German and French. Trigram models were used to compute the test perplexity.

	WER	CER	WSR	KSR
English/Spanish	31.1	21.7	28.7	8.9
Spanish/English	34.9	24.7	30.0	9.3
English/French	61.6	49.2	60.7	22.6
French/English	58.0	48.2	61.6	22.2
English/German	68.0	56.9	67.6	25.6
German/English	59.5	50.6	62.0	21.9

Table 5: Comparison of post-editing effort in MT scenario (WER/CER) and the interactive-editing effort in CAT scenario (WSR/KSR). Non-monotone search and 1-best hypothesis is used.

sonable interaction times. In this approach, the length of the proposed extension is variable in function of the expected benefit of the human translator.

In (Och et al., 2003) the Alignment-Templates translation model is used. To achieve fast response time, it proposes to use a word hypothesis graph as an efficient search space representation. This word graph is precalculated before the user interactions.

In (Civera et al., 2004) finite state transducers are presented as a candidate technology in the CAT paradigm. These transducers are inferred using the GIATI technique (Casacuberta and Vidal, 2004). To solve the real-time constraints a word hypothesis graph is used. The N-best configuration is proposed.

In (Bender et al., 2005) the use of a word hypothesis graph is compared with the direct use of the translation model. The combination of two strategies is also proposed.

8 Conclusions

Phrase-based models have been used for interactive CAT in this work. We show how SMT can be used, with slight adaptations, in a CAT system. A prototype has been developed in the framework of the TransType2 project (SchlumbergerSema S.A. et al., 2001).

The experimental results have proved that the systems based on such models achieve a good performance, possibly, allowing a saving of human effort with respect to the classical post-editing operation. However, this fact must be checked by actual users.

The main critical aspect of the interactive CAT system is the response time. To deal with this issue, other proposals are based on the construction of a word graphs. This method can reduce the generation capability of the fully fledged translation model (Och et al., 2003; Bender et al., 2005). The main contribution of the present proposal is a new decoding algorithm, that combines monotone and non-monotone search. It runs fast enough and the construction of word graph is not necessary.

Acknowledgments

This work has been partially supported by the Spanish project TIC2003-08681-C02-02 the IST Programme of the European Union under grant IST-2001-32091. The authors wish to thank the anonymous reviewers for their criticisms and suggestions.

References

S. Barrachina, O. Bender, F. Casacuberta, J. Civera, E. Cubel, S. Khadivi, A. Lagarda, H. Net, J. Tomás, E.Vidal, and J.M. Vilar. 2006. Statistical approaches to computer-assisted translation. *In preparation.*

O. Bender, S. Hasan, D. Vilar, R. Zens, and H. Ney. 2005. Comparison of generation strategies for interactive machine translation. In *Proceedings of EAMT 2005 (10th Annual Conference of the European Association for Machine Translation)*, pages 30–40, Budapest, Hungary, May.

A. L. Berger, P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, J. R. Gillett, A. S. Kehler, and R. L. Mercer. 1996. Language translation apparatus and method of using context-based translation models. United States Patent, No. 5510981, April.

P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

F. Casacuberta and E. Vidal. 2004. Machine translation with inferred stochastic finite-state transducers. *Computational Linguistics*, 30(2):205–225.

J. Civera, J. M. Vilar, E. Cubel, A. L. Lagarda, S. Barrachina, E. Vidal, F. Casacuberta, D. Picó, and J. González. 2004. From machine translation to computer assisted translation using finite-state models. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP04)*, Barcelona, Spain.

G. Foster, P. Isabelle, and P. Plamondon. 1996. Word completion: A first step toward target-text mediated IMT. In *COLING '96: The 16th Int. Conf. on Computational Linguistics*, pages 394–399, Copenhagen, Denmark, August.

G. Foster, P. Langlais, and G. Lapalme. 2002. User-friendly text prediction for translators. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP02)*, pages 148–155, Philadelphia, USA, July.

P. Koehn, F. J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Human Language Technology and North American Association for Computational Linguistics Conference (HLT/NAACL)*, pages 48–54, Edmonton, Canada, June.

E Macklovitch. 2004. The contribution of end-users to the transtype2 project. volume 3265 of *Lecture Notes in Computer Science*, pages 197–207. Springer-Verlag.

D. Marcu and W. Wong. 2002. A phrase-based joint probability model for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Philadelphia, USA, July.

F. J. Och and H. Ney. 2000. Improved statistical alignment models. In *Proc. of the 38th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 440–447, Hong Kong, October.

F. J. Och and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, March.

F. J. Och, R. Zens, and H. Ney. 2003. Efficient search for interactive statistical machine translation. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 387.–393, Budapest, Hungary, April.

F. J. Och. 2002. *Statistical Machine Translation: From Single-Word Models to Alignment Templates*. Ph.D. thesis, Computer Science Department, RWTH Aachen, Germany, October.

SchlumbergerSema S.A., Instituto Tecnológico de Informática, Rheinisch Westfälische Technische Hochschule Aachen Lehrstul für Informatik VI, Recherche Appliquée en Linguistique Informatique Laboratory University of Montreal, Celer Soluciones, Société Gamma, and Xerox Research Centre Europe. 2001. TT2. TransType2 - computer assisted translation. Project technical annex.

J. Tomás and F. Casacuberta. 2001. Monotone statistical translation using word groups. In *Procs. of the Machine Translation Summit VIII*, pages 357–361, Santiago de Compostela, Spain.

J. Tomás and F. Casacuberta. 2004. Statistical machine translation decoding using target word reordering. In *Structural, Syntactic, and Statistical Pattern Recongnition*, volume 3138 of *Lecture Notes in Computer Science*, pages 734–743. Springer-Verlag.

J. Tomás, J. Lloret, and F. Casacuberta. 2005. Phrase-based alignment models for statistical machine translation. In *Pattern Recognition and Image Analysis*, volume 3523 of *Lecture Notes in Computer Science*, pages 605–613. Springer-Verlag.

R. Zens, F. J. Och, and H. Ney. 2002. Phrase-based statistical machine translation. *Advances in Artificial Inteligence*, LNAI 2479(25):18–32, September.

Using Word Support Model to Improve Chinese Input System

Jia-Lin Tsai
Tung Nan Institute of Technology, Department of Information Management
Taipei 222, Taiwan
tsaijl@mail.tnit.edu.tw

Abstract

This paper presents a word support model (WSM). The WSM can effectively perform homophone selection and syllable-word segmentation to improve Chinese input systems. The experimental results show that: (1) the WSM is able to achieve tonal (syllables input with four tones) and toneless (syllables input without four tones) syllable-to-word (STW) accuracies of 99% and 92%, respectively, among the converted words; and (2) while applying the WSM as an adaptation processing, together with the Microsoft Input Method Editor 2003 (MSIME) and an optimized bigram model, the average tonal and toneless STW improvements are 37% and 35%, respectively.

1 Introduction

According to (Becker, 1985; Huang, 1985; Gu et al., 1991; Chung, 1993; Kuo, 1995; Fu et al., 1996; Lee et al., 1997; Hsu et al., 1999; Chen et al., 2000; Tsai and Hsu, 2002; Gao et al., 2002; Lee, 2003; Tsai, 2005), the approaches of Chinese input methods (i.e. Chinese input systems) can be classified into two types: (1) ***keyboard based approach***: including phonetic and pinyin based (Chang et al., 1991; Hsu et al., 1993; Hsu, 1994; Hsu et al., 1999; Kuo, 1995; Lua and Gan, 1992), arbitrary codes based (Fan et al., 1988) and structure scheme based (Huang, 1985); and (2) ***non-keyboard based approach***: including optical character recognition (OCR) (Chung, 1993), online handwriting (Lee et al., 1997) and speech recognition (Fu et al., 1996; Chen et al., 2000). Currently, the most popular Chinese input system is phonetic and pinyin based approach, because Chinese people are taught to write phonetic and pinyin syllables of each Chinese character in primary school.

In Chinese, each Chinese word can be a mono-syllabic word, such as "鼠(mouse)", a bi-syllabic word, such as "袋鼠(kangaroo)", or a multi-syllabic word, such as "米老鼠(Mickey mouse)." The corresponding phonetic and pinyin syllables of each Chinese word is called syllable-words, such as "dai4 shu3" is the pinyin syllable-word of "袋鼠(kangaroo)." According to our computation, the {minimum, maximum, average} words per each distinct mono-syllable-word and poly-syllable-word (including bi-syllable-word and multi-syllable-word) in the CKIP dictionary (Chinese Knowledge Information Processing Group, 1995) are {1, 28, 2.8} and {1, 7, 1.1}, respectively. The CKIP dictionary is one of most commonly-used Chinese dictionaries in the research field of Chinese natural language processing (NLP). Since the size of problem space for syllable-to-word (STW) conversion is much less than that of syllable-to-character (STC) conversion, the most pinyin-based Chinese input systems (Hsu, 1994; Hsu et al., 1999; Tsai and Hsu, 2002; Gao et al., 2002; Microsoft Research Center in Beijing; Tsai, 2005) are addressed on STW conversion. On the other hand, STW conversion is the main task of Chinese Language Processing in typical Chinese speech recognition systems (Fu et al., 1996; Lee et al., 1993; Chien et al., 1993; Su et al., 1992).

As per (Chung, 1993; Fong and Chung, 1994; Tsai and Hsu, 2002; Gao et al., 2002; Lee, 2003; Tsai, 2005), ***homophone selection*** and ***syllable-word segmentation*** are two critical problems in developing a Chinese input system. Incorrect homophone selection and syllable-word seg-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 842–849,
Sydney, July 2006.

mentation will directly influence the STW conversion accuracy. Conventionally, there are two approaches to resolve the two critical problems: (1) ***linguistic approach***: based on syntax parsing, semantic template matching and contextual information (Hsu, 1994; Fu et al., 1996; Hsu et al., 1999; Kuo, 1995; Tsai and Hsu, 2002); and (2) ***statistical approach***: based on the n-gram models where n is usually 2, i.e. bigram model (Lin and Tsai, 1987; Gu et al., 1991; Fu et al., 1996; Ho et al., 1997; Sproat, 1990; Gao et al., 2002; Lee 2003). From the studies (Hsu 1994; Tsai and Hsu, 2002; Gao et al., 2002; Kee, 2003; Tsai, 2005), the linguistic approach requires considerable effort in designing effective syntax rules, semantic templates or contextual information, thus, it is more user-friendly than the statistical approach on understanding why such a system makes a mistake. The statistical language model (SLM) used in the statistical approach requires less effort and has been widely adopted in commercial Chinese input systems.

In our previous work (Tsai, 2005), a word-pair (WP) identifier was proposed and shown a simple and effective way to improve Chinese input systems by providing tonal and toneless STW accuracies of 98.5% and 90.7% on the identified poly-syllabic words, respectively. In (Tsai, 2005), we have shown that the WP identifier can be used to reduce the over weighting and corpus sparseness problems of bigram models and achieve better STW accuracy to improve Chinese input systems. As per our computation, poly-syllabic words cover about 70% characters of Chinese sentences. Since the identified character ratio of the WP identifier (Tsai, 2005) is about 55%, there are still about 15% improving room left.

The objective of this study is to illustrate a word support model (WSM) that is able to improve our WP-identifier by achieving better identified character ratio and STW accuracy on the identified poly-syllabic words with the same word-pair database. We conduct STW experiments to show the tonal and toneless STW accuracies of a commercial input product (Microsoft Input Method Editor 2003, MSIME), and an optimized bigram model, BiGram (Tsai, 2005), can both be improved by our WSM and achieve better STW improvements than that of these systems with the WP identifier.

The remainder of this paper is arranged as follows. In Section 2, we present an auto word-pair (AUTO-WP) generation used to generate the WP database. Then, we develop a word support model with the WP database to perform STW conversion on identifying words from the Chinese syllables. In Section 3, we report and analyze our STW experimental results. Finally, in Section 4, we give our conclusions and suggest some future research directions.

2 Development of Word Support Model

The system dictionary of our WSM is comprised of 82,531 Chinese words taken from the CKIP dictionary and 15,946 unknown words auto-found in the UDN2001 corpus by a Chinese Word Auto-Confirmation (CWAC) system (Tsai et al., 2003). The UDN2001 corpus is a collection of 4,539624 Chinese sentences extracted from whole 2001 UDN (United Daily News, 2001) Website in Taiwan (Tsai and Hsu, 2002). The system dictionary provides the knowledge of words and their corresponding pinyin syllable-words. The pinyin syllable-words were translated by phoneme-to-pinyin mappings, such as "ㄐㄩ"-to-"ju2."

2.1 Auto-Generation of WP Database

Following (Tsai, 2005), the three steps of auto-generating word-pairs (AUTO-WP) for a given Chinese sentence are as below: (the details of AUTO-WP can be found in (Tsai, 2005))

Step 1. ***Get forward and backward word segmentations***: Generate two types of word segmentations for a given Chinese sentence by forward maximum matching (FMM) and backward maximum matching (BMM) techniques (Chen et al., 1986; Tsai et al., 2004) with the system dictionary.

Step 2. ***Get initial WP set***: Extract all the combinations of word-pairs from the FMM and the BMM segmentations of Step 1 to be the initial WP set.

Step 3. ***Get finial WP set***: Select out the word-pairs comprised of two poly-syllabic words from the initial WP set into the finial WP set. For the final WP set, if the word-pair is not found in the WP data-

base, insert it into the WP database and set its frequency to 1; otherwise, increase its frequency by 1.

2.2 Word Support Model

The four steps of our WSM applied to identify words for a given Chinese syllables is as follows:

Step 1. Input tonal or toneless syllables.

Step 2. Generate all possible word-pairs comprised of two poly-syllabic words for the input syllables to be the WP set of Step 3.

Step 3. Select out the word-pairs that match a word-pair in the WP database to be the WP set. Then, compute the **word support degree** (**WS degree**) for each distinct word of the WP set. The WS degree is defined to be the total number of the word found in the WP set. Finally, arrange the words and their corresponding WS degrees into the WSM set. If the number of words with the same syllable-word and WS degree is greater than one, one of them is randomly selected into the WSM set.

Step 4. Replace words of the WSM set in descending order of WS degree with the input syllables into a WSM-sentence. If no words can be identified in the input syllables, a NULL WSM-sentence is produced.

Table 1 is a step by step example to show the four steps of applying our WSM on the Chinese syllables "sui1 ran2 fu3 shi2 jin4 shi4 sui4 yue4 xi1 xu1(雖然俯拾盡是歲月唏噓)." For this input syllables, we have a WSM-sentence "雖然俯拾盡是歲月唏噓." For the same syllables, outputs of the MSIME, the BiGram and the WP identifier are "雖然腐蝕進士歲月唏噓," "雖然俯拾盡是歲月唏噓" and "雖然 fu3 shi2 近視 sui4 yue4 xi1 xu1."

3 STW Experiments

To evaluate the STW performance of our WSM, we define the STW accuracy, identified character ratio (ICR) and STW improvement, by the following equations:

STW accuracy = # of correct characters / # of total characters. (1)

Identified character ratio (ICR) = # of characters of identified WP / # of total characters in testing sentences. (2)

STW improvement (I) (i.e. STW error reduction rate) = (accuracy of STW system with WP – accuracy of STW system)) / (1 – accuracy of STW system). (3)

Step #	Results
Step.1	sui1 ran2 fu3 shi2 jin4 shi4 sui4 yue4 xi1 xu1 (雖 然 俯 拾 盡 是 歲 月 唏 噓)
Step.2	WP set (word-pair / word-pair frequency) = {雖然-近視/6 (key WP for WP identifier), 俯拾-盡是/4, 雖然-歲月/4, 雖然-盡是/3, 俯拾-唏噓/2, 雖然-俯拾/2, 俯拾-歲月/2, 盡是-唏噓/2, 盡是-歲月/2, 雖然-唏噓/2, 歲月-唏噓/2}
Step.3	WSM set (word / WS degree) = {雖然/5, 俯拾/4, 盡是/4, 歲月/4, 唏噓/4, 近視/1} Replaced word set = 雖然(sui1 ran2), 俯拾(fu3 shi2), 盡是(jin4 shi4), 歲月(sui4 yue4), 唏噓(xi1 xu1)
Step.4	WSM-sentence: 雖然俯拾盡是歲月唏噓

Table 1. An illustration of a WSM-sentence for the Chinese syllables "sui1 ran2 fu3 shi2 jin4 shi4 sui4 yue4 xi1 xu1(雖然俯拾盡是歲月唏噓)."

3.1 Background

To conduct the STW experiments, firstly, use the inverse translator of phoneme-to-character (PTC) provided in GOING system to convert testing sentences into their corresponding syllables. All the error PTC translations of GOING PTC were corrected by post human-editing. Then, apply our WSM to convert the testing input syllables back to their WSM-sentences. Finally, calculate its STW accuracy and ICR by Equations (1) and (2). Note that all test sentences are composed of a string of Chinese characters in this study.

The training/testing corpus, closed/open test sets and system/user WP database used in the following STW experiments are described as below:

(1) Training corpus: We used the UDN2001 corpus as our training corpus, which is a collection of 4,539624 Chinese sentences extracted from whole 2001 UDN (United Daily News, 2001) Website in Taiwan (Tsai and Hsu, 2002).

(2) Testing corpus: The Academia Sinica Balanced (AS) corpus (Chinese Knowledge Information Processing Group, 1996) was selected as our testing corpus. The AS corpus is one of most famous traditional Chinese corpus used in the Chinese NLP research field (Thomas, 2005).

(3) Closed test set: 10,000 sentences were randomly selected from the UDN2001 corpus as the closed test set. The {minimum, maximum, and mean} of characters per sentence for the closed test set are {4, 37, and 12}.

(4) Open test set: 10,000 sentences were randomly selected from the AS corpus as the open test set. At this point, we checked that the selected open test sentences were not in the closed test set as well. The {minimum, maximum, and mean} of characters per sentence for the open test set are {4, 40, and 11}.

(5) System WP database: By applying the AUTO-WP on the UDN2001 corpus, we created 25,439,679 word-pairs to be the system WP database.

(6) User WP database: By applying our AUTO-WP on the AS corpus, we created 1,765,728 word-pairs to be the user WP database.

We conducted the STW experiment in a progressive manner. The results and analysis of the experiments are described in Subsections 3.2 and 3.3.

3.2 STW Experiment Results of the WSM

The purpose of this experiment is to demonstrate the tonal and toneless STW accuracies among the identified words by using the WSM with the system WP database. The comparative system is the WP identifier (Tsai, 2005). Table 2 is the experimental results. The WP database and system dictionary of the WP identifier is same with that of the WSM.

From Table 2, it shows the average tonal and toneless STW accuracies and ICRs of the WSM are all greater than that of the WP identifier. These results indicate that the WSM is a better way than the WP identifier to identify polysyllabic words for the Chinese syllables.

	Closed	Open	Average (ICR)
Tonal (WP)	99.1%	97.7%	98.5% (57.8%)
Tonal (WSM)	99.3%	97.9%	98.7% (71.3%)
Toneless (WP)	94.0%	87.5%	91.3% (54.6%)
Toneless (WSM)	94.4%	88.1%	91.6% (71.0%)

Table 2. The comparative results of tonal and toneless STW experiments for the WP identifier and the WSM.

3.3 STW Experiment Results of Chinese Input Systems with the WSM

We selected Microsoft Input Method Editor 2003 for Traditional Chinese (MSIME) as our experimental commercial Chinese input system. In addition, following (Tsai, 2005), an optimized bigram model called BiGram was developed. The BiGram STW system is a bigram-based model developing by SRILM (Stolcke, 2002) with Good-Turing back-off smoothing (Manning and Schuetze, 1999), as well as forward and backward longest syllable-word first strategies (Chen et al., 1986; Tsai et al., 2004). The system dictionary of the BiGram is same with that of the WP identifier and the WSM.

Table 3a compares the results of the MSIME, the MSIME with the WP identifier and the MSIME with the WSM on the closed and open test sentences. Table 3b compares the results of the BiGram, the BiGram with the WP identifier and the BiGram with the WSM on the closed and open test sentences. In this experiment, the STW output of the MSIME with the WP identifier and the WSM, or the BiGram with the WP identifier and the WSM, was collected by directly replacing the identified words of the WP identifier and the WSM from the corresponding STW output of the MSIME and the BiGram.

	Ms	Ms+WP (I)[a]	Ms+WSM (I)[b]
Tonal	94.5%	95.5% (18.9%)	95.9% (25.6%)
Toneless	85.9%	87.4% (10.1%)	88.3% (16.6%)

[a] STW accuracies and improvements of the words identified by the MSIME (Ms) with the WP identifier

[b] STW accuracies and improvements of the words identified by the MSIME (Ms) with the WSM

Table 3a. The results of tonal and toneless STW experiments for the MSIME, the MSIME with the WP identifier and with the WSM.

	Bi	Bi+WP (I)[a]	Bi+WSM (I)[b]
Tonal	96.0%	96.4% (8.6%)	96.7% (17.1%)
Toneless	83.9%	85.8% (11.9%)	87.5% (22.0%)

[a] STW accuracies and improvements of the words identified by the BiGram (Bi) with the WP identifier

[b] STW accuracies and improvements of the words identified by the BiGram (Bi) with the WSM

Table 3b. The results of tonal and toneless STW experiments for the BiGram, the BiGram with the WP identifier and with the WSM.

From Table 3a, the tonal and toneless STW improvements of the MSIME by using the WP identifier and the WSM are (18.9%, 10.1%) and (25.6%, 16.6%), respectively. From Table 3b, the tonal and toneless STW improvements of the BiGram by using the WP identifier and the WSM are (8.6%, 11.9%) and (17.1%, 22.0%), respectively. (Note that, as per (Tsai, 2005), the differences between the tonal and toneless STW accuracies of the BiGram and the TriGram are less than 0.3%).

Table 3c is the results of the MSIME and the BiGram by using the WSM as an adaptation processing with both system and user WP database. From Table 3c, we get the average tonal and toneless STW improvements of the MSIME and the BiGram by using the WSM as an adaptation processing are 37.2% and 34.6%, respectively.

	Ms+WSM (ICR, I)[a]	Bi+WSM (ICR, I)[b]
Tonal	96.8% (71.4%, 41.7%)	97.3% (71.4%, 32.6%)
Toneless	90.6% (74.6%, 33.2%)	97.3% (74.9%, 36.0%)

[a] STW accuracies, ICRs and improvements of the words identified by the MSIME (Ms) with the WSM

[b] STW accuracies, ICRs and improvements of the words identified by the BiGram (Bi) with the WSM

Table 3c. The results of tonal and toneless STW experiments for the MSIME and the BiGram using the WSM as an adaptation processing.

To sum up the above experiment results, we conclude that the WSM can achieve a better STW accuracy than that of the MSIME, the BiGram and the WP identifier on the identified-words portion. (Appendix A presents two cases of STW results that were obtained from this study).

3.4 Error Analysis

We examine the Top 300 STW conversions in the *tonal* and *toneless* from the open testing results of the BiGram with the WP identifier and the WSM, respectively. As per our analysis, the STW errors are caused by three problems, they are:

(1) ***Unknown word (UW) problem***: For Chinese NLP systems, unknown word extraction is one of the most difficult problems and a critical issue. When an STW error is caused only by the lack of words in the system dictionary, we call it unknown word problem.

(2) ***Inadequate Syllable-Word Segmentation (ISWS) problem***: When an error is caused by ambiguous syllable-word segmentation (including overlapping and combination ambiguities), we call it inadequate syllable-word segmentation problem.

(3) ***Homophone selection problem***: The remaining STW conversion error is homophone selection problem.

Problem	Coverage	
	Tonal WP, WSM	Toneless WP, WSM
UW	3%, 4%	3%, 4%
ISWS	32%, 32%	58%, 56%
HS	65%, 64%	39%, 40%
# of error characters	170, 153	506, 454
# of error characters of mono-syllabic words	100, 94	159, 210
# of error characters of poly-syllabic words	70, 59	347, 244

Table 4. The analysis results of the STW errors from the Top 300 tonal and toneless STW conversions of the BiGram with the WP identifier and the WSM.

Table 4 is the analysis results of the three STW error types. From Table 4, we have three observations:

(1) ***The coverage of unknown word problem for tonal and toneless STW conversions is similar***. In most Chinese input systems, unknown word extraction is not specifically a STW problem, therefore, it is usually taken care of through online and offline manual editing processing (Hsu et al, 1999). The results of Table 4 show that the most STW errors should be caused by ISWS and HS

problems, not UW problem. This observation is similarly with that of our previous work (Tsai, 2005).

(2) ***The major problem of error conversions in tonal and toneless STW systems is different***. This observation is similarly with that of (Tsai, 2005). From Table 4, the major improving targets of tonal STW performance are the HS errors because more than 50% tonal STW errors caused by HS problem. On the other hand, since the ISWS errors cover more than 50% toneless STW errors, the major targets of improving toneless STW performance are the ISWS errors.

(3) ***The total number of error characters of the BiGram with the WSM in tonal and toneless STW conversions are both less than that of the BiGram with the WP identifier***. This observation should answer the question "Why the STW performance of Chinese input systems (MSIME and BiGram) with the WSM is better than that of these systems with the WP-identifier?"

To sum up the above three observations and all the STW experimental results, we conclude that the WSM is able to achieve better STW improvements than that of the WP identifier is because: (1) the identified character ratio of the WSM is 15% greater than that of the WP identifier with the same WP database and dictionary, and meantime (2) the WSM not only can maintain the ratio of the three STW error types but also can reduce the total number of error characters of converted words than that of the WP identifier.

4 Conclusions and Future Directions

In this paper, we present a word support model (WSM) to improve the WP identifier (Tsai, 2005) and support the Chinese Language Processing on the STW conversion problem. All of the WP data can be generated fully automatically by applying the AUTO-WP on the given corpus. We are encouraged by the fact that the WSM with WP knowledge is able to achieve state-of-the-art tonal and toneless STW accuracies of 99% and 92%, respectively, for the identified poly-syllabic words. The WSM can be easily integrated into existing Chinese input systems by identifying words as a post processing. Our experimental results show that, by applying the WSM as an adaptation processing together with the MSIME (a trigram-like model) and the BiGram (an optimized bigram model), the average tonal and toneless STW improvements of the two Chinese input systems are 37% and 35%, respectively.

Currently, our WSM with the mixed WP database comprised of UDN2001 and AS WP database is able to achieve more than 98% identified character ratios of poly-syllabic words in tonal and toneless STW conversions among the UDN2001 and the AS corpus. Although there is room for improvement, we believe it would not produce a noticeable effect as far as the STW accuracy of poly-syllabic words is concerned.

We will continue to improve our WSM to cover more characters of the UDN2001 and the AS corpus by those word-pairs comprised of at least one mono-syllabic word, such as "我們(we)-是(are)". In other directions, we will extend it to other Chinese NLP research topics, especially word segmentation, main verb identification and Subject-Verb-Object (SVO) auto-construction.

References

Becker, J.D. 1985. Typing Chinese, Japanese, and Korean, *IEEE Computer* 18(1):27-34.

Chang, J.S., S.D. Chern and C.D. Chen. 1991. Conversion of Phonemic-Input to Chinese Text Through Constraint Satisfaction, *Proceedings of ICCPOL'91*, 30-36.

Chen, B., H.M. Wang and L.S. Lee. 2000. Retrieval of broadcast news speech in Mandarin Chinese collected in Taiwan using syllable-level statistical characteristics, *Proceedings of the 2000 International Conference on Acoustics Speech and Signal Processing*.

Chen, C.G., Chen, K.J. and Lee, L.S. 1986. A model for Lexical Analysis and Parsing of Chinese Sentences, *Proceedings of 1986 International Conference on Chinese Computing*, 33-40.

Chien, L.F., Chen, K.J. and Lee, L.S. 1993. A Best-First Language Processing Model Integrating the Unification Grammar and Markov Language Model for Speech Recognition Applications, *IEEE Transactions on Speech and Audio Processing*, 1(2):221-240.

Chung, K.H. 1993. *Conversion of Chinese Phonetic Symbols to Character*s, M. Phil. thesis, Department of Computer Science, Hong Kong

University of Science and Technology.

Chinese Knowledge Information Processing Group. 1995. *Technical Report no. 95-02, the content and illustration of Sinica corpus of Academia Sinica*. Institute of Information Science, Academia Sinica.

Chinese Knowledge Information Processing Group. 1996. *A study of Chinese Word Boundaries and Segmentation Standard for Information processing* (in Chinese). Technical Report, Taiwan, Taipei, Academia Sinica.

Fong, L.A. and K.H. Chung. 1994. Word Segmentation for Chinese Phonetic Symbols, *Proceedings of International Computer Symposium*, 911-916.

Fu, S.W.K, C.H. Lee and Orville L.C. 1996. A Survey on Chinese Speech Recognition, *Communications of COLIPS*, 6(1):1-17.

Gao, J., Goodman, J., Li, M. and Lee K.F. 2002. Toward a Unified Approach to Statistical Language Modeling for Chinese, *ACM Transactions on Asian Language Information Processing*, 1(1):3-33.

Gu, H.Y., C.Y. Tseng and L.S. Lee. 1991. Markov modeling of mandarin Chinese for decoding the phonetic sequence into Chinese characters, *Computer Speech and Language* 5(4):363-377.

Ho, T.H., K.C. Yang, J.S. Lin and L.S. Lee. 1997. Integrating long-distance language modeling to phonetic-to-text conversion, *Proceedings of ROCLING X International Conference on Computational Linguistics*, 287-299.

Hsu, W.L. and K.J. Chen. 1993. The Semantic Analysis in GOING - An Intelligent Chinese Input System, *Proceedings of the Second Joint Conference of Computational Linguistics*, Shiamen, 1993, 338-343.

Hsu, W.L. 1994. Chinese parsing in a phoneme-to-character conversion system based on semantic pattern matching, *Computer Processing of Chinese and Oriental Languages* 8(2):227-236.

Hsu, W.L. and Chen, Y.S. 1999. On Phoneme-to-Character Conversion Systems in Chinese Processing, *Journal of Chinese Institute of Engineers*, 5:573-579.

Huang, J.K. 1985. The Input and Output of Chinese and Japanese Characters, *IEEE Computer* 18(1):18-24.

Kuo, J.J. 1995. Phonetic-input-to-character conversion system for Chinese using syntactic connection table and semantic distance, *Computer Processing and Oriental Languages*, 10(2):195-210.

Lee, L.S., Tseng, C.Y., Gu, H..Y., Liu F.H., Chang, C.H., Lin, Y.H., Lee, Y., Tu, S.L., Hsieh, S.H., and Chen C.H. 1993. Golden Mandarin (I) - A Real-Time Mandarin Speech Dictation Machine for Chinese Language with Very Large Vocabulary, *IEEE Transaction on Speech and Audio Proces*sing, 1(2).

Lee, C.W., Z. Chen and R.H. Cheng. 1997. A perturbation technique for handling handwriting variations faced in stroke-based Chinese character classification, *Computer Processing of Oriental Language*s, 10(3):259-280.

Lee, Y.S. 2003. Task adaptation in Stochastic Language Model for Chinese Homophone Disambiguation, *ACM Transactions on Asian Language Information Processing*, 2(1):49-62.

Lin, M.Y. and W.H. Tasi. 1987. Removing the ambiguity of phonetic Chinese input by the relaxation technique, *Computer Processing and Oriental Languages*, 3(1):1-24.

Lua, K.T. and K.W. Gan. 1992. A Touch-Typing Pinyin Input System, *Computer Processing of Chinese and Oriental Languages*, 6:85-94.

Manning, C. D. and Schuetze, H. 1999. *Fundations of Statistical Natural Language Processing*, MIT Press: 191-220.

Microsoft Research Center in Beijing, "http://research.microsoft.com/aboutmsr/labs/beijing/"

Qiao, J., Y. Qiao and S. Qiao. 1984. Six-Digit Coding Method, *Commun. ACM* 33(5):248-267.

Sproat, R. 1990. An Application of Statistical Optimization with Dynamic Programming to Phonemic-Input-to-Character Conversion for Chinese, *Proceedings of ROCLING III*, 379-390.

Stolcke A. 2002. SRILM - An Extensible Language Modeling Toolkit, *Proc. Intl. Conf. Spoken Language Processing, Denver*.

Su, K.Y., Chiang, T.H. and Lin, Y.C. 1992. A Unified Framework to Incorporate Speech and Language Information in Spoken Language Processing, *ICASSP-92*, 185-188.

Thomas E. 2005. The Second International Chinese Word Segmentation Bakeoff, In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, Oct. Jeju, Koera, 123-133.

Tsai, J.L. and W.L. Hsu. 2002. Applying an NVEF Word-Pair Identifier to the Chinese Syllable-to-Word Conversion Problem, *Proceedings of 19th COLING 2002*, 1016-1022.

Tsai, J,L, Sung, C.L. and Hsu, W.L. 2003. Chinese Word Auto-Confirmation Agent, *Proceedings of ROCLING XV*, 175-192.

Tsai, J.L., Hsieh, G. and Hsu, W.L. 2004. Auto-Generation of NVEF knowledge in Chinese, *Computational Linguistics and Chinese Language Processing*, 9(1):41-64.

Tsai, J.L. 2005. Using Word-Pair Identifier to Improve Chinese Input System, *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing, IJCNLP2005*, 9-16.

United Daily News. 2001. On-Line United Daily News, http://udnnews.com/NEWS/

Appendix A. Two cases of the STW results used in this study.

Case I.

(a) Tonal STW results for the Chinese tonal syllables "guan1 yu2 liang4 xing2 suo3 sheng1 zhi1 shi4 shi2" of the Chinese sentence "關於量刑所生之事實"

Methods	STW results
WP set	關於-知識/4 (key WP), 關於-量刑/3, 量刑-事實/1, 關於-事實/1
WSM Set	關於(guan1 yu2)/3, 量刑(liang4 xing2)/2, 事實(shi4 shi2)/2, 知識(zhi1 shi4)/1
WP-sentence	關於 liang4 xing2 suo3 sheng1 知識 shi2
WSM-sentence	關於量刑 suo3 sheng1 zhi1 事實
MSIME	關於量行所生之事實
MSIME+WP	**關於**量行所生**知識**實
MSIME+WSM	**關於量刑**所生之**事實**
BiGram	關於量刑所生之事時
BiGram+WP	**關於**量刑所生**知識**時
BiGram+WSM	**關於量刑**所生之**事實**

(b) Toneless STW results for the Chinese toneless syllables "guan yu liang xing suo sheng zhi shi shi" of the Chinese sentence "關於量刑所生之事實"

Methods	STW results
WP set	關於/實施/4 (key WP), 關於/知識/4, 關於/量刑/3, 兩性/知識/2, 兩性/實施/2, 關於/失事/2, 量刑/事實/1, 關於/兩性/1, 關與/實施/1, 生殖/實施/1, 關於/事實/1, 關於/史實/1
WSM Set	關於(guan yu)/7, 實施(shi shi)/4, 兩性(liang xing)/3, 量刑(liang xing)/2, 知識(zhi shi)/2, 事實(shi shi)/2, 失事(shi shi)/1, 關與(guan yu)/1, 生殖(shengzhi)/1
WP-sentence	關於 liang xing suo sheng zhi 實施
WSM-sentence	關於兩性 suo 生殖實施
MSIME	關於兩性所生之事實
MSIME+WP	**關於**兩性所生之**實施**
MSIME+WSM	**關於兩性**所**生殖實施**
BiGram	貫譽良興所升值施事
BiGram+WP	**關於**良興所升值**實施**
BiGram+WSM	**關於兩性**所**生殖實施**

Case II.

(a) Tonal STW results for the Chinese tonal syllables "you2 yu2 xian3 he4 de5 jia1 shi4" of the Chinese sentence "由於顯赫的家世"

Methods	STW results
WP set	由於/家事/6 (key WP), 顯赫/家世/2, 由於/家世/2 由於/家飾/1, 由於/顯赫/1
WSM set	由於(you2 yu2)/4, 顯赫(xian 3he4)/2, 家世(jia1 shi4)/2, 家事(jia1 shi4)/1
WP-sentence	由於 xian2 he4 de5 家事
WSM-sentence	由於顯赫 de 家世
MSIME	由於顯赫的家事
MSIME+WP	**由於**顯赫的**家事**
MSIME+SWM	**由於顯赫**的**家世**
BiGram	由於顯赫的家事
BiGram+WP	**由於**顯赫的**家事**
BiGram+SWM	**由於顯赫**的**家世**

(b) Toneless STW results for the Chinese toneless syllables "you yu xian he de jia shi" of the Chinese sentence "由於顯赫的家世"

Methods	STW results
WP set	由於-駕駛/14 (key WP), 由於-假釋/6, 由於-家事/6 顯赫/家世/2, 由於/家世/2 由於/家飾/1, 由於/顯赫/1
WSM set	由於(you yu)/6, 顯赫(xian he)/2, 家世(jia shi)/2, 駕駛(jia shi)/1
WP-sentence	由於 xian he de 駕駛
WSM-sentence	由於顯赫 de 家世
MSIME	由於顯赫的架勢
MSIME+WP	**由於**顯赫的**駕駛**
MSIME+SWM	**由於顯赫**的**家世**
BiGram	由於現喝的假實
BiGram+WP	**由於**現喝的**駕駛**
BiGram+SWM	**由於顯赫**的**家世**

Trimming CFG Parse Trees for Sentence Compression Using Machine Learning Approaches

Yuya Unno[1] **Takashi Ninomiya**[2] **Yusuke Miyao**[1] **Jun'ichi Tsujii**[134]
[1]Department of Computer Science, University of Tokyo
[2]Information Technology Center, University of Tokyo
[3]School of Informatics, University of Manchester
[4]SORST, JST
Hongo 7-3-1, Bunkyo-ku, Tokyo, Japan
{unno, yusuke, tsujii}@is.s.u-tokyo.ac.jp
ninomi@r.dl.itc.u-tokyo.ac.jp

Abstract

Sentence compression is a task of creating a short grammatical sentence by removing extraneous words or phrases from an original sentence while preserving its meaning. Existing methods learn statistics on trimming context-free grammar (CFG) rules. However, these methods sometimes eliminate the original meaning by incorrectly removing important parts of sentences, because trimming probabilities only depend on parents' and daughters' non-terminals in applied CFG rules. We apply a maximum entropy model to the above method. Our method can easily include various features, for example, other parts of a parse tree or words the sentences contain. We evaluated the method using manually compressed sentences and human judgments. We found that our method produced more grammatical and informative compressed sentences than other methods.

1 Introduction

In most automatic summarization approaches, text is summarized by extracting sentences from a given document without modifying the sentences themselves. Although these methods have been significantly improved to extract good sentences as summaries, they are not intended to shorten sentences; i.e., the output often has redundant words or phrases. These methods cannot be used to make a shorter sentence from an input sentence or for other applications such as generating headline news (Dorr et al., 2003) or messages for the small screens of mobile devices. We need to compress sentences to obtain short and useful summaries. This task is called *sentence compression*.

While several methods have been proposed for sentence compression (Witbrock and Mittal, 1999; Jing and McKeown, 1999; Vandeghinste and Pan, 2004), this paper focuses on Knight and Marcu's noisy-channel model (Knight and Marcu, 2000) and presents an extension of their method. They developed a probabilistic model for trimming a CFG parse tree of an input sentence. Their method drops words of input sentences but does not change their order or change the words. They use a parallel corpus that contains pairs of original and compressed sentences. The method makes CFG parse trees of both original and compressed sentences and learns trimming probabilities from these pairs. Although their method is concise and well-defined, its accuracy is still unsatisfactory. Their method has two problems. One is that probabilities are calculated only from the frequencies of applied CFG rules, and other characteristics like whether the phrase includes negative words cannot be introduced. The other problem is that the parse trees of original and compressed sentences sometimes do not correspond.

To solve the former problem, we apply a maximum entropy model to Knight and Marcu's model to introduce machine learning features that are defined not only for CFG rules but also for other characteristics in a parse tree, such as the depth from the root node or words it contains. To solve the latter problem, we introduce a novel matching method, *the bottom-up method*, to learn complicated relations of two unmatched trees.

We evaluated each algorithm using the Ziff-Davis corpus, which has long and short sentence pairs. We compared our method with Knight and Marcu's method in terms of F-measures, bigram F-measures, BLEU scores and human judgments.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 850–857,
Sydney, July 2006.

2 Background

2.1 The Noisy-Channel Model for Sentence Compression

Knight and Marcu proposed a sentence compression method using a noisy-channel model (Knight and Marcu, 2000). This model assumes that a long sentence was originally a short one and that the longer sentence was generated because some unnecessary words were added. Given a long sentence l, it finds a short sentence s that maximizes $P(s|l)$. This is equivalent to finding the s that maximizes $P(s) \cdot P(l|s)$ in Bayes' Rule.

The expression $P(s)$ is the source model, which gives the probability that s is the original short string. When s is ungrammatical, $P(s)$ becomes small. The expression $P(l|s)$ is the channel model, which gives the probability that s is expanded to l. When s does not include important words of l, $P(l|s)$ has a low value.

In the Knight and Marcu's model, a probabilistic context-free grammar (PCFG) score and a word-bigram score are incorporated as the source model. To estimate the channel model, Knight and Marcu used the Ziff-Davis parallel corpus, which contains long sentences and corresponding short sentences compressed by humans. Note that each compressed sentence is a subsequence of the corresponding original sentence. They first parse both the original and compressed sentences using a CFG parser to create parse trees. When two nodes of the original and compressed trees have the same non-terminals, and the daughter nodes of the compressed tree are a subsequence of the original tree, they count the node pair as a *joint event*. For example, in Figure 1, the original parse tree contains a rule $r_l = (B \rightarrow D\ E\ F)$, and the compressed parse tree contains $r_s = (B \rightarrow D\ F)$. They assume that r_s was expanded into r_l, and count the node pairs as joint events. The expansion probability of two rules is given by:

$$P_{expand}(r_l|r_s) = \frac{count(joint(r_l, r_s))}{count(r_s)}.$$

Finally, new subtrees grow from new daughter nodes in each expanded node. In Figure 1, $(E\ (G\ g)\ (H\ h))$ grows from E. The PCFG scores, P_{cfg}, of these subtrees are calculated. Then, each probability is assumed to be independent of the others, and the channel model, $P(l|s)$, is calculated as the product of all expansion probabilities of joint events and PCFG scores of new subtrees:

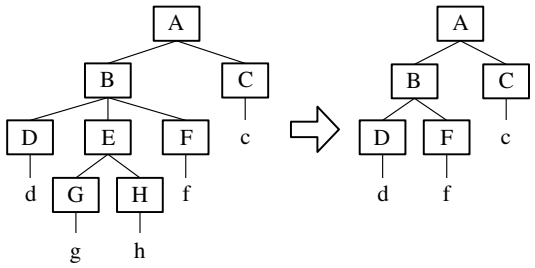


Figure 1: Examples of original and compressed parse trees.

$$P(l|s) = \prod_{(r_l, r_s) \in R} P_{expand}(r_l|r_s) \cdot \prod_{r \in R'} P_{cfg}(r),$$

where R is the set of rule pairs, and R' is the set of generation rules in new subtrees.

To compress an input sentence, they create a tree with the highest score of all possible trees. They pack all possible trees in a shared-forest structure (Langkilde, 2000). The forest structure is represented by an AND-OR tree, and it contains many tree structures. The forest representation saves memory and makes calculation faster because the trees share sub structures, and this can reduce the total number of calculations.

They normalize each log probability using the length of the compressed sentence; that is, they divide the log probability by the length of the compressed sentence.

Turner and Charniak (Turner and Charniak, 2005) added some special rules and applied this method to unsupervised learning to overcome the lack of training data. However their model also has the same problem. McDonald (McDonald, 2006) independently proposed a new machine learning approach. He does not trim input parse trees but uses rich features about syntactic trees and improved performance.

2.2 Maximum Entropy Model

The maximum entropy model (Berger et al., 1996) estimates a probability distribution from training data. The model creates the most "uniform" distribution within the constraints given by users. The distribution with the maximum entropy is considered the most uniform.

Given two finite sets of event variables, $\mathcal{X}$ and $\mathcal{Y}$, we estimate their joint probability distribution, $P(x, y)$. An output, y $(\in \mathcal{Y})$, is produced, and

contextual information, x ($\in \mathcal{X}$), is observed. To represent whether the event (x, y) satisfies a certain feature, we introduce a *feature function*. A feature function f_i returns 1 iff the event (x, y) satisfies the feature i and returns 0 otherwise.

Given training data $\{(x_1, y_1), \cdots, (x_n, y_n)\}$, we assume that the expectation of f_i on the distribution of the model conforms to that on the empirical probability distribution $\tilde{P}(x, y)$. We select the probability distribution that satisfies these constraints of all feature functions and maximizes its entropy, $H(P) = -\sum_{x,y} P(x, y) \log (P(x, y))$.

3 Methods

3.1 Maximum Entropy Model for Sentence Compression

We describe a maximum entropy method as a natural extension of Knight and Marcu's noisy-channel model (Knight and Marcu, 2000). Knight and Marcu's method uses only mother and daughter local relations in CFG parse trees. Therefore, it sometimes eliminates the meanings of the original sentences. For example, their method cannot distinguish "never" and "always" because these two adverbs are assigned the same non-terminals in parse trees. However, if "never" is removed from a sentence, the meaning of the sentence completely changes. Turner and Charniak (Turner and Charniak, 2005) revised and improved Knight and Marcu's algorithm; however, their algorithm also uses only mother and daughter relations and has the same problem. We use other information as feature functions of the maximum entropy model, and this model can deal with many features more appropriately than using simple frequency.

Suppose that we trim a node in the original full parse tree. For example, suppose we have a mother node A and daughter nodes $(B\ C\ D)$ that are derived using a CFG rule. We must leave at least one non-terminal in the daughter nodes. The trim candidates of this rule are the members of the set of subsequences, $\mathcal{Y}$, of $(B\ C\ D)$, or the seven non-terminal sequences below:

$$\mathcal{Y} = \{B, C, D, BC, BD, CD, BCD\}.$$

For each y ($\in \mathcal{Y}$), such as $(B\ C)$, *the trimming probability*, $P(y|\mathcal{Y}) = P_{trim}(A \rightarrow B\ C|A \rightarrow B\ C\ D)$, is calculated by using the maximum entropy model. We assume that these *joint events* are independent of each other and calculate the probability that an original sentence, l, is compressed to s as the product of all trimming probabilities, like in Knight and Marcu's method.

	Description
1	the mother node
2	the current node
3	the daughter node sequence in the original sentence and which daughters are removed
4	the daughter node sequence in the compressed sentence
5	the number of daughter nodes
6	the depth from the root
7	the daughter non-terminals that are removed
8	the daughter terminals that are removed
9	whether the daughters are "negative adverbs", and removed
10	tri-gram of daughter nodes
11	only one daughter exists, and its non-terminal is the same as that of the current node
12	only one daughter exists, and its non-terminal is the same as that of the mother node
13	how many daughter nodes are removed
14	the number of terminals the current node contains
15	whether the head daughter is removed
16	the left-most and the right-most daughters
17	the left and the right siblings

Table 1: Features for maximum entropy model.

$$P(s|l) = \prod_{(r_s, r_l) \in R} P_{trim}(r_s|r_l),$$

where R is the set of compressed and original rule pairs in joint events. Note that our model does not use Bayes' Rule or any language models.

For example, in Figure 1, the trimming probability is calculated as below:

$$P(s|l) = P_{trim}(A \rightarrow B\ C|A \rightarrow B\ C) \cdot P_{trim}(B \rightarrow D\ F|B \rightarrow D\ E\ F).$$

To represent all summary candidates, we create a compression forest as Knight and Marcu did. We select the tree assigned the highest probability from the forest.

Features in the maximum entropy model are defined for a tree node and its surroundings. When we process one node, or one non-terminal x, we call it the *current* node. We focus on not only x and its *daughter* nodes, but its *mother* node, its *sibling* nodes, terminals of its *subtree* and so on. The features we used are listed in Table 1.

Knight and Marcu divided the log probabilities by the length of the summary. We extend this idea so that we can change the output length flexibly. We introduce a *length parameter*, α, and define a score S_α as $S_\alpha(s) = length(s)^\alpha \log P(s|l)$, where l is an input sentence to be shortened, and s is a

summary candidate. Because $\log P(s|l)$ is negative, short sentences obtain a high score for large α, and long ones get a low score. The parameter α can be negative or positive, and we can use it to control the average length of outputs.

3.2 Bottom-Up Method

As explained in Section 2.1, in Knight and Marcu's method, both original and compressed sentences are parsed, and correspondences of CFG rules are identified. However, when the daughter nodes of a compressed rule are not a subsequence of the daughter nodes in the original one, the method cannot learn this joint event. A complex sentence is a typical example. A complex sentence is a sentence that includes another sentence as a part. An example of a parse tree of a complex sentence and its compressed version is shown in Figure 2. When we extract joint events from these two trees, we cannot match the two root nodes because the sequence of the daughter nodes of the root node of the compressed parse tree, $(NP\ ADVP\ VP\ .)$, is not a subsequence of the daughter nodes of the original parse tree, $(S\ ,\ NP\ VP\ .)$. Turner and Charniak (Turner and Charniak, 2005) solve this problem by appending special rules that are applied when a mother node and its daughter node have the same label. However, there are several types of such problems like Figure 2. We need to extract these structures from a training corpus.

We propose a *bottom-up method* to solve the problem explained above. In our method, only original sentences are parsed, and the parse trees of compressed sentences are extracted from the original parse trees. An example of this method is shown in Figure 3. The original sentence is '*d g h f c*', and its compressed sentence is '*d g c*'. First, each terminal in the parse tree of the original sentence is marked if it exists in the compressed sentence. In the figure, the marked terminals are represented by circles. Second, each non-terminal in the original parse tree is marked if it has at least one marked terminal in its sub-trees. These are represented as bold boxes in the figure. If non-terminals contain marked non-terminals in their sub-trees, these non-terminals are also marked recursively. These marked non-terminals and terminals compose a tree structure like that on the right-hand side in the figure. These non-terminals represent joint events at each node.

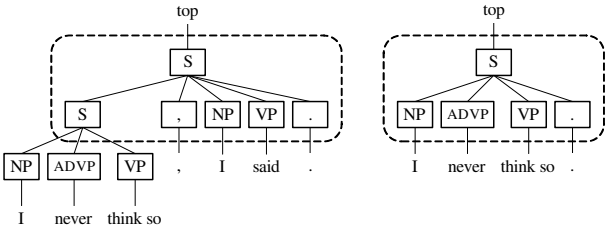


Figure 2: Example of parse tree pair that cannot be matched.

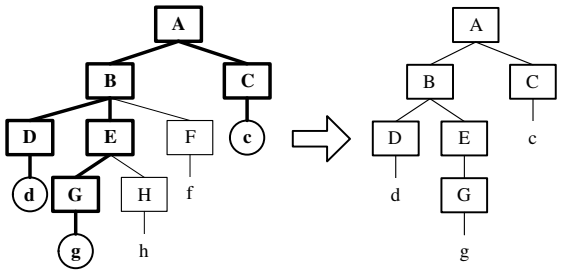


Figure 3: Example of bottom-up method.

Note that this "tree" is not guaranteed to be a grammatical "parse tree" by the CFG grammar. For example, from the tree of Figure 2, $(S\ (S\ \cdots)\ (,\ ,)\ (NP\ \mathrm{I})\ (VP\ \mathrm{said})\ (.\ .))$, a new tree, $(S\ (S\ \cdots)\ (.\ .))$, is extracted. However, the rule $(S\ \rightarrow S\ .)$ is ungrammatical.

4 Experiment

4.1 Evaluation Method

We evaluated each sentence compression method using word F-measures, bigram F-measures, and BLEU scores (Papineni et al., 2002). BLEU scores are usually used for evaluating machine translation quality. A BLEU score is defined as the weighted geometric average of n-gram precisions with length penalties. We used from unigram to 4-gram precisions and uniform weights for the BLEU scores.

ROUGE (Lin, 2004) is a set of recall-based criteria that is mainly used for evaluating summarization tasks. ROUGE-N uses average N-gram recall, and ROUGE-1 is word recall. ROUGE-L uses the length of the longest common subsequence (LCS) of the original and summarized sentences. In our model, the length of the LCS is equal to the number of common words, and ROUGE-L is equal to the unigram F-measure because words are not rearranged. ROUGE-L and ROUGE-1 are supposed to be appropriate for the headline gener-

ation task (Lin, 2004). This is not our task, but it is the most similar task in his paper.

We also evaluated the methods using human judgments. The evaluator is not the author but not a native English speaker. The judgment used the same criteria as those in Knight and Marcu's methods. We performed two experiments. In the first experiment, evaluators scored from 1 to 5 points the grammaticality of the compressed sentence. In the second one, they scored from 1 to 5 points how well the compressed sentence contained the important words of the original one.

We used the parallel corpus used in Ref. (Knight and Marcu, 2000). This corpus consists of sentence pairs extracted automatically from the Ziff-Davis corpus, a set of newspaper articles about computer products. This corpus has 1087 sentence pairs. Thirty-two of these sentences were used for the human judgments in Knight and Marcu's experiment, and the same sentences were used for our human judgments. The rest of the sentences were randomly shuffled, and 527 sentence pairs were used as a training corpus, 263 pairs as a development corpus, and 264 pairs as a test corpus.

To parse these corpora, we used Charniak and Johnson's parser (Charniak and Johnson, 2005).

4.2 Settings of Two Experiments

We experimented with/without goal sentence length for summaries.

In the first experiment, the system was given only a sentence and no sentence length information. The sentence compression problem without the length information is a general task, but evaluating it is difficult because the correct length of a summary is not generally defined even by humans. The following example shows this.

Original:"A font, on the other hand, is a subcategory of a typeface, such as Helvetica Bold or Helvetica Medium."

Human: "A font is a subcategory of a typeface, such as Helvetica Bold."

System: "A font is a subcategory of a typeface."

The "such as" phrase is removed in this system output, but it is not removed in the human summary. Neither result is wrong, but in such situations, the evaluation score of the system decreases. This is because the compression rate of each algorithm is different, and evaluation scores are affected by the lengths of system outputs. For this reason, results with different lengths cannot be compared easily. We therefore examined the relations between the average compression ratios and evaluation scores for all methods by changing the system summary length with the different *length parameter* α introduced in Section 3.1.

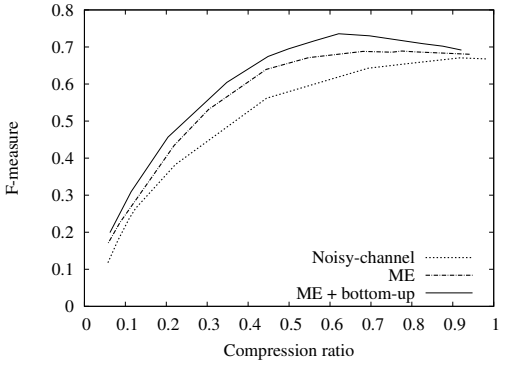


Figure 4: F-measures and compression ratios.

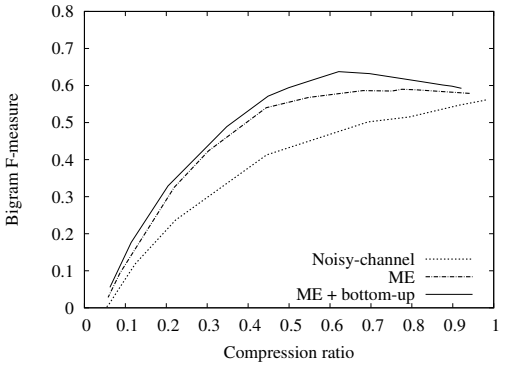


Figure 5: Bigram F-measures and compression ratios.

In the second experiment, the system was given a sentence and the length for the compressed sentence. We compressed each input sentence to the length of the sentence in its goal summary. This sentence compression problem is easier than that in which the system can generate sentences of any length. We selected the highest-scored sentence from the sentences of length l. Note that the recalls, precisions and F-measures have the same scores in this setting.

4.3 Results of Experiments

The results of the experiment without the sentence length information are shown in Figure 4, 5 and 6. *Noisy-channel* indicates the results of the noisy-channel model, *ME* indicates the results of the maximum-entropy method, and *ME + bottom-up* indicates the results of the maximum-entropy

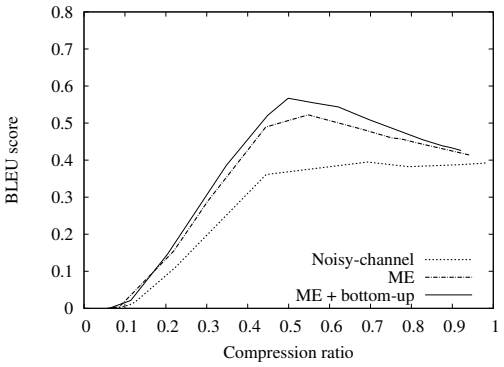


Figure 6: BLEU scores and compression ratios.

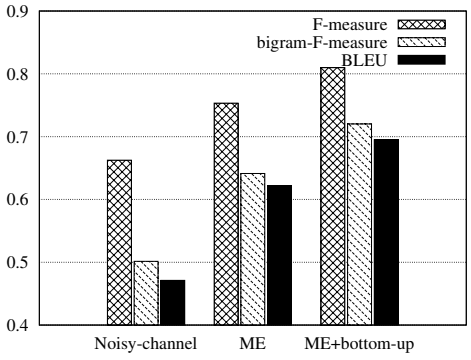


Figure 7: Results of experiments with length information.

method with the bottom-up method. We used the length parameter, α, introduced in Section 3.1, and obtained a set of summaries with different average lengths. We plotted the compression ratios and three scores in the figures. In these figures, a compression ratio is the ratio of the total number of words in compressed sentences to the total number of words in the original sentences.

In these figures, our maximum entropy methods obtained higher scores than the noisy-channel model at all compression ratios. The maximum entropy method with the bottom-up method obtain the highest scores on these three measures.

The results of the experiment with the sentence length information are shown in Figure 7. In this experiment, the scores of the maximum entropy methods were higher than the scores of the noisy-channel model. The maximum entropy method with the bottom-up method achieved the highest scores on each measure.

The results of the human judgments are shown in Table 2. In this experiment, each length of output is same as the length of goal sentence. The

Method	Grammar	Importance
Human	4.94	4.31
Noisy-channel	3.81	3.38
ME	3.88	3.38
ME + bottom-up	4.22	4.06

Table 2: Results of human judgments.

maximum entropy with the bottom-up method obtained the highest scores of the three methods. We did t-tests (5% significance). Between the noisy-channel model and the maximum entropy with the bottom-up method, importance is significantly different but grammaticality is not. Between the human and the maximum entropy with the bottom-up method, grammaticality is significantly different but importance is not. There are no significant differences between the noisy-channel model and the maximum entropy model.

4.3.1 Problem of Negative Adverbs

One problem of the noisy-channel model is that it cannot distinguish the meanings of removed words. That is, it sometimes removes semantically important words, such as "not" and "never", because the expansion probability depends only on non-terminals of parent and daughter nodes.

For example, our test corpus includes 15 sentences that contain "not". The noisy-channel model removed six "not"s, and the meanings of the sentences were reversed. However, the two maximum entropy methods removed only one "not" because they have "negative adverb" as a feature in their models. The first example in Table 3 shows one of these sentences. In this example, only *Noisy-channel* removed "not".

4.3.2 Effect of Bottom-Up Method

Our bottom-up method achieved the highest accuracy, in terms of F-measures, bigram F-measures, BLEU scores and human judgments. The results were fairly good, especially when it summarized *complex sentences*, which have sentences as parts. The second example in Table 3 is a typical complex sentence. In this example, only *ME + bottom-up* correctly remove "he said".

Most of the complex sentences were correctly compressed by the bottom-up method, but a few sentences like the third example in Table 3 were not. In this example, the original sentence was parsed as shown in Figure 8 (left). If this sentence is compressed to the human output, its parse tree has to be like that in Figure 8 (middle) using

Original	a file or application '' alias '' similar in effect to the ms-dos path statement provides a visible icon in folders where an aliased application does not actually reside .
Human	a file or application alias provides a visible icon in folders where an aliased application does not actually reside .
Noisy-channel	a similar in effect to ms-dos statement provides a visible icon in folders where an aliased application does reside .
ME	a or application alias statement provides a visible icon in folders where an aliased application does not actually reside .
ME + bottom-up	a file or application statement provides a visible icon in folders where an aliased application does not actually reside .

Original	the user can then abort the transmission , he said .
Human	the user can then abort the transmission .
Noisy-channel	the user can abort the transmission said .
ME	the user can abort the transmission said .
ME + bottom-up	the user can then abort the transmission .

Original	it is likely that both companies will work on integrating multimedia with database technologies .
Human	both companies will work on integrating multimedia with database technologies .
Noisy-channel	it is likely that both companies will work on integrating .
ME	it is likely that both companies will work on integrating .
ME + bottom-up	it is will work on integrating multimedia with database technologies .

Table 3: Examples of compressed sentences.

our method. When a parse tree is too long from the root to the leaves like this, some nodes are trimmed but others are not because we assume that each trimming probability is independent. The compressed sentence is ungrammatical, as in the third example in Table 3.

We have to constrain such ungrammatical sentences or introduce another rule that reconstructs a short tree as in Figure 8 (right). That is, we introduce a new transformation rule that compresses $(A_1\ (B\ (C\ (A_2\ \cdots))))$ to $(A_2\ \cdots)$.

4.4 Comparison with Original Results

We compared our results with Knight and Marcu's original results. They implemented two methods: one is the noisy-channel model and the other is a decision-based model. Each model produced 32 compressed sentences, and we calculated F-measures, bigram F-measures, and BLEU scores.

We used the length parameter $\alpha = 0.5$ for the maximum-entropy method and $\alpha = -0.25$ for

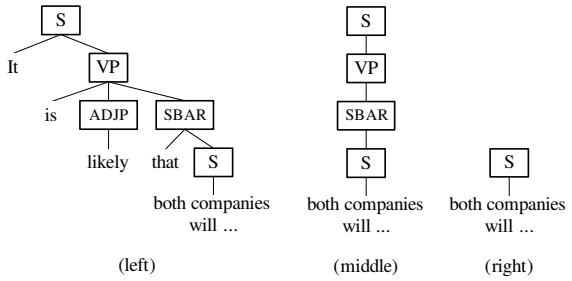


Figure 8: Parse trees of complicated complex sentences.

Method	Comp.	F-measure	bigram F-measure	BLEU
Noisy-channel	70.19%	68.80	55.96	44.54
Decision-based	57.26%	71.25	61.93	58.21
ME	66.51%	73.10	62.86	53.51
ME + bottom-up	58.14%	78.58	70.30	65.26
Human	53.59%			

Table 4: Comparison with original results.

the maximum-entropy method with the bottom-up method. These two values were determined using experiments on the development set, which did not contain the 32 test sentences.

The results are shown in Table 4. *Noisy-channel* indicates the results of Knight and Marcu's noisy-channel model, and *Decision-based* indicates the results of Knight and Marcu's decision-based model. *Comp.* indicates the compression ratio of each result. Our two methods achieved higher accuracy than the noisy-channel model. The results of the decision-based model and our maximum-entropy method were not significantly different. Our maximum-entropy method with the bottom-up method achieved the highest accuracy.

4.5 Corpus Size and Output Accuracy

In general, using more training data improves the accuracy of outputs and using less data results in low accuracy. Our experiment has the problem that the training corpus was small. To study the relation between training corpus size and accuracy, we experimented using different training corpus sizes and compared accuracy of the output.

Figure 9 shows the relations between training corpus size and three scores, F-measures, bigram F-measures and BLEU scores, when we used the maximum entropy method with the bottom-up method. This graph suggests that the accuracy in-

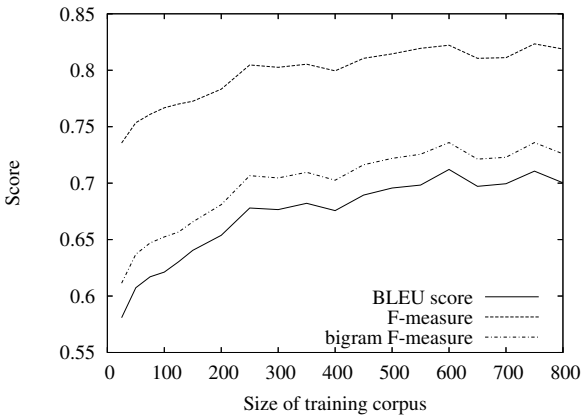


Figure 9: Relation between training corpus size and evaluation score.

creases when the corpus size is increased. Over about 600 sentences, the increase becomes slower.

The graph shows that the training corpus was large enough for this study. However, if we introduced other specific features, such as lexical features, a larger corpus would be required.

5 Conclusion

We presented a maximum entropy model to extend the sentence compression methods described by Knight and Marcu (Knight and Marcu, 2000). Our proposals are two-fold. First, our maximum entropy model allows us to incorporate various characteristics, such as a mother node or the depth from a root node, into a probabilistic model for determining which part of an input sentence is removed. Second, our bottom-up method of matching original and compressed parse trees can match tree structures that cannot be matched using Knight and Marcu's method.

The experimental results show that our maximum entropy method improved the accuracy of sentence compression as determined by three evaluation criteria: F-measures, bigram F-measures and BLEU scores. Using our bottom-up method further improved accuracy and produced short summaries that could not be produced by previous methods. However, we need to modify this model to appropriately process more complicated sentences because some sentences were not correctly summarized. Human judgments showed that the maximum entropy model with the bottom-up method provided more grammatical and more informative summaries than other methods.

Though our training corpus was small, our experiments demonstrated that the data was sufficient. To improve our approaches, we can introduce more feature functions, especially more semantic or lexical features, and to deal with these features, we need a larger corpus.

Acknowledgements

We would like to thank Prof. Kevin Knight and Prof. Daniel Marcu for providing their parallel corpus and the experimental results.

References

A. L. Berger, V. J. Della Pietra, and S. A. Della Pietra. 1996. A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, 22(1):39–71.

E. Charniak and M. Johnson. 2005. Coarse-to-Fine n-Best Parsing and MaxEnt Discriminative Reranking. In *Proc. of ACL'05*, pages 173–180.

B. Dorr, D. Zajic, and R. Schwartz. 2003. Hedge Trimmer: A Parse-and-Trim Approach to Headline Generation. In *Proc. of DUC 2003*, pages 1–8.

H. Jing and K. R. McKeown. 1999. The decomposition of human-written summary sentences. In *Proc. of SIGIR'99*, pages 129–136.

K. Knight and D. Marcu. 2000. Statistics-Based Summarization - Step One: Sentence Compression. In *Proc. of AAAI/IAAI'00*, pages 703–710.

I. Langkilde. 2000. Forest-Based Statistical Sentence Generation. In *Proc. of NAACL'00*, pages 170–177.

C. Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out: Proc. of ACL'04 Workshop*, pages 74–81.

R. McDonald. 2006. Discriminative Sentence Compression with Soft Syntactic Evidence. In *Proc. of EACL'06*.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proc. of ACL'02*, pages 311–318.

J. Turner and E. Charniak. 2005. Supervised and Unsupervised Learning for Sentence Compression. In *Proc. of ACL'05*, pages 290–297.

V. Vandeghinste and Y. Pan. 2004. Sentence Compression for Automated Subtitling: A Hybrid Approach. In *Text Summarization Branches Out: Proc. of ACL'04 Workshop*, pages 89–95.

M. J. Witbrock and V. O. Mittal. 1999. Ultra-Summarization: A Statistical Approach to Generating Highly Condensed Non-Extractive Summaries. In *Proc. of SIGIR'99*, pages 315–316.

Word Vectors and Two Kinds of Similarity

Akira Utsumi and Daisuke Suzuki
Department of Systems Engineering
The University of Electro-Communications
1-5-1 Chofugaoka, Chofushi, Tokyo 182-8585, Japan
utsumi@se.uec.ac.jp, dajie@utm.se.uec.ac.jp

Abstract

This paper examines what kind of similarity between words can be represented by what kind of word vectors in the vector space model. Through two experiments, three methods for constructing word vectors, i.e., LSA-based, cooccurrence-based and dictionary-based methods, were compared in terms of the ability to represent two kinds of similarity, i.e., taxonomic similarity and associative similarity. The result of the comparison was that the dictionary-based word vectors better reflect taxonomic similarity, while the LSA-based and the cooccurrence-based word vectors better reflect associative similarity.

1 Introduction

Recently, geometric models have been used to represent words and their meanings, and proven to be highly useful both for many NLP applications associated with semantic processing (Widdows, 2004) and for human modeling in cognitive science (Gärdenfors, 2000; Landauer and Dumais, 1997). There are also good reasons for studying geometric models in the field of computational linguistics. First, geometric models are cost-effective in that it takes much less time and less effort to construct large-scale geometric representation of word meanings than it would take to construct dictionaries or thesauri. Second, they can represent the implicit knowledge of word meanings that dictionaries and thesauri cannot do. Finally, geometric representation is easy to revise and extend.

A vector space model is the most commonly used geometric model for the meanings of words. The basic idea of a vector space model is that words are represented by high-dimensional vectors, i.e., *word vectors*, and the degree of semantic similarity between any two words can be easily computed as a cosine of the angle formed by their vectors.

A number of methods have been proposed for constructing word vectors. Latent semantic analysis (LSA) is the most well-known method that uses the frequency of words in a fraction of documents to assess the coordinates of word vectors and singular value decomposition (SVD) to reduce the dimension. LSA was originally put forward as a document indexing technique for automatic information retrieval (Deerwester et al., 1990), but several studies (Landauer and Dumais, 1997) have shown that LSA successfully mimics many human behaviors associated with semantic processing. Other methods use a variety of other information: cooccurrence of two words (Burgess, 1998; Schütze, 1998), occurrence of a word in the sense definitions of a dictionary (Kasahara et al., 1997; Niwa and Nitta, 1994) or word association norms (Steyvers et al., 2004).

However, despite the fact that there are different kinds of similarity between words, or different relations underlying word similarity such as a synonymous relation and an associative relation, no studies have ever examined the relationship between methods for constructing word vectors and the type of similarity involved in word vectors in a systematic way. Some studies on word vectors have compared the performance among different methods on some specific tasks such as semantic disambiguation (Niwa and Nitta, 1994) and cued/free recall (Steyvers et al., 2004), but it is not at all clear whether there are essential differences in the quality of similarity among word vectors constructed by different methods, and if so, what kind of similarity is involved in what kind of word vectors. Even in the field of cognitive psychology, although geometric models of similarity such as multidimensional scaling have long been studied and debated (Nosofsky, 1992), the possibility that different methods for word vectors may cap-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 858–865,
Sydney, July 2006.

ture different kinds of word similarity has never been addressed.

This study, therefore, aims to examine the relationship between the methods for constructing word vectors and the type of similarity in a systematic way. Especially this study addresses three methods, *LSA-based*, *cooccurrence-based*, and *dictionary-based methods*, and two kinds of similarity, *taxonomic similarity* and *associative similarity*. Word vectors constructed by these methods are compared in the performance of two tasks, i.e., multiple-choice synonym test and word association, which measure the degree to which they reflect these two kinds of similarity.

2 Two Kinds of Similarity

In this study, we divide word similarity into two categories: taxonomic similarity and associative similarity. Taxonomic similarity, or categorical similarity, is a kind of semantic similarity between words in the same level of categories or clusters of the thesaurus, in particular synonyms, antonyms, and other coordinates. Associative similarity, on the other hand, is a similarity between words that are associated with each other by virtue of semantic relations other than taxonomic one such as a collocational relation and a proximity relation. For example, the word *writer* and the word *author* are taxonomically similar because they are synonyms, while the word *writer* and the word *book* are associatively similar because they are associated by virtue of an agent-subject relation.

This dichotomy of similarity is practically important. Some tasks such as automatic thesaurus updating and paraphrasing need assessing taxonomic similarity, while some other tasks such as affective Web search and semantic disambiguation require assessing associative similarity rather than taxonomic similarity. This dichotomy is also psychologically motivated. Many empirical studies on word searches and speech disorders have revealed that words in the mind (i.e., mental lexicon) are organized by these two kinds of similarity (Aitchison, 2003). This dichotomy is also essential to some cognitive processes. For example, metaphors are perceived as being more apt when their constituent words are associatively more similar but categorically dissimilar (Utsumi et al., 1998). These psychological findings suggest that people distinguish between these two kinds of similarity in certain cognitive processes.

3 Constructing Word Vectors

3.1 Overview

In this study, word vectors (or word spaces) are constructed in the following way. First, all content words t_i in a corpus are represented as m-dimensional feature vectors $\boldsymbol{w}_i$.

$$\boldsymbol{w}_i = (w_{i1}, w_{i2}, \cdots, w_{im}) \tag{1}$$

Each element w_{ij} is determined by statistical analysis of the corpus, whose methods will be described in Section 3.3. A matrix M is then constructed using n feature vectors as rows.

$$M = \begin{pmatrix} \boldsymbol{w}_1 \\ \vdots \\ \boldsymbol{w}_n \end{pmatrix} \tag{2}$$

Finally, the dimension of row vectors $\boldsymbol{w}_i$ is reduced from m to k by means of a SVD technique. As a result, any words are represented as k-dimensional vectors.

3.2 Corpus

In this study, we employ three kinds of Japanese corpora: newspaper articles, novels and a dictionary. As a newspaper corpus, we use 4 months' worth of Mainichi newspaper articles published in 1999. They consist of 500,182 sentences in 251,287 paragraphs, and words vectors are constructed for 53,512 words that occur three times or more in these articles. Concerning a corpus of novels, we use a collection of 100 Japanese novels "Shincho Bunko No 100 Satsu" consisting of 475,782 sentences and 230,392 paragraphs. Word vectors are constructed for 46,666 words that occur at least three times. As a Japanese dictionary, we use "Super Nihongo Daijiten" published by Gakken, from which 89,007 words are extracted for word vectors.

3.3 Methods for Computing Vector Elements

LSA-based method (LSA)

In the LSA-based method, a vector element w_{ij} is assessed as a tf-idf score of a word t_i in a piece s_j of document.

$$w_{ij} = tf_{ij} \times \left(\log \frac{m}{df_i} + 1 \right) \tag{3}$$

In this formula, tf_{ij} denotes the number of times the word t_i occurs in a piece of text s_j, and df_i denotes the number of pieces in which the word t_i occurs. As a unit of text piece s_j, we consider

a sentence and a paragraph. Hence, for example, when a sentence is used as a unit, the dimension of feature vectors $\boldsymbol{w}_i$ is equal to the number of sentences in a corpus. We also use two corpora, i.e., newspapers and novels, and thus we obtain four different word spaces by the LSA-based method.

Cooccurrence-based method (COO)

In the cooccurrence-based method, a vector element w_{ij} is assessed as the number of times words t_i and t_j occur in the same piece of text, and thus M is an $n \times n$ symmetric matrix. As in the case of the LSA-based method, we use two units of text piece (i.e., a sentence or a paragraph) and two corpora (i.e., newspapers or novels), thus resulting in four different word spaces.

Note that this method is similar to Schütze's (1998) method for constructing a semantic space in that both are based on the word cooccurrence, not on the word frequency. However they are different in that Schütze's method uses the cooccurrence with frequent content words chosen as indicators of primitive meanings. Burgess's (1998) "Hyperspace Analogue to Language (HAL)" is also based on the word cooccurrence but does not use any technique of dimensionality reduction.

Dictionary-based method (DIC)

In the dictionary-based method, a vector element w_{ij} is assessed by the following formula:

$$w_{ij} = \left(f_{ij} + \alpha \sqrt{\sum_k f_{ik} f_{kj}} + \beta f_{ji} \right) \times \log \frac{n}{df_j} \quad (4)$$

where f_{ij} denotes the number of times the word t_j occurs in the sense definitions of the word t_i, and df_j denotes the number of words whose sense definitions contain the word t_j. The second term in parentheses in Equation (4) means the square root of the number of times the word t_j occurs in a collection of sense definitions for any words that are included in the sense definitions of the word t_i, while the third term means the number of times t_i occurs in the sense definitions of t_j. The parameters α and β are positive real constants expressing the weights for these information. (Following Kasahara et al. (1997), these parameters are set to 0.2 in this paper.)

Equation (4) was originally put forward by Kasahara et al. (1997), but our dictionary-based method differs from their method in terms of how the dimensions are reduced. Their method groups together the dimensions for words in the same category of a thesaurus, but our method uses SVD as we will described next.

3.4 Reducing Dimensions

Using a SVD technique, a matrix M is factorized as the product of three matrices $U\Sigma V^T$, where the diagonal matrix Σ consists of r singular values that are arranged in nonincreasing order such that r is the rank of M. When we use a $k \times k$ matrix Σ_k consisting of the largest k singular values, the matrix M is approximated by $U_k \Sigma_k V_k^T$, where the i-th row of U_k corresponds to a k-dimensional "reduced word vector" for the word t_i.

4 Experiment 1: Synonym Judgment

4.1 Method

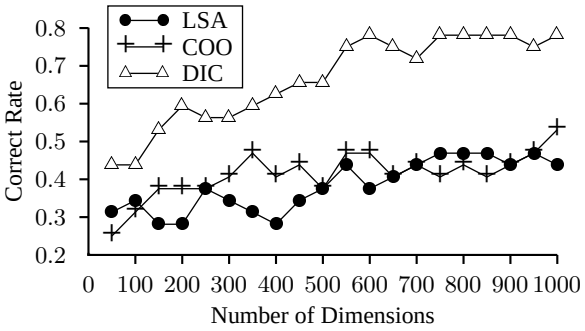
In order to compare different word vectors in terms of the ability to judge taxonomic similarity between words, we conducted a synonym judgment experiment using a standard multiple-choice synonym test. Each item of a synonym test consisted of a stem word and five alternative words from which the test-taker was asked to choose one with the most similar meaning to the stem word.

In the experiment, we used 32 items from the synonym portions of Synthetic Personality Inventory (SPI) test, which has been widely used for employment selection in Japanese companies. These items were selected so that all the vector spaces could contain the stem word and at least four of the five alternative words. For comparison purpose, we also used 38 antonym test items chosen from the same SPI test. Furthermore, in order to obtain a more reliable, unbiased result, we automatically constructed 200 test items in such a way that we chose the stem word randomly, one correct alternative word randomly from words in the same deepest category of a Japanese thesaurus as the stem word, and other four alternatives from words in other categories. As a Japanese thesaurus, we used "Goi-Taikei" (Ikehara et al., 1999).

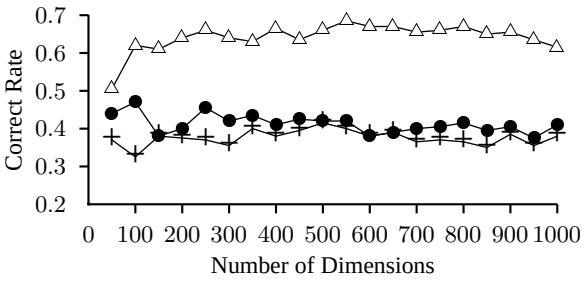
In the computer simulation, the computer's choices were determined by computing cosine similarity between the stem word and each of the five alternative words using the vector spaces and choosing the word with the highest similarity.

4.2 Results and Discussion

For each of the nine vector spaces, the synonym judgment simulation described above was con-



(a) SPI test items



(b) Computer-generated test items

Figure 1: Correct rates of synonym tests

ducted and the percentage of correct choices was calculated. This process was repeated using 20 numbers of dimensions, i.e., every 50 dimensions between 50 and 1000.

Figure 1 shows the percentage of correct choices for the three methods of matrix construction. Concerning the LSA-based method (denoted by LSA) and the cooccurrence-based method (denoted by COO), Figure 1 plots the correct rates for the word vectors derived from the paragraphs of the newspaper corpus. (Such combination of corpus and text unit was optimal among all combinations, which will be justified later in this section.) The most important result shown in Figure 1 is that, regardless of the number of dimensions, the dictionary-based word vectors outperformed the other kinds of vectors on both SPI and computer-generated test items. This result thus suggests that *the dictionary-based vector space reflects taxonomic similarity between words better than the LSA-based and the correlation-based spaces.*

Another interesting finding is that there was no clear peak in the graphs of Figure 1. For SPI test items, correct rates of the three methods increased linearly as the number of dimensions increased, $r = .86$ for the LSA-based method, $r = .72$ for the correlation-based method and $r = .93$ for the dictionary-based method (all $ps < .0001$), while correct rates for computer-generated test items

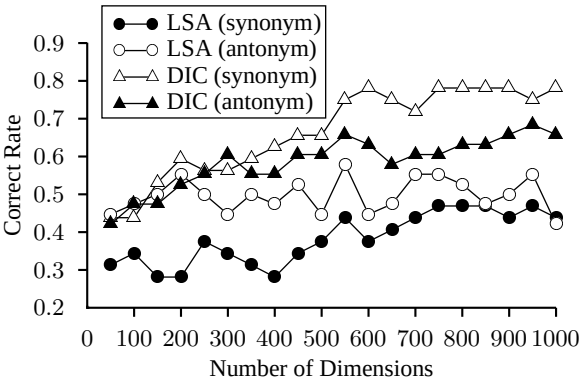


Figure 2: Synonym versus antonym judgment

were steady. Our finding of the absence of any obvious optimal dimensions is in a sharp contrast to Landauer and Dumais's (1997) finding that the LSA word vectors with 300 dimensions achieved the maximum performance of 53% correct rate in a similar multiple-choice synonym test. Note that their maximum performance was a little better than that of our LSA vectors, but still worse than that of our dictionary-based vectors.

Figure 2 shows the performance of the LSA-based and the dictionary-based methods in antonym judgment, together with the result of synonym judgment. (Since the performance of the cooccurrence-based method did not differ from that of the LSA-based method, the correct rates of the cooccurrence-based method are not plotted in this figure.) The dictionary-based method also outperformed the LSA-based method in antonym judgment but their difference was much smaller than that of synonym judgment; at 200 or lower dimensions LSA-based method was better than the dictionary-based method. Interestingly, *the dictionary-based word vectors yielded better performance in synonym judgment than in antonym judgment, while the LSA-based vectors showed better performance in antonym judgment.* These contrasting results may be attributed to the difference of corpus characteristics. Dictionary's definitions for antonymous words are likely to involve different words so that the differences between their meanings can be made clear. On the other hand, in newspaper articles (or literary texts), context words with which antonymous words occur are likely to overlap because their meanings are about the same domain.

Finally, we show the results of comparison among four combinations of corpora and text units for the LSA-based and the cooccurrence-based

Table 1: Comparison of mean correct rate among the combinations of two corpora and two text units

	Newspaper		Novel	
Method	Para	Sent	Para	Sent
SPI test				
LSA	**0.383**	0.366	0.238	0.369
COO	**0.413**	0.369	0.255	0.280
Computer-generated test				
LSA	**0.410**	0.377	0.346	0.379
COO	**0.375**	0.363	0.311	0.310

Note. Para = Paragraph; Sent = Sentence.

methods. Table 1 lists mean correct rates of SPI test and computer-generated test averaged over all the numbers of dimensions. Regardless of construction methods and test items, the word vectors constructed using newspaper paragraphs achieved the best performance, which are denoted by boldfaces. Concerning an effect of corpus difference, the newspaper corpus was superior to the literary corpus. The difference of text units did not have a clear influence on the performance of word spaces.

5 Experiment 2: Word Association

5.1 Method

In order to compare the ability of the word spaces to judge associative similarity, we conducted a word association experiment using a Japanese word association norm "Renso Kijunhyo" (Umemoto, 1969). This free-association database was developed based on the responses of 1,000 students to 210 stimulus words. For example, when given the word *writer* as a stimulus, students listed the words shown in Table 2. (Table 2 also shows the original words in Japanese.)

For the simulation experiment, we selected 176 stimulus words that all the three corpora contained. These stimuli had 27 associate words on average. We then removed any associate words that were synonymous with the stimulus word (e.g., *author* in Table 2), since the purpose of this experiment was to examine the ability to assess associative similarity between words. Whether or not each associate is synonymous with the stimulus was determined according to whether they belong to the same deepest category of a Japanese thesaurus "Goi-Taikei" (Ikehara et al., 1999).

In the computer simulation, cosine similarity

Table 2: Associates for the stimulus word *writer*

Stimulus: writer（作家）				
Associates:				
novel 小説	pen ペン	literary work 著作		painter 画家
book 本	author 作者	best-seller ベストセラー		money 金
write 書く	literature 文学	play 劇	art 芸術	work 作品
popular 流行	human 人	book 書物	paper 紙	pencil 鉛筆
lucrative 儲かる	writing 文章	mystery 推理	music 音楽	

between the stimulus word and each of all the other words included in the vector space was computed, and all the words were sorted in descending order of similarity. The top i words were then chosen as associates.

The ability of word spaces to mimic human word association was evaluated on mean precision. Precision is the ratio of the number of human-produced associates chosen by computer to the number i of computer-chosen associates. A precision score was calculated every time a new human-produced associate was found in the top i words when i was incremented by 1, and after that mean precision was calculated as the average of all these precision scores. It must be noted here that, in order to eliminate the bias caused by the difference in the number of contained words among word spaces, we conducted the simulation using 46,000 words that we randomly chose for each corpus so that they could include all the human-produced associates.

Although this computational method of producing associates is sufficient for the present purpose, it may be inadequate to model the psychological process of free association. Some empirical studies of word association (Nelson et al., 1998) revealed that frequent or familiar words were highly likely to be produced as associates, but our methods for constructing word vectors may not directly address such frequency effect on word association. Hence, we conducted an additional experiment in which only familiar words were used for computing similarity to a given stimulus word, i.e., less familiar words were not used as candidates of as-

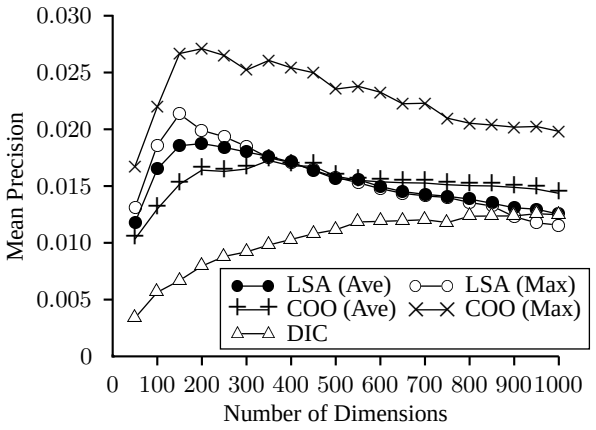


Figure 3: Mean precision of word association judgment

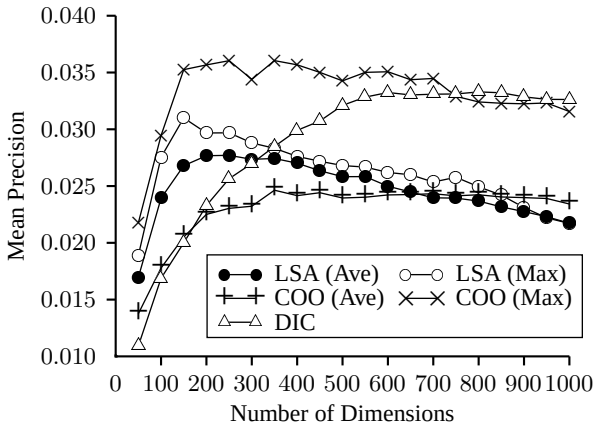


Figure 4: Average precision of word association judgment for familiar words

sociates. For a measure of word familiarity, we used word familiarity scores (ranging from 1 to 7) provided by "Nihongo no Goitaikei" (Amano and Kondo, 2003). Using this data, we selected the words whose familiarity score is 5 or higher as familiar ones.

5.2 Results and Discussion

For each of the nine vector spaces, the association judgment simulation was conducted and the mean precision was calculated. As in the synonym judgment experiment, this process was repeated by every 50 dimensions between 50 and 1000.

Figure 3 shows the result of word association experiment. For the LSA-based and the cooccurrence-based methods, two kinds of mean precision were plotted: the average of mean precision scores for the four word vectors and the maximum score among them. (As we will show in Table 3, the LSA-based method achieved the maximum precision when sentences of the newspaper corpus were used, while the performance of the cooccurrence-based method was maximal when paragraphs of the newspaper corpus were used.) The overall result was that the dictionary-based word vectors yielded the worst performance, as opposed to the result of synonym judgment. There was no big difference in performance between the LSA-based method and the cooccurrence-based method, but the maximal cooccurrence-based vectors (constructed from newspaper paragraphs) considerably outperformed the other kinds of word vectors. [1] These results clearly show that *the LSA-based and the cooccurrence-based vector spaces reflect associative similarity between words more than the dictionary-based space.*

The relation between the number of dimensions and the performance in association judgment was quite different from the relation observed in the synonym judgment experiment. Although the score of the dictionary-based vectors increased as the dimension of the vectors increased as in the case of synonym judgment, the scores of both LSA-based and cooccurrence-based vectors had a peak around 200 dimensions, as Landauer and Dumais (1997) demonstrated. This finding seems to suggest that some hundred dimensions may be enough to represent the knowledge of associative similarity.

Figure 4 shows the result of the additional experiment in which familiarity effects were taken into account. As compared to the result without familiarity filtering, there was a remarkable improvement of the performance of the dictionary-based method; the dictionary-based method outperformed the LSA-based method at 350 or higher dimensions and the cooccurrence-based method at 800 or higher dimensions. This may be because word occurrence in the sense definitions of a dictionary does not reflect the actual frequency or familiarity of words, and thus the dictionary-based method may possibly overestimate the similarity of infrequent or unfamiliar words. On the other hand, since the corpus of newspaper articles or novels is likely to reflect actual word frequency, the vector spaces derived from these corpora represent the similarity of infrequent words as appropriately as that of familiar words. [2]

The result that the cooccurrence-based word

[1] These results were replicated even when all the human-produced associates including synonymous ones were used for assessing the precision scores.

Table 3: Comparison of mean precision among the combinations of two corpora and two text units

	Newspaper		Novel	
Method	Para	Sent	Para	Sent
All associates				
LSA	0.016	**0.017**	0.015	0.015
COO	**0.023**	0.018	0.012	0.008
Familiar associates				
LSA	**0.0261**	0.0258	0.024	0.023
COO	**0.033**	0.027	0.018	0.014

Note. Para = Paragraph; Sent = Sentence.

vectors constructed from newspaper paragraphs achieved the best performance was again obtained in the additional experiment. This consistent result indicates that *the cooccurrence-based method is particularly useful for representing the knowledge of associative similarity between words.* The relation between the number of dimensions and mean precision was unchanged even if a familiarity effect was considered.

Finally, Table 3 shows the comparison result among four kinds of word vectors constructed from different corpora and text units in the experiment with and without familiarity filtering. The listed values are mean precisions averaged over all the 20 numbers of dimensions. As in the case of synonym judgment experiment, word vectors constructed from newspaper paragraphs achieved the best performance, although only the LSA-based vectors had the highest precision when they were derived from sentences of newspaper articles. As in the case of synonym judgment, the newspaper corpus showed better performance than the novel corpus, and especially the cooccurrence-based method showed a fairly large difference in performance between two corpora. This finding seems to suggest that word cooccurrence in a newspaper corpus is more likely to reflect associative similarity.

6 Semantic Network and Similarity

As related work, Steyvers and Tenenbaum (2005) examined the properties of semantic network, another important geometric model for word meanings. They found that three kinds of semantic networks — WordNet, Roget's thesaurus, and word associations — had a small-world structure and a scale-free pattern of connectivity, but semantic networks constructed from the LSA-based vector spaces did not have these properties. They interpreted this finding as indicating a limitation of the vector space model such as LSA to model human knowledge of word meanings.

However, we can interpret their finding in a different way by considering a possibility that different semantic networks may capture different kinds of word similarity. Scale-free networks have a common characteristic that a small number of nodes are connected to a very large number of other nodes (Barabási and Albert, 1999). In the semantic networks, such "hub" nodes correspond to basic and highly polysemous words such as *make* and *money*, and these words are likely to be taxonomically similar to many other words. Hence if semantic networks reflect in large part taxonomic similarity between words, they are likely to have a scale-free structure. On the other hand, since it is less likely to assume that only a few words are associatively similar to a large number of other words, semantic networks reflecting associative similarity may not have a scale-free structure. Taken together, Steyvers and Tenenbaum's (2005) finding can be reinterpreted as suggesting that WordNet and Roget's thesaurus better reflect taxonomic similarity, while the LSA-based word vectors better reflect associative similarity, which is consistent with our finding.

7 Conclusion

Through two simulation experiments, we obtained the following findings:

- The dictionary-based word vectors better reflect the knowledge of taxonomic similarity, while the LSA-based and the cooccurrence-based word vectors better reflect the knowledge of associative similarity. In particular, the cooccurrence-based vectors are useful for representing associative similarity.
- The dictionary-based vectors yielded better performance in synonym judgment, but the LSA-based vectors showed better performance in antonym judgment.
- These kinds of word vectors showed the distinctive patterns of the relationship between

[2]Indeed, the dictionary-based vector spaces contained a larger number of unfamiliar words than the other spaces; 63% of words in the dictionary were judged as unfamiliar, while only 42% and 50% of words in the newspapers and the novels were judged as unfamiliar.

the number of dimensions of word vectors and their performance.

We are now extending this work to examine in more detail the relationship between various kinds of word vectors and the quality of word similarity involved in these vectors. It would be interesting for further work to develop a method for extracting the knowledge of a specific similarity from the word space, e.g., extracting the knowledge of taxonomic similarity from the dictionary-based word space. Vector negation (Widdows, 2003) may be a useful technique for this purpose. At the same time we are also interested in a method for combining different word spaces into one space, e.g., combining the dictionary-based and the LSA-based spaces into one coherent word space. Additionally we are trying to simulate cognitive processes such as metaphor comprehension (Utsumi, 2006).

Acknowledgment

This research was supported in part by Grant-in-Aid for Scientific Research(C) (No.17500171) from Japan Society for the Promotion of Science.

References

Jean Aitchison. 2003. *Words in the Mind: An Introduction to the Mental Lexicon, 3rd Edition*. Oxford, Basil Blackwell.

Shigeaki Amano and Kimihisa Kondo, editors. 2003. *Nihongo-no Goitokusei CD-ROM (Lexical properties of Japanese)*. Sanseido, Tokyo.

Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *Science*, 286:509–512.

Curt Burgess. 1998. From simple associations to the building blocks of language: Modeling meaning in memory with the HAL model. *Behavior Research Methods, Instruments, & Computers*, 30(2):188–198.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas L. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society For Information Science*, 41(6):391–407.

Peter G¨ardenfors. 2000. *Conceptual Spaces: The Geometry of Thought*. MIT Press.

Satoru Ikehara, Masahiro Miyazaki, Satoshi Shirai, Akio Yokoo, Hiromi Nakaiwa, Kentaro Ogura, Yoshifumi Ooyama, and Yoshihiko Hayashi. 1999. *Goi-Taikei: A Japanese Lexicon CDROM*. Iwanami Shoten, Tokyo.

Kaname Kasahara, Kazumitsu Matsuzawa, and Tsutomu Ishikawa. 1997. A method for judgment of semantic similarity between daily-used words by using machine readable dictionaries. *Transactions of Information Processing Society of Japan*, 38(7):1272–1283. in Japanese.

Thomas K. Landauer and Susan T. Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*, 104:211–240.

Douglas L. Nelson, Cathy L. McEvoy, and Thomas A. Schreiber. 1998. The university of south fbrida word association, rhyme, and word fragment norms. http://www.usf.edu/FreeAssociation/.

Yoshiki Niwa and Yoshihiko Nitta. 1994. Co-occurrence vectors from corpora vs. distance vectors from dictionaries. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING94)*, pages 304–309.

Robert M. Nosofsky. 1992. Similarity scaling and cognitive process models. *Annual Review of Psychology*, 43:25–53.

Hinrich Sch¨utze. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123.

Mark Steyvers and Joshua B. Tenenbaum. 2005. The large-scale structure of semantic network: Statistical analyses and a model of semantic growth. *Cognitive Science*, 29(1):41–78.

Mark Steyvers, Richard M. Shiffrin, and Douglas L. Nelson. 2004. Word association spaces for predicting semantic similarity effects in episodic memory. In Alice F. Healy, editor, *Experimental Cognitive Psychology and Its Applications*. American Psychological Association, 2004.

Takao Umemoto. 1969. *Renso Kijunhyo (Free Association Norm)*. Tokyo Daigaku Shuppankai, Tokyo.

Akira Utsumi, Koichi Hori, and Setsuo Ohsuga. 1998. An affective-similarity-based method for comprehending attributional metaphors. *Journal of Natural Language Processing*, 5(3):3–32.

Akira Utsumi. 2006. Computational exploration of metaphor comprehension processes. In *Proceedings of the 28th Annual Meeting of the Cognitive Science Society (CogSci 2006)*.

Dominic Widdows. 2003. Orthogonal negation in vector spaces for modelling word-meanings and document retrieval. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 136–143.

Dominic Widdows. 2004. *Geometry and Meaning*. CSLI Publications.

Finding Synonyms Using Automatic Word Alignment and Measures of Distributional Similarity

Lonneke van der Plas & Jörg Tiedemann
Alfa-Informatica
University of Groningen
P.O. Box 716
9700 AS Groningen
The Netherlands
{vdplas,tiedeman}@let.rug.nl

Abstract

There have been many proposals to extract semantically related words using measures of distributional similarity, but these typically are not able to distinguish between synonyms and other types of semantically related words such as antonyms, (co)hyponyms and hypernyms. We present a method based on automatic word alignment of parallel corpora consisting of documents translated into multiple languages and compare our method with a monolingual syntax-based method. The approach that uses aligned multilingual data to extract synonyms shows much higher precision and recall scores for the task of synonym extraction than the monolingual syntax-based approach.

1 Introduction

People use multiple ways to express the same idea. These alternative ways of conveying the same information in different ways are referred to by the term *paraphrase* and in the case of single words sharing the same meaning we speak of *synonyms*. Identification of synonyms is critical for many NLP tasks. In information retrieval the information that people ask for with a set of words may be found in in a text snippet that comprises a completely different set of words. In this paper we report on our findings trying to automatically acquire synonyms for Dutch using two different resources, a large monolingual corpus and a multilingual parallel corpus including 11 languages.

A common approach to the automatic extraction of semantically related words is to use distributional similarity. The basic idea behind this is that similar words share similar contexts. Systems based on distributional similarity provide ranked lists of semantically related words according to the similarity of their contexts. Synonyms are expected to be among the highest ranks followed by (co)hyponyms and hypernyms, since the highest degree of semantic relatedness next to identity is synonymy.

However, this is not always the case. Several researchers (Curran and Moens (2002), Lin (1998), van der Plas and Bouma (2005)) have used large monolingual corpora to extract distributionally similar words. They use grammatical relations[1] to determine the context of a target word. We will refer to such systems as *monolingual syntax-based* systems. These systems have proven to be quite successful at finding semantically related words. However, they do not make a clear distinction between synonyms on the one hand and related words such as antonyms, (co)hyponyms, hypernyms etc. on the other hand.

In this paper we have defined context in a multilingual setting. In particular, translations of a word into other languages found in parallel corpora are seen as the (translational) context of that word. We assume that words that share translational contexts are semantically related. Hence, relatedness of words is measured using distributional similarity in the same way as in the monolingual case but with a different type of context. Finding translations in parallel data can be approx-

[1] One can define the context of a word in a non-syntactic monolingual way, that is as the document in which it occurs or the n words surrounding it. From experiments we have done and also building on the observations made by other researchers (Kilgarriff and Yallop, 2000) we can state that this approach generates a type of semantic similarity that is of a *looser* kind, an associative kind,for example *doctor* and *disease*. These words are typically not good candidates for synonymy.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 866–873,
Sydney, July 2006.

imated by automatic word alignment. We will refer to this approach as *multilingual alignment-based* approaches. We expect that these translations will give us synonyms and less semantically related words, because translations typically do not expand to hypernyms, nor (co)hyponyms, nor antonyms. The word *apple* is typically not translated with a word for *fruit* nor *pear*, and neither is *good* translated with a word for *bad*.

In this paper we use both monolingual syntax-based approaches and multilingual alignment-based approaches and compare their performance when using the same similarity measures and evaluation set.

2 Related Work

Monolingual syntax-based distributional similarity is used in many proposals to find semantically related words (Curran and Moens (2002), Lin (1998), van der Plas and Bouma (2005)).

Several authors have used a monolingual parallel corpus to find paraphrases (Ibrahim et al. (2003), Barzilay and McKeown (2001)). However, bilingual parallel corpora have mostly been used for tasks related to word sense disambiguation such as target word selection (Dagan et al., 1991) and separation of senses (Dyvik, 1998). The latter work derives relations such as synonymy and hyponymy from the separated senses by applying the method of semantic mirrors.

Turney (2001) reports on an PMI and IR driven approach that acquires data by querying a Web search engine. He evaluates on the TOEFL test in which the system has to select the synonym among 4 candidates.

Lin et al. (2003) try to tackle the problem of identifying synonyms among distributionally related words in two ways: Firstly, by looking at the overlap in translations of semantically similar words in multiple bilingual dictionaries. Secondly, by looking at patterns specifically designed to filter out antonyms. They evaluate on a set of 80 synonyms and 80 antonyms from a thesaurus.

Wu and Zhou's (2003) paper is most closely related to our study. They report an experiment on synonym extraction using bilingual resources (an English-Chinese dictionary and corpus) as well as monolingual resources (an English dictionary and corpus). Their monolingual corpus-based approach is very similar to our monolingual corpus-based approach. The bilingual approach is different from ours in several aspects. Firstly, they do not take the corpus as the starting point to retrieve word alignments, they use the bilingual dictionary to retrieve multiple translations for each target word. The corpus is only employed to assign probabilities to the translations found in the dictionary. Secondly, the authors use a parallel corpus that is bilingual whereas we use a multilingual corpus containing 11 languages in total. The authors show that the bilingual method outperforms the monolingual methods. However a combination of different methods leads to the best performance.

3 Methodology

3.1 Measuring Distributional Similarity

An increasingly popular method for acquiring semantically similar words is to extract distributionally similar words from large corpora. The underlying assumption of this approach is that semantically similar words are used in similar contexts. The contexts a given word is found in, be it a syntactic context or an alignment context, are used as the features in the vector for the given word, the so-called *context vector*. The vector contains frequency counts for each feature, i.e., the multiple contexts the word is found in.

Context vectors are compared with each other in order to calculate the distributional similarity between words. Several measures have been proposed. Curran and Moens (2002) report on a large-scale evaluation experiment, where they evaluated the performance of various commonly used methods. Van der Plas and Bouma (2005) present a similar experiment for Dutch, in which they tested most of the best performing measures according to Curran and Moens (2002). Pointwise Mutual Information (I) and *Dice*† performed best in their experiments. *Dice* is a well-known combinatorial measure that computes the ratio between the size of the intersection of two feature sets and the sum of the sizes of the individual feature sets. *Dice*† is a measure that incorporates weighted frequency counts.

$$Dice\dagger = \frac{2\sum_f min(I(W_1, f), I(W_2, f))}{\sum_f I(W_1, f) + I(W_2, f)}$$

,where f is the feature
W_1 and W_2 are the two words that are being compared,
and I is a weight assigned to the frequency counts.

3.2 Weighting

We will now explain why we use weighted frequencies and which formula we use for weighting.

The information value of a cell in a word vector (which lists how often a word occurred in a specific context) is not equal for all cells. We will explain this using an example from monolingual syntax-based distributional similarity. A large number of nouns can occur as the subject of the verb *have*, for instance, whereas only a few nouns may occur as the object of *squeeze*. Intuitively, the fact that two nouns both occur as subject of *have* tells us less about their semantic similarity than the fact that two nouns both occur as object of *squeeze*. To account for this intuition, the frequency of occurrence in a vector can be replaced by a weighted score. The weighted score is an indication of the amount of information carried by that particular combination of a noun and its feature.

We believe that this type of weighting is beneficial for calculating similarity between word alignment vectors as well. Word alignments that are shared by many different words are most probably mismatches.

For this experiment we used Pointwise Mutual Information (I) (Church and Hanks, 1989).

$$I(W,f) = log\frac{P(W,f)}{P(W)P(f)}$$

,where W is the target word

P(W) is the probability of seeing the word

P(f) is the probability of seeing the feature

P(W,f) is the probability of seeing the word and the feature together.

3.3 Word Alignment

The multilingual approach we are proposing relies on automatic word alignment of parallel corpora from Dutch to one or more target languages. This alignment is the basic input for the extraction of the *alignment context* as described in section 5.2.2. The alignment context is then used for measuring distributional similarity as introduced above.

For the word alignment, we apply standard techniques derived from statistical machine translation using the well-known IBM alignment models (Brown et al., 1993) implemented in the open-source tool GIZA++ (Och, 2003). These models can be used to find links between words in a source language and a target language given sentence aligned parallel corpora. We applied standard settings of the GIZA++ system without any optimisation for our particular input. We also used plain text only, i.e. we did not apply further preprocessing except tokenisation and sentence splitting. Additional linguistic processing such as lemmatisation and multi-word unit detection might help to improve the alignment but this is not part of the present study.

The alignment models produced are asymmetric and several heuristics exist to combine directional word alignments to improve alignment accuracy. We believe, that precision is more crucial than recall in our approach and, therefore, we apply a very strict heuristics namely we compute the intersection of word-to-word links retrieved by GIZA++. As a result we obtain partially word-aligned parallel corpora from which translational context vectors are built (see section 5.2.2). Note, that the intersection heuristics allows one-to-one word links only. This is reasonable for the Dutch part as we are only interested in single words and their synonyms. However, the distributional context of these words defined by their alignments is strongly influenced by this heuristics. Problems caused by this procedure will be discussed in detail in section 7 of our experiments.

4 Evaluation Framework

In the following, we describe the data used and measures applied.

The evaluation method that is most suitable for testing with multiple settings is one that uses an available resource for synonyms as a gold standard. In our experiments we apply automatic evaluation using an existing hand-crafted synonym database, Dutch EuroWordnet (EWN, Vossen (1998)).

In EWN, one synset consists of several synonyms which represent a single sense. Polysemous words occur in several synsets. We have combined for each target word the EWN synsets in which it occurs. Hence, our gold standard consists of a list of all nouns found in EWN and their corresponding synonyms extracted by taking the union of all synsets for each word. Precision is then calculated as the percentage of candidate synonyms that are truly synonyms according to our gold standard. Recall is the percentage of the synonyms according to EWN that are indeed found by the system. We have extracted randomly from all synsets in EWN 1000 words with a frequency

above 4 for which the systems under comparison produce output.

The drawback of using such a resource is that coverage is often a problem. Not all words that our system proposes as synonyms can be found in Dutch EWN. Words that are not found in EWN are discarded.[2]. Moreover, EWN's synsets are not exhaustive. After looking at the output of our best performing system we were under the impression that many correct synonyms selected by our system were classified as incorrect by EWN. For this reason we decided to run a human evaluation over a sample of 100 candidate synonyms classified as incorrect by EWN.

5 Experimental Setup

In this section we will describe results from the two synonym extraction approaches based on distributional similarity: one using syntactic context and one using translational context based on word alignment and the combination of both. For both approaches, we used a cutoff n for each row in our word-by-context matrix. A word is discarded if the row marginal is less than n. This means that each word should be found in any context at least n times else it will be discarded. We refer to this by the term *minimum row frequency*. The cutoff is used to make the feature space manageable and to reduce noise in the data. [3]

5.1 Distributional Similarity Based on Syntactic Relations

This section contains the description of the synonym extraction approach based on distributional similarity and syntactic relations. Feature vectors for this approach are constructed from syntactically parsed monolingual corpora. Below we describe the data and resources used, the nature of the context applied and the results of the synonym extraction task.

5.1.1 Data and Resources

As our data we used the Dutch CLEF QA corpus, which consists of 78 million words of Dutch newspaper text (Algemeen Dagblad and NRC Handelsblad 1994/1995). The corpus was parsed automatically using the Alpino parser (van der Beek et al., 2002; Malouf and van Noord, 2004). The result of parsing a sentence is a dependency graph according to the guidelines of the Corpus of Spoken Dutch (Moortgat et al., 2000).

subject-verb	*cat_eat*
verb-object	*feed_cat*
adjective-noun	*black_cat*
coordination	*cat_dog*
apposition	*cat_Garfield*
prep. complement	*go+to_work*

Table 1: Types of dependency relations extracted

grammatical relation	# pairs
subject	507K
object	240K
adjective	289K
coordination	400 K
apposition	109K
prep. complement	84K
total	1629K

Table 2: Number of word-syntactic-relation pairs (types) per dependency relation with frequency $>$ 1.

5.1.2 Syntactic Context

We have used several grammatical relations: subect, object, adjective, coordination, apposition and prepositional complement. Examples are given in table 1. Details on the extraction can be found in van der Plas and Bouma (2005). The number of pairs (types) consisting of a word and a syntactic relation found are given in table 2. We have discarded pairs that occur less than 2 times.

5.2 Distributional Similarity Based on Word Alignment

The alignment approach to synonym extraction is based on automatic word alignment. Context vectors are built from the alignments found in a parallel corpus. Each aligned word type is a feature in the vector of the target word under consideration. The alignment frequencies are used for weighting the features and for applying the frequency cutoff. In the following section we describe the data and resources used in our experiments and finally the results of this approach.

[2]Note that we use the part of EWN that contains only nouns

[3]We have determined the optimum in F-score for the alignment-based method, the syntax-based method and the combination independently by using a development set of 1000 words that has no overlap with the test set used in evaluation. The minimum row frequency was set to 2 for all alignment-based methods. It was set to 46 for the syntax-based method and the combination of the two methods.

5.2.1 Data and Resources

Measures of distributional similarity usually require large amounts of data. For the alignment method we need a parallel corpus of reasonable size with Dutch either as source or as target language. Furthermore, we would like to experiment with various languages aligned to Dutch. The freely available Europarl corpus (Koehn, 2003) includes 11 languages in parallel, it is sentence aligned, and it is of reasonable size. Thus, for acquiring Dutch synonyms we have 10 language pairs with Dutch as the source language. The Dutch part includes about 29 million tokens in about 1.2 million sentences. The entire corpus is sentence aligned (Tiedemann and Nygaard, 2004) which is a requirement for the automatic word alignment described below.

5.2.2 Alignment Context

Context vectors are populated with the links to words in other languages extracted from automatic word alignment. We applied GIZA++ and the intersection heuristics as explained in section . From the word aligned corpora we extracted *word type links*, pairs of source and target words with their alignment frequency attached. Each aligned target word type is a feature in the (translational) context of the source word under consideration.

Note that we rely entirely on automatic processing of our data. Thus, results from the automatic word alignments include errors and their precision and recall is very different for the various language pairs. However, we did not assess the quality of the alignment itself which would be beyond the scope of this paper.

As mentioned earlier, we did not include any linguistic pre-processing prior to the word alignment. However, we post-processed the alignment results in various ways. We applied a simple lemmatizer to the list of bilingual word type links in order to 1) reduce data sparseness, and 2) to facilitate our evaluation based on comparing our results to existing synonym databases. For this we used two resources: CELEX – a linguistically annotated dictionary of English, Dutch and German (Baayen et al., 1993), and the Dutch snowball stemmer implementing a suffix stripping algorithm based on the Porter stemmer. Note that lemmatization is only done for Dutch. Furthermore, we removed word type links that include non-alphabetic characters to focus our investigations on 'real words'. In order to reduce alignment noise, we also applied a frequency threshold to remove alignments that occur only once. Finally, we restricted our study to Dutch nouns. Hence, we extracted word type links for all words tagged as noun in CELEX. We also included words which are not found at all in CELEX assuming that most of them will be productive noun constructions.

From the remaining word type links we populated the context vectors as described earlier. Table 3 shows the number of context elements extracted in this manner for each language pair considered from the Europarl corpus[4]

	#word-transl. pairs		#word-transl. pairs
DA	104K	FR	90K
DE	133K	IT	96K
EL	60K	PT	86K
EN	119K	SV	97K
ES	119K	ALL	994K
FI	89K		

Table 3: Number of word-translation pairs for different languages with alignment frequency > 1

6 Results and Discussion

Table 4 shows the precision recall en F-score for the different methods. The first 10 rows refer to the results for all language pairs individually. The 11th row corresponds to the setting in which all alignments for all languages are combined. The penultimate row shows results for the syntax-based method and the last row the combination of the syntax-based and alignment-based method.

Judging from the precision, recall and F-score in table 4 Swedish is the best performing language for Dutch synonym extraction from parallel corpora. It seems that languages that are similar to the target language, for example in word order, are good candidates for finding synonyms at high precision rates. Also the fact that Dutch and Swedish both have one-word compounds avoids mistakes that are often found with the other languages. However, judging from recall (and F-score) French is not a bad candidate either. It is possible that languages that are lexically different from the target language provide more synonyms. The fact that Finnish and Greek do not gain high scores might be due to the fact that there are only a limited amount of translational contexts (with a frequency > 1) available for these language (as is shown in table 3). The reasons are twofold.

[4] abbreviations taken from the ISO-639 2-letter codes

	# candidate synonyms								
	1			2			3		
	Prec	Rec	F-sc	Prec	Rec	F-sc	Prec	Rec	F-sc
DA	19.8	5.1	8.1	15.5	7.6	10.2	13.3	9.4	11.0
DE	21.2	5.4	8.6	16.1	7.9	10.6	13.1	9.3	10.9
EL	18.2	4.5	7.2	14.0	6.5	8.9	11.8	7.9	9.4
EN	19.5	5.3	8.3	14.7	7.8	10.2	12.4	9.7	10.9
ES	18.4	5.0	7.9	14.7	7.8	10.2	12.1	9.4	10.6
FI	18.0	3.9	6.5	14.3	5.6	8.1	12.1	6.5	8.5
FR	20.3	5.5	8.7	15.8	8.3	10.9	13.0	10.1	11.4
IT	18.7	4.9	7.8	14.7	7.5	9.9	12.3	9.2	10.5
PT	17.7	4.8	7.6	14.0	7.4	9.7	11.6	8.9	10.1
SV	22.3	5.6	9.0	16.4	7.9	10.7	13.3	9.3	10.9
ALL	**22.5**	**6.4**	**10.0**	**16.6**	**9.4**	**12.0**	**13.7**	**11.5**	**12.5**
SYN	8.8	2.5	3.9	6.9	4.0	5.09	5.9	5.1	5.5
COMBI	19.9	5.8	8.9	14.5	8.4	10.6	11.7	10.1	10.9

Table 4: Precision, recall and F-score (%) at increasing number of candidate synonyms

Firstly, for Greek and Finnish the Europarl corpus contains less data. Secondly, the fact that Finnish is a language that has a lot of cases for nouns, might lead to data sparseness and worse accuracy in word alignment.

The results in table 4 also show the difference in performance between the multilingual alignment-method and the syntax-based method. The monolingual alignment-based method outperforms the syntax-based method by far. The syntax-based method that does not rely on scarce multilingual resources is more portable and also in this experiment it makes use of more data. However, the low precision scores of this method are not convincing. Combining both methods does not result in better performance for finding synonyms. This is in contrast with the results reported by Wu and Zhou (2003). This might well be due to the more sophisticated method they use for combining different methods, which is a weighted combination.

The precision scores are in line with the scores reported by Wu and Zhou (2003) in a similar experiment discussed under related work. The recall we attain however is more than three times higher. These differences can be due to differences between their approach such as starting from a bilingual dictionary for acquiring the translational context versus using automatic word alignments from a large multilingual corpus directly. Furthermore, the different evaluation methods used make comparison between the two approaches difficult. They use a combination of the English WordNet (Fellbaum, 1998) and Roget thesaurus (Roget, 1911) as a gold standard in their evaluations. It is obvious that a combination of these resources leads to larger sets of synonyms. This could explain the relatively low recall scores. It does however not explain the similar precision scores.

We conducted a human evaluation on a sample of 100 candidate synonyms proposed by our best performing system that were classified as incorrect by EWN. Ten evaluators (authors excluded) were asked to classify the pairs of words as synonyms or non-synonyms using a web form of the format *yes/no/don't know*. For 10 out of the 100 pairs all ten evaluators agreed that these were synonyms. For 37 of the 100 pairs more than half of the evaluators agreed that these were synonyms. We can conclude from this that the scores provided in our evaluations based on EWN (table 4) are too pessimistic. We believe that the actual precision scores lie 10 to 37 % higher than the 22.5 % reported in table 4. Over and above, this indicates that we are able to extract automatically synonyms that are not yet covered by available resources.

7 Error Analysis

In table 5 some example output is given for the method combining word alignments of all 10 foreign languages as opposed to the monolingual syntax-based method. These examples illustrate the general patterns that we discovered by looking into the results for the different methods.

The first two examples show that the syntax-

	ALIGN(ALL)	SYNTAX
consensus *consensus*	eensgezindheid *consensus*	evenwicht *equilibrium*
herfst *autumn*	najaar *autumn*	winter *winter*
eind *end*	einde *end*	begin *beginning*
armoede *poverty*	armoedebestrijding *poverty reduction*	werkloosheid *unemployment*
alcohol *alcohol*	alcoholgebruik *alcohol consumption*	drank *liquor*
bes *berry*	charme *charm*	perzik *peach*
definitie *definition*	definie *define+incor.stemm.*	criterium *criterion*
verlamming *paralysis*	lam *paralysed*	verstoring *disturbance*

Table 5: Example candidate synonyms at 1st rank and their translations in italics

based method often finds semantically related words whereas the alignment-based method finds synonyms. The reasons for this are quite obvious. Synonyms are likely to receive identical translations, words that are only semantically related are not. A translator would not often translate *auto* (*car*) with *vrachtwagen* (*truck*). However, the two words are likely to show up in identical syntactic relations, such as being the object of *drive* or appearing in coordination with *motorcycle*.

Another observation that we made is that the syntax-based method often finds antonyms such as *begin* (*beginning*) for the word *einde* (*end*). Explanations for this are in line with what we said about the semantically related words: Synonyms are likely to receive identical translations, antonyms are not but they do appear in similar syntactic contexts.

Compounds pose a problem for the alignment-method. We have chosen intersection as alignment method. It is well-known that this method cannot cope very well with the alignment of compounds because it only allows one-to-one word links. Dutch uses many one-word compounds that should be linked to multi-word counterparts in other languages. However, using intersection we obtain only partially correct alignments and this causes many mistakes in the distributional similarity algorithm. We have given some examples in rows 4 and 5 of table 5.

We have used the distributional similarity score only for ranking the candidate synonyms. In some cases it seems that we should have used it to set a threshold such as in the case of *berry* and *charm*. These two words share one translational context : the article *el* in Spanish. The distributional similarity score in such cases is often very low. We could have filtered some of these mistakes by setting a threshold.

One last observation is that the alignment-based method suffers from incorrect stemming and the lack of sufficient part-of-speech information. We have removed all context vectors that were built for a word that was registered in CELEX with a PoS-tag different from 'noun'. But some words are not found in CELEX and although they are not of the word type 'noun' their context vectors remain in our data. They are stemmed using the snowball stemmer. The candidate synonym *definie* is a corrupted verbform that is not found in CELEX. *Lam* is ambiguous between the noun reading that can be translated in English with *lamb* and the adjective *lam* which can be translated with *paralysed*. This adjective is related to the word *verlamming* (*paralysis*), but would have been removed if the word was correctly PoS-tagged.

8 Conclusions

Parallel corpora are mostly used for tasks related to WSD. This paper shows that multilingual word alignments can be applied to acquire synonyms automatically without the need for resources such as bilingual dictionaries. A comparison with a monolingual syntax-based method shows that the alignment-based method is able to extract synonyms with much greater precision and recall. A human evaluation shows that the synonyms the alignment-based method finds are often missing in EWN. This leads us to believe that the precision scores attained by using EWN as a gold standard are too pessimistic. Furthermore it is good news that we seem to be able to find synonyms that are not yet covered by existing resources.

The precision scores are still not satisfactory and we see plenty of future directions. We would like to use linguistic processing such as PoS-tagging for word alignment to increase the accuracy of the alignment itself, to deal with compounds more effectively and to be able to filter out proposed synonyms that are of a different word class than the target word. Furthermore we would like to make use of the distributional similarity score to set a threshold that will remove a lot of errors. The last thing that remains for future work is to find a more adequate way to combine the

syntax-based and the alignment-based methods.

Acknowledgements

This research was carried out in the project *Question Answering using Dependency Relations*, which is part of the research program for *Interact ive Multimedia Information Extraction*, IMIX, financed by NWO, the Dutch Organisation for Scientific Research.

References

R.H. Baayen, R. Piepenbrock, and H. van Rijn. 1993. The CELEX lexical database (CD-ROM). Linguistic Data Consortium, University of Pennsylvania,Philadelphia.

Regina Barzilay and Kathleen McKeown. 2001. Extracting paraphrases from a parallel corpus. In *Meeting of the Association for Computational Linguistics*, pages 50–57.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–296.

K.W. Church and P. Hanks. 1989. Word association norms, mutual information and lexicography. *Proceedings of the 27th annual conference of the Association of Computational Linguistics*, pages 76–82.

J.R. Curran and M. Moens. 2002. Improvements in automatic thesaurus extraction. In *Proceedings of the Workshop on Unsupervised Lexical Acquisition*, pages 59–67.

Ido Dagan, Alon Itai, and Ulrike Schwall. 1991. Two languages are more informative than one. In *Meeting of the Association for Computational Linguistics*, pages 130–137.

Helge Dyvik. 1998. Translations as semantic mirrors. In *Proceedings of Workshop Multilinguality in the Lexicon II, ECAI 98, Brighton, UK*, pages 24–44.

C. Fellbaum. 1998. Wordnet, an electronic lexical database. MIT Press.

A. Ibrahim, B. Katz, and J. Lin. 2003. Extracting structural paraphrases from aligned monolingual corpora.

A. Kilgarriff and C. Yallop. 2000. What's in a thesaurus? In *Proceedings of the Second Conference on Language Resource an Evaluation*, pages 1371–1379.

Philipp Koehn. 2003. Europarl: A multilingual corpus for evaluation of machine translation. unpublished draft, available from http://people.csail.mit.edu/koehn/publications/europarl/.

Dekang Lin, Shaojun Zhao, Lijuan Qin, and Ming Zhou. 2003. Identifying synonyms among distributionally similar words. In *IJCAI*, pages 1492–1493.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *COLING-ACL*, pages 768–774.

Robert Malouf and Gertjan van Noord. 2004. Wide coverage parsing with stochastic attribute value grammars. In *IJCNLP-04 Workshop Beyond Shallow Analyses - Formalisms and stati stical modeling for deep analyses*, Hainan.

Michael Moortgat, Ineke Schuurman, and Ton van der Wouden. 2000. CGN syntactische annotatie. Internal Project Report Corpus Gesproken Nederlands, see http://lands. let.kun.nl/cgn.

Franz Josef Och. 2003. GIZA++: Training of statistical translation models. Available from http://www.isi.edu/~och/GIZA++.html.

P. Roget. 1911. Thesaurus of English words and phrases.

Jörg Tiedemann and Lars Nygaard. 2004. The OPUS corpus - parallel & free. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal.

Peter D. Turney. 2001. Mining the Web for synonyms: PMI–IR versus LSA on TOEFL. *Lecture Notes in Computer Science*, 2167:491–502.

Leonoor van der Beek, Gosse Bouma, and Gertjan van Noord. 2002. Een brede computationele grammatica voor het Nederlands. *Nederlandse Taalkunde*, 7(4):353–374.

Lonneke van der Plas and Gosse Bouma. 2005. Syntactic contexts for finding semantically similar words. Proceedings of the Meeting of Computational Linguistics in the Netherlands (CLIN).

P. Vossen. 1998. Eurowordnet a multilingual database with lexical semantic networks.

Hua Wu and Ming Zhou. 2003. Optimizing synonym extraction using monolingual and bilingual resources. In *Proceedings of the Second International Workshop on Paraphrasing: Paraphrase Acquisition and Applications (IWP2003)*, Sapporo, Japan.

Word Alignment for Languages with Scarce Resources Using Bilingual Corpora of Other Language Pairs

Haifeng Wang Hua Wu Zhanyi Liu
Toshiba (China) Research and Development Center
5/F., Tower W2, Oriental Plaza, No.1, East Chang An Ave., Dong Cheng District
Beijing, 100738, China
{wanghaifeng, wuhua, liuzhanyi}@rdc.toshiba.com.cn

Abstract

This paper proposes an approach to improve word alignment for languages with scarce resources using bilingual corpora of other language pairs. To perform word alignment between languages L1 and L2, we introduce a third language L3. Although only small amounts of bilingual data are available for the desired language pair L1-L2, large-scale bilingual corpora in L1-L3 and L2-L3 are available. Based on these two additional corpora and with L3 as the pivot language, we build a word alignment model for L1 and L2. This approach can build a word alignment model for two languages even if no bilingual corpus is available in this language pair. In addition, we build another word alignment model for L1 and L2 using the small L1-L2 bilingual corpus. Then we interpolate the above two models to further improve word alignment between L1 and L2. Experimental results indicate a relative error rate reduction of 21.30% as compared with the method only using the small bilingual corpus in L1 and L2.

1 Introduction

Word alignment was first proposed as an intermediate result of statistical machine translation (Brown et al., 1993). Many researchers build alignment links with bilingual corpora (Wu, 1997; Och and Ney, 2003; Cherry and Lin, 2003; Zhang and Gildea, 2005). In order to achieve satisfactory results, all of these methods require a large-scale bilingual corpus for training. When the large-scale bilingual corpus is unavailable, some researchers acquired class-based alignment rules with existing dictionaries to improve word alignment (Ker and Chang, 1997). Wu et al. (2005) used a large-scale bilingual corpus in general domain to improve domain-specific word alignment when only a small-scale domain-specific bilingual corpus is available.

This paper proposes an approach to improve word alignment for languages with scarce resources using bilingual corpora of other language pairs. To perform word alignment between languages L1 and L2, we introduce a third language L3 as the pivot language. Although only small amounts of bilingual data are available for the desired language pair L1-L2, large-scale bilingual corpora in L1-L3 and L2-L3 are available. Using these two additional bilingual corpora, we train two word alignment models for language pairs L1-L3 and L2-L3, respectively. And then, with L3 as a pivot language, we can build a word alignment model for L1 and L2 based on the above two models. Here, we call this model an *induced model*. With this induced model, we perform word alignment between languages L1 and L2 even if no parallel corpus is available for this language pair. In addition, using the small bilingual corpus in L1 and L2, we train another word alignment model for this language pair. Here, we call this model an *original model*. An *interpolated model* can be built by interpolating the induced model and the original model.

As a case study, this paper uses English as the pivot language to improve word alignment between Chinese and Japanese. Experimental results show that the induced model performs better than the original model trained on the small Chinese-Japanese corpus. And the interpolated model further improves the word alignment results, achieving a relative error rate reduction of

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 874–881,
Sydney, July 2006.

21.30% as compared with results produced by the original model.

The remainder of this paper is organized as follows. Section 2 discusses the related work. Section 3 introduces the statistical word alignment models. Section 4 describes the parameter estimation method using bilingual corpora of other language pairs. Section 5 presents the interpolation model. Section 6 reports the experimental results. Finally, we conclude and present the future work in section 7.

2 Related Work

A shared task on word alignment was organized as part of the ACL 2005 Workshop on Building and Using Parallel Texts (Martin et al., 2005). The focus of the task was on languages with scarce resources. Two different subtasks were defined: *Limited resources* and *Unlimited resources*. The former subtask only allows participating systems to use the resources provided. The latter subtask allows participating systems to use any resources in addition to those provided.

For the subtask of unlimited resources, Aswani and Gaizauskas (2005) used a multi-feature approach for many-to-many word alignment on English-Hindi parallel corpora. This approach performed local word grouping on Hindi sentences and used other methods such as dictionary lookup, transliteration similarity, expected English words, and nearest aligned neighbors. Martin et al. (2005) reported that this method resulted in absolute improvements of up to 20% as compared with the case of only using limited resources. Tufis et al. (2005) combined two word aligners: one is based on the limited resources and the other is based on the unlimited resources. The unlimited resource consists of a translation dictionary extracted from the alignment of Romanian and English WordNet. Lopez and Resnik (2005) extended the HMM model by integrating a tree distortion model based on a dependency parser built on the English side of the parallel corpus. The latter two methods produced comparable results with those methods using limited resources. All the above three methods use some language dependent resources such as dictionary, thesaurus, and dependency parser. And some methods, such as transliteration similarity, can only be used for very similar language pairs.

In this paper, besides the limited resources for the given language pair, we make use of large amounts of resources available for other language pairs to address the alignment problem for languages with scarce resources. Our method does not need language-dependent resources or deep linguistic processing. Thus, it is easy to adapt to any language pair where a pivot language and corresponding large-scale bilingual corpora are available.

3 Statistical Word Alignment

According to the IBM models (Brown et al., 1993), the statistical word alignment model can be generally represented as in equation (1).

$$\Pr(\mathbf{a},\mathbf{f}\mid\mathbf{c}) = \frac{\Pr(\mathbf{a},\mathbf{f}\mid\mathbf{c})}{\sum_{\mathbf{a}'}\Pr(\mathbf{a}',\mathbf{f}\mid\mathbf{c})} \tag{1}$$

Where, **c** and **f** represent the source sentence and the target sentence, respectively[1].

In this paper, we use a simplified IBM model 4 (Al-Onaizan et al., 1999), which is shown in equation (2). This version does not take into account word classes in Brown et al. (1993).

$$\begin{aligned}\Pr(\mathbf{a},\mathbf{f}\mid\mathbf{c}) &= \binom{m-\phi_0}{\phi_0} p_0^{m-2\phi_0} p_1^{\phi_0} \cdot \\ &\prod_{i=1}^{l} n(\phi_i \mid c_i) \cdot \prod_{j=1}^{m} t(f_j \mid c_{a_j}) \cdot \\ &\Big(\prod_{j=1,a_j\neq 0}^{m} ([j = h(a_j)] \cdot d_1(j - \odot_{i-1})) + \\ &\prod_{j=1,a_j\neq 0}^{m} ([j \neq h(a_j)] \cdot d_{>1}(j - p(j)))\Big)\end{aligned} \tag{2}$$

l, m are the lengths of the source sentence and the target sentence respectively.

j is the position index of the target word.

a_j is the position of the source word aligned to the j^{th} target word.

ϕ_i is the fertility of c_i.

p_0, p_1 are the fertility probabilities for c_0, and $p_0 + p_1 = 1$.

$t(f_j \mid c_{a_j})$ is the word translation probability.

$n(\phi_i \mid c_i)$ is the fertility probability.

$d_1(j - \odot_{i-1})$ is the distortion probability for the head word of the cept.

$d_{>1}(j - p(j))$ is the distortion probability for the non-head words of the cept.

[1] This paper uses **c** and **f** to represent a Chinese sentence and a Japanese sentence, respectively. And **e** represents an English sentence.

$h(i) = \min_k \{k : i = a_k\}$ is the head of cept *i*.

$p(j) = \max_{k<j} \{k : a_j = a_k\}$.

$\odot_i$ is the center of cept *i*.

During the training process, IBM model 3 is first trained, and then the parameters in model 3 are employed to train model 4. For convenience, we describe model 3 in equation (3). The main difference between model 3 and model 4 lies in the calculation of distortion probability.

$$\Pr(\mathbf{a},\mathbf{f} \mid \mathbf{c}) = \binom{m-\phi_0}{\phi_0} p_0^{m-2\phi_0} p_1^{\phi_0} \cdot \prod_{i=1}^{l} n(\phi_i \mid c_i) \cdot \prod_{i=1}^{l} \phi_i! \cdot \prod_{j=1}^{m} t(f_j \mid c_{a_j}) \cdot \prod_{j=1, a_j \neq 0}^{m} d(j \mid a_j, l, m) \quad (3)$$

4 Parameter Estimation Using Bilingual Corpora of Other Language Pairs

As shown in section 3, the word alignment model mainly has three kinds of parameters that must be specified, including the translation probability, the fertility probability, and the distortion probability. The parameters are usually estimated by using bilingual sentence pairs in the desired languages, namely Chinese and Japanese here. In this section, we describe how to estimate the parameters without using the Chinese-Japanese bilingual corpus. We introduce English as the pivot language, and use the Chinese-English and English-Japanese bilingual corpora to estimate the parameters of the Chinese-Japanese word alignment model. With these two corpora, we first build Chinese-English and English-Japanese word alignment models as described in section 3. Then, based on these two models, we estimate the parameters of Chinese-Japanese word alignment model. The estimated model is named *induced model*.

The following subsections describe the method to estimate the parameters of Chinese-Japanese alignment model. For reversed Japanese-Chinese word alignment, the parameters can be estimated with the same method.

4.1 Translation Probability

Basic Translation Probability

We use the translation probabilities trained with Chinese-English and English-Japanese corpora to estimate the Chinese-Japanese probability as shown in equation (4). In (4), we assume that the translation probability $t_{\mathrm{EJ}}(f_j \mid e_k, c_i)$ is independent of the Chinese word c_i .

$$\begin{aligned} t_{\mathrm{CJ}}(f_j \mid c_i) &= \sum_{e_k} t_{\mathrm{EJ}}(f_j \mid e_k, c_i) \cdot t_{\mathrm{CE}}(e_k \mid c_i) \\ &= \sum_{e_k} t_{\mathrm{EJ}}(f_j \mid e_k) \cdot t_{\mathrm{CE}}(e_k \mid c_i) \end{aligned} \quad (4)$$

Where $t_{\mathrm{CJ}}(f_j \mid c_i)$ is the translation probability for Chinese-Japanese word alignment. $t_{\mathrm{EJ}}(f_j \mid e_k)$ is the translation probability trained using the English-Japanese corpus. $t_{\mathrm{CE}}(e_k \mid c_i)$ is the translation probability trained using the Chinese-English corpus.

Cross-Language Word Similarity

In any language, there are ambiguous words with more than one sense. Thus, some noise may be introduced by the ambiguous English word when we estimate the Chinese-Japanese translation probability using English as the pivot language. For example, the English word "bank" has at least two senses, namely:

bank1 - a financial organization
bank2 - the border of a river

Let us consider the Chinese word:

河岸 - bank2 (the border of a river)

And the Japanese word:

銀行 - bank1 (a financial organization)

In the Chinese-English corpus, there is high probability that the Chinese word "河岸(bank2)" would be translated into the English word "bank". And in the English-Japanese corpus, there is also high probability that the English word "bank" would be translated into the Japanese word "銀行(bank1)".

As a result, when we estimate the translation probability using equation (4), the translation probability of "銀行(bank1)" given "河岸(bank2)" is high. Such a result is not what we expect.

In order to alleviate this problem, we introduce cross-language word similarity to improve translation probability estimation in equation (4). The cross-language word similarity describes how likely a Chinese word is to be translated into a Japanese word with an English word as the pivot. We make use of both the Chinese-English corpus and the English-Japanese corpus to calculate the cross language word similarity between a Chinese word *c* and a Japanese word *f* given an

Input: An English word e, a Chinese word c, and a Japanese word f;
The Chinese-English corpus; The English-Japanese corpus.

(1) Construct Set 1: identify those Chinese-English sentence pairs that include the given Chinese word c and English word e, and put the English sentences in the pairs into Set 1.

(2) Construct Set 2: identify those English-Japanese sentence pairs that include the given English word e and Japanese word f, and put the English sentences in the pairs into Set 2.

(3) Construct the feature vectors V_{CE} and V_{EJ} of the given English word using all other words as context in Set 1 and Set 2, respectively.

$V_{CE} =< (e_1, ct_{11}), (e_2, ct_{12}), ..., (e_n, ct_{1n}) >$

$V_{EJ} =< (e_1, ct_{21}), (e_2, ct_{22}), ..., (e_n, ct_{2n}) >$

Where ct_{ij} is the frequency of the context word e_j. $ct_{ij} = 0$ if e_j does not occur in Set i.

(4) Given the English word e, calculate the cross-language word similarity between the Chinese word c and the Japanese word f as in equation (5)

$$sim(c, f; e) = \cos(V_{CE}, V_{EJ}) = \frac{\sum_j ct_{1j} \cdot ct_{2j}}{\sqrt{\sum_j (ct_{1j})^2} \cdot \sqrt{\sum_j (ct_{2j})^2}} \quad (5)$$

Output: The cross language word similarity $sim(c, f; e)$ of the Chinese word c and the Japanese word f given the English word e

Figure 1. Similarity Calculation

English word *e*. For the ambiguous English word *e*, both the Chinese word *c* and the Japanese word *f* can be translated into *e*. The sense of an instance of the ambiguous English word *e* can be determined by the context in which the instance appears. Thus, the cross-language word similarity between the Chinese word *c* and the Japanese word *f* can be calculated according to the contexts of their English translation *e*. We use the feature vector constructed using the context words in the English sentence to represent the context. So we can calculate the cross-language word similarity using the feature vectors. The detailed algorithm is shown in figure 1. This idea is similar to translation lexicon extraction via a bridge language (Schafer and Yarowsky, 2002).

For example, the Chinese word "河岸" and its English translation "bank" (the border of a river) appears in the following Chinese-English sentence pair:

(a) *他们沿着河岸走回家。*

(b) *They walked home along the river bank.*

The Japanese word "銀行" and its English translation "bank" (a financial organization) appears in the following English-Japanese sentence pair:

(c) *He has plenty of money in the bank.*

(d) *彼は銀行預金が相当ある。*

The context words of the English word "bank" in sentences (b) and (c) are quite different. The difference indicates the cross language word similarity of the Chinese word "河岸" and the Japanese word "銀行" is low. So they tend to have different senses.

Translation Probability Embedded with Cross Language Word Similarity

Based on the cross language word similarity calculation in equation (5), we re-estimate the translation probability as shown in (6). Then we normalize it in equation (7).

The word similarity of the Chinese word "河岸 (bank2)" and the Japanese word "銀行 (bank1)" given the word English word "bank" is low. Thus, using the updated estimation method, the translation probability of "銀行 (bank1)" given "河岸(bank2)" becomes low.

$$t'_{CJ}(f_j \mid c_i) = \sum_{e_k} (t_{EJ}(f_j \mid e_k) \cdot t_{CE}(e_k \mid c_i) \cdot sim(c_i, f_j; e_k)) \quad (6)$$

$$t_{CJ}(f_j \mid c_i) = \frac{t'_{CJ}(f_j \mid c_i)}{\sum_{f'} t'_{CJ}(f' \mid c_i)} \quad (7)$$

4.2 Fertility Probability

The induced fertility probability is calculated as shown in (8). Here, we assume that the probabil-

ity $n_{\mathrm{EJ}}(\phi_i \mid e_k, c_i)$ is independent of the Chinese word c_i.

$$\begin{aligned} & n_{\mathrm{CJ}}(\phi_i \mid c_i) \\ &= \sum_{e_k} n_{\mathrm{EJ}}(\phi_i \mid e_k, c_i) \cdot t_{\mathrm{CE}}(e_k \mid c_i) \\ &= \sum_{e_k} n_{\mathrm{EJ}}(\phi_i \mid e_k) \cdot t_{\mathrm{CE}}(e_k \mid c_i) \end{aligned} \tag{8}$$

Where, $n_{\mathrm{CJ}}(\phi_i \mid c_i)$ is the fertility probability for the Chinese-Japanese alignment. $n_{\mathrm{EJ}}(\phi_i \mid e_k)$ is the trained fertility probability for the English-Japanese alignment.

4.3 Distortion Probability in Model 3

With the English language as a pivot language, we calculate the distortion probability of model 3. For this probability, we introduce two additional parameters: one is the position of English word and the other is the length of English sentence. The distortion probability is estimated as shown in (9).

$$\begin{aligned} & d_{\mathrm{CJ}}(j \mid i, l, m) \\ &= \sum_{k,n} \Pr(j, k, n \mid i, l, m) \\ &= \sum_{k,n} \Pr(j \mid k, n, i, l, m) \cdot \Pr(k, n \mid i, l, m) \\ &= \sum_{k,n} (\Pr(j \mid k, n, i, l, m) \cdot \\ &\qquad \Pr(k \mid n, i, l, m) \cdot \Pr(n \mid i, l, m)) \end{aligned} \tag{9}$$

Where, $d_{\mathrm{CJ}}(j \mid i, l.m)$ is the estimated distortion probability. k is the introduced position of an English word. n is the introduced length of an English sentence.

In the above equation, we assume that the position probability $\Pr(j \mid k, n, i, l, m)$ is independent of the position of the Chinese word and the length of the Chinese sentence. And we assume that the position probability $\Pr(k \mid n, i, l, m)$ is independent of the length of Japanese sentence. Thus, we rewrite these two probabilities as follows.

$$\Pr(j \mid k, n, i, l, m) \approx \Pr(j \mid k, n, m) = d_{\mathrm{EJ}}(j \mid k, n, m)$$

$$\Pr(k \mid i, l, m, n) \approx \Pr(k \mid i, l, n) = d_{\mathrm{CE}}(k \mid i, l, n)$$

For the length probability, the English sentence length n is independent of the word positions i. And we assume that it is uniformly distributed. Thus, we take it as a constant, and rewrite it as follows.

$$\Pr(n \mid i, l, m) = \Pr(n \mid l, m) = \text{constant}$$

According to the above three assumptions, we ignore the length probability $\Pr(n \mid l, m)$. Equation (9) is rewritten in (10).

$$\begin{aligned} & d_{\mathrm{CJ}}(j \mid i, l.m) \\ &= \sum_{k,n} d_{\mathrm{EJ}}(j \mid k, n, m) \cdot d_{\mathrm{CE}}(k \mid i, l, n) \end{aligned} \tag{10}$$

4.4 Distortion Probability in Model 4

In model 4, there are two parameters for the distortion probability: one for head words and the other for non-head words.

Distortion Probability for Head Words

The distortion probability $d_1(j - \odot_{i-1})$ for head words represents the relative position of the head word of the i^{th} cept and the center of the $(i\text{-}1)^{\text{th}}$ cept. Let $\Delta j = j - \odot_{i-1}$, then Δj is independent of the absolute position. Thus, we estimate the distortion probability by introducing another relative position $\Delta j'$ of English words, which is shown in (11).

$$\begin{aligned} & d_{1,\mathrm{CJ}}(\Delta j = j - \odot_{i-1}) \\ &= \sum_{\Delta j'} d_{1,\mathrm{CE}}(\Delta j') \cdot \Pr_{\mathrm{EJ}}(\Delta j \mid \Delta j') \end{aligned} \tag{11}$$

Where, $d_{1,\mathrm{CJ}}(\Delta j = j - \odot_{i-1})$ is the estimated distortion probability for head words in Chinese-Japanese alignment. $d_{1,\mathrm{CE}}(\Delta j')$ is the distortion probability for head word in Chinese-English alignment. $\Pr_{\mathrm{EJ}}(\Delta j \mid \Delta j')$ is the translation probability of relative Japanese position given relative English position.

In order to simplify $\Pr_{\mathrm{EJ}}(\Delta j \mid \Delta j')$, we introduce j' and $\odot_{i'-1}$ and let $\Delta j' = j' - \odot_{i'-1}$, where j' and $\odot_{i'-1}$ are positions of English words. We rewrite $\Pr_{\mathrm{EJ}}(\Delta j \mid \Delta j')$ in (12).

$$\begin{aligned} & \Pr_{\mathrm{EJ}}(\Delta j \mid \Delta j') \\ &= \Pr_{\mathrm{EJ}}(j - \odot_{i-1} \mid j' - \odot_{i'-1}) \\ &= \sum_{\substack{j, \odot_{i-1}: j - \odot_{i-1} = \Delta j \\ j', \odot_{i'-1}: j' - \odot_{i'-1} = \Delta j'}} \Pr_{\mathrm{EJ}}(j, \odot_{i-1} \mid j', \odot_{i'-1}) \end{aligned} \tag{12}$$

The English word in position j' is aligned to the Japanese word in position j, and the English word in position $\odot_{i'-1}$ is aligned to the Japanese word in position $\odot_{i-1}$.

We assume that j and $\odot_{i-1}$ are independent, j only depends on j', and $\odot_{i-1}$ only depends on $\odot_{i'-1}$. Then $\Pr_{\mathrm{EJ}}(j, \odot_{i-1} \mid j', \odot_{i'-1})$ can be estimated as shown in (13).

$$\Pr_{EJ}(j, \odot_{i-1} \mid j', \odot_{i'-1}) = \Pr_{EJ}(j \mid j') \cdot \Pr_{EJ}(\odot_{i-1} \mid \odot_{i'-1}) \quad (13)$$

Both of the two parameters in (13) represent the position translation probabilities. Thus, we can estimate them from the distortion probability in model 3. $\Pr_{EJ}(j \mid j')$ is estimated as shown in (14). And $\Pr_{EJ}(\odot_{i-1} \mid \odot_{i'-1})$ can be estimated in the same way. In (14), we also assume that the sentence length distribution $\Pr(l, m \mid j')$ is independent of the word position and that it is uniformly distributed.

$$\begin{aligned} \Pr_{EJ}(j \mid j') &= \sum_{l,m} \Pr_{EJ}(j, l, m \mid j') \\ &= \sum_{l,m} d_{EJ}(j \mid j', l, m) \cdot \Pr(l, m \mid j') \\ &= \sum_{l,m} d_{EJ}(j \mid j', l, m) \end{aligned} \quad (14)$$

Distortion Probability for Non-Head Words

The distortion probability $d_{>1}(j - p(j))$ describes the distribution of the relative position of non-head words. In the same way, we introduce relative position $\Delta j'$ of English words, and model the probability in (15).

$$d_{>1,CJ}(\Delta j = j - p(j)) = \sum_{\Delta j'} d_{>1,CE}(\Delta j') \cdot \Pr_{EJ}(\Delta j \mid \Delta j') \quad (15)$$

$d_{>1,CJ}(\Delta j = j - p(j))$ is the estimated distortion probability for the non-head words in Chinese-Japanese alignment. $d_{>1,CE}(\Delta j')$ is the distortion probability for non-head words in Chinese-English alignment. $\Pr_{EJ}(\Delta j \mid \Delta j')$ is the translation probability of the relative Japanese position given the relative English position.

In fact, $\Pr_{EJ}(\Delta j \mid \Delta j')$ has the same interpretation as in (12). Thus, we introduce two parameters j' and $p(j')$ and let $\Delta j' = j' - p(j')$, where j' and $p(j')$ are positions of English words. The final distortion probability for non-head words can be estimated as shown in (16).

$$d_{>1,CJ}(\Delta j = j - p(j)) = \sum_{\Delta j'} (d_{>1,CE}(\Delta j') \cdot \sum_{\substack{j, p(j): j - p(j) = \Delta j \\ j', p(j'): j' - p(j') = \Delta j'}} \Pr_{EJ}(j \mid j') \cdot \Pr_{EJ}(p(j) \mid p(j'))) \quad (16)$$

5 Interpolation Model

With the Chinese-English and English-Japanese corpora, we can build the induced model for Chinese-Japanese word alignment as described in section 4. If we have small amounts of Chinese-Japanese corpora, we can build another word alignment model using the method described in section 3, which is called the *original model* here. In order to further improve the performance of Chinese-Japanese word alignment, we build an interpolated model by interpolating the induced model and the original model.

Generally, we can interpolate the induced model and the original model as shown in equation (17).

$$\Pr(\mathbf{a}, \mathbf{f} \mid \mathbf{c}) = \lambda \cdot \Pr_{O}(\mathbf{a}, \mathbf{f} \mid \mathbf{c}) + (1 - \lambda) \cdot \Pr_{I}(\mathbf{a}, \mathbf{f} \mid \mathbf{c}) \quad (17)$$

Where $\Pr_{O}(\mathbf{a}, \mathbf{f} \mid \mathbf{c})$ is the original model trained from the Chinese-Japanese corpus, and $\Pr_{I}(\mathbf{a}, \mathbf{f} \mid \mathbf{c})$ is the induced model trained from the Chinese-English and English-Japanese corpora. λ is an interpolation weight. It can be a constant or a function of $\mathbf{f}$ and $\mathbf{c}$.

In both model 3 and model 4, there are mainly three kinds of parameters: translation probability, fertility probability and distortion probability. These three kinds of parameters have their own interpretation in these two models. In order to obtain fine-grained interpolation models, we interpolate the three kinds of parameters using different weights, which are obtained in the same way as described in Wu et al. (2005). λ_t represents the weights for translation probability. λ_n represents the weights for fertility probability. λ_{d3} and λ_{d4} represent the weights for distortion probability in model 3 and in model 4, respectively. λ_{d4} is set as the interpolation weight for both the head words and the non-head words. The above four weights are obtained using a manually annotated held-out set.

6 Experiments

In this section, we compare different word alignment methods for Chinese-Japanese alignment. The "Original" method uses the original model trained with the small Chinese-Japanese corpus. The "Basic Induced" method uses the induced model that employs the basic translation probability without introducing cross-language word similarity. The "Advanced Induced" method uses the induced model that introduces the cross-language word similarity into the calculation of the translation probability. The "Interpolated" method uses the interpolation of the word alignment models in the "Advanced Induced" and "Original" methods.

6.1 Data

There are three training corpora used in this paper: Chinese-Japanese (CJ) corpus, Chinese-English (CE) Corpus, and English-Japanese (EJ) Corpus. All of these tree corpora are from general domain. The Chinese sentences and Japanese sentences in the data are automatically segmented into words. The statistics of these three corpora are shown in table 1. "# Source Words" and "# Target Words" mean the word number of the source and target sentences, respectively.

Language Pairs	#Sentence Pairs	# Source Words	# Target Words
CJ	21,977	197,072	237,834
CE	329,350	4,682,103	4,480,034
EJ	160,535	1,460,043	1,685,204

Table 1. Statistics for Training Data

Besides the training data, we also have held-out data and testing data. The held-out data includes 500 Chinese-Japanese sentence pairs, which is used to set the interpolated weights described in section 5. We use another 1,000 Chinese-Japanese sentence pairs as testing data, which is not included in the training data and the held-out data. The alignment links in the held-out data and the testing data are manually annotated. Testing data includes 4,926 alignment links[2].

6.2 Evaluation Metrics

We use the same metrics as described in Wu et al. (2005), which is similar to those in (Och and Ney, 2000). The difference lies in that Wu et al. (2005) took all alignment links as sure links.

If we use S_G to represent the set of alignment links identified by the proposed methods and S_C to denote the reference alignment set, the methods to calculate the precision, recall, f-measure, and alignment error rate (AER) are shown in equations (18), (19), (20), and (21), respectively. It can be seen that the higher the f-measure is, the lower the alignment error rate is. Thus, we will only show precision, recall and AER scores in the evaluation results.

$$precision = \frac{|S_G \cap S_C|}{|S_G|} \quad (18)$$

$$recall = \frac{|S_G \cap S_C|}{|S_C|} \quad (19)$$

$$fmeasure = \frac{2|S_G \cap S_C|}{|S_G| + |S_C|} \quad (20)$$

$$AER = 1 - \frac{2|S_G \cap S_C|}{|S_G| + |S_C|} = 1 - fmeasure \quad (21)$$

6.3 Experimental Results

We use the held-out data described in section 6.1 to set the interpolation weights in section 5. λ_t is set to 0.3, λ_n is set to 0.1, λ_{d3} for model 3 is set to 0.5, and λ_{d4} for model 4 is set to 0.1. With these parameters, we get the lowest alignment error rate on the held-out data.

For each method described above, we perform bi-directional (source to target and target to source) word alignment and obtain two alignment results. Based on the two results, we get a result using "refined" combination as described in (Och and Ney, 2000). Thus, all of the results reported here describe the results of the "refined" combination. For model training, we use the GIZA++ toolkit[3].

Method	Precision	Recall	AER
Interpolated	0.6955	0.5802	0.3673
Advanced Induced	0.7382	0.4803	0.4181
Basic Induced	0.6787	0.4602	0.4515
Original	0.6026	0.4783	0.4667

Table 2. Word Alignment Results

The evaluation results on the testing data are shown in table 2. From the results, it can be seen that both of the two induced models perform better than the "Original" method that only uses the limited Chinese-Japanese sentence pairs. The "Advanced Induced" method achieves a relative error rate reduction of 10.41% as compared with the "Original" method. Thus, with the Chinese-English corpus and the English-Japanese corpus, we can achieve a good word alignment results even if no Chinese-Japanese parallel corpus is available. After introducing the cross-language word similarity into the translation probability, the "Advanced Induced" method achieves a relative error rate reduction of 7.40% as compared with the "Basic Induced" method. It indicates that cross-language word similarity is effective in the calculation of the translation probability. Moreover, the "interpolated" method further improves the result, which achieves relative error

[2] For a non one-to-one link, if *m* source words are aligned to *n* target words, we take it as one alignment link instead of *m*∗*n* alignment links.

[3] It is located at http://www.fjoch.com/ GIZA++.html.

rate reductions of 12.51% and 21.30% as compared with the "Advanced Induced" method and the "Original" method.

7 Conclusion and Future Work

This paper presented a word alignment approach for languages with scarce resources using bilingual corpora of other language pairs. To perform word alignment between languages L1 and L2, we introduce a pivot language L3 and bilingual corpora in L1-L3 and L2-L3. Based on these two corpora and with the L3 as a pivot language, we proposed an approach to estimate the parameters of the statistical word alignment model. This approach can build a word alignment model for the desired language pair even if no bilingual corpus is available in this language pair. Experimental results indicate a relative error reduction of 10.41% as compared with the method using the small bilingual corpus.

In addition, we interpolated the above model with the model trained on the small L1-L2 bilingual corpus to further improve word alignment between L1 and L2. This interpolated model further improved the word alignment results by achieving a relative error rate reduction of 12.51% as compared with the method using the two corpora in L1-L3 and L3-L2, and a relative error rate reduction of 21.30% as compared with the method using the small bilingual corpus in L1 and L2.

In future work, we will perform more evaluations. First, we will further investigate the effect of the size of corpora on the alignment results. Second, we will investigate different parameter combination of the induced model and the original model. Third, we will also investigate how simpler IBM models 1 and 2 perform, in comparison with IBM models 3 and 4. Last, we will evaluate the word alignment results in a real machine translation system, to examine whether lower word alignment error rate will result in higher translation accuracy.

References

Yaser Al-Onaizan, Jan Curin, Michael Jahr, Kevin Knight, John Lafferty, Dan Melamed, Franz-Josef Och, David Purdy, Noah A. Smith, and David Yarowsky. 1999. Statistical Machine Translation Final Report. *Johns Hopkins University Workshop*.

Niraj Aswani and Robert Gaizauskas. 2005. Aligning Words in English-Hindi Parallel Corpora. In *Proc. of the ACL 2005 Workshop on Building and Using Parallel Texts: Data-driven Machine Translation and Beyond*, pages 115-118.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2): 263-311.

Colin Cherry and Dekang Lin. 2003. A Probability Model to Improve Word Alignment. In *Proc. of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-2003)*, pages 88-95.

Sue J. Ker and Jason S. Chang. 1997. A Class-based Approach to Word Alignment. *Computational Linguistics*, 23(2): 313-343.

Adam Lopez and Philip Resnik. 2005. Improved HMM Alignment Models for Languages with Scarce Resources. In *Proc. of the ACL-2005 Workshop on Building and Using Parallel Texts: Data-driven Machine Translation and Beyond*, pages 83-86.

Joel Martin, Rada Mihalcea, and Ted Pedersen. 2005. Word Alignment for Languages with Scarce Resources. In *Proc. of the ACL-2005 Workshop on Building and Using Parallel Texts: Data-driven Machine Translation and Beyond*, pages 65-74.

Charles Schafer and David Yarowsky. 2002. Inducing Translation Lexicons via Diverse Similarity Measures and Bridge Languages. In *Proc. of the 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*, pages 1-7.

Dan Tufis, Radu Ion, Alexandru Ceausu, and Dan Stefanescu. 2005. Combined Word Alignments. In *Proc. of the ACL-2005 Workshop on Building and Using Parallel Texts: Data-driven Machine Translation and Beyond*, pages 107-110.

Franz Josef Och and Hermann Ney. 2000. Improved Statistical Alignment Models. In *Proc. of the 38th Annual Meeting of the Association for Computational Linguistics (ACL-2000)*, pages 440-447.

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19-51.

Dekai Wu. 1997. Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora. *Computational Linguistics*, 23(3):377-403.

Hua Wu, Haifeng Wang, and Zhanyi Liu. 2005. Alignment Model Adaptation for Domain-Specific Word Alignment. In *Proc. of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005)*, pages 467-474.

Hao Zhang and Daniel Gildea. 2005. Stochastic Lexicalized Inversion Transduction Grammar for Alignment. In *Proc. of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005)*, pages 475-482.

Combining Statistical and Knowledge-based Spoken Language Understanding in Conditional Models

Ye-Yi Wang, Alex Acero, Milind Mahajan
Microsoft Research
One Microsoft Way
Redmond, WA 98052, USA
{yeyiwang,alexac,milindm}@microsoft.com

John Lee
Spoken Language Systems
MIT CSAIL
Cambridge, MA 02139, USA
jsylee@csail.mit.edu

Abstract

Spoken Language Understanding (SLU) addresses the problem of extracting semantic meaning conveyed in an utterance. The traditional knowledge-based approach to this problem is very expensive -- it requires joint expertise in natural language processing and speech recognition, and best practices in language engineering for every new domain. On the other hand, a statistical learning approach needs a large amount of annotated data for model training, which is seldom available in practical applications outside of large research labs. A generative HMM/CFG composite model, which integrates easy-to-obtain domain knowledge into a data-driven statistical learning framework, has previously been introduced to reduce data requirement. The major contribution of this paper is the investigation of integrating prior knowledge and statistical learning in a conditional model framework. We also study and compare conditional random fields (CRFs) with perceptron learning for SLU. Experimental results show that the conditional models achieve more than 20% relative reduction in slot error rate over the HMM/CFG model, which had already achieved an SLU accuracy at the same level as the best results reported on the ATIS data.

1 Introduction

Spoken Language Understanding (SLU) addresses the problem of extracting meaning conveyed in an utterance. Traditionally, the problem is solved with a knowledge-based approach, which requires joint expertise in natural language processing and speech recognition, and best practices in language engineering for every new domain. In the past decade many statistical learning approaches have been proposed, most of which exploit generative models, as surveyed in (Wang, Deng et al., 2005). While the data-driven approach addresses the difficulties in knowledge engineering, it requires a large amount of labeled data for model training, which is seldom available in practical applications outside of large research labs. To alleviate the problem, a generative HMM/CFG composite model has previously been introduced (Wang, Deng et al., 2005). It integrates a knowledge-based approach into a statistical learning framework, utilizing prior knowledge to compensate for the dearth of training data. In the ATIS evaluation (Price, 1990), this model achieves the same level of understanding accuracy (5.3% error rate on standard ATIS evaluation) as the best system (5.5% error rate), which is a semantic parsing system based on a manually developed grammar.

Discriminative training has been widely used for acoustic modeling in speech recognition (Bahl, Brown et al., 1986; Juang, Chou et al., 1997; Povey and Woodland, 2002). Most of the methods use the same generative model framework, exploit the same features, and apply discriminative training for parameter optimization. Along the same lines, we have recently exploited conditional models by directly porting the HMM/CFG model to Hidden Conditional Random Fields (HCRFs) (Gunawardana, Mahajan et al., 2005), but failed to obtain any improvement. This is mainly due to the vast parameter space, with the parameters settling at local optima. We then simplified the original model structure by removing the hidden variables, and introduced a number of important overlapping and non-homogeneous features. The resulting Conditional Random Fields (CRFs) (Lafferty, McCallum et al., 2001) yielded a 21% relative improvement in SLU accuracy. We also applied a much simpler perceptron learning algorithm on the conditional model and observed improved SLU accuracy as well.

In this paper, we will first introduce the generative HMM/CFG composite model, then discuss the problem of directly porting the model to HCRFs, and finally introduce the CRFs and

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 882–889,
Sydney, July 2006.

the features that obtain the best SLU result on ATIS test data. We compare the CRF and perceptron training performances on the task.

2 Generative Models

The HMM/CFG composite model (Wang, Deng et al., 2005) adopts a pattern recognition approach to SLU. Given a word sequence W, an SLU component needs to find the semantic representation of the meaning M that has the maximum *a posteriori* probability $\Pr(M|W)$:

$$\hat{M} = \arg\max_{M} \Pr(M|W)$$
$$= \arg\max_{M} \Pr(W|M) \cdot \Pr(M)$$

The composite model integrates domain knowledge by setting the topology of the prior model, $\Pr(M)$, according to the domain semantics; and by using PCFG rules as part of the lexicalization model $\Pr(W|M)$.

The domain semantics define an application's semantic structure with semantic frames. Figure 1 shows a simplified example of three semantic frames in the ATIS domain. The two frames with the "toplevel" attribute are also known as commands. The "filler" attribute of a slot specifies the semantic object that can fill it. Each slot may be associated with a CFG rule, and the filler semantic object must be instantiated by a word string that is covered by that rule. For example, the string "Seattle" is covered by the "City" rule in a CFG. It can therefore fill the ACity (ArrivalCity) or the DCity (DepartureCity) slot, and instantiate a Flight frame. This frame can then fill the Flight slot of a ShowFlight frame. Figure 2 shows a semantic representation according to these frames.

```
<frame name="ShowFlight" toplevel="1">
    <slot name="Flight" filler="Flight"/>
</frame>
<frame name="GroundTrans" toplevel="1">
    <slot name="City" filler="City"/>
</frame>
<frame name="Flight">
    <slot name="DCity" filler="City"/>
    <slot name="ACity" filler="City"/>
</frame>
```

Figure 1. Simplified domain semantics for the ATIS domain.

The semantic prior model comprises the HMM topology and state transition probabilities. The topology is determined by the domain semantics, and the transition probabilities can be estimated from training data. Figure 3 shows the topology of the underlying states in the statistical model for the semantic frames in Figure 1. On top is the transition network for the two top-level commands. At the bottom is a zoomed-in view for the "Flight" sub-network. State 1 and state 4 are called precommands. State 3 and state 6 are called postcommands. States 2, 5, 8 and 9 represent slots. A slot is actually a three-state sequence — the slot state is preceded by a preamble state and followed by a postamble state, both represented by black circles. They provide contextual clues for the slot's identity.

```
<ShowFlight>
    <Flight>
        <DCity filler="City">Seattle</DCity>
        <ACity filler="City">Boston</ACity>
    </Flight>
</ShowFlight>
```

Figure 2. The semantic representation for "Show me the flights departing from Seattle arriving at Boston" is an instantiation of the semantic frames in Figure 1.

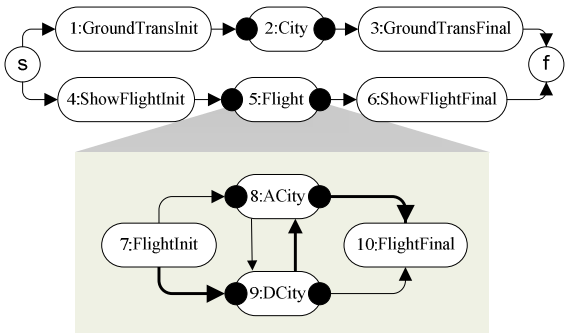


Figure 3. The HMM/CFG model's state topology, as determined by the semantic frames in Figure 1.

The lexicalization model, $\Pr(W|M)$, depicts the process of sentence generation from the topology by estimating the distribution of words emitted by a state. It uses state-dependent n-grams to model the precommands, postcommands, preambles and postambles, and uses knowledge-based CFG rules to model the slot fillers. These rules help compensate for the dearth of domain-specific data. In the remainder of this paper we will say a string is "covered by a CFG non-terminal (NT)", or equivalently, is "CFG-covered for *s*" if the string can be parsed by the CFG rule corresponding to the slot *s*.

Given the semantic representation in Figure 2, the state sequence through the model topology in

Figure 3 is deterministic, as shown in Figure 4. However, the words are not aligned to the states in the shaded boxes. The parameters in their corresponding n-gram models can be estimated with an EM algorithm that treats the alignments as hidden variables.

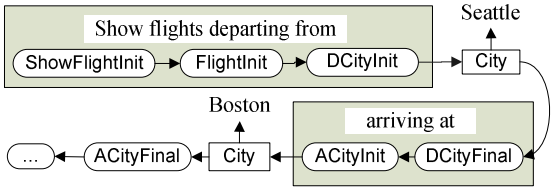


Figure 4. Word/state alignments. The segmentation of the word sequences in the shaded region is hidden.

The HMM/CFG composite model was evaluated in the ATIS domain (Price, 1990). The model was trained with ATIS3 category A training data (~1700 annotated sentences) and tested with the 1993 ATIS3 category A test sentences (470 sentences with 1702 reference slots). The slot insertion-deletion-substitution error rate (SER) of the test set is 5.0%, leading to a 5.3% semantic error rate in the standard end-to-end ATIS evaluation, which is slightly better than the best manually developed system (5.5%). Moreover, a steep drop in the error rate is observed after training with only the first two hundred sentences. This demonstrates that the inclusion of prior knowledge in the statistical model helps alleviate the data sparseness problem.

3 Conditional Models

We investigated the application of conditional models to SLU. The problem is formulated as assigning a label l to each element in an observation $\mathbf{o}$. Here, $\mathbf{o}$ consists of a word sequence $\mathbf{o}_1^\tau$ and a list of CFG non-terminals (NT) that cover its subsequences, as illustrated in Figure 5. The task is to label "two" as the "Num-of-tickets" slot of the "ShowFlight" command, and "Washington D.C." as the ArrivalCity slot for the same command. To do so, the model must be able to resolve several kinds of ambiguities:

1. Filler/non-filler ambiguity, e.g., "two" can either fill a Num-of-tickets slot, or its homonym "to" can form part of the preamble of an ArrivalCity slot.
2. CFG ambiguity, e.g., "Washington" can be CFG-covered as either City or State.
3. Segmentation ambiguity, e.g., [Washington] [D.C.] vs. [Washington D.C.].
4. Semantic label ambiguity, e.g., "Washington D.C." can fill either an ArrivalCity or DepartureCity slot.

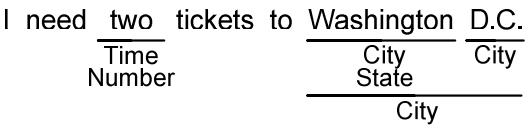


Figure 5. The observation includes a word sequence and the subsequences covered by CFG non-terminals.

3.1 CRFs and HCRFs

Conditional Random Fields (CRFs) (Lafferty, McCallum et al., 2001) are undirected conditional graphical models that assign the conditional probability of a state (label) sequence s_1^τ with respect to a vector of features $\mathbf{f}(s_1^\tau, \mathbf{o}_1^\tau)$. They are of the following form:

$$p(s_1^\tau \mid \mathbf{o}; \lambda) = \frac{1}{z(\mathbf{o}; \lambda)} \exp\left(\lambda \cdot \mathbf{f}(s_1^\tau, \mathbf{o})\right). \quad (1)$$

Here $z(\mathbf{o}; \lambda) = \sum_{s_1^\tau} \exp\left(\lambda \cdot \mathbf{f}(s_1^\tau, \mathbf{o})\right)$ normalizes the distribution over all possible state sequences. The parameter vector λ is trained conditionally (discriminatively). If we assume that s_1^τ is a Markov chain given $\mathbf{o}$ and the feature functions only depend on two adjacent states, then

$$p(s_1^\tau \mid \mathbf{o}; \lambda) = \frac{1}{z(\mathbf{o}; \lambda)} \exp\left(\sum_k \lambda_k \sum_{t=1}^{\tau} f_k(s^{(t-1)}, s^{(t)}, \mathbf{o}, t)\right) \quad (2)$$

In some cases, it may be natural to exploit features on variables that are not directly observed. For example, a feature for the Flight preamble may be defined in terms of an observed word and an unobserved state in the shaded region in Figure 4:

$$f_{\text{FlightInit,flights}}(s^{(t-1)}, s^{(t)}, \mathbf{o}, t) = \begin{cases} 1 & \text{if } s^{(t)} = \text{FlightInit} \wedge \mathbf{o}^t = \text{flights}; \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

In this case, the state sequence s_1^τ is only partially observed in the meaning representation $M : M(s_5) = \text{"DCity"} \wedge M(s_8) = \text{"ACity"}$ for the words "Seattle" and "Boston". The states for the remaining words are hidden. Let $\Gamma(M)$ represent the set of all state sequences that satisfy the constraints imposed by M. To obtain the conditional probability of M, we need to sum over all possible labels for the hidden states:

$$p(M \mid \mathbf{o}; \lambda) = \frac{1}{z(\mathbf{o}; \lambda)} \sum_{s_1^\tau \in \Gamma(M)} \exp\left(\sum_k \lambda_k \sum_{t=1}^{\tau} f_k(s^{(t-1)}, s^{(t)}, \mathbf{o}, t) \right)$$

CRFs with features dependent on hidden state variables are called Hidden Conditional Random Fields (HCRFs). They have been applied to tasks such as phonetic classification (Gunawardana, Mahajan et al., 2005) and object recognition (Quattoni, Collins et al., 2004).

3.2 Conditional Model Training

We train CRFs and HCRFs with gradient-based optimization algorithms that maximize the log posterior. The gradient of the objective function is

$$\nabla_\lambda L(\lambda) = \mathbf{E}_{\tilde{P}(\mathbf{l},\mathbf{o})P(s_1^\tau \mid \mathbf{l},\mathbf{o})}\left[\mathbf{f}(s_1^\tau, \mathbf{o}); \lambda\right] - \mathbf{E}_{\tilde{P}(\mathbf{o})P(s_1^\tau \mid \mathbf{o})}\left[\mathbf{f}(s_1^\tau, \mathbf{o}); \lambda\right]$$

which is the difference between the conditional expectation of the feature vector given the observation sequence and label sequence, and the conditional expectation given the observation sequence alone. With the Markov assumption in Eq. (2), these expectations can be computed using a forward-backward-like dynamic programming algorithm. For CRFs, whose features do not depend on hidden state sequences, the first expectation is simply the feature counts given the observation and label sequences. In this work, we applied stochastic gradient descent (SGD) (Kushner and Yin, 1997) for parameter optimization. In our experiments on several different tasks, it is faster than L-BFGS (Nocedal and Wright, 1999), a quasi-Newton optimization algorithm.

3.3 CRFs and Perceptron Learning

Perceptron training for conditional models (Collins, 2002) is an approximation to the SGD algorithm, using feature counts from the Viterbi label sequence in lieu of expected feature counts. It eliminates the need of a forward-backward algorithm to collect the expected counts, hence greatly speeds up model training. This algorithm can be viewed as using the minimum margin of a training example (i.e., the difference in the log conditional probability of the reference label sequence and the Viterbi label sequence) as the objective function instead of the conditional probability:

$$L'(\lambda) = \log P(\mathbf{l} \mid \mathbf{o}; \lambda) - \max_{\mathbf{l}'} \log P(\mathbf{l}' \mid \mathbf{o}; \lambda)$$

Here again, **o** is the observation and **l** is its reference label sequence. In perceptron training, the parameter updating stops when the Viterbi label sequence is the same as the reference label sequence. In contrast, the optimization based on the log posterior probability objective function keeps pulling probability mass from all incorrect label sequences to the reference label sequence until convergence.

In both perceptron and CRF training, we average the parameters over training iterations (Collins, 2002).

4 Porting HMM/CFG Model to HCRFs

In our first experiment, we would like to exploit the discriminative training capability of a conditional model without changing the HMM/CFG model's topology and feature set. Since the state sequence is only partially labeled, an HCRF is used to model the conditional distribution of the labels.

4.1 Features

We used the same state topology and features as those in the HMM/CFG composite model. The following indicator features are included:

Command prior features capture the *a priori* likelihood of different top-level commands:

$$f_c^{PR}(s^{(t-1)}, s^{(t)}, \mathbf{o}, t) = \begin{cases} 1 & \text{if } t{=}0 \wedge \mathrm{C}(s^{(t)}) = c \\ 0 & \text{otherwise} \end{cases}, \forall c \in \text{CommandSet}$$

Here C(s) stands for the name of the command that corresponds to the transition network containing state s.

State Transition features capture the likelihood of transition from one state to another:

$$f_{s_1,s_2}^{TR}(s^{(t-1)}, s^{(t)}, \mathbf{o}, t) = \begin{cases} 1 & \text{if } s^{(t-1)} = s_1, s^{(t)} = s_2 \\ 0 & \text{otherwise} \end{cases},$$

where $s_1 \to s_2$ is a legal transition according to the state topology.

Unigram and *Bigram* features capture the likelihoods of words emitted by a state:

$$f^{UG}_{s,w}(s^{(t-1)}, s^{(t)}, \mathbf{o}^{\tau}_1, t) = \begin{cases} 1 & \text{if } s^{(t)} = s \wedge \mathbf{o}^t = w \\ 0 & \text{otherwise} \end{cases},$$

$$f^{BG}_{s,w_1,w_2}(s^{(t-1)}, s^{(t)}, \mathbf{o}^{\tau}_1, t) = \begin{cases} 1 & \text{if } s^{(t-1)} = s \wedge s^{(t)} = s \wedge \mathbf{o}^{t-1} = w_1 \wedge \mathbf{o}^t = w_2 \\ 0 & \text{otherwise} \end{cases},$$

$$\forall s \mid \neg \text{isFiller}(s); \forall w, w_1 w_2 \in \text{TrainingData}$$

The condition isFiller(s_1) restricts s_1 to be a slot state and not a pre- or postamble state.

4.2 Experiments

The model is trained with SGD with the parameters initialized in two ways. The *flat start* initialization sets all parameters to 0. The *generative model* initialization uses the parameters trained by the HMM/CFG model.

Figure 6 shows the test set slot error rates (SER) at different training iterations. With the flat start initialization (top curve), the error rate never comes close to the 5% baseline error rate of the HMM/CFG model. With the generative model initialization, the error rate is reduced to 4.8% at the second iteration, but the model quickly gets over-trained afterwards.

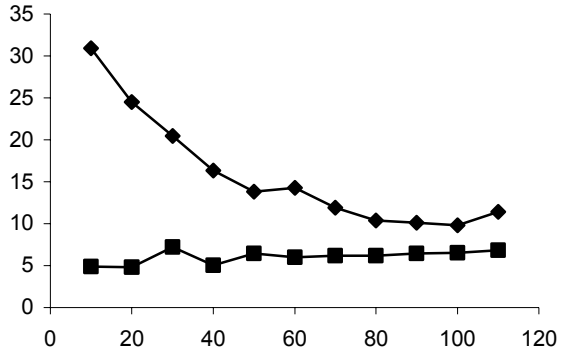


Figure 6. Test set slot error rates (in %) at different training iterations. The top curve is for the flat start initialization, the bottom for the generative model initialization.

The failure of the direct porting of the generative model to the conditional model can be attributed to the following reasons:

- The conditional log-likelihood function is no longer a convex function due to the summation over hidden variables. This makes the model highly likely to settle on a local optimum. The fact that the flat start initialization failed to achieve the accuracy of the generative model initialization is a clear indication of the problem.
- In order to account for words in the test data, the n-grams in the generative model are properly smoothed with back-offs to the uniform distribution over the vocabulary. This results in a huge number of parameters, many of which cannot be estimated reliably in the conditional model, given that model regularization is not as well studied as in n-grams.
- The hidden variables make parameter estimation less reliable, given only a small amount of training data.

5 CRFs for SLU

An important lesson we have learned from the previous experiment is that we should not think generatively when applying conditional models. While it is important to find cues that help identify the slots, there is no need to exhaustively model the generation of every word in a sentence. Hence, the distinctions between pre- and postcommands, and pre- and postambles are no longer necessary. Every word that appears between two slots is labeled as the preamble state of the second slot, as illustrated in Figure 7. This labeling scheme effectively removes the hidden variables and simplifies the model to a CRF. It not only expedites model training, but also prevents parameters from settling at a local optimum, because the log conditional probability is now a convex function.

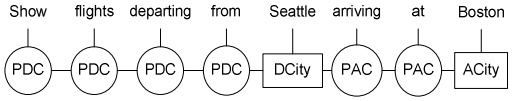


Figure 7. Once the slots are marked in the simplified model topology, the state sequence is fully marked, leaving no hidden variables and resulting in a CRF. Here, PAC stands for "preamble for arrival city," and PDC for "preamble for departure city."

The command prior and state transition features (with fewer states) are the same as in the HCRF model. For unigrams and bigrams, only those that occur in front of a CFG-covered string are considered. If the string is CFG-covered for slot *s*, then the unigram and bigram features for the preamble state of *s* are included. Suppose the words "that departs" occur at positions $t-1$ and t in front of the word "Seattle," which is CFG-covered by the non-terminal **City**. Since **City** can fill a DepartureCity or ArrivalCity slot, the four following features are introduced:

$$f^{UG}_{\text{PDC,that}}(s^{(t-1)}, s^{(t)}, \mathbf{o}^{\tau}_1, t) = f^{UG}_{\text{PAC,that}}(s^{(t-1)}, s^{(t)}, \mathbf{o}^{\tau}_1, t) = 1$$

And

$$f^{BG}_{\text{PDC,that,departs}}(s^{(t-1)}, s^{(t)}, \mathbf{o}^{\tau}_1, t) =$$
$$f^{BG}_{\text{PAC,that,departs}}(s^{(t-1)}, s^{(t)}, \mathbf{o}^{\tau}_1, t) = 1$$

Formally,

$$f^{UG}_{s,w}(s^{(t-1)}, s^{(t)}, \mathbf{o}^{\tau}_1, t) = \begin{cases} 1 & \text{if } s^{(t)} = s \wedge \mathbf{o}^t = w \\ 0 & \text{otherwise} \end{cases},$$

$$f^{BG}_{s,w_1,w_2}(s^{(t-1)}, s^{(t)}, \mathbf{o}^{\tau}_1, t) = \begin{cases} 1 & \text{if } s^{(t-1)} = s^{(t)} = s \wedge \mathbf{o}^{t-1} = w_1 \wedge \mathbf{o}^t = w_2 \\ 0 & \text{otherwise} \end{cases},$$

$\forall s \mid \neg\text{isFiller}(s)$;

$\forall w, w_1 w_2 \mid$ in the training data, w and $w_1 w_2$ appears in front of sequence that is CFG-covered for s.

5.1 Additional Features

One advantage of CRFs over generative models is the ease with which overlapping features can be incorporated. In this section, we describe three additional feature sets.

The first set addresses a side effect of not modeling the generation of every word in a sentence. Suppose a preamble state has never occurred in a position that is confusable with a slot state *s*, and a word that is CFG-covered for *s* has never occurred as part of the preamble state in the training data. Then, the unigram feature of the word for that preamble state has weight 0, and there is thus no penalty for mislabeling the word as the preamble. This is one of the most common errors observed in the development set. The *chunk coverage for preamble words* feature introduced to model the likelihood of a CFG-covered word being labeled as a preamble:

$$f^{CC}_{c,NT}(s^{(t-1)}, s^{(t)}, \mathbf{o}, t) = \begin{cases} 1 & \text{if } \mathrm{C}(s^{(t)}) = c \wedge \text{covers}(NT, \mathbf{o}^t) \wedge \text{isPre}(s^{(t)}) \\ 0 & \text{otherwise} \end{cases}$$

where isPre(s) indicates that s is a preamble state.

Often, the identity of a slot depends on the preambles of the previous slot. For example, "at two PM" is a DepartureTime in "flight from Seattle to Boston at two PM", but it is an ArrivalTime in "flight departing from Seattle arriving in Boston at two PM." In both cases, the previous slot is ArrivalCity, so the state transition features are not helpful for disambiguation. The identity of the time slot depends not on the ArrivalCity slot, but on its preamble. Our second feature set, *previous-slot context*, introduces this dependency to the model:

$$f^{PC}_{s_1,s_2,w}(s^{(t-1)}, s^{(t)}, \mathbf{o}, t) = \begin{cases} 1 & \text{if } s^{(t-1)} = s_1 \wedge s^{(t)} = s_2 \wedge w \in \Theta(s_1, \mathbf{o}, t-1) \\ & \wedge\ \text{isFiller}(s_1) \wedge\ \text{Slot}(s_1) \neq \text{Slot}(s_2) \\ 0 & \text{otherwise} \end{cases}$$

Here Slot(s) stands for the slot associated with the state s, which can be a filler state or a preamble state, as shown in Figure 7. $\Theta(s_1, \mathbf{o}, t-1)$ is the set of k words (where k is an adjustable window size) in front of the longest sequence that ends at position $t-1$ and that is CFG-covered by $\text{Slot}(s_1)$.

The third feature set is intended to penalize erroneous segmentation, such as segmenting "Washington D.C." into two separate City slots. The *chunk coverage for slot boundary* feature is activated when a slot boundary is covered by a CFG non-terminal *NT*, i.e., when words in two consecutive slots ("Washington" and "D.C.") can also be covered by one single slot:

$$f^{SB}_{c,NT}(s^{(t-1)}, s^{(t)}, \mathbf{o}, t) = \begin{cases} 1 & \text{if } \mathrm{C}(s^{(t)}) = c \wedge \text{covers}(NT, \mathbf{o}^t_{t-1}) \\ & \wedge\ \text{isFiller}(s^{(t-1)}) \wedge\ \text{isFiller}(s^{(t)}) \\ & \wedge s^{(t-1)} \neq s^{(t)} \\ 0 & \text{otherwise} \end{cases}$$

This feature set shares its weights with the *chunk coverage features for preamble words*, and does not introduce any new parameters.

Features	# of Param.	SER
Command Prior	6	
+State Transition	+1377	18.68%
+Unigrams	+14433	7.29%
+Bigrams	+58191	7.23%
+Chunk Cov Preamble Word	+156	6.87%
+Previous-Slot Context	+290	5.46%
+Chunk Cov Slot Boundaries	+0	3.94%

Table 1. Number of additional parameters and the slot error rate after each new feature set is introduced.

5.2 Experiments

Since the objective function is convex, the optimization algorithm does not make any significant difference on SLU accuracy. We

trained the model with SGD. Other optimization algorithm like Stochastic Meta-Decent (Vishwanathan, Schraudolph et al., 2006) can be used to speed up the convergence. The training stopping criterion is cross-validated with the development set.

Table 1 shows the number of new parameters and the slot error rate (SER) on the test data, after each new feature set is introduced. The new features improve the prediction of slot identities and reduce the SER by 21%, relative to the generative HMM/CFG composite model.

The figures below show in detail the impact of the n-gram, previous-slot context and chunk coverage features. The chunk coverage feature has three settings: 0 stands for no chunk coverage features; 1 for chunk coverage features for preamble words only; and 2 for both words and slot boundaries.

Figure 8 shows the impact of the order of n-gram features. Zero-order means no lexical features for preamble states are included. As the figure illustrates, the inclusion of CFG rules for slot filler states and domain-specific knowledge about command priors and slot transitions have already produced a reasonable SER under 15%. Unigram features for preamble states cut the error by more than 50%, while the impact of bigram features is not consistent -- it yields a small positive or negative difference depending on other experimental parameter settings.

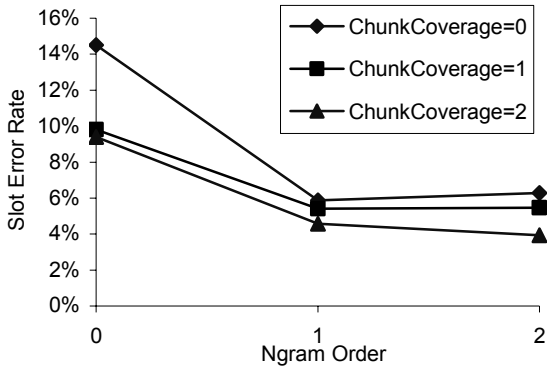


Figure 8. Effects of the order of n-grams on SER. The window size for the previous-slot context features is 2.

Figure 9 shows the impact of the CFG chunk coverage feature. Coverage for both preamble words and slot boundaries help improve the SLU accuracy.

Figure 10 shows the impact of the window size for the previous-slot context feature. Here, 0 means that the previous-slot context feature is not used. When the window size is *k*, the *k* words in front of the longest previous CFG-covered word sequence are included as the previous-slot unigram context features. As the figure illustrates, this feature significantly reduces SER, while the window size does not make any significant difference.

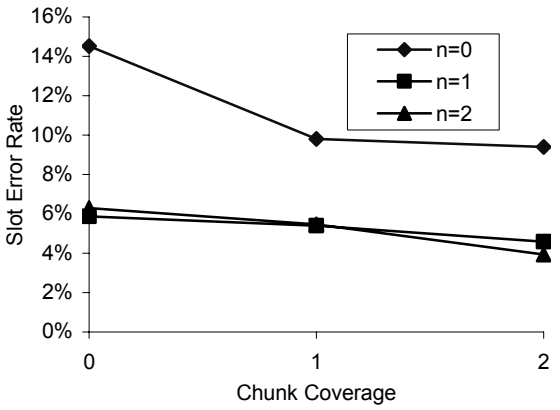


Figure 9. Effects of the chunk coverage feature. The window size for the previous-slot context feature is 2. The three lines correspond to different n-gram orders, where 0-gram indicates that no preamble lexical features are used.

It is important to note that overlapping features like f^{CC}, f^{SB} and f^{PC} could not be easily incorporated into a generative model.

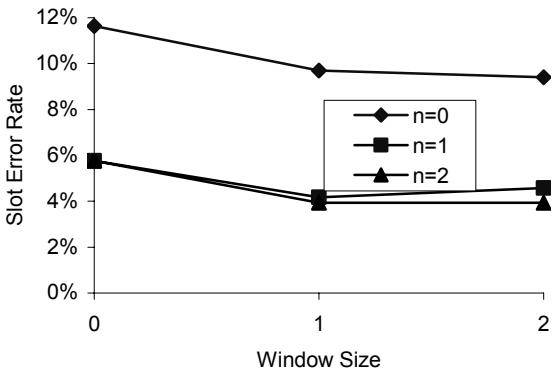


Figure 10. Effects of the window size of the previous-slot context feature. The three lines represent different orders of n-grams (0, 1, and 2). Chunk coverage features for both preamble words and slot boundaries are used.

5.3 CRFs vs. Perceptrons

Table 2 compares the perceptron and CRF training algorithms, using chunk coverage features for both preamble words and slot boundaries, with which the best accuracy results

are achieved. Both improve upon the 5% baseline SER from the generative HMM/CFG model. CRF training outperforms the perceptron in most settings, except for the one with unigram features for preamble states and with window size 1 -- the model with the fewest parameters. One possible explanation is as follows. The objective function in CRFs is a convex function, and so SGD can find the single global optimum for it. In contrast, the objective function for the perceptron, which is the difference between two convex functions, is not convex. The gradient ascent approach in perceptron training is hence more likely to settle on a local optimum as the model becomes more complicated.

	PSWSize=1		PSWSize=2	
	Perceptron	CRFs	Perceptron	CRFs
n=1	3.76%	4.11%	4.23%	3.94%
n=2	4.76%	4.41%	4.58%	3.94%

Table 2. Perceptron vs. CRF training. Chunk coverage features are used for both preamble words and slot boundaries. PSWSize stands for the window size of the previous-slot context feature. N is the order of the n-gram features.

The biggest advantage of perceptron learning is its speed. It directly counts the occurrence of features given an observation and its reference label sequence and Viterbi label sequence, with no need to collect expected feature counts with a forward-backward-like algorithm. Not only is each iteration faster, but fewer iterations are required, when using SLU accuracy on a cross-validation set as the stopping criterion. Overall, perceptron training is 5 to 8 times faster than CRF training.

6 Conclusions

This paper has introduced a conditional model framework that integrates statistical learning with a knowledge-based approach to SLU. We have shown that a conditional model reduces SLU slot error rate by more than 20% over the generative HMM/CFG composite model. The improvement was mostly due to the introduction of new overlapping features into the model. We have also discussed our experience in directly porting a generative model to a conditional model, and demonstrated that it may not be beneficial at all if we still think generatively in conditional modeling; more specifically, replicating the feature set of a generative model in a conditional model may not help much. The key benefit of conditional models is the ease with which they can incorporate overlapping and non-homogeneous features. This is consistent with the finding in the application of conditional models for POS tagging (Lafferty, McCallum et al., 2001). The paper also compares different training algorithms for conditional models. In most cases, CRF training is more accurate, however, perceptron training is much faster.

References

Bahl, L., P. Brown, et al. 1986. Maximum mutual information estimation of hidden Markov model parameters for speech recognition. IEEE International Conference on Acoustics, Speech, and Signal Processing.

Collins, M. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. EMNLP, Philadelphia, PA.

Gunawardana, A., M. Mahajan, et al. 2005. Hidden conditional random fields for phone classification. Eurospeech, Lisbon, Portugal.

Juang, B.-H., W. Chou, et al. 1997. "Minimum classification error rate methods for speech recognition." IEEE Transactions on Speech and Audio Processing 5(3): 257-265.

Kushner, H. J. and G. G. Yin. 1997. Stochastic approximation algorithms and applications, Springer-Verlag.

Lafferty, J., A. McCallum, et al. 2001. Conditional random fields: probabilistic models for segmenting and labeling sequence data. ICML.

Nocedal, J. and S. J. Wright. 1999. Numerical optimization, Springer-Verlag.

Povey, D. and P. C. Woodland. 2002. Minimum phone error and I-smoothing for improved discriminative training. IEEE International Conference on Acoustics, Speech, and Signal Processing.

Price, P. 1990. Evaluation of spoken language system: the ATIS domain. DARPA Speech and Natural Language Workshop, Hidden Valley, PA.

Quattoni, A., M. Collins and T. Darrell. 2004. Conditional Random Fields for Object Recognition. NIPS.

Vishwanathan, S. V. N., N. N. Schraudolph, et al. 2006. Accelerated Training of conditional random fields with stochastic meta-descent. The Learning Workshop, Snowbird, Utah.

Wang, Y.-Y., L. Deng, et al. 2005. "Spoken language understanding --- an introduction to the statistical framework." IEEE Signal Processing Magazine 22(5): 16-31.

Sinhala Grapheme-to-Phoneme Conversion and Rules for Schwa Epenthesis

Asanka Wasala, Ruvan Weerasinghe and Kumudu Gamage
Language Technology Research Laboratory
University of Colombo School of Computing
35, Reid Avenue, Colombo 07, Sri Lanka
{awasala,kgamage}@webmail.cmb.ac.lk, arw@ucsc.cmb.ac.lk

Abstract

This paper describes an architecture to convert Sinhala Unicode text into phonemic specification of pronunciation. The study was mainly focused on disambiguating schwa-/ə/ and /a/ vowel epenthesis for consonants, which is one of the significant problems found in Sinhala. This problem has been addressed by formulating a set of rules. The proposed set of rules was tested using 30,000 distinct words obtained from a corpus and compared with the same words manually transcribed to phonemes by an expert. The Grapheme-to-Phoneme (G2P) conversion model achieves 98 % accuracy.

1 Introduction

The conversion of Text-to-Speech (TTS) involves many important processes. These processes can be divided mainly in to three parts; text analysis, linguistic analysis and waveform generation (Black and Lenzo, 2003). The text analysis process is responsible for converting the non-textual content into text. This process also involves tokenization and normalization of the text. The identification of words or chunks of text is called text-tokenization. Text normalization establishes the correct interpretation of the input text by expanding the abbreviations and acronyms. This is done by replacing the non-alphabetic characters, numbers, and punctuation with appropriate text strings depending on the context. The linguistic analysis process involves finding the correct pronunciation of words, and assigning prosodic features (eg. phrasing, intonation, stress) to the phonemic string to be spoken. The final process of a TTS system is waveform generation which involves the production of an acoustic digital signal using a particular synthesis approach such as formant synthesis, articulatory synthesis or waveform concatenation (Lemmetty, 1999). The text analysis and linguistic analysis processes together are known as the Natural Language Processing (NLP) component, while the waveform generation process is known as the Digital Signal Processing (DSP) component of a TTS System (Dutoit, 1997).

Finding correct pronunciation for a given word is one of the first and most significant tasks in the linguistic analysis process. The component which is responsible for this task in a TTS system is often named the Grapheme-To-Phoneme (G2P), Text-to-Phone or Letter-To-Sound (LTS) conversion module. This module accepts a word and generates the corresponding phonemic transcription. Further, this phonemic transcription can be annotated with appropriate prosodic markers (Syllables, Accents, Stress etc) as well.

In this paper, we describe the implementation and evaluation of a G2P conversion model for a Sinhala TTS system. A Sinhala TTS system is being developed based on *Festival*, the open source speech synthesis framework. Letter to sound conversion for Sinhala usually has simple one to one mapping between orthography and phonemic transcription for most Sinhala letters. However some G2P conversion rules are proposed in this paper to complement the generation of more accurate phonemic transcription.

The rest of this paper is organized as follows: Section 2 gives an overview of the Sinhala phonemic inventory and the Sinhala writing system, Section 3 briefly discusses G2P conversion approaches. Section 4 describes the schwa epenthesis issue peculiar to Sinhala and Section 5 explains the Sinhala G2P conversion architecture.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 890–897,
Sydney, July 2006.

Section 6 gives experimental results and our discussion on it. The work is summarized in the final section.

2 Sinhala Phonemic Inventory and Writing System

2.1 The Sinhala Phonemic Inventory

Sinhala is the official language of Sri Lanka and the mother tongue of the majority - 74% of its population. Spoken Sinhala contains 40 segmental phonemes; 14 vowels and 26 consonants as classified below in Table 1 and Table 2 (Karunatillake, 2004).

There are two nasalized vowels occurring in two or three words in Sinhala. They are /ã/, /ã:/, /æ̃/ and /æ̃:/ (Karunatillake, 2004). Spoken Sinhala also has following Diphthongs; /iu/, /eu/, /æu/, /ou/, /au/, /ui/, /ei/, /æi/, /oi/ and /ai/ (Disanayaka, 1991).

	Front		Central		Back	
	Short	Long	Short	Long	Short	Long
High	i	i:			u	u:
Mid	e	e:	ə	ə:	o	o:
Low	æ	æ:	a	a:		

Table 1. Spoken Sinhala Vowel Classification.

		Lab.	Den.	Alv.	Ret.	Pal.	Vel.	Glo.
Stops	Voiceless	p	t		ʈ		k	
	Voiced	b	d		ɖ		g	
Affricates	Voiceless					c		
	Voiced					ɟ		
Pre-nasalized voiced stops		ɓ	ɗ		ᶑ		g̃	
Nasals		m		n		ɲ	ŋ	
Trill				r				
Lateral				l				
Spirants		f	s			ʃ		h
Semivowels		w				j		

Table 2*. Spoken Sinhala Consonant Classification.

A separate sign for vowel /ə/ is not provided by the Sinhala writing system. In terms of distribution, the vowel /ə/ does not occur at the beginning of a syllable except in the conjugational variants of verbs formed from the verbal stem /kərə/ *(to do)*. In contrast to this, though the letter "ඥ", which symbolizes the consonant sound /ɲ/ exists, it is not considered a phoneme in Sinhala.

2.2 The Sinhala Writing System

The Sinhala character set has 18 vowels, and 42 consonants as shown in Table 3.

Vowels and corresponding vowel modifiers (within brackets):
අ ආ(ා) ඇ(ැ) ඈ(ෑ) ඉ(ි) ඊ(ී) උ(ු) ඌ(ූ) ඍ(ෘ) ඎ(ෲ) ඏ(ෟ) ඐ(ෳ) එ(ෙ) ඒ(ේ) ඓ(ෛ) ඔ(ො) ඕ (ෝ) ඖ(ෞ)

Consonants:
ක ඛ ග ඝ ඞ ඟ ච ඡ ජ ඣ ඤ ඥ ඦ ට ඨ ඩ ඪ ණ ඬ ත ථ ද ධ න ඳ ප ඵ බ භ ම ඹ ය ර ල ව ශ ෂ ස හ ළ ෆ ං ඃ

Special symbols: ් ා ෘ ෲ
Inherent vowel remover (*Hal* marker): ්

Table 3. Sinhala Character Set.

Sinhala characters are written left to right in horizontal lines. Words are delimited by a space in general. Vowels have corresponding full-character forms when they appear in an absolute initial position of a word. In other positions, they appear as 'strokes' and, are used with consonants to denote vowel modifiers. All vowels except "ඍ" /iru:/, are able to occur in word initial positions (Disanayaka, 1995). The vowel /ə/ and /ə:/ occurs only in loan words of English origin. Since there are no special symbols to represent them, frequently the "අ" vowel is used to symbolize them (Karunatillake, 2004).

All consonants occur in word initial position except /ŋ/ and nasals (Disanayaka, 1995). The symbols "ණ", and "ළ" represent the retroflex nasal /ɳ/ and the retroflex lateral /ɭ/ respectively. But they are pronounced as their respective alveolar counterparts "න"-/n/ and "ල"-/l/. Similarly, the symbol "ෂ" representing the retroflex sibilant /ʂ/, is pronounced as the palatal sibilant "ශ"-/ʃ/. The corresponding aspirated symbols of letters ක, ග, ච, ජ, ට, ඩ, ත, ද, ප, බ namely ඛ, ඝ, ඡ, ඣ, ඨ, ඪ, ථ, ඵ, භ respectively are pronounced like the corresponding unaspirates (Karunatillake, 2004). When consonants are combined with /r/ or /j/, special conjunct symbols are used. "ර්"-/r/ immediately following a consonant can be marked by the symbol "්ර" added to the bottom of the consonant preceding it. Similarly, "ය්"-/j/, immediately following consonant can be marked by the symbol "්ය"

* Lab. – Labial, Den. – Dental, Alv. – Alveolar, Ret. – Retroflex, Pal. – Palatal, Vel. – Velar and Glo. – Glottal.

added to the right-hand side of the consonant preceding it (Karunatillake, 2004). "ඔ" /ilu/ and "ඕ" /ilu:/ do not occur in contemporary Sinhala (Disanayaka, 1995). Though there are 60 symbols in Sinhala (Disanayaka, 1995), only 42 symbols are necessary to represent Spoken Sinhala (Karunatillake, 2004).

3 G2P Conversion Approaches

The issue of mapping textual content into phonemic content is highly language dependent. Three main approaches of G2P conversion are; use of a pronunciation dictionary, use of well defined language-dependent rules and data-driven methods (El-Imam and Don, 2005).

One of the easiest ways of G2P conversion is the use of a lexicon or pronunciation dictionary. A lexicon consists of a large list of words together with their pronunciation. There are several limitations to the use of lexicons. It is practically impossible to construct such to cover the whole vocabulary of a language owing to Zipfian phenomena. Though a large lexicon is constructed, one would face other limitations such as efficient access, memory storage etc. Most lexicons often do not include many proper names, and only very few provide pronunciations for abbreviations and acronyms. Only a few lexicons provide distinct entries for morphological productions of words. In addition, pronunciations of some words differ based on the context and their parts-of-speech. Further, an enormous effort has to be made to develop a comprehensive lexicon. In practical scenarios, speech synthesizers as well as speech recognizers need to be able to produce the pronunciation of words that are not in the lexicon. Names, morphological productivity and numbers are the three most important cases that cause the use of lexica to be impractical (Jurafsky and Martin, 2000).

To overcome these difficulties, rules can be specified on how letters can be mapped to phonemes. In this way, the size of the lexicon can be reduced as only to contain exceptions to the rules. In contrast to the above fact, some systems rely on using very large lexicons, together with a set of letter-to-sound conversion rules to deal with words which are not found in the lexicon (Black and Lenzo, 2003).

These language and context dependent rules are formulated using phonetic and linguistic knowledge of a particular language. The complexity of devising a set of rules for a particular language is dependent on the degree of correspondence between graphemes and phonemes. For some languages such as English and French, the relationship is complex and require large numbers of rules (El-Imam and Don, 2005; Damper et al., 1998), while some languages such as Urdu (Hussain, 2004), and Hindi (Ramakishnan et al., 2004; Choudhury, 2003) show regular behavior and thus pronunciation can be modeled by defining fairly regular simple rules.

Data-driven methods are widely used to avoid tedious manual work involving the above approaches. In these methods, G2P rules are captured by means of various machine learning techniques based on a large amount of training data. Most previous data-driven approaches have been used for English. Widely used data-driven approaches include, Pronunciation by Analogy (PbA), Neural Networks (Damper et al., 1998), and Finite-State-Machines (Jurafsky and Martin, 2000). Black et al. (1998) discussed a method for building general letter-to-sound rules suitable for any language, based on training a CART – decision tree.

4 Schwa Epenthesis in Sinhala

G2P conversion problems encountered in Sinhala are similar to those encountered in the Hindi language (Ramakishnan et al., 2004). All consonant graphemes in Sinhala are associated with an inherent vowel schwa-/ə/ or /a/ which is not represented in orthography. Vowels other than /ə/ and /a/ are represented in orthographic text by placing specific vowel modifier diacritics around the consonant grapheme. In the absence of any vowel modifier for a particular consonant grapheme, there is an ambiguity of associating /ə/ or /a/ as the vowel modifier. The inherent vowel association in Sinhala can be distinguished from Hindi. In Hindi the only possible association is schwa vowel where as in Sinhala either of vowel-/a/ or schwa-/ə/ can be associated with a consonant. Native Sinhala speakers are naturally capable of choosing the association of the appropriate vowel (/ə/ or /a/) in context. Moreover, linguistic rules describing the transformation of G2P, is rarely found in literature, with available literature not providing any precise procedure suitable for G2P conversion of contemporary Sinhala. Automating the G2P conversion process is a difficult task due to the ambiguity of choosing between /ə/ and /a/.

A similar phenomenon is observed in Hindi and Malay as well. In Hindi, the "deletion of the schwa vowel *(in some cases)*" is successfully

solved by using rule based algorithms (Choudhury 2003; Ramakishnan et al., 2004). In Malay, the character 'e' can be pronounced as either vowel /e/ or /ə/, and rule based algorithms are used to address this ambiguity (El-Imam and Don, 2005).

In our research, a set of rules is proposed to disambiguate epenthesis of /a/ and /ə/, when associating with consonants. Unlike in Hindi, in Sinhala, the schwa is not deleted, instead always inserted. Hence, this process is named "Schwa Epenthesis" in this paper.

5 Sinhala G2P Conversion Architecture

An architecture is proposed to convert Sinhala Unicode text into phonemes encompassing a set of rules to handle schwa epenthesis. The G2P architecture developed for Sinhala is identical to the Hindi G2P architecture (Ramakishnan et al., 2004). The input to the system is normalized Sinhala Unicode text. The G2P engine first maps all characters in the input word into corresponding phonemes by using the letter-to-phoneme mapping table below (Table 4).

අ	/a/	ඔ ,ොෟ	/o/	ඩ	/ɖ/	ෆ	/f/
ආ,ා	/a:/	ඕ,ෝ	/o:/	ත,ථ	/t/	ාෘ	/ru:/
ඇ,ැ	/æ/	ඖ,ෞ	/ou/	ද,ධ	/d/		
ඈ,ෑ	/æ:/	ක,ඛ	/k/	ඳ	/ð/		
ඉ ,ි	/i/	ග,ඝ	/g/	ප,ඵ	/p/		
ඊ,ී	/i:/	ඞ,ං	/ŋ/	බ,භ	/b/		
උ,ු	/u/	ඟ	/ɠ/	ම	/m/		
ඌ,ූ	/u:/	ච,ඡ	/c/	ඹ	/ƀ/		
සෘ	/ri/	ජ,ඣ	/ɟ/	ය	/j/		
ෘ	/ru/	ඤ	/ɲ/	ර	/r/		
ඵ	/ilu/	ඥ	/jɲ/	ල,ළ	/l/		
ඹෟ	/ilu:/	ඡ	/ʄ/	ව	/w/		
එ ,ෙ	/e/	ට,ඨ	/ʈ/	ශ,ෂ	/ʃ/		
ඒ,ේ	/e:/	ඩ,ඪ	/ɖ/	ස	/s/		
ඓ,ෛ	/ai/	න,ණ	/n/	හ,ඃ	/h/		

Table 4. G2P Mapping Table

The mapping procedure is given in section 5.1. Then, a set of rules are applied to this phonemic string in a specific order to obtain a more accurate version. This phonemic string is then compared with the entries in the exception lexicon. If a matching entry is found, the correct pronunciation form of the text is obtained from the lexicon, otherwise the resultant phonemic string is returned. Hence, the final output of G2P model is the phonemic transcription of the input text.

5.1 G2P Mapping Procedure

Each tokenized word represented by Unicode normalization form is analyzed by individual graphemes from left to right. By using the G2P mapping table (Table 4), corresponding phonemes are obtained. As in the given example Figure 1, no mappings are required for the Zero-Width-Joiner and diacritic *Hal* marker "්" *(Halant)* which is used to remove the inherent vowel in a consonant.

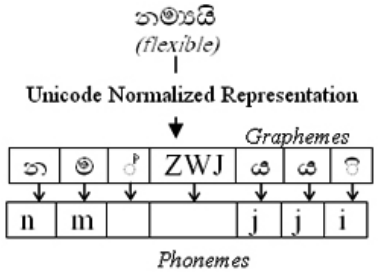


Figure 1. G2P Mapping (Example).

The next step is epenthesis of schwa-/ə/ for consonants. In Sinhala, the tendency of associating a /ə/ with consonant is very much higher than associating vowel /a/. Therefore, initially, all plausible consonants are associated with /ə/. To obtain the accurate pronunciation, the assigned /ə/ is altered to /a/ or vice versa by applying the set of rules given in next section. However, when associating /ə/ with consonants, /ə/ should associate only with consonant graphemes excluding the graphemes "ං", "ඞ" and "ඃ", which do not contain any vowel modifier or diacritic *Hal* marker. In the above example, only /n/ and first /j/ are associated with schwa, because other consonants violate the above principle. When schwa is associated with appropriate consonants, the resultant phonemic string for the given example (section 5.1) is; /nəmjəji/.

5.2 G2P Conversion Rules

It is observed that resultant phoneme strings from the above procedure should undergo several modifications in terms of schwa assignments into vowel /a/ or vice versa, in order to obtain the accurate pronunciation of a particular word. Guided by the literature (Karunatillake, 2004), it was noticed that these modifications can be carried out by formulating a set of rules.

The G2P rules were formulated with the aid of phonological rules described in the linguistic literature (Karunatillake, 2004) and by a comprehensive word search analysis using the *UCSC*

Sinhala corpus BETA (2005). Some of these existing phonological rules were altered in order to reflect the observations made in the corpus word analysis and to achieve more accurate results. The proposed new set of rules is empirically shown to be effective and can be conveniently implemented using regular expressions.

Each rule given below is applied from left to right, and the presented order of the rules is to be preserved. Except for rule #1, rule #5, rule #6 and rule #8, all other rules are applied repeatedly many times to a single word until the conditions presented in the rules are satisfied.

Rule #1: If the nucleus of the first syllable is a schwa, the schwa should be replaced by vowel /a/ (Karunatillake, 2004), except in the following situations;

(a) The syllable starts with /s/ followed by /v/. (ie. /sv/)

(b) The first syllable starts with /k/ where as, /k/ is followed by /ə/ and subsequently /ə/ is preceded by /r/. (ie. /kər/)

(c) The word consists of a single syllable having CV structure (eg. /də/)

Rule #2:

(a) If /r/ is preceded by any consonant, followed by /ə/ and subsequently followed by /h/, then /ə/ should be replaced by /a/.

(/[consonant]rəh/->/[consonant]rah/)

(b) If /r/ is preceded by any consonant, followed by /ə/ and subsequently followed by any consonant other than /h/, then /ə/ should be replaced by /a/.

(/[consonant]rə[!h]/->/[consonant]ra[!h]/)

(c) If /r/ is preceded by any consonant, followed by /a/ and subsequently followed by any consonant other than /h/, then /a/ should be replaced by /ə/.

(/[consonant]ra[!h]/->/[consonant]rə!h]/)

(d) If /r/ is preceded by any consonant, followed by /a/ and subsequently followed by /h/, then /a/ is retained.

(/[consonant]ra[h]/->/[consonant]ra[h]/)

Rule #3: If any vowel in the set {/a/, /e/, /æ/, /o/, /ə/} is followed by /h/ and subsequently /h/ is preceded by schwa, then schwa should replaced by vowel /a/.

Rule #4: If schwa is followed by a consonant cluster, the schwa should be replaced by /a/ (Karunatillake, 2004).

Rule #5: If /ə/ is followed by the word final consonant, it should be replaced by /a/, except in the situations where the word final consonant is /r/, /b/, /ɖ/ or /ʈ/.

Rule #6: At the end of a word, if schwa precedes the phoneme sequence /ji/, the schwa should be replaced by /a/ (Karunatillake, 2004).

Rule #7: If the /k/ is followed by schwa, and subsequent phonemes are /r/ or /l/ followed by /u/, then schwa should be replaced by phoneme /a/. (ie. /kə(r|l)u/->/ka(r|l)u/)

Rule #8: Within the given context of following words, /a/ found in phoneme sequence /kal/, (the left hand side of the arrow) should be changed to /ə/ as shown in the right hand side.

- /kal(a:|e:|o:)y/->/kəl(a:|e:|o:)y/
- /kale(m|h)(u|i)/->/kəle(m|h)(u|i)/
- /kaləh(u|i)/->/kəleh(u|i)/
- /kalə/->/kələ/

The above rules handle the schwa epenthesis problem. The corresponding diphthongs (refer section 2) are then obtained by processing the resultant phonetized string. This string is again analyzed from left to right, and the phoneme sequences given in the first column of Table 5 are replaced by the diphthong, represented in the second column.

Phoneme Sequence	Diphthong
/i/ /w/ /u/	/iu/
/e/ /w/ /u/	/eu/
/æ/ /w/ /u/	/æu/
/o/ /w/ /u/	/ou/
/a/ /w/ /u/	/au/
/u/ /j/ /i/	/ui/
/e/ /j/ /i/	/ei/
/æ/ /j/ /i/	/æi/
/o/ /j/ /i/	/oi/
/a/ /j/ /i/	/ai/

Table 5. Diphthong Mapping Table.

The application of the above rules for the given example (section 5.1) is illustrated in Figure 2.

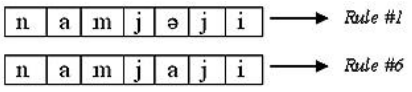


Figure 2. Application of G2P Rules – An Example.

6 Results and Discussion

Text obtained from the category *"News Paper> Feature Articles > Other"* of the *UCSC Sinhala* corpus was chosen for testing due to the heterogeneous nature of these texts and hence perceived better representation of the language in this part of the corpus[*]. A list of distinct words was first extracted, and the 30,000 most frequently occurring words chosen for testing.

The overall accuracy of our G2P module was calculated at 98%, in comparison with the same words correctly transcribed by an expert.

Since this is the first known documented work on implementing a G2P scheme for Sinhala, its contribution to the existing body of knowledge is difficult to evaluate. However, an experiment was conducted in order to arrive at an approximation of the scale of this contribution.

It was first necessary, to define a baseline against which this work could be measured. While this could be done by giving a single default letter-to-sound mapping for any Sinhala letter, owing to the near universal application of rule #1 in Sinhala words (22766 of the 30000 words used in testing), the baseline was defined by the application of this rule in addition to the 'default mapping'. This baseline gives us an error of approximately 24%. Since the proposed solution reduces this error to 2%, this work can claim to have improved performance by 22%.

An error analysis revealed the following types of errors (Table 6):

Error description	**# of words**
Compound words- (ie. Single words formed by combining 2 or more distinct words; such as in the case of the English word "thereafter").	382
Foreign (mainly English) words directly encoded in Sinhala. eg. ෆැෂන් - fashion, කැම්පස් - campus.	116
Other	118

Table 6. Types of Errors.

The errors categorized as "Other" are given below with clarifications:

- The modifier used to denote long vowel "ආ" /a:/ is "ා" which is known as "Aela-pilla". eg. consonant "ක්" /k/ associates with "ා" /a:/ to produce grapheme "කා" is pronounced as /ka:/. The above exercise revealed some 37 words end without vowel modifier "ා", but are usually pronounced with the associated long vowel /a:/. In the following examples, each input word is listed first, followed by the erroneous output of G2P conversion, and correct transcription.

  "අම්ම"(mother) -> /ammə/ -> /amma:/
  "අක්ක"*(sister)* -> /akkə/ -> /akka:/
  "ගත්ත"*(taken)*-> /gattə/ -> /gatta:/

- There were 27 words associated with erroneous conversion of words having the letter "හ", which corresponds to phoneme /h/. The study revealed this letter shows an unusual behavior in G2P conversion.

- The modifier used to denote vowel "ඍ" - "ෘ" is known as *"Geta-pilla"*. When this vowel appears as the initial letter of a word, it is pronounced as /ri/ as in "ඍණ" /rinə/ *(minus)*. When the corresponding vowel modifier appears in a middle of a word most of the time it is pronounced as /ru/ (Disanayaka, 2000). eg. "කෘතිය" *(book)*is pronounced as /krutijə/, "පෘෂ්ඨය" *(surface)* - /pruʃʈəjə/, "උත්කෘෂ්ට" *(excellent)*-/utkruʃʈə/. But 13 words were found as exceptions of this general rule. In those words, the "ෘ" is pronounced as /ur/ rather than /ru/. eg. "ප්‍රවෘත්ති" *(news)*-/prəwurti/,"සමෘද්ධි"*(prosperity)*-/samurdi/, "විවෘත" *(opened)* - /wiwurtə/.

- In general, vowel modifiers "ැ" *(Adha-pilla)*, "ෑ" *(Diga Adha-pilla)* symbolizes the vowel "ඇ" /æ/ and "ඈ" /æ:/ respectively. eg. consonant "ක්" /k/ combines with vowel modifier "ැ" to create "කැ" which is pronounced as /kæ/. Few words were found where this rule is violated. In such words, the vowel modifiers "ැ" and "ෑ" represent vowels "උ"- /u/, and "ඌ"- /u:/ respectively. eg. "ජනශ්‍රැති" *(legend)* - /ʄanəʃruti/, "ක්‍රෑර" *(cruel)* - /kru:rə/.

- The verbal stem "කර" *(to do)* is pronounced as /kərə/. Though there are many words starting with the same verbal stem, there are a few other words differently pronounced as /karə/ or /kara/. eg. "කරත්තය" *(cart)* /karattəyə/, "කරවල" *(dried fish)* /karəvələ/.

[*] This accounts for almost two-thirds of the size of this version of the corpus.

- A few of the remaining errors are due to homographs; “වන” - /vanə/, /vənə/; “කල” -/kalə/, /kələ/; “කර” - /karə/, /kərə/.

The above error analysis itself shows that the model can be extended. Failures in the current model are mostly due to compound words and foreign words directly encoded in Sinhala (1.66%). The accuracy of the G2P model can be increased significantly by incorporating a method to identify compound words and transcribe them accurately. If the constituent words of a compound word can be identified and separated, the same set of rules can be applied for each constituent word, and the resultant phonetized strings combined to obtain the correct pronunciation. The same problem is observed in the Hindi language too. Ramakishnan et al. (2004) proposed a procedure for extracting compound words from a Hindi corpus. The utilization of compound word lexicon in their rule-based G2P conversion module improved the accuracy of G2P conversion by 1.6% (Ramakishnan et al., 2004). In our architecture, the most frequently occurring compound words and foreign words are dealt with the aid of an exceptions lexicon. Homographs are also disambiguated using the most frequently occurring words in Sinhala. Future improvements of the architecture will include incorporation of a compound word identification and phonetization module.

7 Conclusion

In this paper, the problem of Sinhala grapheme-to-phoneme conversion is addressed with a special focus on dealing with the schwa epenthesis. The proposed G2P conversion mechanism will be useful in various applications in the speech domain. To the best of our knowledge no other documented evidence has been reported for Sinhala grapheme-to-phoneme conversion in the literature. There are no other approaches available for the transcription of Sinhala text that provides a platform for comparison of the proposed rule-based method. The empirical evidence from a wide spectrum Sinhala corpus indicates that the proposed model can account for nearly 98% of cases accurately.

The proposed G2P module is fully implemented in Sinhala TTS being developed at Language Technology Research Lab, UCSC. A demonstration tool of the proposed G2P module integrated with Sinhala syllabification algorithm proposed by Weerasinghe et al. (2005) is available for download from:

http://www.ucsc.cmb.ac.lk/ltrl/downloads.html

Acknowledgement

This work has been supported through the PAN Localization Project, (http://www.PANL10n.net) grant from the International Development Research Center (IDRC), Ottawa, Canada, administered through the Center for Research in Urdu Language Processing, National University of Computer and Emerging Sciences, Pakistan. The authors would like to thank Sinhala Language scholars Prof. R.M.W. Rajapaksha, and Prof. J.B. Dissanayake for their invaluable support and advice throughout the study. Special thanks to Dr. Sarmad Hussain (NUCES, Pakistan) for his guidance and advices. We also wish to acknowledge the contribution of Mr. Viraj Welgama, Mr. Dulip Herath, and Mr. Nishantha Medagoda of Language Technology Research Laboratory of the University of Colombo School of Computing, Sri Lanka.

References

Alan W. Black and Kevin A. Lenzo. 2003. *Building Synthetic Voices*, Language Technologies Institute, Carnegie Mellon University and Cepstral LLC. Retrieved from http://festvox.org/bsv/

Alan W. Black, Kevin Lenzo, and Vincent Pagel. 1998. Issues in Building General Letter to Sound Rules. *In Proc. of the 3rd ESCA Workshop on Speech Synthesis*, pages 77–80.

Monojit Choudhury. 2003. Rule-Based Grapheme to Phoneme Mapping for Hindi Speech Synthesis, *presented at the 90th Indian Science Congress of the International Speech Communication Association (ISCA)*, Bangalore.

R.I. Damper, Y. Marchand, M.J. Adamson and K. Gustafson. 1998. Comparative Evaluation of Letter-to-Sound Conversion Techniques for English Text-to-Speech Synthesis. *In Proc. Third ESCA/COCOSDA Workshop on Speech Synthesis*, pages 53- 58, Blue Mountains, NSW, Australia.

J.B. Disanayaka. 1991. *The Structure of Spoken Sinhala*, National Institute of Education, Maharagama.

J.B. Disanayaka. 2000. *Basaka Mahima: 2, Akuru ha pili*, S. Godage & Bros., 661, P. D. S. Kularathna Mawatha, Colombo 10.

J.B. Disanayaka. 1995. *Grammar of Contemporary Literary Sinhala - Introduction to Grammar,*

Structure of Spoken Sinhala, S. Godage & Bros., 661, P. D. S. Kularathna Mawatha, Colombo 10.

T. Dutoit. 1997. *An Introduction to Text-to-Speech Synthesis,* Kluwer Academic Publishers, Dordrecht, Netherlands.

Yousif A. El-Imam and Zuraidah M. Don. 2005. Rules and Algorithms for Phonetic Transcription of Standard Malay**,** *IEICE Trans Inf & Syst*, E88-D 2354-2372.

Sarmad Hussain. 2004. Letter-to-Sound Conversion for Urdu Text-to-Speech System, *Proceedings of Workshop on "Computational Approaches to Arabic Script-based Languages," COLING 2004,* p. 74-49, Geneva, Switzerland.

Daniel Jurafsky and James H. Martin. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Pearson Education (Singapore) Pte. Ltd, Indian Branch, 482 F.I.E. Patparganj, Delhi 110 092, India.

W.S. Karunatillake. 2004. *An Introduction to Spoken Sinhala*, 3rd edn., M.D. Gunasena & Co. ltd., 217, Olcott Mawatha, Colombo 11.

Sami Lemmetty. 1999. *Review of Speech Synthesis Technology*, MSc. thesis, Helsinki University of Technology.

A.G. Ramakishnan, Kalika Bali, Partha Pratim Talukdar N. and Sridhar Krishna. 2004. Tools for the Development of a Hindi Speech Synthesis System, *In 5th ISCA Speech Synthesis Workshop*, Pittsburgh. pages 109-114.

Ruvan Weerasinghe, Asanka Wasala and Kumudu Gamage. 2005. A Rule Based Syllabification Algorithm for Sinhala, *Proceedings of 2nd International Joint Conference on Natural Language Processing (IJCNLP-05)*, p. 438-449, Jeju Island, Korea.

*UCSC Sinhala Corpus BETA***.** 2005. Retrieved August 30, 2005, from University of Colombo School of Computing, Language Technology Research Laboratory Web site:

http://www.ucsc.cmb.ac.lk/ltrl/downloads.html

From Prosodic Trees to Syntactic Trees

Andi Wu
GrapeCity Inc.
andi.wu@grapecity.com

Kirk Lowery
Westminster Hebrew Institute
klowery@whi.wts.edu

Abstract

This paper describes an ongoing effort to parse the Hebrew Bible. The parser consults the bracketing information extracted from the cantillation marks of the Masoetic text. We first constructed a cantillation treebank which encodes the prosodic structures of the text. It was found that many of the prosodic boundaries in the cantillation trees correspond, directly or indirectly, to the phrase boundaries of the syntactic trees we are trying to build. All the useful boundary information was then extracted to help the parser make syntactic decisions, either serving as hard constraints in rule application or used probabilistically in tree ranking. This has greatly improved the accuracy and efficiency of the parser and reduced the amount of manual work in building a Hebrew treebank.

Introduction

The text of the Hebrew Bible (HB) has been carefully studied throughout the centuries, with detailed lexical, phonological and morphological analysis available for every verse of HB. However, very few attempts have been made at a verse-by-verse syntactic analysis. The only known effort in this direction is the Hebrew parser built by George Yaeger (Yaeger 1998, 2002), but the analysis is still incomplete in the sense that not all syntactic units are recognized and the accuracy of the trees are yet to be checked.

Since a detailed syntactic analysis of HB is of interest to both linguistic and biblical studies, we launched a project to build a treebank of the Hebrew Bible. In this project, the trees are automatically generated by a parser and then manually checked in a tree editor. Once a tree has been edited or approved, its phrase boundaries are recorded in a database. When the same verse is parsed again, the existing brackets will force the parser to produce trees whose brackets are exactly the same as those of the manually approved trees. Compared with traditional approaches to treebanking where the correct structure is preserved in a set of tree files, our approach has much more agility. In the event of design/format changes, we can automatically regenerate the trees according to the new specifications without manually touching the trees. The bracketing information will persist through the updates and the basic structure of the trees will remain correct regardless of the changes in the details of trees. We call this a "dynamic treebank" where, instead of maintaining a set of trees, we maintain a parser/grammar, a dictionary, a set of sentences, and a database of bracketing information. The trees can be generated at any time.

Since our parser/grammar can consult known phrase boundaries to build trees, its performance can be greatly improved if large amounts of bracketing information are available. Human inspection and correction can provide those boundaries, but the amount of manual work can be reduced significantly if there is an existing source of bracketing information for us to use. Fortunately, a great deal of such information can be obtained from the cantillation marks of the Hebrew text.

1 The cantillation treebank

1.1 Cantillation marks

The text of HB has been systematically annotated for more than a thousand years. By the end of the 9th century, a group of Jewish scholars known as the Masoretes had developed a system for

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 898–904,
Sydney, July 2006.

marking the structures of the Bible verses. The system contains a set of cantillation marks [1] which indicate the division and subdivision of each verse, very much like the punctuation marks or the brackets we use to mark constituent structures. At that time, those cantillation marks were intended to record the correct way of reading or chanting the Hebrew text: how to group words into phrases and where to put pauses between intonational units. In the eyes of modern linguists, the hierarchical structures thus marked can be best understood as a prosodic representation of the verses (Dresher 1994).

There are two types of cantillation marks: the conjunctive marks which group multiple words into single units and the disjunctive marks which divide and subdivide a verse in a binary fashion. The marking of Genesis 1:1, for example, is equivalent to the bracketing shown below. (English words are used here in place of Hebrew to make it easier for non-Hebrew-speakers to understand. OM stands for object marker.)

```
( ( ( In beginning )
    ( created God )
  )
  ( ( OM
      ( the heavens )
    )
    (  ( and OM )
       ( the earth )
    )
  )
)
```

This analysis resembles the prosodic structure in Selkirk (1984) and the performance structure in Gee and Grosjean (1983).

1.2. Parsing the prosodic structure

The cantillation system in the Mesoretic text is a very complex one with dozens of diacritic symbols and complicated annotation rules. As a result, only a few trained scholars can decipher them and their practical use has been very limited. In order to make the information encoded by this system more accessible to both humans and machines, we built a treebank where the prosodic structures of HB verses are explicitly represented as trees in XML format (Wu & Lowery, 2006).

There have been quite a few studies of the Masoretic cantillation system. After reviewing the existing analyses, such as Wickes (1881), Price (1990), Richter (2004) and Jacobson (2002), we adopted the binary analysis of British and Foreign Bible Society (BFBS 2002) which is based on the principle of dichotomy of Wicks (1881). The binary trees thus generated are best for extracting brackets that are syntactically significant.

We found all the binary rules that underlie the annotation and coded them in a context-free grammar. This CFG was then used by the parser to automatically generate the prosodic trees. The input text to the parser was the MORPH database developed by Groves & Lowery (2006) where the the cantillation marks are represented as numbers in its Michigan Code text.

The following is the prosodic tree generated for Genesis 1:1, displayed in English glosses in the tree editor (going right-to-left according to the writing convention of Hebrew):

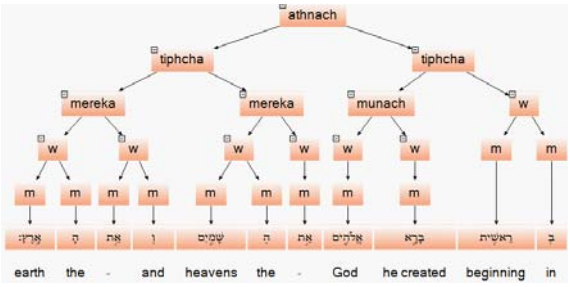


Figure 1

The node labels “athnach”, “tiphcha”, “mereka” and “munach” in this tree are names of the cantillation marks that indicate the types of boundaries between the two chunks they dominate. Different types of boundaries have different (relative) boundary strengths. The “m” nodes are morphemes and the “w” nodes are words.

[1] The cantillation marks show how a text is to be sung. See http://en.wikipedia.org/wiki/Cantillation.

1.3. A complete prosodic treebank

Since the Mesoretic annotation is supposed to mark the structure of every verse unambiguously, we expect to parse every verse successfully with exactly one tree assigned to it, given that (1) the annotation is perfectly correct and (2) the CFG grammars correctly encoded the annotation rules. The actual results were close to our expectation: all the 23213 verses were successfully parsed, of which 23099 received exactly one complete tree. The success rate is 99.5 percent. The 174 verses that received multiple parse trees all have words that carry more than one cantillation mark. This can of course create boundary ambiguities and result in multiple parse trees. We have good reasons to believe that the grammars we used are correct. We would have failed to parse some verses if the grammars had been incomplete and we would have gotten multiple trees for a much greater number of verses if the grammars had been ambiguous.

2 Phrase boundary extraction

Now that a cantillation treebank is available, we can get brackets from those trees and use them in syntactic parsing. Although prosodic structures are not syntactic structures, they do correspond to each other in some systematic ways. Just as there are ways to transform syntactic structures to prosodic structures (e.g. Abney 1992), prosodic structures can also provide clues to syntactic structures. As we have discovered, some of the brackets in the cantillation trees can be directly mapped to syntactic boundaries, some can be mapped after some adjustment, and some have no syntactic significance at all.

2.1 Direct correspondences

Direct correspondences are most likely to be found at the clausal level. Almost all the clause boundaries can be found in the cantillation trees. Take Genesis 1:3 as an example:

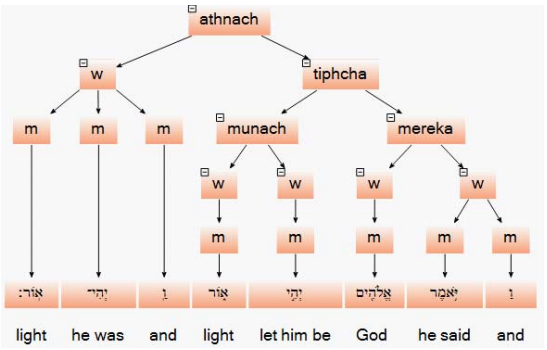


Figure 2

Here, the verse is first divided into two clauses: “God said let there be light” and “there was light”. The first clause is further divided into “God said” and “let there be light”. Such bracketing will prevent the wrong analysis where “let there be light” and “there was light” are conjoined to serve as the object of “God said”. Given the fact that there are no punctuation marks in HB, it is very difficult for the parser to rule out the wrong parse without the help of the cantillatioin information.

Coordination is another area where the cantillation brackets are of great help. The syntactic ambiguity associated with coordination is well-known, but the ambiguity can often be resolved with help of prosodic cues. This is indeed what we find in the cantillation treebank. In Genesis 24:35, for example, we find the following sequence of words: “male servants and maid servants and camels and donkeys”. Common sense tells us that there are only two possible analyses for this sequence: (1) a flat structure where the four NPs are sisters, or (2) “male servant” conjoins with “maid servant, “camels” conjoins with “donkeys”, and then the two conjoined NPs are further conjoined as sisters. However, the computer is faced with 14 different choices. Fortunately, the cantillation tree can help us pick the correct structure:

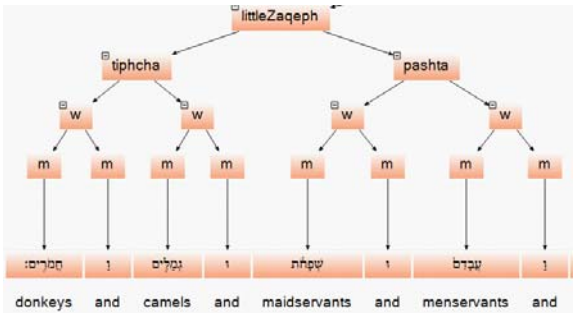


Figure 3

The brackets extracted from this tree will force the parser to produce only the second analysis above.

Good correspondences are also found for most base NPs and PPs. Here is an example from Genesis 1:4, which means "God separated the light from the darkness":

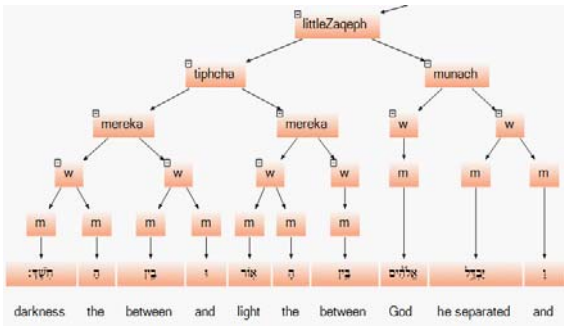


Figure 4

As we can see, the noun phrases and prepositional phrases all have corresponding brackets in this tree.

2.2 Indirect correspondences

Now we turn to prosodic structures that can be adjusted to correspond to syntactic structures. They usually involve the use of function words such as conjunctions, prepositions and determiners. Syntactically, these words are supposed to be attached to complete NPs, often resulting in trees where those single words are sisters to large NP chunks. Such "unbalanced" trees are rarely found in prosodic structures, however, where a sentence tends to be divided into chunks of similar length for better rhythm and flow of speech.

This is certainly the case in the HB cantillation treebank. It must have already been noticed in the example trees we have seen so far that the conjunction "and" is always attached to the word that immediately follows it. As a matter of fact, the conjunction and the following word are often treated as a single word for phonological reasons.

Prepositions are also traditionally treated as part of the following word. It is therefore not a surprise to find trees of the following kind:

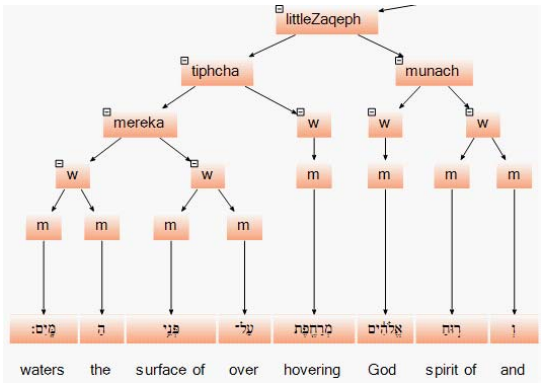


Figure 5

In this tree, the preposition "over" is attached to "surface of" instead of "surface of the waters". We also see the conjunction "and" is attached to "spirit of" rather than to the whole clause.

A similar situation is found for determiners, as can be seen in this sub-tree where "every of" is attached to "crawler of" instead of "crawler of the ground".

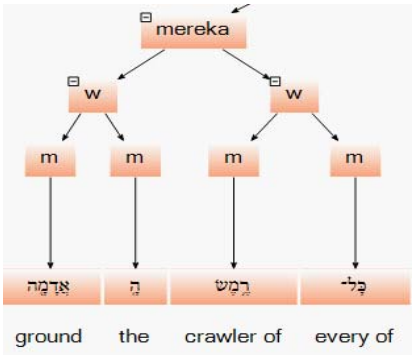


Figure 6

In all these cases, the differences between prosodic structures and syntactic structures are systematic and predictable. All of them can be adjusted to correspond better to syntactic structures by raising the function words out of their current positions and re-attach them to some higher nodes.

2.3 Extracting the boundaries

In the bracket extraction phase, we go through all the sub-trees and get their beginning and ending positions in the form of *(begin, end)*. Given the tree in Figure 6, for example, we can extract the following brackets: (*n*, *n+3*), (*n*, *n+1*), (*n+2*, *n+3*), where *n* is the position of the first word in the sub-tree.

For cases of indirect correspondence discussed in 2.2, we automatically adjust the brackets by removing the ones around the function word and its following word and adding a pair of brackets that start from the word following the function word and end in the last word of the phrase. After this adjustment, the brackets extracted from Figure 6 will become (*n*, *n+3*), *(n+1*, *n+3*) and (*n+2*, *n+3*). This in effect transforms this tree to the one in Figure 7 which corresponds better to its syntactic structure:

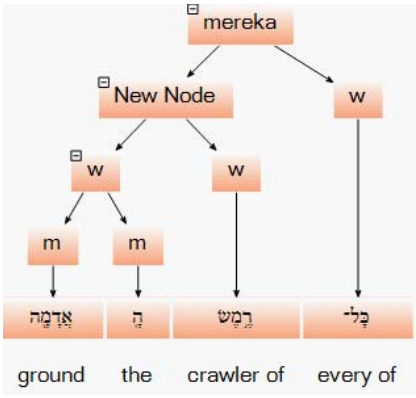


Figure 7

For trees that start with “and”, we detach “and” and re-attach it to the highest node that covers the phrase starting with “and”. After this and other adjustments, the brackets we extract from Figure 5 will be:

(*n*, *n+7*)
(*n+1*, *n+7*)
(*n+1*, *n+2*)
(*n+3*, *n+7*)
(*n+4*, *n+7*)
(*n+5*, *n+7*)
(*n+6*, *n+7*)

These brackets transform the tree into the one in Figure 8:

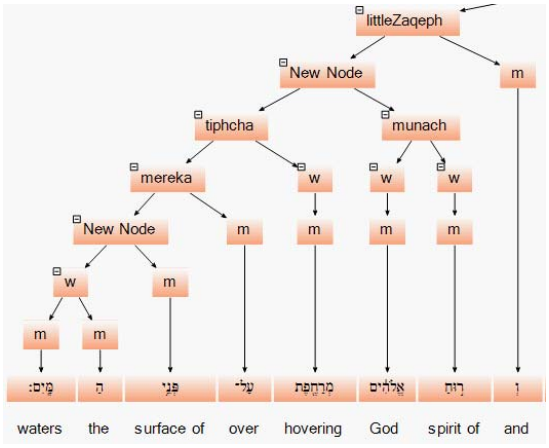


Figure 8

The cantillation trees also contain brackets that are not related to syntactic structures at all. Since it is difficult to identify those useless brackets automatically, we just leave them alone and let them be extracted anyway. Fortunately, as we will see in the next section, the parser does not depend completely on the extracted bracketing information. The useless brackets can simply be ignored in the parsing process.

3 Building a syntactic treebank

As we mentioned earlier, we use a parser to generate the treebank. This parser uses an augmented context-free grammar that encodes the grammatical knowledge of Biblical Hebrew. Each rule in this grammar has a number of grammatical conditions which must be satisfied in order for the rule to apply. In addition, it may have a bracketing condition which can either block the application of a rule or force a rule to apply.

Besides serving as conditions in rule application,

the bracketing information is also used to rank trees in cases where more than one tree is generated.

3.1 Brackets as rule conditions

Bracketing information is used in some grammar rules to guide the parser in making syntactic decisions. In those rules, we have conditions that look at the beginning position and ending position of the sub-tree to be produced by the rule and check to see if those bracket positions are found in our phrase boundary database. The sub-tree will be built only if the bracketing conditions are satisfied.

There are two types of bracketing conditions. One type serves as the necessary *and* sufficient condition for rule application. These conditions work in *disjunction* with grammatical conditions. A rule will apply when either the grammatical conditions or the bracketing conditions are satisfied. This is where the bracketing condition can force a rule to apply regardless of the grammatical conditions. The brackets consulted by this kind of conditions must be the manually approved ones or the automatically extracted ones that are highly reliable. Such conditions make it possible for us to override grammatical conditions that are too strict and build the structures that are known to be correct.

The other type of bracketing conditions serves as the necessary conditions only. They work in *conjunction* with grammatical conditions to determine the applicability of a rule. The main function of those bracketing conditions is to block structures that the grammatical conditions fail to block because of lack of information. However, they cannot force a rule to apply. The sub-tree to be produced the rule will be built only if both the grammatical conditions and the bracketing conditions are met.

The overall design of the rules and conditions are meant to build a linguistically motivated Hebrew grammar that is independent of the cantillation treebank while making use of its prosodic information.

3.2 Brackets for tree ranking

The use of bracketing conditions greatly reduces the number of trees the parser generates. In fact, many verses yield a single parse only. However, there are still cases where multiple trees are generated. In those cases, we use the bracketing information to help rank the trees.

During tree ranking, the brackets of each tree are compared with the brackets in the cantillation trees to find the number of mismatches. Trees that have fewer mismatches are ranked higher than trees that have more mismatches. In most cases, the top-ranking tree is the correct parse.

Theoretically, it should be possible to remove all the bracketing conditions from the rules, let the parser produce all possible trees, and use the bracketing information solely at the tree-ranking stage to select the correct trees. We can even use machine learning techniques to build a statistical parser. However, a Treebank of the Bible requires 100% accuracy but none of the statistical models are capable of that standard yet. As long as 100% accuracy is not guaranteed, manual checking will be required to fix all the individual errors. Such case-by-case fixes are easy to do in our current approach but are very difficult in statistical models.

3.3 Evaluation

Since only a very small fraction of the trees generated by our parser have been manually verified, there is not yet a complete golden standard to objectively evaluate the accuracy of the parser. However, some observations are obvious:

(1) The parsing process can become intractable without the bracketing conditions. We tried parsing with those conditions removed from the rules to see how many more trees we will get. It turned out that parsing became so slow that we had to terminate it before it was finished. This shows that the bracketing conditions are playing an indispensable role in making syntactic decisions.

(2) The number of edits needed to correct the trees in manual checking is minimal. Most trees generated by the machine are basically correct and only a few touches are necessary to make them perfect.
(3) The boundary information extracted from the cantillation tree could take a long time to create if done by hand, and a great deal of manual work is saved by using the brackets from the cantillation treebank.

Conclusion

In this paper, we have demonstrated the use of prosodic information in syntactic parsing in a treebanking project. There are correlations between prosodic structures and syntactic structures. By using a parser that consults the prosodic phrase boundaries, the cost of building the treebank can be minimized.

References

Abney, S (1992) Prosodic Structure, Performance Structure and Phrase Structure. In *Proceedings, Speech and Natural. Language Workshop*, pp. 425-428.

BFBS (2002) The Masoretes and the Punctuation of Biblical Hebrew. British & Foreign Bible Society, Machine Assisted Translation Team.

Dresher, B.E. (1994) The Prosodic Basis of the Tiberian Hebrew System of Accents. In *Language*, Vol. 70, No 1, pp. 1-52.

Gee J. P. & F. Grosjean (2002) The Masoretes and the Punctuation of Biblical Hebrew. British & Foreign Bible Society, Machine Assisted Translation Team.

Groves, A & K. Lowery, eds. (2006). *The Westminster Hebrew Bible Morphology Database*. Philadelphia: Westminster Hebrew Institute.

Jacobson, J.R. (2002) *Chanting the Hebrew Bible*. The Jewish Publication Society, Philadelphia..

Price, J. (1990) *The Syntax of Masoretic Accents in the Hebrew Bible*. The Edwin Mellen Press, Lewiston/Queenston/Lampeter.

Richter, H (2004) Hebrew Cantillation Marks and Their Encoding. Published at http://www.lrz-muenchen.de/~hr/teamim/

Selkirk, E. (1984) *Phonology and Syntax: The Relation between Sound and Structure*. Cambridge, MA: MIT Press.

Wickes, W. (1881) *Two Treatises on the Accentuation of the Old Testament*. Reprint by KTAV, New York, 1970..

Wu, A & K. Lowery (2006) A Hebrew Tree Bank Based on Cantillation Marks. In *Proceedings of LREC 2006*.

Yaeger, G (1998) Layered Parsing: a Principled Bottom-up Parsing Formalism for Classical Biblical Hebrew, a working paper, ASTER Institute, Point Pleasant, NJ.

Yaeger, G (2002) A Layered Parser Implementation of a Schema of Clause Types in Classical Biblical Hebrew, SBL Conference, Toronto, Ontario, Canada.

A Grammatical Approach to Understanding Textual Tables using Two-Dimensional SCFGs

Dekai WU[1] **Ken Wing Kuen LEE**
Human Language Technology Center
HKUST
Department of Computer Science and Engineering
University of Science and Technology
Clear Water Bay, Hong Kong
{dekai,cswkl}@cs.ust.hk

Abstract

We present an elegant and extensible model that is capable of providing semantic interpretations for an unusually wide range of textual tables in documents. Unlike the few existing table analysis models, which largely rely on relatively *ad hoc* heuristics, our linguistically-oriented approach is systematic and grammar based, which allows our model (1) to be concise and yet (2) recognize a wider range of data models than others, and (3) disambiguate to a significantly finer extent the underlying semantic interpretation of the table in terms of data models drawn from relation database theory. To accomplish this, the model introduces Viterbi parsing under two-dimensional stochastic CFGs. The cleaner grammatical approach facilitates not only greater coverage, but also grammar extension and maintenance, as well as a more direct and declarative link to semantic interpretation, for which we also introduce a new, cleaner data model. In disambiguation experiments on recognizing relevant data models of unseen web tables from different domains, a blind evaluation of the model showed 60% precision and 80% recall.

1 Introduction

Natural language processing has historically tended to emphasize understanding of linear strings—sentences, paragraphs, discourse structure. The vast body of work that focuses on text understanding is often seen as an approximation of spoken language understanding. Yet real-life text is actually heavily dependent on visual layout and formatting, which compensate for cues normally found in spoken language but are absent in text. As Scott (2003) reiterated in the opening ACL'03 invited talk: "The overlay of graphics on text is in many ways equivalent to the overlay of prosody on speech... Just as prosody undoubtedly contributes to the meaning of utterances, so too does a text's graphical presentation contribute to its meaning. However... few natural language understanding systems use graphical presentational features to aid interpretation..." (Power *et al.*, 2003).

Nowhere is this more evident than in the widespread use of tables in real-world, unsimplified text documents. Tables have a comparable or greater complexity as other elements of text. Unfortunately, in mainstream NLP it is not uncommon for tables to be regarded as a somehow "degenerate" form of text, unworthy of the same degree of attention as the rest of the text. But as we will discuss, the degree of ambiguity in table understanding is at least as great as for many sense and attachment problems. Many of the same mechanisms used for understanding linear text are also required for table understanding. The same division of surface syntax and underlying semantics is found.

Indeed, to perceive the limitations of existing table understanding models, we may distinguish several very different levels of table analysis tasks. In **table classification**, the table is classified into one of several coarse categories (in the extreme case, some models simply predict whether the purpose of the table is for page layout versus tabular data). In **table synactic recognition**, the surface types of individual cells or block regions are labeled (e.g., as heading or data) but the underlying semantic relationships between the table elements remain unrecognized and usually highly ambiguous (i.e., no logical relations between the elements

[1]The authors would like to thank the Hong Kong Research Grants Council (RGC) for supporting this research in part through grants RGC6083/99E, RGC6256/00E, and DAG03/04.EG09.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 905–912,
Sydney, July 2006.

in the table are assigned). In contrast, in **table semantic interpretation**, the exact logical relations between the elements in the table must be recognized (e.g., by associating the table and/or subregions thereof with precise table schemas in relational database style).

Existing table understanding work largely lies at the level of superficial table classification or syntactic recognition. Rarely, if ever, are precise logical relations assigned between the elements in the table. *Ad hoc* heuristic approaches tend to rule, rather than linguistic approaches.

On the other hand, in the linguistic approach advocated by Scott (2003) and (Power *et al.*, 2003), tables were not considered. The various physical presentation elements discussed included headings, captions, and bulleted lists—all of which exhibit numerous similarities to tabular elements. Possibly, tables were not considered because they are difficult to describe adequately within the expressiveness of common linguistic formalisms like CFGs.

The work presented here aims to address this problem. Our model provides an enabling foundation toward a linguistic approach by first shifting to a two-dimensional CFG framework. This permits us to construct a grammar where all the rules are meaningfully discriminative, such that—unlike existing table understanding models—any analysis of a table includes a full parse tree that assigns precise data model labels to all its regions (including nested subregions) thereby specifying the logical relations between the table's elements. Additionally, probabilities on the production rules support thresholding (or ranking) of the alternative candidate table interpretation hypotheses.

As with many natural language phenomena, a full model of disambiguation must ultimately integrate lexical semantics. However, in this research step we focus on the question of how much semantic interpretation can be performed on the basis of other features, in the absence of a lexical or ontological model. Just as syntax and morphology and prosody alone already permit much recognition and disambiguation of semantic roles and argument structure to be done for sentence, the same can be done for tables. At the same time, we believe future integration of lexical semantics will be facilitated by the grammatical framework of our model.

One way to think about this is that we wish to model what you might be able to recognize from a "Martian" table such as that in Table 1. The non-Martian reader relies solely on knowledge of alphabets and numbers, can spot font and formatting clues, and is familiar with the conventions (i.e., grammars) of tables in general.

Table 1: Example "Martian" table (see text).

Pbje	Kwe	Zxc	Amc
Hoer	15 - 18	17 - 20	19 - 23
NQ	85 - 95%	70 - 90%	75 - 95%
Ncowifl	Djhi	Djhi	Rubzlx

You might reasonably interpret this table as a collection of vertical records with an attributes header column *(Pbje, Hoer, NQ, Ncowifl)* on the left. You might additionally interpret it as a table that contains an record key header row *(Kwe, Zxc, Amc)* along with the attributes header column *(Pbje, Hoer, NQ, Ncowifl)*. You might assign the latter interpretation a slightly higher probability, noticing the slightly longer form of *Pbje* compared to *Kwe*, *Zxc*, and *Amc*. On the other hand, even without reading English, you could reject the interpretation as a collection of horizontal records under the header attributes row *(Pbje, Kwe, Zxc, Amc)*, since each row contains different forms and types, in a pattern that is consistent across columns. Other interpretations are also possible, but unlikely given the regularity of the patterns.

Thus by analyzing the structure of a table, the reader would form a hypothesis about its data model, providing a semantic interpretation that allows the reader to extract information from the table. As can be seen from the restored original English version of the same example in Table 2, the most likely interpretation was predicted even without access to specific lexical knowledge. We aim to show that a fairly useful baseline level of semantic interpretation accuracy can already be achieved, even with relatively little lexical and ontological knowledge.

We model these alternative hypotheses for the interpretation of ambiguous tables as competing parses. Just as with ordinary parsing and semantic interpretation, the reader often builds multiple competing interpretations of the same table.

Note that many previous models do not even distinguish between the alternative possible interpretations in the Martian example. Existing mod-

els such as Hurst (2000) and Yang (2002) interpret tables with the same structural layout simply by assigning them same data model, which stops short of recognizing that it is necessary to rank multiple competing interpretations that entail different sets of logical relations.

In contrast, our proposed model is capable of producing multiple competing parses indicating different semantic interpretations of tables having the same structural layout, by selecting specific data models for the table and its subregions.

2 Data Models for Specifying Semantic Interpretations

To begin, some formal basis is needed to facilitate precise specification of the alternative semantic interpretations of a table, such that the exact logical relations between its elements are unambiguously specified. This will enable us to then design a table understanding model that attempts to map any given table (and recursively, its subregions) to alternative data models depending on which is most appropriate.

The set of data models we define below is a more comprehensive and precise inventory than found in the previous table analysis models discussed in this paper. It describes all the common conventional patterns of logical relations we have found in the course of empirically analyzing tables from corpora. One advantage of this inventory of data models arises from our appropriation of relational database theory wherever possible to help describe the form of the data models (Silberschatz *et al.*, 2002), allowing broad coverage of different table types without sacrificing precision as to the logical relations between entities.

Each data model assigns a clear semantics in terms of logical relations between the table elements, thereby allowing extraction of relational facts. In contrast, previous work on table analysis tends to either classify a table using only one single limited data model (e.g., Hurst (2000)), or using data models which essentially are merely surface layout types whose semantics are vague and ambiguous (e.g., Yang (2002), Yang and Luk (2002), Wang *et al.* (2000), Yoshida *et al.* (2001)).

A table is a logical view of a collection of interrelated items usually presented as a row-column structure such that the reader's ability to access and compare information can be enhanced, as also noted by Wang (1996). From a database management system perspective, each table can be considered as a (tiny) database. Like a program, the reader accesses the data. As a result, we consider that every table must correspond to a data model, and this model determines how the reader extracts information from the table.

Table 2: Example from Table 1 in its original version, with the English words restored.

Date	Thu	Fri	Sat
Temp	15 - 18	17 - 20	19 - 23
RH	85 - 95%	70 - 90%	75 - 95%
Weather	Cool	Cool	Cloudy

Each data model has a schema which, as we shall see below, may or may not surface (partially or completely) as a subset of cells in the table that describe attributes. Recognizing the data models of a table correctly therefore also implies that both attribute-value pairings and table structures have been recognized.

At the top level, we categorize the data models into three broad types:

- Flat model: A table is interpreted as a database table in non-1NF normal relational model.
- Nested model: A table is interpreted as a database table in an object-relational model, which allow complex types such as nested relations and concept hierarchy.
- Dimensional data model: A table (usually cross-tabular) is interpreted as a data cube (multidimensional table) in a multidimensional data model.

We now consider each of these types of data models in turn.

2.1 Flat model

A flat model is used for the semantic interpretation of any table as a relational database table in non-1NF. For example, tables such as Tables 2 and 3 are often interpreted by humans in terms of flat models. It is obvious that Table 3 can be viewed as a relational database table with a schema (Pos, Teams, Pld, Pts) and three records, because the table's surface form resembles how records are stored in a relational database tables. Similarly, Table 2 resembles a relational database table, but transposed to a vertical orientation, with the first

Table 3: Example of a ranking table, which is typically laid out in a flat relational model.

Pos	Teams	Pld	Pts
1.	Chelsea	38	95
2.	Arsenal	38	83
3.	Man United	38	77

column as the schema (Date, Temp, RH, Weather) and other columns as data records.

The flat model is closest to the 1-dimensional table approach used by the majority of previous models, but our approach designates the flat model as a semantic representation, in contrast to the previous models which see 1-dimensional tables merely as a syntactic surface form (e.g., Yang (2002), Yang and Luk (2002), Wang *et al.* (2000), Yoshida *et al.* (2001)). While such previous models only recognize tables that are physically laid out in this form, our approach clearly delineates an explicit separation of syntax and semantics, which provides greater flexibility allowing any table to be interpreted as a flat model, regardless of its surface form (though the flat model interpretation is more common for some surface forms than others).

As an example showing that any kind of table can be categorized as flat model, consider Table 6. Even such a table can be semantically interpreted as a flat model because related attributes can join together to form a composite attribute, though humans would less naturally choose this semantic interpretation. Certainly there are hierarchical relationship between attributes; for example, Ass1 is a subtype of Assignments. However, it is also valid to consider the attributes along a hierarchical path as one composite attribute. For example, "Mark -> Assignments -> Ass1" becomes the single attribute "Mark-Assignments-Ass1". Then the complete flat model schema is (Year, Team, Mark-Assignments-Ass1, Mark-Assignments-Ass2, Mark-Assignments-Ass3, Mark-Examinations-Midterm, Mark-Examinations-Final), and the first record is (1991, Winter, 85, 80, 75, 60, 75, 75).

2.2 Nested model

With the exception of Hurst (2000), previous work has not generally considered nested models in explicit fashion. Hurst (2000)'s model is based on Wang (1996)'s abstract table model, in which attributes may be related in a hierarchical way. On the other hand, Wang *et al.* (2000) oversimplistically considers nested models as 1-dimensional, thus missing the correct relationships between attributes and values.

A nested model can be seen as a generalization of the flat model, in which attributes may be related through composition or inheritance. Table 6 is naturally interpreted as a nested data model because the attributes have an inheritance relationship. The corresponding schema is (Year, Team, Mark (Assignments (Ass1, Ass2, Ass3), Examinations (Midterm, Final, Grade)).

A nested model is not appropriate for tables without hierarchical structure, such as Table 2 and Table 3.

2.3 Dimensional model

Our approach also nicely handles dimensional models, which are generally handled quite weakly in previous models. A dimensional model refers to a table, such as the table in Table 4, that resembles a view of collection of data stored in multidimensional data model. A multidimensional data cube, as described in the database literature (e.g., Han and Kamber (2000), Chaudhuri and Dayal (1997)), consists of a set of numeric measures (though in fact the data need not be numeric), each of which is determined by a set of dimensions. Each dimension is described by a set of attributes. For example, Table 5 can be semantically interpreted using the multidimensional data model depicted in Figure 1. Likewise, the cross-tabular table in Table 4 can also be semantically interpreted using the same multidimensional data model in Figure 1. The value of the first three columns in Table 5 are the dimension attributes and the revenue values are the measures.

In contrast, among previous models, Yang (2002) produces a semantically incorrect recognition of a multidimensional table that inappropriately presents the attributes in hierarchical structure. Yang and Luk (2002) and Wang *et al.* (2000) only recognize the simplest 2-dimensional case and apparently cannot handle 3 or more dimensions. Yoshida *et al.* (2001) only handle 1-dimensional cases.

A dimensional model is an inappropriate interpretation for non-cross-tabular tables, such as Table 2 and Table 3. A dimensional model is also not valid for tables such as Table 6. Semantically, it is not possible for "Assignments" and "Midterm"

Table 4: Example table showing revenue according to Location = {Vancouver, Victoria}, Type = {Phone, Computer} and Time = {2001, 2002}, using a tabular view of a 3-dimensional data cube.

	Vancouver		Victoria	
	Phone	Computer	Phone	Computer
2001	845	1078	818	968
2002	943	1130	894	1024

Table 5: Example relational database table containing the same logical information as Table 4.

Location	*Type*	*Time*	*Revenue*
Vancouver	Phone	2001	845
Vancouver	Phone	2002	943
Vancouver	Computer	2001	1078
Vancouver	Computer	2002	1130
Victoria	Phone	2001	818
Victoria	Phone	2002	894
Victoria	Computer	2001	968
Victoria	Computer	2002	1024

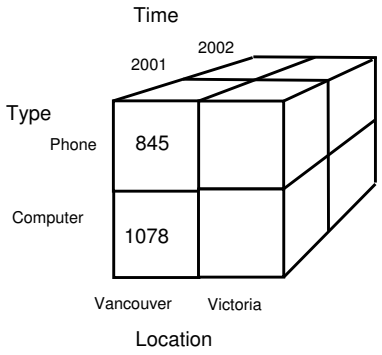


Figure 1: Multidimensional data cube corresponding to Tables 4 and 5.

to belong to different dimensions because it is incorrect to determine the score by both "Assignments" and "Midterm". Syntactically, the texts in the last attribute row of Table 6 are all unique; however, the last attribute row of the table in Table 4 is a repeating sequence of ("Phone", "Computer"). Therefore, to a non-English reader, an English cross-tabular table which possess repeating sequences in the attribute rows is likely to be semantically interpreted as a dimensional model, while a cross-tabular table which does not have this property is likely to be interpreted as a nested model.

Table 6: Example table of grades.

Year	Team	Mark					
		Assignments			Examinations		
		Ass1	Ass2	Ass3	Midterm	Final	Grade
1991	Winter	85	80	75	60	75	75
	Spring	80	65	75	60	70	70
	Fall	80	85	75	55	80	75
1992	Winter	85	80	70	70	75	75
	Spring	80	80	70	70	75	75
	Fall	75	70	65	60	80	70

3 A 2D SCFG Model for Table Analysis

In this section, we will present our two-dimensional SCFG parsing model for table analysis which has several advantages over the ad hoc approaches. First, the probabilistic grammar approach permits a cleaner encapsulation and generalization of the kind of knowledge that previous models attempted to capture within their ad hoc heuristics. Most previous works (e.g. Yang (2002), Yang and Luk (2002), Hurst (2000), Hurst (2002)) gradually built up their ad hoc heuristics manually by inspecting some set of training samples. This approach may work if tables are from limited domains of similar nature. However, like text documents, the syntactic layout of textual tables may be determined by its context as well as its language. For instance, it is natural for an Arabic reader to read an Arabic table taking the rightmost column as the attribute column, instead of the leftmost column. Yoshida *et al.* (2001) use machine-learning techniques to analyze nine types of table structures, all 1-dimensional. Our grammar-based approach allows the model to be readily adapted to different situations by applying different sets of grammar rules.

Another advantage is that grammatical approach can make more accurate decisions while being simpler to implement, because it requires only a single integrated parsing process to complete the entire table analysis. This includes classifying the functions of each cell (as attribute or value), pairing attributes and values, and identifying the structure and the data model of a table. In contrast, previous works require several stages to complete the entire analysis, introducing complex

problems that are difficult to resolve, such as premature commitment to incorrect early-stage decisions.

To our knowledge Wang *et al.* (2000) is the only textual table analysis model that uses a grammar to describe table structures. However, in that case, only a simple template matching analyzer is used. Their grammar notation is unable to show both physical structure and the semantics of a table at the same time in a hierarchical manner. In contrast, information such as "a data block contains three rows of data cell" can be stored in the parse tree constructed by our parsing model.

Outside of the table understanding literature, there exists a different 2D parsing technique called PLEX (Feder, 1971), (Costagliola *et al.*, 1994) which allows an object to have finite sets of attaching points. PLEX is used to generate 2D diagrams such as molecular structures, circuit diagrams and flow charts in a grammatical way. However, we consider it too complex and computationally expensive for our application because it does not exploit that fact that a textual table cell only has at most four attaching points in fixed directions.

Our parser is a two-dimensional extension of the conventional probabilistic chart parsing algorithm (Lari and Young, 1990), (Goodman, 1998). Intuitively, consider a sentence as a vector of tokens that will be parsed horizontally; then a table is a matrix of tokens (like a crossword puzzle) that will be parsed both horizontally and vertically. Because of this, our parser must run in both directions. We achieve this by employing a grammar notation that specifies the direction of parsing.

The two-dimensional grammar notation includes of a set of nonterminals, terminals, and two generation operators "–>" and "|->". Let X be a nonterminal and Y, Z, be two symbols which may be either nonterminals or terminals. Then:

- X –> Y Z denotes a horizontal production rule saying that the nonterminal X horizontally generates two symbols Y and Z.
- X |-> Y Z denotes a vertical production rule saying that the nonterminal X vertically generates two symbols Y and Z.
- X –> Y or X |-> Y equivalently denote a unary production rule saying that the nonterminal X generates a symbol Y.

We assume that all rules are binary without loss of generality, since any grammar can be mechanically binarized without materially changing the parse tree structure, just as in the case of ordinary 1D grammars.

The operators "–>" and "|->" control the generation direction. In term of table analysis, a nonterminal represents a matrix of tokens and a terminal represents a single token. Sub-matrices generated by a horizontal rule will have same height but not necessarily same width; similarly, submatrices generated by a vertical rule will have same width but not necessarily same height. In other words, a matrix is partitioned into two halves by the binary production rule.

Probabilities are placed on each rule, as in ordinary 1D SCFGs. They are used to eliminate parses falling below a threshold, which also helps to reduce the time complexity in practice.

Parsing with two-dimensional grammars can be conceptualized most easily via parse tree examples. Figure 2 shows a complete parse tree for parsing the table in Table 7 into a flat model. Figure 3 is a portion of a parse tree for parsing the table in Table 8 into a nested model, while Figure 4 is a portion of parse trees for parsing Table 7 into a dimensional model. The following is the grammar fragment that gives the parse tree as Figure 2:

```
T1-1H |-> FlatModel
FlatModel |-> FlatSchema Records
FlatSchema --> CompositeAttribute FlatSchema
FlatSchema --> CompositeAttribute
Records |-> Record Records
Records |-> Record
Record --> Data Record
Record --> Record
```

Note that the internal nodes of the parse trees serve to label subregions with data models, thus assigning a semantic interpretation specifying the exact logical relations between table elements. None of the previous models construct declarative parse trees like these, which are necessary for many types of subsequent analysis, including information extraction applications.

4 Experimental Method

To the best of our knowledge, unfortunately none of the table corpora mentioned in previous work are available to the public. Thus, it was necessary to construct a corpus for our experiments. We collected a large sample of tables by issuing Google searches with a list of random keywords, for example, *census age*, *confusion matrix*, *data table*, *movie ranking*, *MSFT*, *school ranking*, *telephone plan*, *tsunami numbers*, *weather report*, and

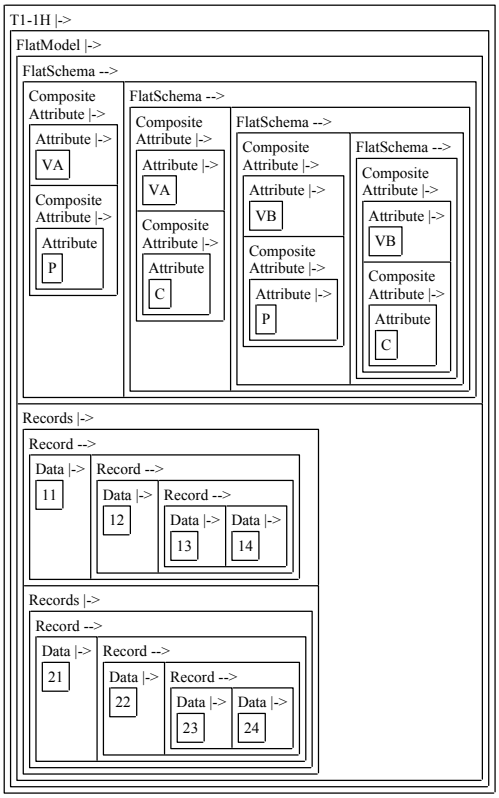


Figure 2: A parse tree for a flat model.

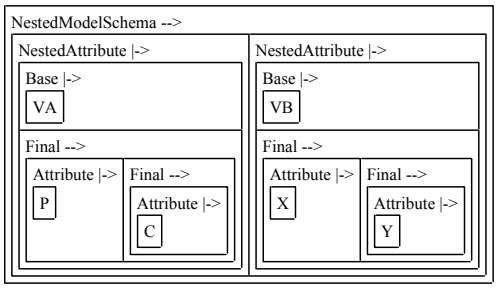


Figure 3: A partial parse for a nested model.

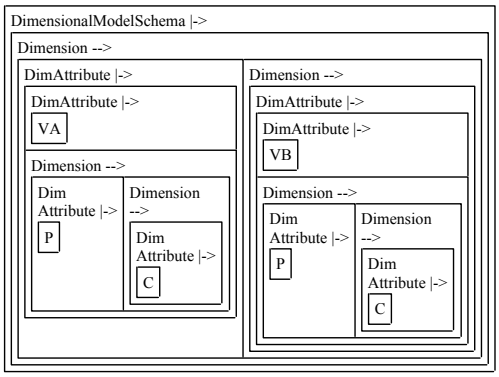


Figure 4: A partial parse for a dimensional model.

so on. Tables were extracted from the collected sample, automatically cleaned, and tokenized into two-dimensional array of tokens.

Table 7: Example table for Figures 2 and 4.

VA		VB	
P	C	P	C
11	12	13	14
21	22	23	24

Table 8: Example table for Figure 3.

VA		VB	
P	C	X	Y
11	12	13	14
21	22	23	24

Table 9: Example table showing a floor legend.

6	School of Business & Management
5	Department of Biochemistry
4	Classrooms 4202 - 4205
3	Department of Computer Science
3	Department of Mathematics

For the blind evaluation, a human annotator independently manually annotated a randomly chosen sample of 45 tables from the collection. All tables in the evaluation sample were previously unseen test cases, never inspected prior to the construction of the two-dimensional grammar.

Each tokenized table was tagged by the human judge with a list of types T relevant to the table. The relevance is defined as follows: a data model is relevant to a table if and only if the human would agree that such a data model would naturally be hypothesized as an interpretation for that table (analogously to the way that word senses are manually annotated for WSD evaluations). Each type is a tuple of the form (R, O, S), where R is the relevant data model, O is the reading orientation of R, and S is a boolean saying if a schema (i.e. attributes) exist in the table. Thus, Table 2 would be tagged as {(flat, vertical, true)} while the table in Table 4 would be tagged as {(flat, horizontal, true), (flat, vertical, true), (dimensional, _, true)}. But Table 9 may be tagged as {(flat, horizontal, false)}. The exceptions are that both the nested model and the dimensional model always have a schema, while the dimensional model does not have orientation. In cases where multiple legitimate readings were possible, the table was tagged

Table 10: Experimental results.

Precision	Recall
0.60	0.80

with multiple types. A total of 92 relevant types were generated from the tokenized tables.

We processed the tokenized tables with the two-dimensional SCFG parser, and computed the precision and the recall rates against the judge's lists of tags for all the test cases.

5 Results and Discussion

The experimental results are summarized in Table 10. All tables could be parsed; in general, it is very rare for any table to be rejected by the parser, since the grammar permits so many different configurations that can be recursively composed.

Unfortunately it is impossible to compare results directly against previous models, since neither those models nor the data they evaluated on are available.

Moreover, it is difficult to compare with previous models as our evaluation criteria are more stringent than in earlier work. Most previous work evaluated the performance in terms of the (vaguer and less demanding) criteria of number of correct attribute-value pairings. Such an evaluation approach gives unduly high weight to large repetitive tables, and neglects structural errors in the analysis of the table. In contrast, our approach gives equal weight to all tables regardless of how many entries they contain, requires semantically valid structural analyses, and yet still accepts any parse that yields the correct attribute-value pairings (since the tagging of the test set includes all legitimate types when there are multiple valid alternatives).

The fact that precision was lower than recall is due to the fact that many tables were wrongly interpreted as tables without schema or in wrong orientations. The current grammar has difficulty distinguishing attributes from values. Significant improvement can be obtained by using constraints to limit the number of incorrect parses, a strategy we are currently implementing.

6 Conclusion

We have introduced a framework to support a more linguistically-oriented approach to finer interpretation of tables, using two-dimensional stochastic CFGs with Viterbi parsing to find appropriate semantic interpretations of textual tables in terms of different data models. This approach yields a concise model that at the same time facilitates broader coverage than existing models, and is more easily scalable and maintainable. We also introduce a cleaner and richer data model to represent semantic interpretations, and illustrate how it systematically captures a wider range of table types. Without such a data model, the right attribute-value relations caanot be extracted from a table, even if surface elements like "header" and "data" are correctly labeled as previous models attempted to do. Our experiments show that even without other ontological and linguistic knowledge, excellent semantic interpretation accuracy can be obtained by parsing with a two-dimensional grammar based on these data models, by using a wide variety of surface features in the terminal symbols. We plan next to extend the model by incorporating ontological and linguistic knowledge for additional disambiguation leverage.

References

Surajit Chaudhuri and Umesh Dayal. An overview of data warehousing and OLAP technology. *ACM SIGMOD Record*, 26(1), March 1997.

Gennaro Costagliola, Andrea De Lucia, and Sergio Orefice. Towards efficient parsing of diagrammatic languages. In *AVI '94: Proceedings of the workshop on Advanced visual interfaces*, pages 162–171, New York, NY, USA, 1994. ACM Press.

Jerome Feder. Plex languages. *Information Sciences*, 3:225–241, 1971.

Joshua T. Goodman. *Parsing Inside-Out*. PhD thesis, Harvard University, 1998.

Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 1 edition, 2000.

Matthew Francis Hurst. *The Interpretation of Tables in Texts*. PhD thesis, The University of Edinburgh, 2000.

Matthew Hurst. Classifying table elements in html. In *The 11th International World Wide Web Conference, Hawaii, USA*, 2002.

K. Lari and S. J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–36, 1990.

Richard Power, Donia Scott, and Nadjet Bouayad-Agha. Document structure. *Computational Linguistics*, 29(4):211–260, Dec 2003.

Donia Scott. Layout in NLP: The case for document structure (invited talk). In *41st Annual Meeting of the Association for Computational Linguistics (ACL-2003)*, Aug 2003.

Abraham Silberschatz, Henry F. Korth, and S. Sudarshan. *Database System Concepts*. McGraw-Hill, 4th edition, 2002.

H. L. Wang, S. H. Wu, I. C. Wang, C. L. Sung, W. L. Hsu, and W. K. Shih. Semantic search on internet tabular information extraction for answering queries. In *CIKM '00: Proceedings of the ninth international conference on Information and knowledge management*, pages 243–249, New York, NY, USA, 2000. ACM Press.

Xinxin Wang. *Tabular Abstraction, Editing, and Formatting*. PhD thesis, The University of Waterloo, Waterloo, Ontario, Canada, 1996.

Yingchen Yang and Wo-Shun Luk. A framework for web table mining. In *WIDM '02: Proceedings of the 4th international workshop on Web information and data management*, pages 36–42, New York, NY, USA, 2002. ACM Press.

Yingchen Yang. Web table mining and database discovery. Master's thesis, Simon Fraser University, August 2002.

M. Yoshida, K. Torisawa, and J. Tsujii. A method to integrate tables of the world wide web, 2001.

Boosting Statistical Word Alignment Using Labeled and Unlabeled Data

Hua Wu Haifeng Wang Zhanyi Liu
Toshiba (China) Research and Development Center
5/F., Tower W2, Oriental Plaza, No.1, East Chang An Ave., Dong Cheng District
Beijing, 100738, China
{wuhua, wanghaifeng, liuzhanyi}@rdc.toshiba.com.cn

Abstract

This paper proposes a semi-supervised boosting approach to improve statistical word alignment with limited labeled data and large amounts of unlabeled data. The proposed approach modifies the supervised boosting algorithm to a semi-supervised learning algorithm by incorporating the unlabeled data. In this algorithm, we build a word aligner by using both the labeled data and the unlabeled data. Then we build a pseudo reference set for the unlabeled data, and calculate the error rate of each word aligner using only the labeled data. Based on this semi-supervised boosting algorithm, we investigate two boosting methods for word alignment. In addition, we improve the word alignment results by combining the results of the two semi-supervised boosting methods. Experimental results on word alignment indicate that semi-supervised boosting achieves relative error reductions of 28.29% and 19.52% as compared with supervised boosting and unsupervised boosting, respectively.

1 Introduction

Word alignment was first proposed as an intermediate result of statistical machine translation (Brown et al., 1993). In recent years, many researchers build alignment links with bilingual corpora (Wu, 1997; Och and Ney, 2003; Cherry and Lin, 2003; Wu et al., 2005; Zhang and Gildea, 2005). These methods *unsupervisedly* train the alignment models with unlabeled data.

A question about word alignment is whether we can further improve the performances of the word aligners with available data and available alignment models. One possible solution is to use the boosting method (Freund and Schapire, 1996), which is one of the ensemble methods (Dietterich, 2000). The underlying idea of boosting is to combine simple "rules" to form an ensemble such that the performance of the single ensemble is improved. The AdaBoost (**Ada**ptive **Boost**ing) algorithm by Freund and Schapire (1996) was developed for supervised learning. When it is applied to word alignment, it should solve the problem of building a reference set for the unlabeled data. Wu and Wang (2005) developed an *unsupervised* AdaBoost algorithm by automatically building a pseudo reference set for the unlabeled data to improve alignment results.

In fact, large amounts of unlabeled data are available without difficulty, while labeled data is costly to obtain. However, labeled data is valuable to improve performance of learners. Consequently, *semi-supervised learning*, which combines both labeled and unlabeled data, has been applied to some NLP tasks such as word sense disambiguation (Yarowsky, 1995; Pham et al., 2005), classification (Blum and Mitchell, 1998; Thorsten, 1999), clustering (Basu et al., 2004), named entity classification (Collins and Singer, 1999), and parsing (Sarkar, 2001).

In this paper, we propose a *semi-supervised* boosting method to improve statistical word alignment with both limited labeled data and large amounts of unlabeled data. The proposed approach modifies the supervised AdaBoost algorithm to a semi-supervised learning algorithm by incorporating the unlabeled data. Therefore, it should address the following three problems. The first is to build a word alignment model with both labeled and unlabeled data. In this paper, with the labeled data, we build a *supervised model* by directly estimating the parameters in

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 913–920,
Sydney, July 2006.

the model instead of using the Expectation Maximization (EM) algorithm in Brown et al. (1993). With the unlabeled data, we build an *unsupervised model* by estimating the parameters with the EM algorithm. Based on these two word alignment models, an *interpolated model* is built through linear interpolation. This interpolated model is used as a learner in the semi-supervised AdaBoost algorithm. The second is to build a reference set for the unlabeled data. It is automatically built with a modified "refined" combination method as described in Och and Ney (2000). The third is to calculate the error rate on each round. Although we build a reference set for the unlabeled data, it still contains alignment errors. Thus, we use the reference set of the labeled data instead of that of the entire training data to calculate the error rate on each round.

With the interpolated model as a learner in the semi-supervised AdaBoost algorithm, we investigate two boosting methods in this paper to improve statistical word alignment. The first method uses the unlabeled data only in the interpolated model. During training, it only changes the distribution of the labeled data. The second method changes the distribution of both the labeled data and the unlabeled data during training. Experimental results show that both of these two methods improve the performance of statistical word alignment.

In addition, we combine the final results of the above two semi-supervised boosting methods. Experimental results indicate that this combination outperforms the unsupervised boosting method as described in Wu and Wang (2005), achieving a relative error rate reduction of 19.52%. And it also achieves a reduction of 28.29% as compared with the supervised boosting method that only uses the labeled data.

The remainder of this paper is organized as follows. Section 2 briefly introduces the statistical word alignment model. Section 3 describes parameter estimation method using the labeled data. Section 4 presents our semi-supervised boosting method. Section 5 reports the experimental results. Finally, we conclude in section 6.

2 Statistical Word Alignment Model

According to the IBM models (Brown et al., 1993), the statistical word alignment model can be generally represented as in equation (1).

$$\Pr(\mathbf{a},\mathbf{f}\mid\mathbf{e})=\frac{\Pr(\mathbf{a},\mathbf{f}\mid\mathbf{e})}{\sum_{\mathbf{a}'}\Pr(\mathbf{a}',\mathbf{f}\mid\mathbf{e})} \quad (1)$$

Where **e** and **f** represent the source sentence and the target sentence, respectively.

In this paper, we use a simplified IBM model 4 (Al-Onaizan et al., 1999), which is shown in equation (2). This simplified version does not take into account word classes as described in Brown et al. (1993).

$$\begin{aligned}\Pr(\mathbf{a},\mathbf{f}\mid\mathbf{e})=&\binom{m-\phi_0}{\phi_0}p_0^{m-2\phi_0}p_1^{\phi_0}\cdot\\&\prod_{i=1}^{l}n(\phi_i\mid e_i)\cdot\prod_{j=1}^{m}t(f_j\mid e_{a_j})\cdot\\&\Big(\prod_{j=1,a_j\neq 0}^{m}([j=h(a_j)]\cdot d_1(j-c_{\rho_{a_j}}))+\\&\prod_{j=1,a_j\neq 0}^{m}([j\neq h(a_j)]\cdot d_{>1}(j-p(j)))\Big)\end{aligned} \quad (2)$$

l, m are the lengths of the source sentence and the target sentence respectively.

j is the position index of the target word.

a_j is the position of the source word aligned to the j^{th} target word.

ϕ_i is the number of target words that e_i is aligned to.

p_0, p_1 are the fertility probabilities for e_0, and $p_0+p_1=1$.

$t(f_j\mid e_{a_j})$ is the word translation probability.

$n(\phi_i\mid e_i)$ is the fertility probability.

$d_1(j-c_{\rho_{a_j}})$ is the distortion probability for the head word of cept[1] *i*.

$d_{>1}(j-p(j))$ is the distortion probability for the non-head words of cept *i*.

$h(i)=\min_k\{k: i=a_k\}$ is the head of cept *i*.

$p(j)=\max_{k<j}\{k: a_j=a_k\}$.

ρ_i is the first word before e_i with non-zero fertility.

c_i is the center of cept *i*.

3 Parameter Estimation with Labeled Data

With the labeled data, instead of using EM algorithm, we directly estimate the three main parameters in model 4: translation probability, fertility probability, and distortion probability.

[1] A cept is defined as the set of target words connected to a source word (Brown et al., 1993).

3.1 Translation Probability

The translation probability is estimated from the labeled data as described in (3).

$$t(f_j \mid e_i) = \frac{count(e_i, f_j)}{\sum_{f'} count(e_i, f')} \quad (3)$$

Where $count(e_i, f_j)$ is the occurring frequency of e_i aligned to f_j in the labeled data.

3.2 Fertility Probability

The fertility probability $n(\phi_i \mid e_i)$ describes the distribution of the numbers of words that e_i is aligned to. It is estimated as described in (4).

$$n(\phi_i \mid e_i) = \frac{count(\phi_i, e_i)}{\sum_{\phi'} count(\phi', e_i)} \quad (4)$$

Where $count(\phi_i, e_i)$ describes the occurring frequency of word e_i aligned to ϕ_i target words in the labeled data.

p_0 and p_1 describe the fertility probabilities for e_0. And p_0 and p_1 sum to 1. We estimate p_0 directly from the labeled data, which is shown in (5).

$$p_0 = \frac{\#\,Aligned - \#\,Null}{\#\,Aligned} \quad (5)$$

Where $\#\,Aligned$ is the occurring frequency of the target words that have counterparts in the source language. $\#\,Null$ is the occurring frequency of the target words that have no counterparts in the source language.

3.3 Distortion Probability

There are two kinds of distortion probability in model 4: one for head words and the other for non-head words. Both of the distortion probabilities describe the distribution of relative positions Thus, if we let $\Delta j_1 = j - c_{\rho_i}$ and $\Delta j_{>1} = j - p(j)$, the distortion probabilities for head words and non-head words are estimated in (6) and (7) with the labeled data, respectively.

$$d_1(\Delta j_1) = \frac{\sum_{j, c_{\rho_i}} \delta(\Delta j_1, j - c_{\rho_i})}{\sum_{\Delta j_1'} \sum_{j', c_{\rho_{i'}}} \delta(\Delta j_1', j' - c_{\rho_{i'}})} \quad (6)$$

$$d_{>1}(\Delta j_{>1}) = \frac{\sum_{j, p(j)} \delta(\Delta j_{>1}, j - p(j))}{\sum_{\Delta j_{>1}'} \sum_{j', p(j')} \delta(\Delta j_{>1}', j' - p(j'))} \quad (7)$$

Where $\delta(x, y) = 1$ if $x = y$. Otherwise, $\delta(x, y) = 0$.

4 Boosting with Labeled Data and Unlabeled Data

In this section, we first propose a semi-supervised AdaBoost algorithm for word alignment, which uses both the labeled data and the unlabeled data. Based on the semi-supervised algorithm, we describe two boosting methods for word alignment. And then we develop a method to combine the results of the two boosting methods.

4.1 Semi-Supervised AdaBoost Algorithm for Word Alignment

Figure 1 shows the semi-supervised AdaBoost algorithm for word alignment by using labeled and unlabeled data. Compared with the supervised Adaboost algorithm, this semi-supervised AdaBoost algorithm mainly has five differences.

Word Alignment Model

The first is the word alignment model, which is taken as a learner in the boosting algorithm. The word alignment model is built using both the labeled data and the unlabeled data. With the labeled data, we train a supervised model by directly estimating the parameters in the IBM model as described in section 3. With the unlabeled data, we train an unsupervised model using the same EM algorithm in Brown et al. (1993). Then we build an interpolation model by linearly interpolating these two word alignment models, which is shown in (8). This interpolated model is used as the model M_l described in figure 1.

$$\begin{aligned} &\Pr(\mathbf{a}, \mathbf{f} \mid \mathbf{e}) \\ &= \lambda \cdot \Pr_S(\mathbf{a}, \mathbf{f} \mid \mathbf{e}) + (1 - \lambda) \cdot \Pr_U(\mathbf{a}, \mathbf{f} \mid \mathbf{e}) \end{aligned} \quad (8)$$

Where $\Pr_S(\mathbf{a}, \mathbf{f} \mid \mathbf{e})$ and $\Pr_U(\mathbf{a}, \mathbf{f} \mid \mathbf{e})$ are the trained supervised model and unsupervised model, respectively. λ is an interpolation weight. We train the weight in equation (8) in the same way as described in Wu et al. (2005).

Pseudo Reference Set for Unlabeled Data

The second is the reference set for the unlabeled data. For the unlabeled data, we automatically build a *pseudo* reference set. In order to build a reliable pseudo reference set, we perform bi-directional word alignment on the training data using the interpolated model trained on the first round. Bi-directional word alignment includes alignment in two directions (source to

Input: A training set S_T including m bilingual sentence pairs;
The reference set R_T for the training data;
The reference sets R_L and R_U ($R_L, R_U \subseteq R_T$) for the labeled data S_L and the unlabeled data S_U respectively, where $S_T = S_U \cup S_L$ and $S_U \cap S_L = \text{NULL}$;
A loop count L.

(1) Initialize the weights:
$w_1(i) = 1/m, \quad i = 1,...,m$
(2) For $l = 1$ to L, execute steps (3) to (9).
(3) For each sentence pair i, normalize the weights on the training set:
$p_l(i) = w_l(i) / \sum_j w_l(j), \quad i = 1,...,m$
(4) Update the word alignment model M_l based on the weighted training data.
(5) Perform word alignment on the training set with the alignment model M_l:
$h_l = M_l(p_l)$
(6) Calculate the error of h_l with the reference set R_L: $\varepsilon_l = \sum_i p_l(i) \cdot \alpha(i)$
Where $\alpha(i)$ is calculated as in equation (9).
(7) If $\varepsilon_l > 1/2$, then let $L = l - 1$, and end the training process.
(8) Let $\beta_l = \varepsilon_l / (1 - \varepsilon_l)$.
(9) For all i, compute new weights:
$w_{l+1}(i) = w_l(i) \cdot (k + (n-k) \cdot \beta_l) / n$
where, n represents n alignment links in the i^{th} sentence pair. k represents the number of error links as compared with R_T.

Output: The final word alignment result for a source word e:

$$h_F(e) = \arg\max_f RS(e, f) = \arg\max_f \sum_{l=1}^{L} (\log \frac{1}{\beta_l}) \cdot WT_l(e, f) \cdot \delta(h_l(e), f)$$

Where $\delta(x, y) = 1$ if $x = y$. Otherwise, $\delta(x, y) = 0$. $WT_l(e, f)$ is the weight of the alignment link (e, f) produced by the model M_l, which is calculated as described in equation (10).

Figure 1. The Semi-Supervised Adaboost Algorithm for Word Alignment

target and target to source) as described in Och and Ney (2000). Thus, we get two sets of alignment results A_1 and A_2 on the unlabeled data. Based on these two sets, we use a modified "refined" method (Och and Ney, 2000) to construct a pseudo reference set R_U.

(1) The intersection $I = A_1 \cap A_2$ is added to the reference set R_U.
(2) We add $(e, f) \in A_1 \cup A_2$ to R_U if a) is satisfied or both b) and c) are satisfied.
   a) Neither e nor f has an alignment in R_U and $p(f|e)$ is greater than a threshold δ_1.
   $$p(f|e) = \frac{count(e, f)}{\sum_{f'} count(e, f')}$$
   Where $count(e, f)$ is the occurring frequency of the alignment link (e, f) in the bi-directional word alignment results.
   b) (e, f) has a horizontal or a vertical neighbor that is already in R_U.
   c) The set $R_U \cup (e, f)$ does not contain alignments with both horizontal and vertical neighbors.

Error of Word Aligner

The third is the calculation of the error of the individual word aligner on each round. For word alignment, a sentence pair is taken as a sample. Thus, we calculate the error rate of each sentence pair as described in (9), which is the same as described in Wu and Wang (2005).

$$\alpha(i) = 1 - \frac{2|S_W \cap S_R|}{|S_W| + |S_R|} \quad (9)$$

Where S_W represents the set of alignment links of a sentence pair i identified by the individual interpolated model on each round. S_R is the reference alignment set for the sentence pair.

With the error rate of each sentence pair, we calculate the error of the word aligner on each round. Although we build a pseudo reference set R_U for the unlabeled data, it contains alignment errors. Thus, the weighted sum of the error rates of sentence pairs in the labeled data instead of that in the entire training data is used as the error of the word aligner.

Weights Update for Sentence Pairs

The forth is the weight update for sentence pairs according to the error and the reference set. In a sentence pair, there are usually several word alignment links. Some are correct, and others may be incorrect. Thus, we update the weights according to the number of correct and incorrect alignment links as compared with the reference set, which is shown in step (9) in figure 1.

Weights for Word Alignment Links

The fifth is the weights used when we construct the final ensemble. Besides the weight $\log(1/\beta_l)$, which is the confidence measure of the l^{th} word aligner, we also use the weight $WT_l(e, f)$ to measure the confidence of each alignment link produced by the model M_l. The weight $WT_l(e, f)$ is calculated as shown in (10). Wu and Wang (2005) proved that adding this weight improved the word alignment results.

$$WT_l(e, f) = \frac{2 \times count(e, f)}{\sum_{f'} count(e, f') + \sum_{e'} count(e', f)} \quad (10)$$

Where $count(e, f)$ is the occurring frequency of the alignment link (e, f) in the word alignment results of the training data produced by the model M_l.

4.2 Method 1

This method only uses the labeled data as training data. According to the algorithm in figure 1, we obtain $S_T = S_L$ and $R_T = R_L$. Thus, we only change the distribution of the labeled data. However, we build an unsupervised model using the unlabeled data. On each round, we keep this unsupervised model unchanged, and we rebuild the supervised model by estimating the parameters as described in section 3 with the weighted training data. Then we interpolate the supervised model and the unsupervised model to obtain an interpolated model as described in section 4.1. The interpolated model is used as the alignment model M_l in figure 1. Thus, in this interpolated model, we use both the labeled and unlabeled data. On each round, we rebuild the interpolated model using the rebuilt supervised model and the unchanged unsupervised model. This interpolated model is used to align the training data.

According to the reference set of the labeled data, we calculate the error of the word aligner on each round. According to the error and the reference set, we update the weight of each sample in the labeled data.

4.3 Method 2

This method uses both the labeled data and the unlabeled data as training data. Thus, we set $S_T = S_L \cup S_U$ and $R_T = R_L \cup R_U$ as described in figure 1. With the labeled data, we build a supervised model, which is kept unchanged on each round.[2] With the weighted samples in the training data, we rebuild the unsupervised model with EM algorithm on each round. Based on these two models, we built an interpolated model as described in section 4.1. The interpolated model is used as the alignment model M_l in figure 1. On each round, we rebuild the interpolated model using the unchanged supervised model and the rebuilt unsupervised model. Then the interpolated model is used to align the training data.

Since the training data includes both labeled and unlabeled data, we need to build a pseudo reference set R_U for the unlabeled data using the method described in section 4.1. According to the reference set R_L of the labeled data, we calculate the error of the word aligner on each round. Then, according to the pseudo reference set R_U and the reference set R_L, we update the weight of each sentence pair in the unlabeled data and in the labeled data, respectively.

There are four main differences between Method 2 and Method 1.

(1) On each round, Method 2 changes the distribution of both the labeled data and the unlabeled data, while Method 1 only changes the distribution of the labeled data.
(2) Method 2 rebuilds the unsupervised model, while Method 1 rebuilds the supervised model.
(3) Method 2 uses the labeled data instead of the entire training data to estimate the error of the word aligner on each round.
(4) Method 2 uses an automatically built pseudo reference set to update the weights for the sentence pairs in the unlabeled data.

4.4 Combination

In the above two sections, we described two semi-supervised boosting methods for word alignment. Although we use interpolated models

[2] In fact, we can also rebuild the supervised model according to the weighted labeled data. In this case, as we know, the error of the supervised model increases. Thus, we keep the supervised model unchanged in this method.

for word alignment in both Method 1 and Method 2, the interpolated models are trained with different weighted data. Thus, they perform differently on word alignment. In order to further improve the word alignment results, we combine the results of the above two methods as described in (11).

$$h_{3,\mathrm{F}}(e) = \arg\max_{f}(\lambda_1 \cdot RS_1(e, f) + \lambda_2 \cdot RS_2(e, f)) \quad (11)$$

Where $h_{3,\mathrm{F}}(e)$ is the combined hypothesis for word alignment. $RS_1(e, f)$ and $RS_2(e, f)$ are the two ensemble results as shown in figure 1 for Method 1 and Method 2, respectively. λ_1 and λ_2 are the constant weights.

5 Experiments

In this paper, we take English to Chinese word alignment as a case study.

5.1 Data

We have two kinds of training data from general domain: Labeled Data (LD) and Unlabeled Data (UD). The Chinese sentences in the data are automatically segmented into words. The statistics for the data is shown in Table 1. The labeled data is manually word aligned, including 156,421 alignment links.

Data	# Sentence Pairs	# English Words	# Chinese Words
LD	31,069	255,504	302,470
UD	329,350	4,682,103	4,480,034

Table 1. Statistics for Training Data

We use 1,000 sentence pairs as testing set, which are not included in LD or UD. The testing set is also manually word aligned, including 8,634 alignment links in the testing set[3].

5.2 Evaluation Metrics

We use the same evaluation metrics as described in Wu et al. (2005), which is similar to those in (Och and Ney, 2000). The difference lies in that Wu et al. (2005) take all alignment links as sure links.

If we use S_{G} to represent the set of alignment links identified by the proposed method and S_{C} to denote the reference alignment set, the methods to calculate the precision, recall, f-measure, and alignment error rate (AER) are shown in equations (12), (13), (14), and (15). It can be seen that the higher the f-measure is, the lower the alignment error rate is.

$$precision = \frac{|S_{\mathrm{G}} \cap S_{\mathrm{C}}|}{|S_{\mathrm{G}}|} \quad (12)$$

$$recall = \frac{|S_{\mathrm{G}} \cap S_{\mathrm{C}}|}{|S_{\mathrm{C}}|} \quad (13)$$

$$fmeasure = \frac{2\times|S_{\mathrm{G}} \cap S_{\mathrm{C}}|}{|S_{\mathrm{G}}|+|S_{\mathrm{C}}|} \quad (14)$$

$$AER = 1 - \frac{2\times|S_{\mathrm{G}} \cap S_{\mathrm{C}}|}{|S_{\mathrm{G}}|+|S_{\mathrm{C}}|} = 1 - fmeasure \quad (15)$$

5.3 Experimental Results

With the data in section 5.1, we get the word alignment results shown in table 2. For all of the methods in this table, we perform bi-directional (source to target and target to source) word alignment, and obtain two alignment results on the testing set. Based on the two results, we get the "refined" combination as described in Och and Ney (2000). Thus, the results in table 2 are those of the "refined" combination. For EM training, we use the GIZA++ toolkit[4].

Results of Supervised Methods

Using the labeled data, we use two methods to estimate the parameters in IBM model 4: one is to use the EM algorithm, and the other is to estimate the parameters directly from the labeled data as described in section 3. In table 2, the method "Labeled+EM" estimates the parameters with the EM algorithm, which is an unsupervised method without boosting. And the method "Labeled+Direct" estimates the parameters directly from the labeled data, which is a supervised method without boosting. "Labeled+EM+Boost" and "Labeled+Direct+Boost" represent the two supervised boosting methods for the above two parameter estimation methods.

Our methods that directly estimate parameters in IBM model 4 are better than that using the EM algorithm. "Labeled+Direct" is better than "Labeled+EM", achieving a relative error rate reduction of 22.97%. And "Labeled+Direct+Boost" is better than "Labeled+EM+Boost", achieving a relative error rate reduction of 22.98%. In addition, the two boosting methods perform better than their corresponding methods without

[3] For a non one-to-one link, if *m* source words are aligned to *n* target words, we take it as one alignment link instead of *m***n* alignment links.

[4] It is located at http://www.fjoch.com/ GIZA++.html.

Method	Precision	Recall	F-Measure	AER
Labeled+EM	0.6588	0.5210	0.5819	0.4181
Labeled+Direct	0.7269	0.6609	0.6924	0.3076
Labeled+EM+Boost	0.7384	0.5651	0.6402	0.3598
Labeled+Direct+Boost	0.7771	0.6757	0.7229	0.2771
Unlabeled+EM	0.7485	0.6667	0.7052	0.2948
Unlabeled+EM+Boost	0.8056	0.7070	0.7531	0.2469
Interpolated	0.7555	0.7084	0.7312	0.2688
Method 1	0.7986	0.7197	0.7571	0.2429
Method 2	0.8060	0.7388	0.7709	0.2291
Combination	0.8175	0.7858	0.8013	0.1987

Table 2. Word Alignment Results

boosting. For example, "Labeled+Direct+Boost" achieves an error rate reduction of 9.92% as compared with "Labeled+Direct".

Results of Unsupervised Methods

With the unlabeled data, we use the EM algorithm to estimate the parameters in the model. The method "Unlabeled+EM" represents an unsupervised method without boosting. And the method "Unlabeled+EM+Boost" uses the same unsupervised Adaboost algorithm as described in Wu and Wang (2005).

The boosting method "Unlabeled+EM+Boost" achieves a relative error rate reduction of 16.25% as compared with "Unlabeled+EM". In addition, the unsupervised boosting method "Unlabeled+EM+Boost" performs better than the supervised boosting method "Labeled+Direct+Boost", achieving an error rate reduction of 10.90%. This is because the size of labeled data is too small to subject to data sparseness problem.

Results of Semi-Supervised Methods

By using both the labeled and the unlabeled data, we interpolate the models trained by "Labeled+Direct" and "Unlabeled+EM" to get an interpolated model. Here, we use "interpolated" to represent it. "Method 1" and "Method 2" represent the semi-supervised boosting methods described in section 4.2 and section 4.3, respectively. "Combination" denotes the method described in section 4.4, which combines "Method 1" and "Method 2". Both of the weights λ_1 and λ_2 in equation (11) are set to 0.5.

"Interpolated" performs better than the methods using only labeled data or unlabeled data. It achieves relative error rate reductions of 12.61% and 8.82% as compared with "Labeled+Direct" and "Unlabeled+EM", respectively.

Using an interpolation model, the two semi-supervised boosting methods "Method 1" and "Method 2" outperform the supervised boosting method "Labeled+Direct+Boost", achieving a relative error rate reduction of 12.34% and 17.32% respectively. In addition, the two semi-supervised boosting methods perform better than the unsupervised boosting method "Unlabeled+EM+Boost". "Method 1" performs slightly better than "Unlabeled+EM+Boost". This is because we only change the distribution of the labeled data in "Method 1". "Method 2" achieves an error rate reduction of 7.77% as compared with "Unlabeled+EM+Boost". This is because we use the interpolated model in our semi-supervised boosting method, while "Unlabeled+EM+Boost" only uses the unsupervised model.

Moreover, the combination of the two semi-supervised boosting methods further improves the results, achieving relative error rate reductions of 18.20% and 13.27% as compared with "Method 1" and "Method 2", respectively. It also outperforms both the supervised boosting method "Labeled+Direct+Boost" and the unsupervised boosting method "Unlabeled+EM+Boost", achieving relative error rate reductions of 28.29% and 19.52% respectively.

Summary of the Results

From the above result, it can be seen that all boosting methods perform better than their corresponding methods without boosting. The semi-supervised boosting methods outperform the supervised boosting method and the unsupervised boosting method.

6 Conclusion and Future Work

This paper proposed a semi-supervised boosting algorithm to improve statistical word alignment with limited labeled data and large amounts of unlabeled data. In this algorithm, we built an interpolated model by using both the labeled data

and the unlabeled data. This interpolated model was employed as a learner in the algorithm. Then, we automatically built a pseudo reference for the unlabeled data, and calculated the error rate of each word aligner with the labeled data. Based on this algorithm, we investigated two methods for word alignment. In addition, we developed a method to combine the results of the above two semi-supervised boosting methods.

Experimental results indicate that our semi-supervised boosting method outperforms the unsupervised boosting method as described in Wu and Wang (2005), achieving a relative error rate reduction of 19.52%. And it also outperforms the supervised boosting method that only uses the labeled data, achieving a relative error rate reduction of 28.29%. Experimental results also show that all boosting methods outperform their corresponding methods without boosting.

In the future, we will evaluate our method with an available standard testing set. And we will also evaluate the word alignment results in a machine translation system, to examine whether lower word alignment error rate will result in higher translation accuracy.

References

Yaser Al-Onaizan, Jan Curin, Michael Jahr, Kevin Knight, John Lafferty, Dan Melamed, Franz-Josef Och, David Purdy, Noah A. Smith, and David Yarowsky. 1999. Statistical Machine Translation Final Report. *Johns Hopkins University Workshop.*

Sugato Basu, Mikhail Bilenko, and Raymond J. Mooney. 2004. Probabilistic Framework for Semi-Supervised Clustering. In *Proc. of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004)*, pages 59-68.

Avrim Blum and Tom Mitchell. 1998. Combing Labeled and Unlabeled Data with Co-training. In *Proc. of the 11th Conference on Computational Learning Theory (COLT-1998)*, pages1-10.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2): 263-311.

Colin Cherry and Dekang Lin. 2003. A Probability Model to Improve Word Alignment. In *Proc. of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-2003)*, pages 88-95.

Michael Collins and Yoram Singer. 1999. Unsupervised Models for Named Entity Classification. In *Proc. of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-1999)*, pages 100-110.

Thomas G. Dietterich. 2000. Ensemble Methods in Machine Learning. In *Proc. of the First International Workshop on Multiple Classifier Systems (MCS-2000)*, pages 1-15.

Yoav Freund and Robert E. Schapire. 1996. Experiments with a New Boosting Algorithm. In *Proc. of the 13th International Conference on Machine Learning (ICML-1996)*, pages 148-156.

Franz Josef Och and Hermann Ney. 2000. Improved Statistical Alignment Models. In *Proc. of the 38th Annual Meeting of the Association for Computational Linguistics (ACL-2000)*, pages 440-447.

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19-51.

Thanh Phong Pham, Hwee Tou Ng, and Wee Sun Lee 2005. Word Sense Disambiguation with Semi-Supervised Learning. In *Proc. of the 20th National Conference on Artificial Intelligence (AAAI 2005)*, pages 1093-1098.

Anoop Sarkar. 2001. Applying Co-Training Methods to Statistical Parsing. In *Proc. of the 2nd Meeting of the North American Association for Computational Linguistics(NAACL-2001)*, pages 175-182.

Joachims Thorsten. 1999. Transductive Inference for Text Classification Using Support Vector Machines. In *Proc. of the 16th International Conference on Machine Learning (ICML-1999)*, pages 200-209.

Dekai Wu. 1997. Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora. *Computational Linguistics*, 23(3): 377-403.

Hua Wu and Haifeng Wang. 2005. Boosting Statistical Word Alignment. In *Proc. of the 10th Machine Translation Summit*, pages 313-320.

Hua Wu, Haifeng Wang, and Zhanyi Liu. 2005. Alignment Model Adaptation for Domain-Specific Word Alignment. In *Proc. of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005)*, pages 467-474.

David Yarowsky. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proc. of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL-1995)*, pages 189-196.

Hao Zhang and Daniel Gildea. 2005. Stochastic Lexicalized Inversion Transduction Grammar for Alignment. In *Proc. of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005)*, pages 475-482.

Aligning Features with Sense Distinction Dimensions

[1] Nianwen Xue, [2] Jinying Chen, [3] Martha Palmer
[1]CSLR and [3]Department of Linguistics
University of Colorado
Boulder, CO, 80309
{Nianwen.Xue,Martha.Palmer}@colorado.edu
[2] Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA, 19104
jinying@cis.upenn.edu

Abstract

In this paper we present word sense disambiguation (WSD) experiments on ten highly polysemous verbs in Chinese, where significant performance improvements are achieved using rich linguistic features. Our system performs significantly better, and in some cases substantially better, than the baseline on all ten verbs. Our results also demonstrate that features extracted from the output of an automatic Chinese semantic role labeling system in general benefited the WSD system, even though the amount of improvement was not consistent across the verbs. For a few verbs, semantic role information actually hurt WSD performance. The inconsistency of feature performance is a general characteristic of the WSD task, as has been observed by others. We argue that this result can be explained by the fact that word senses are partitioned along different dimensions for different verbs and the features therefore need to be tailored to particular verbs in order to achieve adequate accuracy on verb sense disambiguation.

1 Introduction

Word sense disambiguation, the determination of the correct sense of a polysemous word from a number of possible senses based on the context in which it occurs, is a continuing obstacle to high performance natural language processing applications. There are several well-documented factors that make accurate WSD particularly challenging. The first has to do with how senses are defined. The English data used for the SENSEVAL exercises, arguably the most widely used data to train and test WSD systems, are annotated based on very fine-grained distinctions defined in WordNet (Fellbaum, 1998), with human inter-annotator agreement at a little over seventy percent and the top-ranked systems' performances falling between 60%~70% (Palmer, *et al*., 2001; Mihalcea *et al.*, 2004). The second source of difficulty for accurate WSD comes from how senses are distributed. It is often the case that a polysemous word has a dominant sense or several dominant senses that occur with high frequency and not enough instances can be found for its low frequency senses in the currently publicly available data. There are on-going efforts to address these issues. For example, the sense annotation component of the OntoNotes project (Hovy, *et al.*, 2006) attempts to create a large-scale coarse-grained sense-annotated corpus with senses defined based on explicit linguistic criteria. These problems will be alleviated when resources like this are available to the general NLP community. There have already been experiments that show such coarse-grained senses lead to substantial improvement in system performance (Palmer *et al*, 2006).

The goal of our experiments is to explore the implications of a related and yet separate problem, specifically the extent to which the linguistic criteria used to define senses are related to what features need to be used in machine-learning systems. There are already published results that show WSD for different syntactic categories may need different types of features. For example, Yarowsky and Florian (2002), in their experiments on SENSEVAL2 English data, showed that sense distinctions of verbs relied more on linguistically motivated features than other parts-of-speech. In this paper,

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 921–928,
Sydney, July 2006.

we will go one step further and show that even for words of the same syntactic category senses are often defined along different dimensions based on different criteria. One direct implication of this observation for supervised machine-learning approaches to WSD is that the features have to be customized for different word categories, or even for different words of the same category. This supports previous arguments for word-specific feature design and parametric modeling for WSD tasks (Chen and Palmer, 2005; Hoste *et al*. 2002). We report experiments on ten highly polysemous Chinese verbs and show that features are not uniformly useful for all words.

The rest of the paper is organized as follows. In Section 2, we describe our WSD system, focusing on the features we used. We also briefly compare the features we use for Chinese with those used in a similar English WSD system. In Section 3, we present our experimental results and show that although rich linguistic features and features derived from a Chinese Semantic Role Labeling improve the WSD accuracy, the improvement is not uniform across all verbs. We show that this lack of consistency is due to the different dimensions along which the features are defined. In Section 4, we discuss related work. Finally Section 5 concludes this paper and describes future directions.

2 WSD System for Chinese Verbs

Our WSD system uses a smoothed maximum entropy (MaxEnt) model with a Gaussian prior (McCallum, 2002) for learning Chinese verb senses. The primary reason is that the MaxEnt model provides a natural way for combining different features without the assumption of feature independence. Furthermore, smoothing the MaxEnt model with a Gaussian prior is better than other smoothing methods at alleviating the overfitting problem caused by low frequency features (Chen *et al.*, 1999). This model has been applied successfully for English WSD (Dang, 2004; Chen and Palmer, 2005).

The features used by our Chinese WSD system include:

Collocation Features

- Previous and next word (relative to the target verb), w_{-1} and w_1 and their parts-of-speech p_{-1} and p_1

Syntactic Features

- Whether the target verb takes a direct object (i.e., in a transitive use)
- Whether the verb takes a sentential complement
- Whether the verb, if it consists of a single character, occurs at the last position of a compound verb

Semantic Features

- The semantic role information about the verbs
- The semantic categories for the verb's NP arguments from a general Chinese noun Taxonomy

All of these features require some level of preprocessing of the Chinese raw text, which comes without word boundaries. To extract the collocation features the raw text needs to be segmented and POS-tagged; to extract the syntactic and semantic features, the Chinese text needs to be parsed. We use an integrated parser that does segmentation, POS-tagging and parsing in one step. Since part of the sense-tagged data comes from the Chinese Treebank that the parser is trained on, we divide the Chinese Treebank into nine equal-sized portions and parse each portion with a parsing model trained on the other eight portions so that the parser has not seen any of the data it parses. The data that is not from the Chinese Treebank is parsed with a parsing model trained on the entire Chinese Treebank. The parser produces a segmented, POS-tagged and parsed version of the same text to facilitate the extraction of the different types of features. The extraction of the semantic role labels as features requires the use of a semantic role tagger, which we describe in greater detail in Section 2.2.

In addition to using the semantic role labeling information, we also extract another type of semantic features from the verb's NP arguments. These features are top-level semantic categories from a three-level general taxonomy for Chinese nouns, which was created semi-automatically based on two Chinese semantic dictionaries (Chen and Palmer, 2004).

2.1 A Comparison with Our English WSD System

Similar to our English WSD system, which achieved the best published results on SENSEVAL2 English verbs for both fine-grained and coarse-grained senses (Chen and Palmer, 2005), our Chinese WSD system uses the same smoothed MaxEnt machine learning model and linguistically motivated features for Chinese verb sense disambiguation. However, the features used in the two systems differ

somewhat due to the different properties of the two languages .

For example, our English system uses the inflected form and the part-of-speech tag of the target verb as feature. For Chinese we no longer use such features since Chinese words, unlike English ones, do not contain morphology that marks tense.

The collocation features used by our English system include bi-grams and tri-grams of the words that occur within two positions before or after the target verb and their part-of-speech tags. In contrast, our Chinese system extracts collocation features from a narrower, three-word window, with one word immediately before and after the target verb. This decision was made based on two observations about the Chinese language. First, certain single-character Chinese verbs, such as the verbs "出|chu", "开|kai" and "成|cheng" in our experiments, often form a compound with a verb to its immediate left. That verb is often a good indicator of the sense of this verb. An example is given in (1):

(1) 辽宁 已 呈现 **出**
Liaoning already show **completion**

多元化 发展 趋势 。
multidimensional development trend

"Liaoning Province has shown the trend of multidimensional development."

Being the last word of a verb compound is a strong indicator for Sense 8 of the verb "出|chu1" (used after a verb to indicate direction or aspect), as in "呈现|cheng2xian4 出|chu1".

Second, unlike English common nouns that often require determiners such as *the*, *a* or *an*, Chinese common nouns can stand alone. Therefore, the direct object of a verb often occurs right after the verb in Chinese, as shown in (2).

（2）动员 群众 勒紧 腰带 集
mobilize people tighten waistband collect

资 **开** ***公路*** (direct object)。
funds **build** ***highway***

"Mobilize people to tighten their waistbands (i.e., save money) in order to collect funds to build highways."

Based on these observations, we use words surrounding the target verb and their part-of-speech tags as collocation features. A further investigation on the different sizes of the context window (3,5,7,9,11) showed that increasing the window size decreased our system's accuracy.

2.2 Features Based on Automatic Semantic Role Tagging

In a recent paper on the WSD of English verbs, Dang and Palmer (2005) showed that semantic role information significantly improves the WSD accuracy of English verbs for both fine-grained and coarse-grained senses. However, this result assumes the human annotation of the Penn English Propbank (Palmer *et al*, 2005). It seems worthwhile to investigate whether the semantic role information produced by a fully automatic Semantic Role tagger can improve the WSD accuracy on verbs, and test the hypothesis that the senses of a verb have a high correlation to the arguments it takes. To that end, we assigned semantic role labels to the arguments of the target verb with a fully automatic semantic role tagger (Xue and Palmer, 2005) trained on the Chinese Propbank (CPB) (Xue and Palmer, 2003), a corpus annotated with semantic role labels that are similiar in style to the Penn English Propbank. In this annotation, core arguments such as agent or theme are labeled with numbered arguments such as *Arg0* and *Arg1*, up to *Arg5* while adjunct-like elements are assigned functional tags such as *TMP* (for temporal), *MNR*, prefixed by *ArgM*. The Semantic Role tagger takes as input syntactic parses produced by the parser described above as input and produces a list of arguments for each of the sense-tagged target verbs and assigns argument labels to them. Features are extracted from both the core arguments and adjuncts of the target verb. In addition to providing the sematnic role labels (e.g., *Arg0* and *Arg1*) of the extracted core arguments, the Semantic Role tagger also provides Hownet (Dong and Dong, 1991) semantic categories associated with these arguments. (3) shows the arguments for the target verb "打" identified by the Semantic Role tagger:

(3) [$_{\text{ArgM-MNR}}$ 经过 三 年 苦 干] ,
through three year hard work,

[arg0 全 乡] [$_{\text{rel}}$ **打**] 成
whole county **dig** finish

[$_{\text{Arg1}}$ 深 水井] 三 眼 。
deep well three classifier

"The whole county finished digging three deep wells through 3 years of hard work."

Based on the output of the Semantic Role tagger and the Chinese noun taxonomy (as described in Section 2.1), the following features are extracted:

SRL+lex	SRL+HowNet	SRL+Taxonomy
ARG1-水井	ARG1-设施	ARG1_location
ARG0-乡	ARG0-地方	ARG0_location
ARGM|MNR-经过	ARGM|MNR-经受	ARGM|MNR

In this example, semantic role related features include: (1) the head word of the core arguments (ARG1-水井 and ARG0-乡) and the adjunct (ARGM|MNR-经过); (2) the HowNet semantic category for the head word (ARG1-设施, ARG0-地方, ARGM|MNR-经受); (3) the semantic role label of the adjunct (ARGM|MNR); and (4) the top level semantic category from the taxonomy of Chinese nouns for the head word of the NP arguments (ARG1_location and ARG0_location).

3 Experimental Results

The data we used for our experiments are developed as part of the OntoNotes project (Hovy *et al.*, 2006) and they come from a variety of sources. Part of the data is from the Chinese Treebank (Xue *et al*, 2005), which has a combination of Xinhua news and Sinorama News Magazine. Since some verbs have an insufficient number of instances for any meaningful experiments, we also annotated portions of the People's Daily corpus, developed by Peking University. We chose not to use the Chinese WSD dataset used in Senseval 3 [1] because we are mainly interested in investigating how the features used in WSD are related to the criteria used to define the senses of Chinese verbs. The Chinese Senseval dataset includes both nouns and verbs. In addition, the criteria used to define their senses are not made explicit and therefore are not clear to us.

Table 1 summarizes the corpus statistics and the experimental results for the 10 highly polysemous Chinese verbs used in our experiments. The results were obtained by using 5-fold cross validation. The top five verbs are verbs that were identified as difficult verbs in Dang *et al*'s (2002) experiments. The first three columns show the verbs (and their pinyin), the number of instances and the number of senses for each verb in the data. The fourth column shows the sense entropy for each verb in its test data, as calculated in Equation 1.

$$-\sum_{i=1}^{n} P(sense_i)\log P(sense_i) \quad (1)$$

Where n is the number of senses of a verb in our data; $P(sense_i)$ is the probability of the *ith* sense of the verb, which is estimated based on the frequency count of the verb's senses in the data. Sense entropy generally reflects the frequency distribution of senses in the corpus. A verb with an evenly distributed sense distribution tends to have a high entropy value. However, a verb can also have a high sense entropy simply because it is highly polysemous (say, has 20 or more senses) even though the sense distribution may be skewed, with one or two dominant senses. To separate the effects of the number of senses, we also use a normalized sense entropy metric (the sixth column in Table 1), as calculated in Equation 2.

$$\frac{-\sum_{i=1}^{n} P(sense_i)\log P(sense_i)}{-\sum_{i=1}^{n} \frac{1}{n}\log P(\frac{1}{n})} \quad (2)$$

Here a large sense number n corresponds to a high value for the normalization factor $-\sum_{i=1}^{n} \frac{1}{n}\log P(\frac{1}{n})$. Therefore, normalized sense entropy can indicate sense frequency distribution more precisely than sense entropy.

Table 1 (Columns 7 to 10) also shows the experimental results. As we can see, on average, our system achieved about 19% improvement (absolute gain) in accuracy compared to the most frequent sense baseline. Its performance is consistently better than the baseline for all 10 verbs.

3.1 Corpus Statistics and Disambiguation Accuracy

The data in Table 1 shows that verbs with a high normalized sense entropy have the low frequency baselines. Furthermore, this relation is stronger than that between un-normalized sense entropy and the baseline. However, sense entropy is a better predictor for system performance than normalized sense entropy. The reason is intuitive: unlike the baseline, the automatic WSD system, trained on the training data, does not only rely on sense frequency information to predict senses.

[1] http://www.senseval.org/senseval3

	# of instance	# of sense	sense entropy	norm. sense entropy	baseline	all feat	all-SRL
出|chu	271	11	1.12	0.47	74.54	79.70	78.59
恢复|huifu	113	3	0.93	0.84	50.44	69.91	72.57
见|jian	167	7	1.01	0.52	72.46	82.63	82.03
想|xiang	231	6	1.00	0.56	65.80	76.19	77.49
要|yao	254	9	1.56	0.71	33.46	46.46	49.21
成|cheng	161	8	1.38	0.67	43.48	73.29	72.67
打|da	313	21	2.29	0.75	20.77	45.05	32.59
开|kai	382	18	2.31	0.80	19.37	50.00	39.27
通过|tongguo	384	4	0.97	0.70	55.73	81.51	79.17
发展|fazhan	1141	7	0.88	0.45	74.76	79.58	77.56
average		9.4			51.08	70.18	67.13
total	3417						

Table 1 Corpus Statistics and Experimental Results for the 10 Chinese Verbs

The number of senses has a direct impact on how many training instances exist for each verb sense. As a consequence, it is more difficult for the system to make good generalizations from the limited training data that is available for highly polysemous verbs. Therefore, sense entropy, which is based on both sense frequency distribution and polysemy is more appropriate for predicting system accuracy. A related observation is that the system gain (compared with the baseline) is bigger for verbs with a high normalized sense entropy, such as "恢复|huifu", "打|da", "开|kai", and "通过|tongguo", than for other verbs; and the system gain is very small for verbs with low normalized sense entropy and a relatively large number of senses, such as "出|chu" and "发展|fazhan", since they already have high baselines.

3.2 The Effect of Semantic Role Features

When Semantic Role information is used in features, the system's performance on average improves 3.05%, from 67.13% to 70.18% compared with when the features derived from the Semantic Role information is not used. If we look at the system's performance on individual verbs, the results show that adding Semantic Role information as features improves the accuracy of 7 of the 10 verbs. For the remaining 3 verbs, adding semantic role information actually hurts the system's performance. We believe this apparent inconsistency can be explained by looking at how senses are defined for the different verbs. The two verbs that present the most challenge to the system, are "打|da" and "要|yao" While Semantic Role features substantially improve the accuracy of "打|da", they actually hurt the accuracy of "要|yao". For "要|yao", its three most frequent senses account for 86% of its total instances (232 out of 270) and they are the "intend to", "must, should" and "need" senses:

(4) Three most frequent senses of "要|yao"

(a) 双方 表示 要 进一步 合作 。
two sides indicate intend further cooperation

"The two sides indicated that they intended to step up their cooperation."

(b) 路 很 滑， 大家 要 小心 。
road very slippery, everybody should careful

"The road is slippery. Everybody should be careful."

(c) 苏钢 每 年 要 靠
Suzhou Steel Works every year need depend

大运河 运输 原料 。
the Great Canal transport raw material

"Suzhou Steel Works needs to depend on the Great Canal to transport raw material."

Two of the senses, "must" and "need", are used as auxiliary verbs. As such, they do not take arguments in the same way non-auxiliary verbs do. For example, they do not take noun phrases as arguments. As a result, the Semantic Role tagger, which assigns argument labels to head words of noun phrases or clauses, cannot produce a meaningful argument for an auxiliary verb. For the "intend to" sense, even if it is not

an auxiliary verb, it still does not take a noun phrase as an object. Instead, its object is a verb phrase or a clause, depending on the analysis. The correct head word of its argument should be the lower verb, which apparently is not a useful discriminative feature either.

In contrast, the senses of "打|da" are generally defined based on their arguments. The three most frequent senses of "打|da" are "call by telephone", "play" and "fight" and they account for 40% of the "打|da" instances. Some examples are provided in (5)

(5) Top three senses of "打|da"

(a) 你 有过 排长 打
you have queue in long line call

公共 电话 的 经验 吗 ?
public phone DE experience ma

"Do you have the experience of queuing in a line and waiting to make a call with a public phone?"

(b) 几个 值班 人员 围坐
a few on duty personnel sit

一 圈 打 扑克 。
one circle play poker

"A few of the personnel on duty were sitting in a circle and playing poker."

(c) 动员 全 社会 力量 打好
mobilize whole society power fight

扶贫 攻坚 战 。
helping the poor crucial battle

"…mobilize the power of the whole society and fight the crucial battle of helping the poor."

The senses of "打|da" are to a large extent determined by its PATIENT (or *Arg1*) argument, which is generally realized in the object position. The *Arg1* argument usually forms highly coherent lexical classes. For example, the *Arg1* of the "call" sense can be "电话|dianghua/phone, "手机|shouji/cellphone", etc. its *Arg1* argument can be "蓝球|langqiu/basketball", "桥牌|qiaopai/bridge", "游戏|youxi/game", etc for the "play" sense, Finally , for its sense "fight", the *Arg1* argument can be "攻坚|gongjian/crucial 战|zhan/battle", "巷战|xiangzhang/street warfare", "游击战|youjizhan/guerilla warfare", etc.. It's not surprising that recognizing the arguments of "打|da" is crucial in determining its sense.

The accuracy for both verbs is still very low, but for very different reasons. In the case of "要|yao4", the challenge is identifying discriminative features that may not be found in the narrow local context. These could for instance include discourse features. In the case of "打|da', one important reason why the accuracy is still low is because "打|da" is highly polysemous and has over forty senses. Given its large number of senses, the majority of its senses do not have enough instances to train a reasonable model. We believe that more data will improve its WSD accuracy.

There are other dimensions along which verb senses are defined in addition to whether or not a verb is an auxiliary verb and what type of auxiliary verb it is, and what types of arguments it takes. One sense of "出|chu" is a verb particle that indicates the direction or aspect of the main verb that generally immediately precedes it. In this case the most important feature for identifying this sense is the collocation feature.

Our experimental results seem to lend support to a WSD approach where features are tailored to each target word, or at least each class of words, based on a careful analysis of the dimensions along which senses are defined. Automatic feature selection (Blum and Langley, 1997) could also prove useful in providing this type of tailoring. An issue that immediately arises is the feasibility of this approach. At least for Chinese, the task is not too daunting, as the number of highly polysemous verbs is small. Our estimation based on a 250K-word chunk of the Chinese Treebank and a large electronic dictionary in our possession shows only 6% or 384 verb types having four or more definitions in the dictionary. Even for these verbs, the majority of them are not difficult to disambiguate, based on work by Dang *et al*. (2002). Only a small number of these verbs truly need customized features.

4 Related work

There is a large body of literature on WSD and here we only discuss a few that are most relevant to our work. Dang and Palmer (2005) also use predicate-argument information as features in their work on English verbs, but their argument labels are not produced by an automatic SRL system. Rather, their semantic role labels are directly extracted from a human annotated

corpus, the English Proposition Bank (Palmer *et al*, 2005), citing the inadequate accuracy of automatic semantic role labeling systems. In contrast, we used a fully antomated SRL system trained on the Chinese Propbank. Nevertheless, their results show, as ours do, that the use of semantic role labels as features improves the WSD accuracy of verbs.

There are relatively few attempts to use linguistically motivated features for Chinese word sense disambiguation. Niu *et al* (2004) applied a Naive Bayesian model to Chinese WSD and experimented with different window sizes for extracting local and topical features and different types of local features (e.g., bigram templates, local words with position or parts-of-speech information). One basic finding of their experiments is that simply increasing the window size for extracting local features or enriching the set of local features does not improve disambiguation performance. This is consistent with our usage of a small size window for extracting bigram collocation features. Li *et al.* (2005) used sense-tagged *true* bigram collocations [2] as features. These features were obtained from a collocation extraction system that used lexical co-occurrence statistics to extract candidate collocations and then selected *true* collocations by using syntactic dependencies (Xu *et al*., 2003). In their experiments on Chinese nouns and verbs extracted from the People's Daily News and the SENSEVAL3 data set, the Naive Bayesian classifier using *true* collocation features generally performed better than that using simple bigram collocation features (i.e., bigram co-occurence features). It is worth noting that the *true* collocations overlap to a large degree with rich syntactic information used here such as the subject and direct object of a target verb. Therefore, their experiments show evidence that rich linguistic information benefits WSD on Chinese, consistent with our results.

Our work is more closely related to the work of Dang *et al* (2002), who conducted experiments on 28 verbs and achieved an accuracy of 94.2%. However the high accuarcy is largely due to the fact that their verbs are randomly chosen from the Chinese Treebank and some of them are not even polysemous (having a single sense). Extracting features from the gold standard parses also contributed to the high accuracy, although not by much. For 5 of their 28 verbs, their initial experimental results did not break the most frequent sense baseline. They annotated additional data on those five verbs and their system trained on this new data did outperfom the baseline. However, they concluded that the contribution of linguistic motivated features, such as features extracted from a syntactic parse, is insignificant, a finding they attributed to unique properties of Chinese given that the same syntactic features significantly improves the WSD accuracy. Our experimental results show that this conclusion is premature, without a detailed analysis of the senses for the individual verbs.

[2] In their definition, a collocation is a recurrent and conventional fixed expression of words that holds syntactic and semantic relations.

5 Conclusion and future work

We presented experiments with ten highly polysemous Chinese verbs and showed that a previous conclusion that rich linguistic features are not useful for the WSD of Chinese verbs is premature. We demonstrated that rich linguistic features, specifically features based on syntactic and semantic role information, are useful for the WSD of Chinese verbs. We believe that the WSD systems can benefit even more from rich linguistic features as the performance of other NLP tools such as parsers and Semantic Role Taggers improves. Our experimental results also lend support to the position that feature design for WSD should be linked tightly to the study of the criteria that sense distinctions are based on. This position calls for the customization of features for individual verbs based on understanding of the dimensions along which sense distinctions are made and a closer marriage between machine learning and linguistics. We believe this represents a rich area of exploration and we intend to experiment with more verbs with further customization of features, including experimenting with automatic feature selection.

Acknowledgement

This work was supported by National Science Foundation Grant NSF-0415923, Word Sense Disambiguation, the DTO-AQUAINT NBCHC-040036 grant under the University of Illinois subcontract to University of Pennsylvania 2003-07911-01 and the GALE program of the Defense Advanced Research Projects Agency, Contract No. HR0011-06-C-0022. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do

not necessarily reflect the views of the National Science Foundation, the DTO, or DARPA.

References

Avrim L. Blum and Pat Langley. 1997. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97:245-271, 1997.

Jinying Chen and Martha Palmer. 2004. Chinese Verb Sense Discrimination Using an EM Clustering Model with Rich Linguistic Features, In Proc. of the 42nd Annual meeting of the Assoication for Computational Linguistics, ACL-04. July 21-24, Barcelona, Spain

Jinying Chen and Martha Palmer. 2005. Towards Robust High Performance Word Sense Disambiguation of English Verbs Using Rich Linguistic Features. In Proc. of the 2nd International Joint Conference on Natural Language Processing. Jeju Island, Korea, in press.

Stanley. F. Chen and Ronald Rosenfeld. 1999. *A Gaussian Prior for Smoothing Maximum Entropy Modals*. Technical Report CMU-CS-99-108, CMU.

Hoa T. Dang, Ching-yi Chia, Martha Palmer and Fu-Dong Chiou. 2002. Simple Features for Chinese Word Sense Disambiguation. In *Proceedings of COLING-2002, the Nineteenth Int. Conference on Computational Linguistics*, Taipei, Aug.24–Sept.1.

Hoa T. Dang. 2004. *Investigations into the role of lexical semantics in word sense disambiguation*. PhD Thesis. University of Pennsylvania.

Hoa Dang and Martha Palmer. 2005. The role of semantic roles in disambiguating verb senses. In *Proceedings of ACL-05*, Ann Arbor, Michigan.

Zhendong Dong and Qiang Dong, HowNet. 1991. http://www.keenage.com.

Christiane Fellbaum, ed. 1998. *WordNet: An* Electronic *Lexical Database.* Cambridge, MA: MIT Press.

Veronique Hoste, Iris Hendrickx, Walter Daelemans, and Antal van den Bosch. 2002. Parameter optimization for machine-learning of word sense disambiguation. *NLE, Special Issue on Word Sense Disambiguation Systems*, 8(4):311–325.

Eduard Hovy, Mtchchell Marcus, Martha Palmer, Lance Ramshaw and Ralph Weischedel. 2006. OntoNotes: the 90% solution. In Proceedings of the HLT-NAACL 2006, New York City.

Wanyin Li, Qin Lu and Wenjie Li. 2005. Integrating Collocation Features in Chinese Word Sense Disambiguation. In Proceedings of the Fourth Sighan Workshop on Chinese Language Processing. pp: 87-94. Jeju, Korea.

Andrew K. McCallum: MALLET: A Machine Learning for Language Toolkit. http://www.cs.umass.edu/~mccallum/mallet (2002).

Rada Mihalcea, Timothy Chklovski and Adam Kilgarriff. 2004. The Senseval-3 English lexical sample task. In *Proceedings of Senseval-3: The Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*. Barcelona, Spain. July.

Zheng-Yu Niu, Dong-Hong Ji and Chew Lim Tan, Optimizing Feature Set for Chinese Word Sense Disambiguation. 2004. In *Proceedings of the 3rd International Workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL-3)*. Barcelona, Spain.

Martha Palmer, Christiane Fellbaum, Scott Cotton, Lauren Delfs, and Hoa Trang Dang. 2001. English tasks: All-words and verb lexical sample. *Proceedings of Senseval-2: Second International Workshop on Evaluating Word Sense Disambiguation Systems*, Toulouse, France, 21-24.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles, *Computational Linguistics*, 31(1): 71–106.

Martha Palmer, Christiane Fellbaum and Hoa Trang Dang. (to appear, 2006). Making fine-grained and coarse-grained sense distinctions, both manually and automatically. *Natural Language Engineering*.

Ruifeng Xu , Qin Lu, and Yin Li. 2003. An automatic Chinese Collocation Extraction Algorithm Based On Lexical Statistics. In *Proceedings of the NLPKE Workshop*. Beijing, China.

Nianwen Xue, Fei Xia, Fu-Dong Chiou and Martha Palmer. 2005. The Penn Chinese Treebank: Phrase Structure Annotation of a Large Corpus. *Natural Language Engineering*, 11(2):207-238.

Nianwen Xue and Martha Palmer. 2003. Annotating Propositions in the Penn Chinese Treebank, In *Proceedings of the 2nd SIGHAN Workshop on Chinese Language Processing*, in conjunction with ACL'03. Sapporo, Japan.

Nianwen Xue and Martha Palmer. 2005. Automatic Semantic Role Labeling for Chinese Verbs. In Proceedings of the 19th International Joint Conference on Artificial Intelligence. Edinburgh, Scotland.

David Yarowsky and Radu Florian. 2002. Evaluating sense disambiguation across diverse parameter spaces. *Journal of Natural Language Engineering,* 8(4): 293–310.

Word Sense Disambiguation using lexical cohesion in the context

Dongqiang Yang | David M.W. Powers
School of Informatics and Engineering
Flinders University of South Australia
PO Box 2100, Adelaide
Dongqiang.Yang|David.Powers@flinders.edu.au

Abstract

This paper designs a novel lexical hub to disambiguate word sense, using both syntagmatic and paradigmatic relations of words. It only employs the semantic network of WordNet to calculate word similarity, and the Edinburgh Association Thesaurus (EAT) to transform contextual space for computing syntagmatic and other domain relations with the target word. Without any back-off policy the result on the English lexical sample of SENSEVAL-2[1] shows that lexical cohesion based on edge-counting techniques is a good way of unsupervisedly disambiguating senses.

1 Introduction

Word Sense Disambiguation (WSD) is generally taken as an intermediate task like part-of-speech (POS) tagging in natural language processing, but it has not so far achieved the sufficient precision for application as POS tagging (for the history of WSD, cf. Ide and Véronis (1998)). It is partly due to the nature of its complexity and difficulty, and to the widespread disagreement and controversy on its necessity in language engineering, and to the representation of the senses of words, as well as to the validity of its evaluation (Kilgarriff and Palmer, 2000). However the endeavour to automatically achieve WSD has been continuous since the earliest work of the 1950's.

In this paper we specifically investigate the role of semantic hierarchies of lexical knowledge on WSD, using datasets and evaluation methods from SENSEVAL (Kilgarriff and Rosenzweig, 2000) as these are well known and accepted in the community of computational linguistics.

With respect to whether or not they employ the training materials provided, SENSEVAL roughly categorizes the participating systems into "unsupervised systems" and "supervised systems". Those that don't use the training data are not usually truly unsupervised, being based on lexical knowledge bases such as dictionaries, thesauri or semantic nets to discriminate word senses; conversely the "supervised" systems learn from corpora marked up with word senses.

The fundamental assumption, in our "unsupervised" technique for WSD in this paper, is that the similarity of contextual features of the target with the pre-defined features of its sense in the lexical knowledge base provides a quantitative cue for identifying the true sense of the target.

The lexical ambiguity of polysemy and homonymy, whose distinction is however not absolute as sometimes the senses of word may be intermediate, is the main object of WSD. Verbs, with their more flexible roles in a sentence, tend to be more polysemous than nouns, so worsening the computational feasibility. In this paper we disambiguated the sense of a word after its POS tagging has assigned them either a noun or a verb tag. Furthermore, we deal with nouns and verbs separately.

2 Some previous work on WSD using semantic similarity

Sussna (1993) utilized the semantic network of nouns in WordNet to disambiguate term senses to improve the precision of SMART information retrieval at the stage of indexing, in which he assigned two different weights for both directions of edges in the network to compute the similarity of two nodes. He then exploited the moving fixed size window to minimize the sum

[1] http://www.senseval.org/

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 929–936,
Sydney, July 2006.

of all combinations of the shortest distances among target and context words.

Pedersen et al. (2003) extended Lesk's definition method (1986) to discriminate word sense through the definitions of both target and its IS-A relatives, and achieved a better result in the English lexical sample task of SENSEVAL-2, compared with other edge-counting or statistical estimation metrics on WordNet.

Humans carefully select words in a sentence to express harmony or cohesion in order to ease the ambiguity of the sentence. Halliday and Hasan (1976) argued that cohesive chains unite text structure together through reiteration of reference and lexical semantic relations (superordinate and subordinate). Morris and Hirst (1991) suggested building lexical chains is important in the resolution of lexical ambiguity and the determination of coherence and discourse structure. They argued that lexical chains, which cover the multiple semantic relations (systematic and non-systematic), can transform the context setting into the computational one to narrow down the specific meaning of the target, manually realizing this with the help of Roget's Thesaurus. They defined a lexical chain within Roget's very general hierarchy, in which lexical relationships are traced through a common category.

Hirst and St-Onge (1997) define a lexical chain using the syn/antonym and hyper/hyponym links of WordNet to detect and correct malapropisms in context, in which they specified three different weights from extra-strong to medium strong to score word similarity to decide the inserting sequence in the lexical chain. They first computationally employed WordNet to form a "greedy" lexical chain as a substitute of the context to solve the matter of malapropism, where the word sense is decided by its preceding words.

Around the same time, Barzilay and Elhadad (1997) realized a "non-greedy" lexical chain, which determined the word sense after processing of all words, in the context of text summarization.

In this paper we propose an improved lexical chain, the *lexical hub*, that holds the target to be disambiguated as the centre, replacing the usual chain topology used in text summarization and cohesion analysis. In contrast with previous methods we only record the lexical hub of each sense of the target, and we don't keep track of other context words. In other words, after the computation of lexical hub of the target, we can immediately produce the right sense of the target even though the senses of the context words are still in question. We also transform the context surroundings through a word association thesaurus to explore the effect of other semantic relationships such as syntagmatic relation against WSD.

3 Selection of knowledge bases

WordNet (Fellbaum, 1998) provides a fine-grained enumerative semantic net that is commonly used to tag the instances of English target words in the tasks of SENSEVAL with different senses (WordNet synset numbers). WordNet groups related concepts into synsets and links them through IS-A and PART-OF links, emphasizing the vertical interaction between the concepts that is much paradigmatic.

Although WordNet can capture the fine-grained paradigmatic relations of words, another typical word relationship, syntagmatic connectedness, is neglected. The syntagmatic relationship, which is often characterized with different POS tag, and frequently occurs in corpora or human brains, plays a critical part in cross-connecting words from different domains or POS tags.

It should be noted that WordNet 2.0 makes some efforts to interrelate nouns and verbs using their derived lexical forms, placing associated words under the same domain. Although some verbs have derived noun forms that can be mapped onto the noun taxonomy, this mapping only relates the morphological forms of verbs, and still lacks syntagmatic links between words.

The interrelationship of noun and verb hierarchies is far from complete and only a supplement to the primary IS-A and PART-OF taxonomies in WordNet. Moreover as WordNet generally concerns the paradigmatic relations (Fellbaum, 1998), we have to seek for other lexical knowledge sources to compensate for the shortcomings of WordNet in WSD.

The Edinburgh Association Thesaurus[2] (EAT) provides an associative network to account for word relationship in human cognition after collecting the first response words for the stimulus words list (Kiss et al., 1973). Take the words *eat* and *food* for example. There is no direct path between the concepts of these two words in the taxonomy of WordNet (both as noun and verb), except in the gloss of the first and third sense of *eat* to explain '*take in solid food*', or '*take in food*', which glosses are not regularly or care-

[2] http://www.eat.rl.ac.uk/

fully organized in WordNet. However in EAT *eat* is strongly associated with *food*, and when taking *eat* as a stimulus word, 45 out of 100 subjects regarded *food* as the first response.

Yarowsky (1993) indicated that the objects of verbs play a more dominant role than their subjects in WSD and nouns acquire more stable disambiguating information from their noun or adjective modifiers.

In the case of verbs association tests, it is also reported that more than half the response words of verbs (the stimuli) are syntagmatically related (Fellbaum, 1998). In experiments of examining the psychological plausibility of WordNet relationships, Chaffin et al. (1994) stated that only 30.4% of the responses of 75 verb stimuli belongs to verbs, and more than half of the responses are nouns, of which nearly 90% are categorized as the arguments of the verbs.

Sinopalnikova (2004) also reported that there are multiple relationships found in word association thesaurus, such as syntagmatic, paradigmatic relations, domain information etc.

In this paper we only use the straightforward forms of context words separating the effect of syntactic dependence on the WSD. As a supplement of enriching word linkage in the WSD, we retrieve the lexical knowledge from both WordNet and EAT. We first explore the function of semantic hierarchies of WordNet on WSD, and then we transform the context word with EAT to investigate whether other relationships can improve WSD.

4 System design

In order to find semantically related words to cohesively form lexical hubs, we first employ the two word similarity algorithms of Yang and Powers (2005; 2006) that use WordNet to compute noun similarity and verb similarity respectively. We next construct the lexical hub for each target sense to assemble the similarity score between the target and its context words together. The maximum score of these lexical hubs specifically predicts the real sense of the target, also implicitly captures the cohesion and real meaning of the word in its context.

4.1 Similarity metrics on nouns

Yang and Powers (2005) designed a metric,

$$Sim(c1, c2) = \alpha_t * \beta^{\lambda}$$

utilizing both IS-A and PART-OF taxonomies of WordNet to measure noun similarity, and they argued that the similarity of nouns is the maximum of all their concept similarities. They defined the similarity (*Sim*) of two concepts (*c*1 and *c*2) with a link type factor (α_t) to specify the weights of different link types (t) (syn/antonym, hyper/ hyponym, and holo/meronym) in the WordNet, and a path type factor (β_t) to reduce the uniform distance of the single link, along with a depth factor (λ) to restrict the maximum searching distance between concepts. Since their metric on noun similarity is significantly better than some popular measures and even outperforms some subjects on a standard data set, we selected it as a measure on noun similarity in our WSD task.

4.2 Similarity metrics on verbs

Yang and Powers (2006) also redesigned their noun model,

$$Sim(c1, c2) = \alpha_{str} * \alpha_t * \prod_{i=1}^{Dist(c1,c2)} \beta_{t_i}$$

to accommodate verb case, which is harder to deal with in the shallow and incomplete taxonomy of verbs in WordNet. As an enhancement to the uniqueness of verb similarity they also consider three fall-back factors, where if α_{str} is 1 normally but successively falls back to:

- α_{stm}: the verb stem polysemy ignoring sense and form
- α_{der}: the cognate noun hierarchy of the verb
- α_{gls}: the definition of the verb

They also defined two alternate search protocols: rich hierarchy exploration (RHE) with no more than six links and shallow hierarchy exploration (SHE) with no more than two links.

One minor improvement to the verb model in their system comes from comparing the similarity of verbs and nouns using the noun model metric for the derived noun form of verb. It thus allows us to compare nouns and verbs and avoids the limitation of having to have the same POS tag.

4.3 Depth in WordNet

Yang and Powers fine-tuned the parameters of the noun and verb similarity models, finding them relatively insensitive to the precise values, and we have elected to use their recommended values for the WSD task. But it is worth mentioning that their optimal models are achieved in purely verbal data sets, i.e. the similarity score is context-free.

In their models, the depth in the WordNet, i.e. the distance between the synsets of words (λ), is indeed an outside factor which confines the searching scope to the cost of computation and depends on the different applications. If we tuned it using the training data set of SENSEVAL-2 we probably would assign different values and might achieve better results. Note that for both nouns and verbs we employ RHE (rich hierarchy exploration) with λ = 2 making full use of the taxonomy of WordNet and making no use of glosses.

4.4 How to setup the selection standard for the senses

Other than making the most of WSD results, our main motive for this paper is to explore to what extent the semantic relationships will reach accuracy, and to fully acknowledge the contribution of this single attribute working on WSD, which is encouraged by SENSEVAL in order to gain further benefits in this field (Kilgarriff and Palmer, 2000). Without any definition, which is previously surveyed by Lesk (1986) and Pedersen et al. (2003), we screen off the definition factor in the metric of verb similarity, with the intention of focusing on the taxonomies of WordNet.

Assuming that the lexical hub for the right sense would maximize the cohesion with other words in the discourse, we design six different strategies to calculate the lexical hub in its unordered contextual surroundings.

We first put forward three metrics to measure up the similarity of the senses of the target and the context word:

- **The maximized sense similarity**

$$Sim_{max}(T_k, C_i) = \max_j \left(Sim(T_k, C_{i,j}) \right)$$

where T denotes the target, T_k is the *kth* sense of the target; C_i is the *ith* context word in a fixed window size around the target, $C_{i,j}$ the *jth* sense of C_i. Note that T and C can be any noun and verb, along with *Sim* the metrics of Yang and Powers.

- **The average of sense similarity**

$$Sim_{ave}(T_k, C_i) = \sum_{j=1}^{m} Sim(T_k, C_{i,j}) \Big/ \sum_{j=1}^{m} Links(T_k, C_{i,j})$$

where $Links(T_k, C_{i,j})$=1, if $Sim(T_k, C_{i,j})$>0, otherwise 0.

- **The sum of sense similarity**

$$Sim_{sum}(T_k, C_i) = \sum_{j=1}^{m} Sim(T_k, C_{i,j})$$

where m is the total sense number of C_i.

Subsequently we can define six distinctive heuristics to score the lexical hub in the following parts:

- **Heuristic 1 – Sense Norm (HSN)**

$$Sense(T) = \arg\max_k \left(\sum_{i=1}^{l} Sim_{max}(T_k, C_i) \Big/ \sum_{i=1}^{l} Linkw(T_k, C_i) \right)$$

where $Linkw(T_i)$=1 if $Sim_{max}(T_k, C_i)$>0, otherwise 0

- **Heuristic 2 – Sense Max (HSM)**

An unnormalized version of HSN is:

$$Sense(T) = \arg\max_k \left(\sum_{i=1}^{l} Sim_{max}(T_k, C_i) \right)$$

- **Heuristic 3 – Sense Ave (HSA)**

Taking into account all of the links between the target and its context word, the correct sense of the target is:

$$Sense(T) = \arg\max_k \left(\sum_{i=1}^{l} Sim_{ave}(T_k, C_i) \right)$$

- **Heuristic 4 – Sense Sum (HSS)**

The unnormalized version of HSA is:

$$Sense(T) = \arg\max_k \left(\sum_{i=1}^{l} Sim_{sum}(T_k, C_i) \right)$$

- **Heuristic 5 – Word Linkage (HWL)**

The straightforward output of the correct sense of the target in the discourse is to count the maximum number of context words whose similarity scores with the target are larger than zero:

$$Sense(T) = \arg\max_k \left(\sum_{i=1}^{l} Linkw(T_k, C_i) \right)$$

- **Heuristic 6 – Sense Linkage (HSL)**

No matter what kind of relations between the target and its context are, the sense of the target, which is related to the maximum counts of senses of all its context words, is scored as the right meaning:

$$Sense(T) = \arg\max_k \left(\sum_{i=1}^{l} \sum_{j=1}^{m} Links(T_k, C_{i,j}) \right)$$

Therefore the lexical hub of each sense of the target only relies on the interaction of the target and its each context word, rather than of the context words. The implication is that the lexical hub only disambiguates the real sense of the tar-

get other than the real meaning of the context word; the maximum scores or link numbers (on the level of words or senses) in the six heuristics suggest that the correct sense of the target should cohere with as many words or their senses as practicable in the discourse.

When similarity scores are ties we directly produce all of the word senses to prevent us from guessing results. Some WSD systems in SENSEVAL handle tied scores simply using the first sense (in WordNet) of the target as the real sense. It is no doubt that the skewed distribution of word senses in the corpora (the first sense often captures the dominant sense) can benefit the performance of the systems, but at the same time it mixes up the contribution of the semantic hierarchy on WSD in our system.

5 Results

We evaluate the six heuristics on the English lexical sample of SENSEVAL-2, in which each target word has been POS-tagged in the training part. With the absence of taxonomy of adjectives in WordNet we only extract all 29 nouns and all 29 verbs from a total of 73 lexical targets, and then we subcategorize the test dataset into 1754 noun instances and 1806 verb instances. Since the sample of SENSEVAL-2 is manually sense-tagged with the sense number of WordNet 1.7 and our metrics are based on its version 2.0, we translate the sample and answer format into 2.0 in accordance with the system output format.

Finally, we find that each noun target has 5.3 senses on average and each verb target 16.4 senses. Hence the baseline of random selection of senses is the reciprocal of each average sense number, i.e. separately 18.9 percent for nouns and 6 percent for verbs.

In addition, SENSEVAL-2 provides a scoring software with 3 levels of schemes, i.e. fine-grained, coarse-grained and mixed-grained to produce precision and recall rates to evaluate the participating systems. According to the SENSEVAL scoring system, as we always give at least one answer, the precision is identical to the recall under the separate noun and verb datasets. So we just evaluate our systems in light of accuracy. We tested the heuristics with fine-grained precision, which required the exact match of the key to each instance.

5.1 Context

Without any knowledge of domain, frequency and pragmatics to guess, word context is the only way of labeling the real meaning of word. Basically a bag of context words (after morphological analyzing and filtering stop-words) or the fine-grained ones (syntactic role, selection preference etc.) can provide cues for the target. We propose to merely use a bag of words to feed into each heuristic in case of losing any valuable information in the disambiguation, and preventing from any interference of other clues except the semantic hierarchy of WordNet.

The size of the context is not a definitive factor in WSD, Yarowsky (1993) suggested the size of 3 or 4 words for the local ambiguity and 20/50 words for topic ambiguity. He also employed Roget's Thesaurus in 100 words of window to implement WSD (Yarowsky, 1992). To investigate the role of local context and topic context we vary the size of window from one word distance away to the target (left and right) until 100 words away in nouns or 60 in verbs, until there are no increases in the context of each instance.

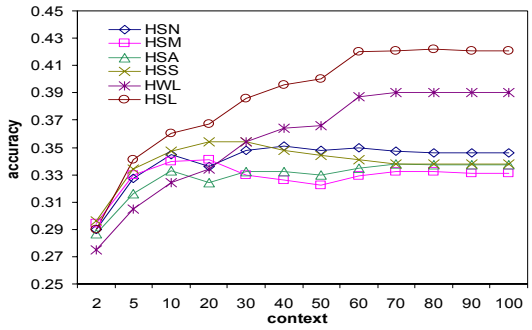


Figure 1: the result of noun disambiguation with different size of context in SENSEVAL 2

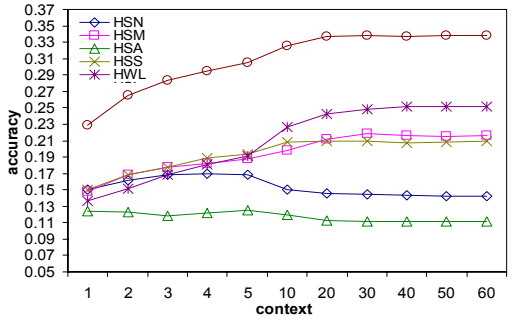


Figure 2: the result of verb disambiguation with different size of context in SENSEVAL 2

Noun and verb disambiguation results are respectively displayed in Figure 1 and 2. Since the performance curves of the heuristics turned into flat and stable (the average standard deviations of the six curves of nouns and verbs is around 0.02 level before 60 and 20, after that approxi-

mately 0.001 level), optimal performance is reached at 60 context words for nouns and 20 words for verbs. These values are used as parameters in subsequent experiments.

5.2 Transformed context (EAT)

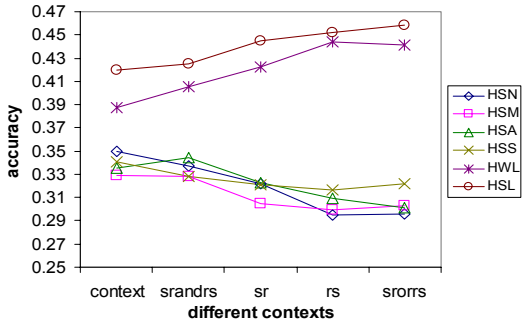


Figure 3: the results of nouns disambiguation of SENSEVAL-2 in the transformed context spaces

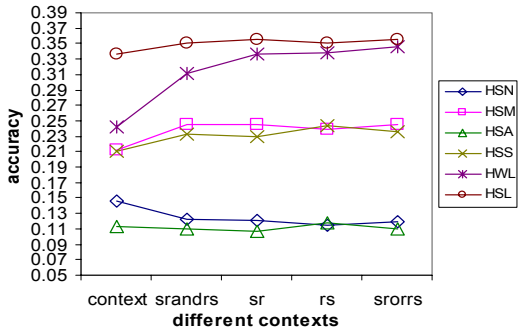


Figure 4: the results of verbs disambiguation of SENSEVAL-2 in the transformed context spaces

Although our metrics can measure the similarity of nouns and verbs through the derived related form of verbs (not from the derived verbs of nouns as a consequence of the shallowness of verb taxonomy of WordNet), we still can't completely rely on WordNet, which focuses on the paradigmatic relations of words, to fully cover the complexity of contextual happenings of words.

Since the word association norm captures both syntagmatic and pragmatic relations in words, we transform the context words of the target into its associated words, which can be retrieved in the EAT, to augment the performance of the lexical hub.

There are two word lists in the EAT: one list takes each head word as a stimulus word, and then collects and ranks all response words according to their frequency of subject consensus; the other list is in the reverse order with the response as a head word and followed by the eliciting stimuli. We denote the stimulus/response set of word as SR, respond/stimulus as RS. Apart from that we symbolize SRANDRS as the intersection of SR and RS, along with SRORRS as the union set of SR and RS. Then for each context word we retrieve its corresponding words in each word list and calculate the similarity between the target and these words including the context words.

As a result we transform the original context space of each target into an enriched context space under the function of SR, RS, SRANDRS or SRORRS.

We take the respective 60 context words of nouns and 20 words of verbs as the reference points for the transferred context experiment, since after that the performance curves of the heuristics turned into flat and stable (the average standard deviations of the six curves of nouns and verbs is around 0.02 level before 60, after that approximately 0.001 level).

After the transformations, the noun and verb results are respectively demonstrated in Figure 3 and 4.

6 Comparison with other techniques.

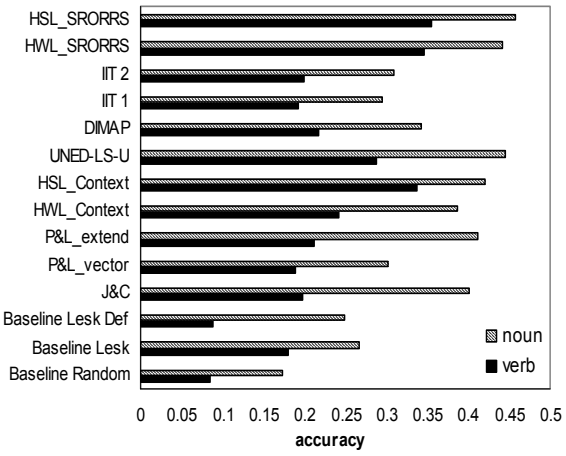


Figure 5: comparisons of HWL and HSL with other unsupervised systems and similarity metrics

Pedersen et al. (2003) in the work of evaluating different similarity techniques based on WordNet, realized two variants of Lesk's methods: extended gloss overlaps (P&L_extend) and gloss vector (P&L_vector), as well as evaluating them in the English lexical sample of SENSEVAL-2. The best edge-counting-based metric that they measured are from Jiang and Conrath (1997) (J&C).

Accordingly, without the transformation of EAT, we compare our results of HWL and HSL (denoted as HWL_Context and HSL_Context) with the above methods (picking up their optimal values). The results are illustrated in Figure 5. At the same time we also list three baselines for unsupervised systems (Kilgarriff and Rosenzweig, 2000), which are Baseline Random (randomly selecting one sense of the target), Baseline Lesk (overlapping between the examples and definitions of and unsupervised systems in SENSEVAL-2 each sense of the target and context words), and its reduced version, i.e. Baseline Lesk Def (only definition).

We further compare HWL and HSL with the intervention of SRORRS of EAT (denoted as HWL_SRORRS and HSL_ SRORRS) with other unsupervised systems that employ no training materials of SENSEVAL-2, which are respectively:

- IIT 1 and IIT 2: extended the WordNet gloss of each sense of the target, along with its superordinate and subordinate node's glosses, without back-off policies.
- DIMAP: employed both WordNet and the New Oxford Dictionary of English. With the first sense as a back-off when tied scores occurred.
- UNED-LS-U: for each sense of the target, they enriched the sense describer through the first five hyponyms of it and a dictionary built from 3200 books from Project Gutenberg. They adopted a back-off policy to the first sense and discarded the senses accounting for less than 10 percent of files in SemCor).

7 Conclusion and discussion

7.1 Local context and topic context

On the analysis of standard deviation of precision on different stage in Figure 1 and 2 we can conclude that the optimum size for HSN to HSS was ±10 words for nouns, reflecting a sensitivity to only local context, whilst HWL and HSL reflected significant improvement up to ±60 reflecting a sensitivity to topical context. In the case of verbs HSA showed little significant context sensitivity, HSN showed some positive sensitivity to local context but increasing beyond ±5 had a negative effect, HSM and HSS to HSL showed some sensitivity to broader topical context but this plateaued around ±20 to 30.

7.2 The analysis of different heuristics.

HWL and HSL were clearly superior for both noun and verb tasks, with the superiority of HSL being significantly greater and more comparable between noun and verb tasks with the difference scarcely reaching significance. These observations remain true with the addition of the EAT information. After transformations with EAT for nouns, HSL and HWL no longer differ significantly in performance, forming a single group with relatively higher precision, whilst the other heuristics clump together into another group with lower precision, reflecting a negative effect from EAT. In the verb case, HWL and HSL, HSM and HSS, and HSN and HSA form three significantly different groups with reference to their precision, reflecting poor performance of both normalized heuristics (HSN and HSA) and a significantly improved result of HWL from the EAT data.

All of this implies that in the lexical hub for WSD, the correct meaning of a word should hold as many links as possible with a relatively large number of context words. These links can be in the level of word form (HWL) or word sense (HSL). HSL achieved the highest precision in both nouns and verbs.

7.3 The interaction of EAT in WSD

For the noun sense disambiguation, the paired two sample for mean of the t-Test showed us that RS and SRORRS transformations can significantly improve the precision of disambiguation of HWL and HSL ($P<0.05$, at the confidence level of 95 percent). All four transformations using EAT for verb disambiguation are significantly better than its straightforward context case on HWL and HSL ($P<0.05$, at the confidence level of 95 percent).

It demonstrated that both the syntagmatic relation and other domain information in the EAT can help discriminate word sense. With the transformation of context surroundings of the target, the similarity metrics can compare the likeness of nouns and verbs, although we can exploit the derived form of word in WordNet to facilitate the comparison.

7.4 Comparison with other methods

The lexical hub reached comparatively higher precision in both nouns (45.8%) and verbs (35.6%). This contrasted with other similarity based methods and the unsupervised systems in SENSEVAL-2. Note that we don't adopt any

back-off policy such as the commonest sense of word used by UNED-LS-U and DIMAP.

Although the noun and verb similarity metrics in this paper are based on edge-counting without any aid of frequency information from corpora, they performed very well in the task of WSD in relation to other information based metrics and definition matching methods. Especially in the verb case, the metric significantly outperformed other metrics.

8 Conclusion and future work

In this paper we defined the lexical hub and proposed its use for processing word sense disambiguation, achieving results that are comparatively better than most unsupervised systems of SENSEVAL-2 in the literature. Since WordNet only organizes the paradigmatic relations of words, unlike previous methods, which are only based on WordNet, we fed the syntagmatic relations of words from the EAT into the noun and verb similarity metrics, and significantly improved the results of WSD, given that no back-off was applied. Moreover, we only utilized the unordered raw context information without any pragmatic knowledge and syntactic information; there is still a lot of work to fuse them in the future research. In terms of the heuristics evaluated, richness of sense or word connectivity is much more important than the strength of individual word or sense linkages. An interesting question is whether these results will be borne out in other datasets. In the forthcoming work we will investigate their validity in the lexical task of SENSEVAL-3.

References

Barzilay, R. and M. Elhadad (1997). Using Lexical Chains for Text Summarization. In the Intelligent Scalable Text Summarization Workshop (ISTS'97), ACL, Madrid, Spain.

Chaffin, R., et al. (1994). The Paradigmatic Organization of Verbs in the Mental Lexicon. Trenton State College.

Fellbaum, C. (1998). Wordnet: An Electronic Lexical Database. Cambridge MA, USA, The MIT Press.

Halliday, M. A. K. and R. Hasan (1976). Cohesion in English. London, London:Longman.

Hirst, G. and D. St-Onge (1997). Lexical Chains as Representations of Context for the Detection and Correction of Malapropisms. Wordnet. C. Fellbaum. Cambridge, MA, The Mit Press.

Ide, N. and J. Véronis (1998). Word Sense Disambiguation: The State of the Art. Computational linguistics 24(1).

Jiang, J. and D. Conrath (1997). Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. In the 10th International Conference on Research in Computational Linguistics (ROCLING), Taiwan.

Kilgarriff, A. and M. Palmer (2000). Introduction, Special Issue on Senseval: Evaluating Word Sense Disambiguation Programs. Computers and the Humanities 34(1-2): 1-13.

Kilgarriff, A. and J. Rosenzweig (2000). Framework and Results for English Senseval. Computers and the Humanities 34(1-2): 15-48.

Kiss, G. R., et al. (1973). The Associative Thesaurus of English and Its Computer Analysis. Edinburgh, University Press.

Lesk, M. (1986). Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Code from an Ice Cream Cone. In the 5th annual international conference on systems documentation, ACM Press.

Morris, J. and G. Hirst (1991). Lexical Cohesion Computed by Thesaural Relations as an Indicator of the Structure of Text. Computational linguistics 17(1).

Pedersen, T., et al. (2003). Maximizing Semantic Relatedness to Perform Word Sense Disambiguation.

Sinopalnikova, A. (2004). Word Association Thesaurus as a Resource for Building Wordnet. In GWC 2004.

Sussna, M. (1993). Word Sense Disambiguation for Free-Text Indexing Using a Massive Semantic Network. In CKIM'93.

Yang, D. and D. M. W. Powers (2005). Measuring Semantic Similarity in the Taxonomy of Wordnet. In the Twenty-Eighth Australasian Computer Science Conference (ACSC2005), Newcastle, Australia, ACS.

Yang, D. and D. M. W. Powers (2006). Verb Similarity on the Taxonomy of Wordnet. In the 3rd International WordNet Conference (GWC-06), Jeju Island, Korea.

Yarowsky, D. (1992). Word Sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora. In the 14th International Conference on Computational Linguistics, Nates, France.

Yarowsky, D. (1993). One Sense Per Collocation. In ARPA Human Language Technology Workshop, Princeton, New Jersey.

Stochastic Discourse Modeling in Spoken Dialogue Systems Using Semantic Dependency Graphs

Jui-Feng Yeh, Chung-Hsien Wu and Mao-Zhu Yang
Department of Computer Science and Information Engineering
National Cheng Kung University
No. 1, Ta-Hsueh Road, Tainan, Taiwan, R.O.C.
{jfyeh, chwu, mzyang}@csie.ncku.edu.tw

Abstract

This investigation proposes an approach to modeling the discourse of spoken dialogue using semantic dependency graphs. By characterizing the discourse as a sequence of speech acts, discourse modeling becomes the identification of the speech act sequence. A statistical approach is adopted to model the relations between words in the user's utterance using the semantic dependency graphs. Dependency relation between the headword and other words in a sentence is detected using the semantic dependency grammar. In order to evaluate the proposed method, a dialogue system for medical service is developed. Experimental results show that the rates for speech act detection and task-completion are 95.6% and 85.24%, respectively, and the average number of turns of each dialogue is 8.3. Compared with the Bayes' classifier and the Partial-Pattern Tree based approaches, we obtain 14.9% and 12.47% improvements in accuracy for speech act identification, respectively.

1 Introduction

It is a very tremendous vision of the computer technology to communicate with the machine using spoken language (Huang et al., 2001; Allen at al., 2001). Understanding of spontaneous language is arguably the core technology of the spoken dialogue systems, since the more accurate information obtained by the machine (Higashinaka et al., 2004), the more possibility to finish the dialogue task. Practical use of speech act theories in spoken language processing (Stolcke et al. 2000; Walker and Passonneau 2001; Wu et al., 2004) have given both insight and deeper understanding of verbal communication. Therefore, when considering the whole discourse, the relationship between the speech acts of the dialogue turns becomes extremely important. In the last decade, several practicable dialogue systems (McTEAR, 2002), such as air travel information service system, weather forecast system, automatic banking system, automatic train timetable information system, and the Circuit-Fix-it shop system, have been developed to extract the user's semantic entities using the semantic frames/slots and conceptual graphs. The dialogue management in these systems is able to handle the dialogue flow efficaciously. However, it is not applicable to the more complex applications such as "Type 5: the natural language conversational applications" defined by IBM (Rajesh and Linda, 2004). In Type 5 dialog systems, it is possible for the users to switch directly from one ongoing task to another. In the traditional approaches, the absence of precise speech act identification without discourse analysis will result in the failure in task switching. The capability for identifying the speech act and extracting the semantic objects by reasoning plays a more important role for the dialog systems. This research proposes a semantic dependency-based discourse model to capture and share the semantic objects among tasks that switch during a dialog for semantic resolution. Besides

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 937–944,
Sydney, July 2006.

acoustic speech recognition, natural language understanding is one of the most important research issues, since understanding and application restriction on the small scope is related to the data structures that are used to capture and store the meaningful items. Wang et al. (Wang et al., 2003) applied the object-oriented concept to provide a new semantic representation including semantic class and the learning algorithm for the combination of context free grammar and N-gram.

Among these approaches, there are two essential issues about dialogue management in natural language processing. The first one is how to obtain the semantic object from the user's utterances. The second is a more effective speech act identification approach for semantic understanding is needed. Since speech act plays an important role in the development of dialogue management for dealing with complex applications, speech act identification with semantic interpretation will be the most important topic with respect to the methods used to control the dialogue with the users. This paper proposes an approach integrating semantic dependency graph and history/discourse information to model the dialogue discourse (Kudo and Matsumoto, 2000; Hacioglu et al., 2003; Gao and Suzuki, 2003). Three major components, such as semantic relation, semantic class and semantic role are adopted in the semantic dependency graph (Gildea and Jurasfky, 2002; Hacioglu and Ward, 2003). The semantic relations constrain the word sense and provide the method for disambiguation. Semantic roles are assigned when the relation established among semantic objects. Both semantic relations and roles are defined in many knowledge resources or ontologies, such as FrameNet (Baker et al., 2004) and HowNet with 65,000 concepts in Chinese and close to 75,000 English equivalents, is a bilingual knowledge-base describing relations between concepts and relations between the attributes of concepts with ontological view (Dong and Dong 2006). Generally speaking, semantic class is defined as a set with the elements that are usually the words with the same semantic interpretation. Hypernyms that are superordinate concepts of the words are usually used as the semantic classes just like the Hypernyms of synsets in WordNet (http://www.cogsci.princeton.edu/~wn/) or definitions of words' primary features in HowNet. Besides, the approach for understanding tries to find the implicit semantic dependency between the concepts and the dependency structure between concepts in the utterance are also taken into consideration. Instead of semantic frame/slot, semantic dependency graph can keep more information for dialogue understanding.

2 Semantic Dependency Graph

Since speech act theory is developed to extract the functional meaning of an utterance in the dialogue (Searle, 1979), discourse or history can be defined as a sequence of speech acts, $H^t = \{SA^1, SA^2, ...SA^{t-1}, SA^t\}$, and accordingly the speech act theory can be adopted for discourse modeling. Based on this definition, the discourse analysis in semantics using the dependency graphs tries to identify the speech act sequence of the discourse. Therefore, discourse modeling by means of speech act identification considering the history is shown in Equation (1). By introducing the hidden variable D_i, representing the i-th possible dependency graph derived from the word sequence W. The dependency relation, r_k, between word w_k and headword w_{kh} is extracted using HowNet and denoted as $DR(w_k, w_{kh}) \equiv r_k$. The dependency graph which is composed of a set of dependency relations in the word sequence *W* is defined as

$$D_i(W) = \{DR_1^i(w_1, w_{1h}), DR_2^i(w_2, w_{2h}), ..., DR_{m-1}^i(w_{m-1}, w_{(m-1)h})\}.$$

The probability of hypothesis SA^t given word sequence W and history H^{t-1} can be described in Equation (1). According to the Bayes' rule, the speech act identification model can be decomposed into two components, $P(SA^t \mid D_i, W, H^{t-1})$ and $P(D_i \mid W, H^{t-1})$, described in the following.

$$\begin{aligned} SA^* &= \arg\max_{SA^t} P(SA^t \mid W, H^{t-1}) \\ &= \arg\max_{SA^t} \sum_{D_i} P(SA^t, D_i \mid W, H^{t-1}) \\ &= \arg\max_{SA^t} \sum_{D_i} P(SA^t \mid D_i, W, H^{t-1}) \times P(D_i \mid W, H^{t-1}) \end{aligned} \quad (1)$$

where SA^* and SA^t are the most probable speech act and the potential speech act at the t-th dialogue turn, respectively. $W=\{w_1, w_2, w_3, \ldots, w_m\}$ denotes the word sequence extracted from the user's utteance without considering the stop words. H^{t-1} is the history representing the previous *t*-1 turns.

2.1 Speech act identification using semantic dependency with discourse analysis

In this analysis, we apply the semantic dependency, word sequence, and discourse analysis to the identification of speech act. Since D_i is the i-th possible dependency graph derived from word sequence W, speech act identification with semantic dependency can be simplified as Equation (2).

$$P\left(SA^t \mid D_i, W, H^{t-1}\right) \cong P\left(SA^t \mid D_i, H^{t-1}\right) \tag{2}$$

According to Bayes' rule, the probability $P\left(SA^t \mid D_i, H^{t-1}\right)$ can be rewritten as:

$$P\left(SA^t \mid D_i, H^{t-1}\right) = \frac{P\left(D_i, H^{t-1} \mid SA^t\right) P\left(SA^t\right)}{\sum_{SA_i} P\left(D_i, H^{t-1} \mid SA_i\right) P\left(SA_i\right)} \tag{3}$$

As the history is defined as the speech act sequence, the joint probability of D_i and H^{t-1} given the speech act SA^t can be expressed as Equation (4). For the problem of data sparseness in the training corpus, the probability, $P\left(D_i, SA^1, SA^2, ..., SA^{t-1} \mid SA^t\right)$, is hard to obtain and the speech act bi-gram model is adopted for approximation.

$$\begin{aligned} & P\left(D_i, H^{t-1} \mid SA^t\right) \\ & = P\left(D_i, SA^1, SA^2, ..., SA^{t-1} \mid SA^t\right) \\ & \cong P\left(D_i, SA^{t-1} \mid SA^t\right) \end{aligned} \tag{4}$$

For the combination of the semantic and syntactic structures, the relations defined in HowNet are employed as the dependency relations, and the hypernym is adopted as the semantic concept according to the primary features of the words defined in HowNet. The headwords are decided by the algorithm based on the part of speech (POS) proposed by Academia Sinica in Taiwan. The probabilities of the headwords are estimated according to the probabilistic context free grammar (PCFG) trained on the Treebank developed by Sinica (Chen et al., 2001). That is to say, the headwords are extracted according to the syntactic structure and the dependency graphs are constructed by the semantic relations defined in HowNet. According to previous definition with independent assumption and the bigram smoothing of the speech act model using the back-off procedure, we can rewrite Equation (4) into Equation (5).

$$\begin{aligned} & P\left(D_i, SA^{t-1} \mid SA^t\right) \\ & = \alpha \prod_{k=1}^{m-1} P(DR_k^i(w_k, w_{kh}), SA^{t-1} \mid SA^t) + \\ & \quad (1-\alpha) \prod_{k=1}^{m-1} P(DR_k^i(w_k, w_{kh}) \mid SA^t) \end{aligned} \tag{5}$$

where α is the mixture factor for normalization.

According to the conceptual representation of the word, the transformation function, $f(\cdot)$, transforms the word into its hypernym defined as the semantic class using HowNet. The dependency relation between the semantic classes of two words will be mapped to the conceptual space. Also the semantic roles among the dependency relations are obtained. On condition that SA^t, SA^{t-1} and the relations are independent, the equation becomes

$$\begin{aligned} & P(DR_k^i(w_k, w_{kh}), SA^{t-1} \mid SA^t) \\ & \cong P(DR_k^i(f(w_k), f(w_{kh})), SA^{t-1} \mid SA^t) \\ & = P(DR_k^i(f(w_k), f(w_{kh})) \mid SA^t) P(SA^{t-1} \mid SA^t) \end{aligned} \tag{6}$$

The conditional probability, $P(DR_k^i(f(w_k), f(w_{kh})) \mid SA^t)$ and $P(SA^{t-1} \mid SA^t)$, are estimated according to Equations (7) and (8), respectively.

$$\begin{aligned} & P(DR_k^i(f(w_k), f(w_{kh})) \mid SA^t) \\ & = \frac{C(f(w_k), f(w_{kh}), r_k, SA^t)}{C(SA^t)} \end{aligned} \tag{7}$$

$$P(SA^{t-1} \mid SA^t) = \frac{C(SA^{t-1}, SA^t)}{C(SA^t)} \tag{8}$$

where $C(\cdot)$ represents the number of events in the training corpus. According to the definitions in Equations (7) and (8), Equation (6) becomes practicable.

2.2 Semantic dependency analysis using word sequence and discourse

Although the discourse can be expressed as the speech act sequence $H^t = \{SA^1, SA^2, ...SA^{t-1}, SA^t\}$, the dependency graph D_i is determined mainly by *W*, but not H^{t-1}. The probability that defines semantic dependency analysis using the words sequence and discourse can be rewritten in the following:

$$\begin{aligned} & P\left(D_i \mid W, H^{t-1}\right) \\ & = P(D_i \mid W, SA^{t-1}, SA^{t-2}, ..., SA^1) \\ & \cong P(D_i \mid W) \end{aligned} \tag{9}$$

and

$$P(D_i \mid W) = \frac{P(D_i, W)}{P(W)}, \tag{10}$$

Seeing that several dependency graphs can be generated from the word sequence *W*, by introducing the hidden factor D_i, the probability $P(W)$ can be the sum of the probabilities $P(D_i, W)$ as Equation (11).

$$P(W) = \sum_{D_i \ : yield(D_i)=W} P(D_i, W) \tag{11}$$

Because D_i is generated from *W*, D_i is the sufficient to represent *W* in semantics. We can estimate the joint probability $P(D_i, W)$ only from the dependency relations D_i. Further, the dependency relations are assumed to be independent with each other and therefore simplified as

$$P(D_i, W) = \prod_{k=1}^{m-1} P(DR_k^i(w_k, w_{kh})) \tag{12}$$

The probability of the dependency relation between words is defined as that between the concepts defined as the hypernyms of the words, and then the dependency rules are introduced. The probability $P(r_k \mid f(w_k), f(w_{kh}))$ is estimated from Equation (13).

$$\begin{aligned} & P(DR_k^i(w_k, w_{kh})) \\ & \equiv P(DR_k^i(f(w_k), f(w_{kh}))) \\ & = P(r_k \mid f(w_k), f(w_{kh})) \\ & = \frac{C(r_k, f(w_k), f(w_{kh}))}{C(f(w_k), f(w_{kh}))} \end{aligned} \tag{13}$$

According to Equations (11), (12) and (13), Equation (10) is rewritten as the following equation.

$$\begin{aligned} P(D_i \mid W) &= \frac{\prod_{k=1}^{m-1} P(DR_k^i(w_k, w_{kh}))}{\sum_{D_i \ : yield(D_i)=W} \prod_{k=1}^{m-1} P(DR_k^i(w_k, w_{kh}))} \\ &= \frac{\prod_{k=1}^{m-1} \frac{C(r_k, f(w_k), f(w_{kh}))}{C(f(w_k), f(w_{kh}))}}{\sum_{D_i \ : yield(D_i)=W} \prod_{k=1}^{m-1} \frac{C(r_k, f(w_k), f(w_{kh}))}{C(f(w_k), f(w_{kh}))}} \end{aligned} \tag{14}$$

where function, $f(\cdot)$, denotes the transformation from the words to the corresponding semantic classes.

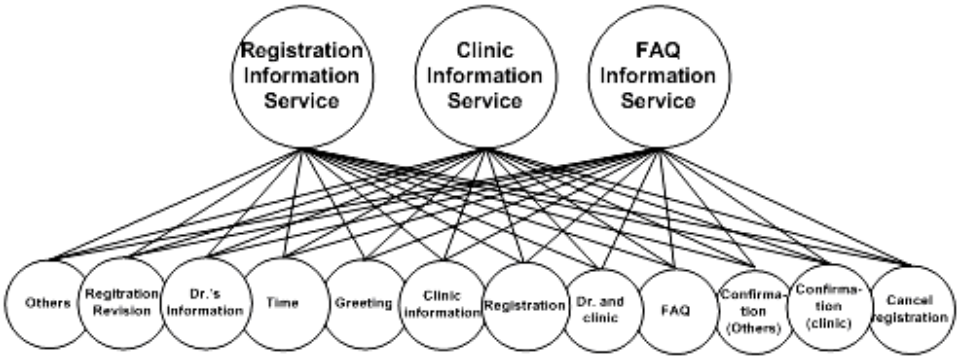


Figure 1. Speech acts corresponding to multiple services in the medical domain

3 Experiments

In order to evaluate the proposed method, a spoken dialogue system for medical domain with multiple services was investigated. Three main services: registration information service, clinic information service, and FAQ information service are used. This system mainly provides the function of on-line registration. For this goal, the health education documents are provided as the FAQ files. And the inference engine about the clinic information according to the patients' syndromes is constructed according to a medical encyclopedia. An example is illustrated as figure 2:

System:午安。需要服務嗎?
Good Afternoon. May I help you?
User:是的，我要看醫生。
Yes, I should be diagnosed by doctor.
System:請問您要掛哪一科?
Which department would you like to make an appointment for?
User:我不知道。
I do not know.
System:請問你有那些症狀？
What symptoms are with you?
User:我喉嚨痛又發燒。而且全身酸痛。
I am with a sore throat and a fever. I ached all over, too.
System:好的，這些症狀多久了？
Well, how long have you had these symptoms?
User:好幾天了。
For several days.
System:建議您掛內科醫師。
My suggestion is that you may visit the Internal Medicine.
User:請問內科何時有門診？
What's the visiting schedule of the Internal Medicine?
System:您可以參考醫院門診表，決定您的時間，我們的時間有…。
You can check the Services Schedule and decide a convenient time for you. The Available time for now is....

Figure 2 An example of dialog

12 Speech acts are defined and shown in Figure 1. Every service corresponds to the 12 speech acts with different probabilities.

The acoustic speech recognition engine embedded in dialog system based on Hidden Markov Models (HMMs) was constructed. The feature vector is parameterized on 26 MFCC coefficients. The decoding strategy is based on a classical Viterbi algorithm. The evaluation results by the character error rate (CER) for a Chinese speech recognition system is 18.3 percent and the vocabulary size of the language is 25,132.

3.1 Analysis of corpus

The training corpus was collected using the on-line recording from National Cheng Kung University Hospital in the first phase and the Wizard-of-Oz method in the second phase. Totally, there are 1,862 dialogues with 13,986 sentences in the corpus. The frequencies of the speech acts used in the system are shown in Figure 3.

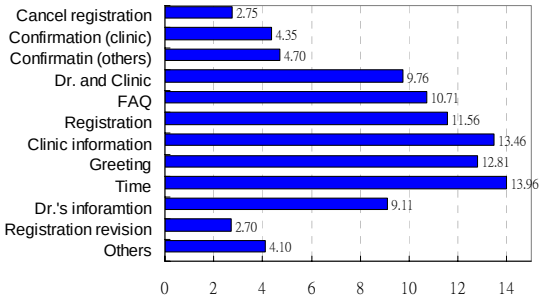


Figure 3 Frequencies for each speech act

The number of dialogue turns is also important to the success of the dialogue task. According to the observation of the corpus, we can find that the dialogues with more than 15 turns usually failed to complete the dialogue, that is to say, the common ground cannot be achieved. These failed dialogues were filtered out from the training corpus before conducting the following experiments. The distribution of the number of turns per dialogue is shown in Figure 4.

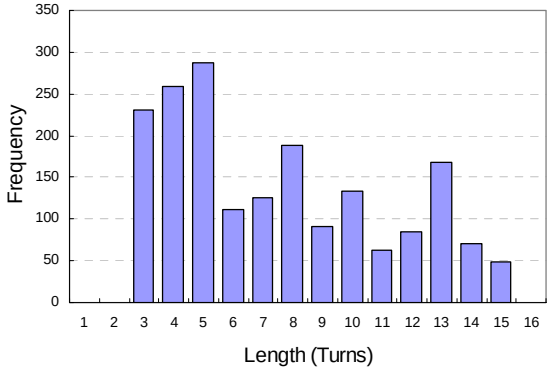


Figure 4. The distribution of the number of turns per dialogue

3.2 Precision of speech act identification related to the corpus size

The size of the training corpus is crucial to the practicability of the proposed method. In this experiment, we analyze the effect of the number of sentences according to the precision rate of the speech act using the semantic dependency graphs with and without the discourse information. From the results, the precision rates for speech act identification achieved 95.6 and 92.4 percentages for the training corpus containing 10,036 and 7,012 sentences using semantic dependency graphs with and without history, respectively. This means that semantic dependency graph with discourse outperforms that without discourse, but more training data are needed to include the discourse for speech act identification. Fig. 5 shows the relationship between the speech act identification rate and the size of the training corpus. From this figure, we can find that more training sentences for the semantic dependency graph with discourse analysis are needed than that without discourse. This implies discourse analysis plays an important role in the identification of the speech act.

3.3 Performance analysis of semantic dependency graph

To evaluate the performance, two systems were developed for comparison. One is based on the Bayes' classifier (Walker et al., 1997), and the other is the use of the partial pattern tree (Wu et al., 2004) to identify the speech act of the user's utterances. Since the dialogue discourse is defined as a sequence of speech acts. The prediction of speech act of the new input utterance becomes the core issue for discourse modeling. The accuracy for speech act identification is shown in Table 1.

According to the observation of the results, semantic dependency graphs obtain obvious improvement compared to other approaches. The reason is that not only the meanings of the words or concepts but also the structural information and the implicit semantic relation defined in the knowledge base are needed to identify the speech act.Besides, taking the discourse into consideration will improve the prediction about the speech act of the new or next utterance. This means the discourse model can improve the accuracy of the speech act identification, that is to say, discourse modeling can help understand the user's desired intension especially when the answer is very short.

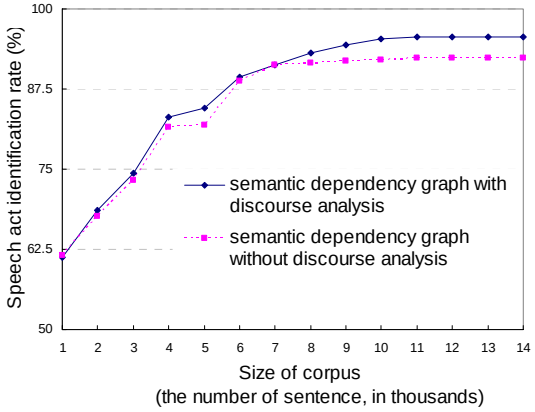


Figure 5. The relation between the speech act identification rate and the size of training corpus

Speech act	Semantic dependency graph		PPT	Bayes' Classifier
	With discourse analysis	Without discourse analysis		
Clinic information (26 sentences)	100 (26)	96.1 (25)	88 (23)	92 (24)
Dr.'s information (42 sentences)	97 (41)	92.8 (39)	66.6 (28)	92.8 (39)
Confirmation(others) (42 sentences)	95 (40)	95 (40)	95 (40)	95 (40)
Others (14 sentences)	57.1 (8)	50 (7)	43 (6)	38 (5)
FAQ (13 sentences)	70 (9)	53.8 (7)	61.5 (8)	46 (6)
Clinic information (135 sentences)	98.5 (133)	96.2 (130)	91.1 (123)	93.3 (126)
Time (38)	94.7 (36)	89.4 (34)	97.3 (37)	92.1 (35)
Registration (75)	100 (75)	100 (75)	86.6 (65)	86.6 (65)
Cancel registration (10)	90 (9)	80 (8)	60 (6)	80 (8)
Average Precision	**95.6**	**92.4**	**85**	**88.1**

Table 1 The accuracy for speech act identification

For example, the user may only say “yes” or “no” for confirmation. The misclassification in speech act will happen due to the limited information. However, it can obtain better interpretation by introducing the semantic dependency relations as well as the discourse information.

To obtain the single measurement, the average accuracy for speech act identification is shown in Table 1. The best approach is the semantic dependency graphs with the discourse. This means the information of the discourse can help speech act identification. And the semantic dependency graph outperforms the traditional approach due to the semantic analysis of words with their corresponding relations.

The success of the dialog lies on the achievement of the common ground between users and machine which is the most important issue in dialogue management. To compare the semantic dependency graph with previous approaches, 150 individuals who were not involved in the development of this project were asked to use the dialogue system to measure the task success rate. To filter out the incomplete tasks, 131 dialogs were employed as the analysis data in this experiment. The results are listed in Table 2.

	SDG[1]	**SDG[2]**	**PPT**	**Bayes’**
Task completion rate	87.2	85.5	79.4	80.2
Number of turns on average	8.3	8.7	10.4	10.5

SDG[1] :With discourse analysis, **SDG[2]** :Without discourse

Table 2 Comparisons on the Task completion rate and the number of dialogue turns between different approaches

We found that the dialogue completion rate and the average length of the dialogs using the dependency graph are better than those using the Bayes’ classifier and partial pattern tree approach. Two main reasons are concluded: First, dependency graph can keep the most important information in the user’s utterance, while in semantic slot/frame approach, the semantic objects not matching the semantic slot/frame are generally filtered out. This approach is able to skip the repetition or similar utterances to fill the same information in different semantic slots. Second, the dependency graph-based approach can provide the inference to help the interpretation of the user’s intension.

For semantic understanding, correct interpretation of the information from the user’s utterances becomes inevitable. Correct speech act identification and correct extraction of the semantic objects are both important issues for semantic understanding in the spoken dialogue systems. Five main categories about medical application, clinic information, Dr.’s information, confirmation for the clinic information, registration time and clinic inference, are analyzed in this experiment.

	SDG	**PPT**	**Bayes’**
Clinic information	95.0	89.5	90.3
Dr.’s information	94.3	71.7	92.4
Confirmation (Clinic)	98.0	98.0	98.0
Clinic	97.3	74.6	78.6
Time	97.6	97.8	95.5

SDG:With discourse analysis

Table 3 Correction rates for semantic object extraction

According to the results shown in Table 3, the worst condition happened in the query for the Dr.’s information using the partial pattern tree. The mis-identification of speech act results in the un-matched semantic slots/frames. This condition will not happen in semantic dependency graph, since the semantic dependency graph always keeps the most important semantic objects according to the dependency relations in the semantic dependency graph instead of the semantic slots. Rather than filtering out the unmatched semantic objects, the semantic dependency graph is constructed to keep the semantic relations in the utterance. This means that the system can preserve most of the user’s information via the semantic dependency graphs. We can observe the identification rate of the speech act is higher for the semantic dependency graph than that for the partial pattern tree and Bayes’ classifier as shown in Table 3.

4 Conclusion

This paper has presented a semantic dependency graph that robustly and effectively deals with a variety of conversational discourse information in the spoken dialogue systems. By modeling the dialogue discourse as the speech act sequence, the predictive method for speech act identification is proposed based on discourse analysis instead of keywords only. According to the corpus analysis, we can find the model proposed in this paper is practicable and effective. The results of the experiments show the semantic dependency graph outperforms those based on the Bayes' rule and partial pattern trees. By integrating discourse analysis this result also shows the improvement obtained not only in the identification rate of speech act but also in the performance for semantic object extraction.

Acknowledgements

The authors would like to thank the National Science Council, Republic of China, for its financial support of this work, under Contract No. NSC 94-2213-E-006-018.

References

J. F. Allen, D. K. Byron, D. M. Ferguson, L. Galescu, and A. Stent. 2001. Towards Conversational Human-Computer Interaction. *AI Magazine*.

C. F. Baker, C. J. Fillmore, and J. B. Lowe. 1998. The Berkeley FrameNet Project. *In Proceedings of COLING/ACL*. 86-90

K. J. Chen, C. R. Huang, F.Y. Chen, C. C. Luo, M. C. Chang, and C.J. Chen. 2001. Sinica Treebank: Design Criteria, representational issues and immplementation. *In Anne Abeille, editor, Building and Using Syntactically Annotated Corpora.* Kluwer. 29-37

Z. Dong and Q. Dong. 2006. HowNet and the computation of meaning. *World Scientific Publishing Co Inc.*

J. Gao, and H. Suzuki. 2003. Unsupervised learning of dependency structure for language modeling. *In Proceedings of ACL 2003*, 521-528.

D. Gildea and D. Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics, 28(3).* 245–288.

K. Hacioglu, S. Pradhan, W. Ward, J. Martin, and D. Jurafsky. 2003. Shallow semantic parsing using support vector machines. *Technical Report TR-CSLR-2003-1, Center for Spoken Language Research, Boulder*, Colorado.

K. Hacioglu and W. Ward. 2003. Target word detection and semantic role chunking using support vector machines. *In HLT-03*.

R. Higashinaka, N. Miyazaki, M. Nakano, and K. Aikawa. 2004. Evaluating Discourse Understanding in Spoken Dialogue Systems. *ACM Transactions on Speech and Language Processing (TSLP)*, Volume 1, 1-20.

X. Huang, A. Acero, and H.-W. Hon. 2001. *Spoken Language Proceeding*. Prentice-Hall,Inc.

T. Kudo and Y. Matsumoto. 2000. Japanese Dependency Structure Analysis Based on Support Vector Machines. *In Proceedings of the EMLNP*. 18–25

M. F. McTEAR. 2002. Spoken Dialogue Technology: Enabling the Conversational User Interface. *ACM Computer Surveys*, Vol 34, No. 1, 90-169..

B. Rajesh, and B. Linda. 2004. Taxonomy of speech-enabled applications (http://www106.ibm.com/developerworks/wireless/library/wi-tax/)

J. Searle. 1979. Expression and Meaning: Studies in the Theory of Speech Acts. New York, Cambridge University Press.

A. Stolcke, K. Ries, N. Coccaro, E. Shriberg, R. Bates, D. Jurafsky, P. Taylor, R. Martin, C. Van Ess-Dykema, and M. Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics 26(3)*, 339--373.

M. A. Walker, D. Litman, C. Kamm, and A. Abella, 1997. PARADISE: a general framework for evaluating spoken dialogue agents. *In Proceedings of the ACL*, 271–280

M. Walker and R. Passonneau. 2001. DATE: a dialogue act tagging scheme for evaluation of spoken dialogue systems. *In Proceedings of the first international conference on Human language technology research*. 1-8.

Y.-Y. Wang and A. Acero. 2003. Combination of CFG and N-gram Modeling in Semantic Grammar Learning, *In Proceedings of the Eurospeech Conference*. Geneva, Switzerland. September 2003.

C.-H. Wu, J.-F. Yeh, and M.-J. Chen. 2004. Speech Act Identification using an Ontology-Based Partial Pattern Tree. *in Proceedings of ICSLP 2004*, Jeju, Korea, 2004.

HAL-based Cascaded Model for Variable-Length Semantic Pattern Induction from Psychiatry Web Resources

Liang-Chih Yu and **Chung-Hsien Wu**
Department of Computer Science and Information Engineering
National Cheng Kung University
Tainan, Taiwan, R.O.C.
{lcyu, chwu}@csie.ncku.edu.tw

Fong-Lin Jang
Department of Psychiatry
Chi-Mei Medical Center
Tainan, Taiwan, R.O.C.
jcj0429@seed.net.tw

Abstract

Negative life events play an important role in triggering depressive episodes. Developing psychiatric services that can automatically identify such events is beneficial for mental health care and prevention. Before these services can be provided, some meaningful semantic patterns, such as <lost, parents>, have to be extracted. In this work, we present a text mining framework capable of inducing variable-length semantic patterns from *unannotated* psychiatry web resources. This framework integrates a cognitive motivated model, *Hyperspace Analog to Language (HAL)*, to represent words as well as combinations of words. Then, a cascaded induction process (CIP) bootstraps with a small set of seed patterns and incorporates *relevance feedback* to iteratively induce more relevant patterns. The experimental results show that by combining the HAL model and relevance feedback, the CIP can induce semantic patterns from the unannotated web corpora so as to reduce the reliance on annotated corpora.

1 Introduction

Depressive disorders have become a major threat to mental health. People in their daily life may suffer from some negative or stressful life events, such as death of a family member, arguments with a spouse, loss of a job, and so forth. Such life events play an important role in triggering depressive symptoms, such as depressed mood, suicide attempts, and anxiety. Therefore, it is desired to develop a system capable of identifying negative life events to provide more effective psychiatric services. For example, through the negative life events, the health professionals can know the background information about subjects so as to make more correct decisions and suggestions. Negative life events are often expressed in natural language segments (e.g., sentences). To identify them, the critical step is to transform the segments into machine-interpretable semantic representation. This involves the extraction of key *semantic patterns* from the segments. Consider the following example.

*Two years ago, I **lost** my **parents**.* (Event)

Since that, I have attempted to kill myself several times. (Suicide)

In this example, the semantic pattern <lost, parents> is constituted by two words, which indicates that the subject suffered from a negative life event that triggered the symptom "Suicide". A semantic pattern can be considered as a semantically plausible combination of k words, where k is the length of the pattern. Accordingly, a semantic pattern may have variable length. In Wu et al.'s study (2005), they have presented a methodology to identify depressive symptoms. In this work, we go a further step to devise a text mining framework for variable-length semantic pattern induction from psychiatry web resources.

Traditional approaches to semantic pattern induction can be generally divided into two streams: knowledge-based approaches and corpus-based approaches (Lehnert et al., 1992; Muslea, 1999). Knowledge-based approaches rely on exploiting expert knowledge to design handcrafted semantic patterns. The major limitations of such approaches include the requirement of significant time and effort on designing the handcrafted patterns. Besides, when applying to a new domain, these patterns have to be redesigned. Such limitations form a knowledge acquisition bottleneck. A possible solution to reducing the problem is to use a general-purpose ontology

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 945–952,
Sydney, July 2006.

such as WordNet (Fellbaum, 1998), or a domain-specific ontology constructed using automatic approaches (Yeh et al., 2004). These ontologies contain rich concepts and inter-concept relations such as hypernymy-hyponymy relations. However, an ontology is a static knowledge resource, which may not reflect the dynamic characteristics of language. For this consideration, we instead refer to the web resources, or more restrictedly, the psychiatry web resources as our knowledge resource.

Corpus-based approaches can automatically learn semantic patterns from domain corpora by applying statistical methods. The corpora have to be annotated with domain-specific knowledge (e.g., events). Then, various statistical methods can be applied to induce variable-length semantic patterns from all possible combinations of words in the corpora. However, statistical methods may suffer from data sparseness problem, thus they require large corpora with annotated information to obtain more reliable parameters. For some application domains, such annotated corpora may be unavailable. Therefore, we propose the use of web resources as the corpora. When facing with the web corpora, traditional corpus-based approaches may be infeasible. For example, it is impractical for health professionals to annotate the whole web corpora. Besides, it is also impractical to enumerate all possible combinations of words from the web corpora, and then search for the semantic patterns.

To address the problems, we take the notion of weakly supervised (Stevenson and Greenwood, 2005) or unsupervised learning (Hasegawa, 2004; Grenager et al., 2005) to develop a framework able to bootstrap with a small set of seed patterns, and then induce more relevant patterns form the *unannotated* psychiatry web corpora. By this way, the reliance on annotated corpora can be significantly reduced. The proposed framework is divided into two parts: *Hyperspace Analog to Language (HAL)* model (Burgess et al., 1998; Bai et al., 2005), and a cascaded induction process (CIP). The HAL model, which is a cognitive motivated model, provides an informative infrastructure to make the CIP capable of learning from unannotated corpora. The CIP treats the variable-length induction task as a cascaded process. That is, it first induces the semantic patterns of length two, then length three, and so on. In each stage, the CIP initializes the set of semantic patterns to be induced based on the better results of the previous stage, rather than enumerating all possible combinations of words. This would be helpful to avoid noisy patterns propagating to the next stage, and the search space can also be reduced.

A crucial step for semantic pattern induction is the representation of words as well as combinations of words. The HAL model constructs a high-dimensional context space for the psychiatry web corpora. Each word in the HAL space is represented as a vector of its context words, which means that the sense of a word can be inferred through its contexts. Such notion is derived from the observation of human behavior. That is, when an unknown word occurs, human beings may determine its sense by referring to the words appearing in the contexts. Based on the cognitive behavior, if two words share more common contexts, they are more semantically similar. To further represent a semantic pattern, the HAL model provides a mechanism to combine its constituent words over the HAL space.

Once the HAL space is constructed, the CIP takes as input a seed pattern per run, and in turn induces the semantic patterns of different lengths. For each length, the CIP first creates the initial set based on the results of the previous stage. Then, the induction process is iteratively performed to induce more patterns relevant to the given seed pattern by comparing their context distributions. In addition, we also incorporate expert knowledge to guide the induction process by using *relevance feedback* (Baeza-Yates and Ribeiro-Neto, 1999), the most popular query reformulation strategy in the information retrieval (IR) community. The induction process is terminated until the termination criteria are satisfied.

In the remainder of this paper, Section 2 presents the overall framework for variable-length semantic pattern induction. Section 3 describes the process of constructing the HAL space. Section 4 details the cascaded induction process. Section 5 summarizes the experiment results. Finally, Section 6 draws some conclusions and suggests directions for future work.

2 Framework for Variable-Length Semantic Pattern Induction

The overall framework, as illustrated in Figure 1, is divided into two parts: the HAL model and the cascaded induction process. First of all, the HAL space is constructed for the psychiatry web corpora after word segmentation. Then, each word in HAL space is evaluated by computing its distance to a given seed pattern. A smaller distance represents that the word is more

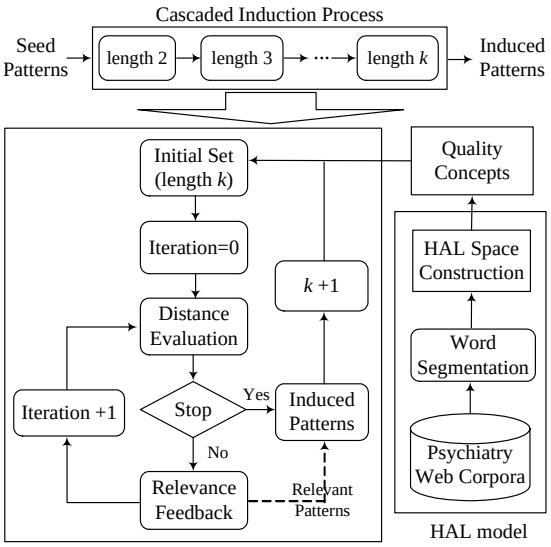


Figure 1. Framework for variable-length semantic pattern induction.

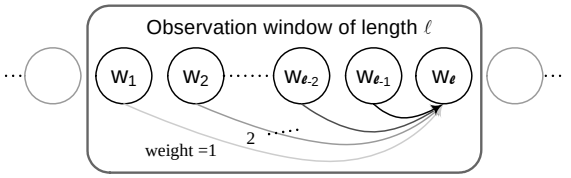


Figure 2. Weighting scheme of the HAL model.

	two	years	ago	I	lost	my	parents
two	0	0	0	0	0	0	0
years	5	0	0	0	0	0	0
ago	4	5	0	0	0	0	0
I	3	4	5	0	0	0	0
lost	2	3	4	5	0	0	0
my	1	2	3	4	5	0	0
parents	0	1	2	3	4	5	0

Table 1. Example of HAL Space (window size=5)

semantically related to the seed pattern. According to the distance measure, the CIP generates *quality concepts*, i.e., a set of semantically related words to the seed pattern. The quality concepts and the better semantic patterns induced in the previous stage are combined to generate the initial set for each length. For example, in the beginning stage, i.e., length two, the initial set is the all possible combinations of two quality concepts. In the later stages, each initial set is generated by adding a quality concept to each of the better semantic patterns. After the initial set for a particular length is created, each semantic pattern and the seed pattern are represented in the HAL space for further computing their distance. The more similar the context distributions between two patterns, the closer they are. Once all the semantic patterns are evaluated, the *relevance feedback* is applied to provide a set of relevant patterns judged by the health professionals. According to the relevant information, the seed pattern can be refined to be more similar to the relevant set. The refined seed pattern will be taken as the reference basis in the next iteration. The induction process for each stage is performed iteratively until no more patterns are judged as relevant or a maximum number of iteration is reached. The relevant set produced at the last iteration is considered as the result of the semantic patterns.

3 HAL Space Construction

The HAL model represents each word in the vocabulary using a vector representation. Each dimension of the vector is a weight representing the strength of association between the target word and its context word. The weights are computed by applying an observation window of length l over the corpus. All words within the window are considered as co-occurring with each other. Thus, for any two words of distance d within the window, the weight between them is computed as $l-d+1$. Figure 2 shows an example. The HAL space views the corpus as a sequence of words. Thus, after moving the window by one word increment over the whole corpus, the HAL space is constructed. The resultant HAL space is an $N \times N$ matrix, where N is the vocabulary size. In addition, each word in the HAL space is called a concept. Table 1 presents the HAL space for the example text "*Two years ago, I lost my parents*."

3.1 Representation of a Single Concept

For each concept in Table 1, the corresponding row vector represents its left context information, i.e., the weights of the words preceding it. Similarly, the corresponding column vector represents its right context information. Accordingly, each concept can be represented by a pair of vectors. That is,

$$c_i = (v_{c_i}^{left}, v_{c_i}^{right}) = \left(\left\langle w_{c_i t_1}^{left}, w_{c_i t_2}^{left}, \ldots, w_{c_i t_N}^{left} \right\rangle, \left\langle w_{c_i t_1}^{right}, w_{c_i t_2}^{right}, \ldots, w_{c_i t_N}^{right} \right\rangle \right), \quad (1)$$

where $v_{c_i}^{left}$ and $v_{c_i}^{right}$ represent the vectors of the left context information and right context information of a concept c_i, respectively, $w_{c_i t_j}$ denotes

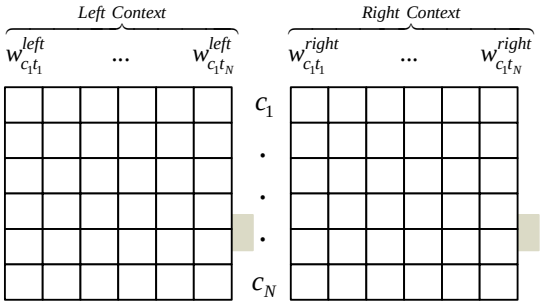


Figure 3. Conceptual representation of the HAL space.

the weight of the *j*-th dimension (t_j) of a vector, and *N* is the dimensionality of a vector, i.e., vocabulary size. The conceptual representation is depicted in Figure 3.

The weighting scheme of the HAL model is frequency-based. For some extremely infrequent words, we consider them as noises and remove them from the vocabulary. On the other hand, a high frequent word tends to get a higher weight, but this does not mean the word is informative, because it may also appear in many other vectors. Thus, to measure the informativeness of a word, the number of the vectors the word appears in should be taken into account. In principle, the more vectors the word appears in, the less information it carries to discriminate the vectors. Here we use a weighting scheme analogous to TF-IDF (Baeza-Yates and Ribeiro-Neto, 1999) to re-weight the dimensions of each vector, as described in Equation (2).

$$w_{c_it_j} = w_{c_it_j} * \log \frac{N_{vector}}{vf(t_j)}, \tag{2}$$

where N_{vector} denotes the total number of vectors, and $vf(t_j)$ denotes the number of vectors with t_j as the dimension. After each dimension is re-weighted, the HAL space is transformed into a probabilistic framework. Accordingly, each weight can be redefined as

$$w_{c_it_j} \equiv P(t_j \mid c_i) = \frac{w_{c_it_j}}{\sum_j w_{c_it_j}}, \tag{3}$$

where $P(t_j \mid c_i)$ is the probability that t_j appears in the vector of c_i.

3.2 Concept Combination

A semantic pattern is constituted by a set of concepts, thus it can be represented through concept combination over the HAL space. This forms a new concept in the HAL space. Let $sp = (c_1,...,c_S)$ be a semantic pattern with *S* constituent concepts, i.e., length *S*. The concept combination is defined as

$$\oplus c_s \equiv ((...(c_1 \oplus c_2) \oplus c_3) \oplus ... \oplus c_S), \tag{4}$$

where $\oplus$ denotes the symbol representing the combination operator over the HAL space, $\oplus c_s$ denotes a new concept generated by the concept combination. The new concept is the representation of a semantic pattern, also a vector representation. That is,

$$\begin{aligned} \oplus c_s &= (v^{left}_{\oplus c_s}, v^{right}_{\oplus c_s}) \\ &= \left(\left\langle w^{left}_{(\oplus c_s)t_1}, \ldots, w^{left}_{(\oplus c_s)t_N} \right\rangle, \left\langle w^{right}_{(\oplus c_s)t_1}, \ldots, w^{right}_{(\oplus c_s)t_N} \right\rangle\right), \end{aligned} \tag{5}$$

The combination operator, $\oplus$, is implemented by the product of the weights of the constituent concepts, described as follows.

$$\begin{aligned} w_{(\oplus c_s)t_j} &= \prod_{s=1}^{S} w_{c_st_j} \\ &= \prod_{s=1}^{S} P(t_j \mid c_s), \end{aligned} \tag{6}$$

where $w_{(\oplus c_s)t_j}$ denotes the weight of the *j*-th dimension of the new concept $\oplus c_s$.

4 Cascaded Induction Process

Given a seed pattern, the CIP is to induce a set of relevant semantic patterns with variable lengths (from 2 to *k*). Let $sp_{seed} = (c_1,...,c_R)$ be a seed pattern of length *R*, and $sp = (c_1,...,c_S)$ be a semantic pattern of length *S*. The formal description of the CIP is presented as

$$\begin{aligned} & sp_{seed} \mid\!- \{sp\} \\ & \equiv (c_1,...,c_R) \mid\!- \{(c_1,...,c_S)\} \quad \text{iff } \forall\, Dist(\oplus c_r, \oplus c_s) \le \lambda, \end{aligned} \tag{7}$$

where $\mid\!-$ denotes the symbol representing the cascaded induction, $\oplus c_r$ and $\oplus c_s$ are the two new concepts representing sp_{seed} and sp, respectively, and $Dist(\bullet, \bullet)$ represents the distance between two semantic patterns. The main steps in the CIP include the *initial set generation*, *distance measure*, and *relevance feedback*.

4.1 Initial Set Generation

The initial set for a particular length contains a set of semantic patterns to be induced, i.e., the search space. Reducing the search space would be helpful for speeding up the induction process,

especially for inducing those patterns with a larger length. For this purpose, we consider that the words and the semantic patterns similar to a given seed pattern are the better candidates for creating the initial sets. Therefore, we generate quality concepts, a set of semantically related words to a seed pattern, as the basis to create the initial set for each length. Thus, each seed pattern will be associated with a set of quality concepts. In addition, the better semantic patterns induced in the previous stage are also considered. The goodness of words and semantic patterns is measured by their distance to a seed pattern. Here, a word is considered as a quality concept if its distance is smaller than the average distance of the vocabulary. Similarly, only the semantic patterns with a distance smaller than the average distance of all semantic patterns in the previous stage are preserved to the next stage. By the way, the semantically unrelated patterns, possibly noisy patterns, will not be propagated to the next stage, and the search space can also be reduced. The principles of creating the initial sets of semantic patterns are summarized as follows.

- In the beginning stage, the aim is to create the initial set for the semantic patterns with length two. Thus, the initial set is the all possible combinations of two quality concepts.
- In the latter stages, each initial set is created by adding a quality concept to each of the better semantic patterns induced in the previous stage.

4.2 Distance Measure

The distance measure is to measure the distance between the seed patterns and semantic patterns to be induced. Let $sp = (c_1, ..., c_S)$ be a semantic pattern and $sp_{seed} = (c_1, ..., c_R)$ be a given seed pattern, their distance is defined as

$$Dist(sp, sp_{seed}) = Dist(\oplus c_s, \oplus c_r), \quad (8)$$

where $Dist(\oplus c_s, \oplus c_r)$ denotes the distance between two semantic patterns in the HAL space. As mentioned earlier, after concept combination, a semantic pattern becomes a new concept in the HAL space, which means the semantic pattern can be represented by its left and right contexts. Thus, the distance between two semantic patterns can be computed through their context distance. Equation (8) thereby can be written as

$$Dist(sp, sp_{seed}) = Dist(v^{left}_{\oplus c_s}, v^{left}_{\oplus c_r}) + Dist(v^{Right}_{\oplus c_s}, v^{Right}_{\oplus c_r}). \quad (9)$$

Because the weights of the vectors are represented using a probabilistic framework, each vector of a concept can be considered as a probabilistic distribution of the context words. Accordingly, we use the *Kullback-Liebler (KL) distance* (Manning and Schütze, 1999) to compute the distance between two probabilistic distributions, as shown in the following.

$$D(v_{\oplus c_s} \| v_{\oplus c_r}) = \sum_{j=1}^{N} P(t_j | \oplus c_s) \log \frac{P(t_j | \oplus c_s)}{P(t_j | \oplus c_r)}, \quad (10)$$

where $D(\bullet \| \bullet)$ denotes the KL distance between two probabilistic distributions. When Equation (10) is ill-conditioned, i.e., zero denominator, the denominator will be set to a small value (10^{-6}). For the consideration of a symmetric distance, we use the divergence measure, shown as follows.

$$Div(v_{\oplus c_s}, v_{\oplus c_r}) = D(v_{\oplus c_s} \| v_{\oplus c_r}) + D(v_{\oplus c_r} \| v_{\oplus c_s}). \quad (11)$$

By this way, the distance between two probabilistic distributions can be computed by their KL divergence. Thus, Equation (9) becomes

$$Dist(v_{\oplus c_s}, v_{\oplus c_r}) = Div(v^{left}_{\oplus c_s}, v^{left}_{\oplus c_r}) + Div(v^{Right}_{\oplus c_s}, v^{Right}_{\oplus c_r}). \quad (12)$$

After each semantic pattern is evaluated, a ranked list is produced for relevance judgment.

4.3 Relevance Feedback

In the induction process, some non-relevant semantic patterns may have smaller distance to a seed pattern, which may decrease the precision of the final results. To overcome the problem, one possible solution is to incorporate expert knowledge to guide the induction process. For this purpose, we use the technique of relevance feedback. In the IR community, the relevance feedback is to enhance the original query from the users by indicating which retrieved documents are relevant. For our task, the relevance feedback is applied after each semantic pattern is evaluated. Then, the health professionals judge which semantic patterns are relevant to the seed pattern. In practice, only the top *n* semantic patterns are presented for relevance judgment. Finally, the semantic patterns judged as relevant are considered to form the relevant set, and the others form the non-relevant set. According to the relevant and non-relevant information, the seed pattern can be refined to be more similar to the relevant set, such that the induction process can induce more relevant patterns and move away from noisy patterns in the future iterations.

The refinement of the seed pattern is to adjust its context distributions (left and right). Such adjustment is based on re-weighting the dimensions of the context vectors of the seed pattern. The dimensions more frequently regarded as relevant patterns are more significant for identifying relevant patterns. Hence, such dimensions of the seed pattern should be emphasized. The significance of a dimension is measured as follows.

$$Sig(t_k) = \frac{\sum_{\oplus c_i \in R} w_{(\oplus c_i)t_k}}{\sum_{\oplus c_j \in \overline{R}} w_{(\oplus c_j)t_k}}, \quad (13)$$

where $Sig(t_k)$ denotes the significance of the dimension t_k, $\oplus c_i$ and $\oplus c_j$ denote the semantic patterns of the relevant set and non-relevant set, respectively, and $w_{(\oplus c_i)t_k}$ and $w_{(\oplus c_j)t_k}$ denote the weights of t_k of $\oplus c_i$ and $\oplus c_j$, respectively. The higher the ratio, the more significant the dimension is. In order to smooth $Sig(t_k)$ to the range from zero to one, the following formula is used:

$$Sig(t_k) = \frac{1}{1 + \left(\sum_{\oplus c_i \in R} w_{(\oplus c_i)t_k} \Big/ \sum_{\oplus c_j \in \overline{R}} w_{(\oplus c_j)t_k} \right)^{-1}}. \quad (14)$$

The corresponding dimension of the seed pattern $sp_{seed} = \oplus c_r$ is then re-weighted by

$$w_{(\oplus c_r)t_k} = w_{(\oplus c_r)t_k} + Sig(t_k). \quad (15)$$

Once the context vectors of the seed pattern are re-weighted, they are also transformed into a probabilistic form using Equation (3). The refined seed pattern will be taken as the reference basis in the next iteration. The relevance feedback is performed iteratively until no more semantic patterns are judged as relevant or a maximum number of iteration is reached. At the same time, the induction process for a particular length is also stopped. The whole CIP process is stopped until the seed patterns are exhausted

5 Experimental Results

To evaluate the performance of the CIP, we built a prototype system and provided a set of seed patterns. The seed patterns were collected by referring to the well-defined instruments for assessing negative life events (Brostedt and Pedersen, 2003; Pagano et al., 2004). A total of 20 seed patterns were selected by the health professionals. Then, the CIP randomly selects one seed pattern per run without replacement from the seed set, and iteratively induces relevant patterns from the psychiatry web corpora. The psychiatry web corpora used here include some professional mental health web sites, such as PsychPark (http://www.psychpark.org) (Bai, 2001) and John Tung Foundation (http://www.jtf.org.tw).

In the following sections, we describe some experiments to in turn examine the effect of using relevance feedback or not, and the coverage on real data using the semantic patterns induced by different approaches. Because the semantic patterns with a length larger than 4 are very rare to express a negative life event, we limit the length *k* to the range of 2 to 4.

5.1 Evaluation on Relevance Feedback

The relevance feedback employed in this study provides the relevant and non-relevant information for the CIP so that it can refine the seed pattern to induce more relevant patterns. The relevance judgment is carried out by three experienced psychiatric physicians. For practical consideration, only the top 30 semantic patterns are presented to the physicians. During relevance judgment, a majority vote mechanism is used to handle the disagreements among the physicians. That is, a semantic pattern is considered as relevant if any two or more physicians judged it as relevant. Finally, the semantic patterns with majority votes are obtained to form the relevant set.

To evaluate the effectiveness of the relevance feedback, we construct three variants of the CIP, *RF(5)*, *RF(10)*, and *RF(20)*, implemented by applying the relevance feedback for 5, 10, and 20 iterations, respectively. These three CIP variants are then compared to the one without using the relevance feedback, denoted as *RF(—)*. We use the evaluation metric, *precision at 30 (prec@30)*, over all seed patterns to examine if the relevance feedback can help the CIP induce more relevant patterns. For a particular seed pattern, prec@n is computed as the number of relevant semantic patterns ranked in the top *n* of the ranked list, divided by *n*. Table 2 presents the results for *k*=2.

The results reveal that the relevance feedback can help the CIP induce more relevant semantic patterns. Another observation indicates that applying the relevance feedback for more iterations

	RF(—)	*RF(5)*	*RF(10)*	*RF(20)*
prec@30	0.203	0.263	0.318	0.387

Table 2. Effect of applying relevance feedback for different number of iterations or not.

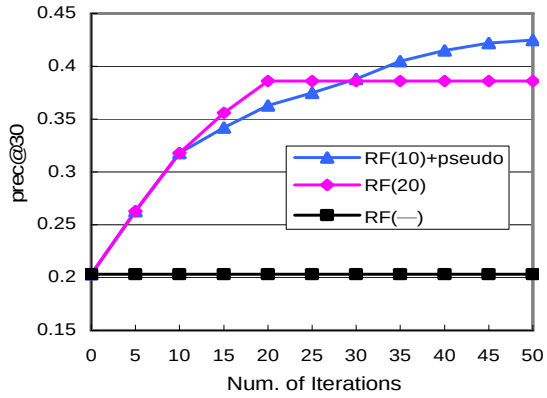


Figure 4. Effect of using the combination of relevance feedback and pseudo-relevance feedback.

can further improve the precision. However, it is usually impractical for experts to involve in the guiding process for too many iterations. Consequently, we further consider *pseudo-relevance feedback* to automate the guiding process. The pseudo-relevance feedback carries out the relevance judgment based on the assumption that the top ranked semantic patterns are more likely to be the relevant ones. Thus, this approach usually relies on setting a threshold or selecting only the top *n* semantic patterns to form the relevant set. However, determining the threshold is not trivial, and the threshold may be different with different seed patterns. Therefore, we apply the pseudo-relevance feedback only after certain expert-guided iterations, rather than applying it throughout the induction process. The notion is that we can get a more reliable threshold value by observing the behavior of the relevant semantic patterns in the ranked list for a few iterations.

To further examine the effectiveness of the combined approach, we additionally construct a CIP variant, *RF(10)+pseudo*, by applying the pseudo-relevance feedback after 10 expert-guided iterations. The threshold is determined by the physicians during their judgments in the 10-th iteration. The results are presented in Figure 4.

The precision of *RF(10)+pseudo* is inferior to that of *RF(20)* before the 25-th iteration. Meanwhile, after the 30-th iteration, *RF(10)+pseudo* achieves higher precision than the other methods. This indicates that the pseudo-relevance feedback can also contribute to semantic pattern induction in the stage without expert intervention.

5.2 Coverage on Real Data

The final results of the semantic patterns are the relevant sets of the last iteration produced by *RF(10)+pseudo*, denoted as SP_{CIP}. Parts of them are shown in Table 3.

Seed Pattern	< boyfriend, argue >
Induced Patterns	<girlfriend, break up>; <friend, fight>

Table 3. Parts of induced semantic patterns.

We compare SP_{CIP} to those created by a corpus-based approach. The corpus-based approach relies on an annotated domain corpus and a learning mechanism to induce the semantic patterns. Thus, we collected 300 consultation records from the PsychPark as the domain corpus, and each sentence in the corpus is annotated with a negative life event or not by the three physicians. After the annotation process, the sentences with negative life events are together to form the training set. Then, we adopt *Mutual Information* (Manning and Schütze, 1999) to learn variable-length semantic patterns. The mutual information between *k* words is defined as

$$MI(w_1,...,w_k) = P(w_1,...,w_k)\log\frac{P(w_1,...,w_k)}{\prod_{i=1}^{k} P(w_i)} \quad (16)$$

where $P(w_1,...w_k)$ is the probability of the *k* words co-occurring in a sentence in the training set, and $P(w_i)$ is the probability of a single word occurring in the training set. Higher mutual information indicates that the *k* words are more likely to form a semantic pattern of length *k*. Here the length *k* also ranges from 2 to 4. For each *k*, we compute the mutual information for all possible combinations of words in the training set, and those with their mutual information above a threshold are selected to be the final results of the semantic patterns, denoted as SP_{MI}. In order to obtain reliable mutual information values, only words with at least the minimum number of occurrences (>5) are considered.

To examine the coverage of SP_{CIP} and SP_{MI} on real data, 15 human subjects are involved in creating a test set. The subjects provide their experienced negative life events in the form of natural language sentences. A total of 69 sentences are collected to be the test set, of which 39 sentences contain a semantic pattern of length two, 21 sentences contain a semantic pattern of length three, and 9 sentences contain a semantic pattern of length four. The evaluation metric used is *out-of-pattern (OOP)* rate, a ratio of unseen patterns occurring in the test set. Thus, the OOP can be defined as the number of test sentences containing the semantic patterns not occurring in the training set, divided by the total number of sentences in the test set. Table 4 presents the results.

	k=2	k=3	k=4
SP_{CIP}	0.36 (14/39)	0.48 (10/21)	0.44 (4/9)
SP_{MI}	0.51 (20/39)	0.62 (13/21)	0.67 (6/9)

Table 4. OOP rate of the CIP and a corpus-based approach.

The results show that the OOP of SP_{MI} is higher than that of SP_{CIP}. The main reason is the lack of a large enough domain corpus with annotated life events. In this circumstance, many semantic patterns, especially for those with a larger length, could not be learned, because the number of their occurrences would be very rare in the training set. With no doubt, one could collect a large amount of domain corpus to reduce the OOP rate. However, increasing the amount of domain corpus also increases the amount of annotation and computation complexity. Our approach, instead, exploits the quality concepts to reduce the search space, also applies the relevance feedback to guide the induction process, thus it can achieve better results with time-limited constraints.

6 Conclusion

This study has presented an HAL-based cascaded model for variable-length semantic pattern induction. The HAL model provides an informative infrastructure for the CIP to induce semantic patterns from the unannotated psychiatry web corpora. Using the quality concepts and preserving the better results from the previous stage, the search space can be reduced to speed up the induction process. In addition, combining the relevance feedback and pseudo-relevance feedback, the induction process can be guided to induce more relevant semantic patterns. The experimental results demonstrated that our approach can not only reduce the reliance on annotated corpora but also obtain acceptable results with time-limited constraints. Future work will be devoted to investigating the detection of negative life events using the induced patterns so as to make the psychiatric services more effective.

References

R. Baeza-Yates and B. Ribeiro-Neto. 1999. *Modern Information Retrieval*. Addison-Wesley, Reading, MA.

Y. M. Bai, C. C. Lin, J. Y. Chen, and W. C. Liu. 2001. Virtual Psychiatric Clinics. *American Journal of Psychiatry*, 158(7):1160-1161.

J. Bai, D. Song, P. Bruza, J. Y. Nie, and G. Cao. 2005. Query Expansion Using Term Relationships in Language Models for Information Retrieval. In *Proc. of the 14th ACM International Conference on Information and Knowledge Management*, pages 688-695.

E. M. Brostedt and N. L. Pedersen. 2003. Stressful Life Events and Affective Illness. *Acta Psychiatrica Scandinavica*, 107:208-215.

C. Burgess, K. Livesay, and K. Lund. 1998. Explorations in Context Space: Words, Sentences, Discourse. *Discourse Processes*. 25(2&3):211-257.

C. Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press.

T. Grenager, D. Klein, and C. D. Manning. 2005. Unsupervised Learning of Field Segmentation Models for Information Extraction. In *Proc. of the 43th Annual Meeting of the ACL*, pages 371-378.

T. Hasegawa, S. Sekine, R. Grishman. 2004. Discovering Relations among Named Entities from Large Corpora. In *Proc. of the 42th Annual Meeting of the ACL*, pages 415-422.

W.Lehnert, C. Cardie, D. Fisher, J. McCarthy, E. Riloff, and S. Soderland. 1992. University of Massachusetts: Description of the CIRCUS System used for MUC-4. In *Proc. of the Fourth Message Understanding Conference*, pages 282-288.

C. Manning and H. Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press. Cambridge, MA.

I. Muslea. 1999. Extraction Patterns for Information Extraction Tasks: A Survey. In *Proc. of the AAAI-99 Workshop on Machine Learning for Information Extraction*, pages 1-6.

M. E. Pagano, A. E. Skodol, R. L. Stout, M. T. Shea, S. Yen, C. M. Grilo, C.A. Sanislow, D. S. Bender, T. H. McGlashan, M. C. Zanarini, and J. G. Gunderson. 2004. Stressful Life Events as Predictors of Functioning: Findings from the Collaborative Longitudinal Personality Disorders Study. *Acta Psychiatrica Scandinavica*, 110:421-429.

M. Stevenson and M. A. Greenwood. 2005. A Semantic Approach to IE Pattern Induction. In *Proc. of the 43th Annual Meeting of the ACL*, pages 379-386.

C. H. Wu, L. C. Yu, and F. L. Jang. 2005. Using Semantic Dependencies to Mine Depressive Symptoms from Consultation Records. *IEEE Intelligent System*, 20(6):50-58.

J. F. Yeh, C. H. Wu, M. J. Chen, and L. C. Yu. 2004. Automated Alignment and Extraction of Bilingual Domain Ontology for Cross-Language Domain-Specific Applications. In *Proc. of the 20th COLING*, pages 1140-1146.

Inducing Word Alignments with Bilexical Synchronous Trees

Hao Zhang and **Daniel Gildea**
Computer Science Department
University of Rochester
Rochester, NY 14627

Abstract

This paper compares different bilexical tree-based models for bilingual alignment. EM training for the new model benefits from the dynamic programming "hook trick". The model produces improved dependency structure for both languages.

1 Introduction

A major difficulty in statistical machine translation is the trade-off between representational power and computational complexity. Real-world corpora for language pairs such as Chinese-English have complex reordering relationships that are not captured by current phrase-based MT systems, despite their state-of-the-art performance measured in competitive evaluations. Synchronous grammar formalisms that are capable of modeling such complex relationships while maintaining the context-free property in each language have been proposed for many years, (Aho and Ullman, 1972; Wu, 1997; Yamada and Knight, 2001; Melamed, 2003; Chiang, 2005), but have not been scaled to large corpora and long sentences until recently.

In Synchronous Context Free Grammars, there are two sources of complexity, grammar branching factor and lexicalization. In this paper we focus on the second issue, constraining the grammar to the binary-branching Inversion Transduction Grammar of Wu (1997). Lexicalization seems likely to help models predict alignment patterns between languages, and has been proposed by Melamed (2003) and implemented by Alshawi et al. (2000) and Zhang and Gildea (2005). However, each piece of lexical information considered by a model multiplies the number of states of dynamic programming algorithms for inference, meaning that we must choose how to lexicalize very carefully to control complexity.

In this paper we compare two approaches to lexicalization, both of which incorporate bilexical probabilities. One model uses bilexical probabilities across languages, while the other uses bilexical probabilities within one language. We compare results on word-level alignment, and investigate the implications of the choice of lexicalization on the specifics of our alignment algorithms. The new model, which bilexicalizes within languages, allows us to use the "hook trick" (Eisner and Satta, 1999) and therefore reduces complexity. We describe the application of the hook trick to estimation with Expectation Maximization (EM). Despite the theoretical benefits of the hook trick, it is not widely used in statistical monolingual parsers, because the savings do not exceed those obtained with simple pruning. We speculate that the advantages may be greater in an EM setting, where parameters to guide pruning are not (initially) available.

In order to better understand the model, we analyze its performance in terms of both agreement with human-annotated alignments, and agreement with the dependencies produced by monolingual parsers. We find that within-language bilexicalization does not improve alignment over cross-language bilexicalization, but does improve recovery of dependencies. We find that the hook trick significantly speeds training, even in the presence of pruning.

Section 2 describes the generative model. The hook trick for EM is explained in Section 3. In Section 4, we evaluate the model in terms of alignment error rate and dependency error rate. We conclude with discussions in Section 5.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 953–960,
Sydney, July 2006.

2 Bilexicalization of Inversion Transduction Grammar

The Inversion Transduction Grammar of Wu (1997) models word alignment between a translation pair of sentences by assuming a binary synchronous tree on top of both sides. Using EM training, ITG can induce good alignments through exploring the hidden synchronous trees from instances of string pairs.

ITG consists of unary production rules that generate English/foreign word pairs e/f:

$$X \rightarrow e/f$$

and binary production rules in two forms that generate subtree pairs, written:

$$X \rightarrow [Y\ Z]$$

and

$$X \rightarrow \langle Y\ Z \rangle$$

The square brackets indicate the right hand side rewriting order is the same for both languages. The pointed brackets indicate there exists a type of syntactic reordering such that the two right hand side constituents rewrite in the opposite order in the second language.

The unary rules account for the alignment links across two sides. Either e or f may be a special null word, handling insertions and deletions. The two kinds of binary rules (called straight rules and inverted rules) build up a coherent tree structure on top of the alignment links. From a modeling perspective, the synchronous tree that may involve inversions tells a generative story behind the word level alignment.

An example ITG tree for the sentence pair *Je les vois / I see them* is shown in Figure 1(left). The probability of the tree is the product rule probabilities at each node:

$$\begin{aligned} P(S \rightarrow A) & \\ & \cdot P(A \rightarrow [C\ B]) \\ & \cdot P(C \rightarrow I/Je) \\ & \cdot P(B \rightarrow \langle C\ C \rangle) \\ & \cdot P(C \rightarrow see/vois) \\ & \cdot P(C \rightarrow them/les) \end{aligned}$$

The structural constraint of ITG, which is that only binary permutations are allowed on each level, has been demonstrated to be reasonable by Zens and Ney (2003) and Zhang and Gildea (2004). However, in the space of ITG-constrained synchronous trees, we still have choices in making the probabilistic distribution over the trees more realistic. The original Stochastic ITG is the counterpart of Stochastic CFG in the bitext space. The probability of an ITG parse tree is simply a product of the probabilities of the applied rules. Thus, it only captures the fundamental features of word links and reflects how often inversions occur.

2.1 Cross-Language Bilexicalization

Zhang and Gildea (2005) described a model in which the nonterminals are lexicalized by English and foreign language word pairs so that the inversions are dependent on lexical information on the left hand side of synchronous rules. By introducing the mechanism of probabilistic head selection there are four forms of probabilistic binary rules in the model, which are the four possibilities created by taking the cross-product of two orientations (straight and inverted) and two head choices:

$$\begin{aligned} X(e/f) &\rightarrow [Y(e/f)\ Z] \\ X(e/f) &\rightarrow [Y\ Z(e/f)] \\ X(e/f) &\rightarrow \langle Y(e/f)\ Z \rangle \\ X(e/f) &\rightarrow \langle Y\ Z(e/f) \rangle \end{aligned}$$

where (e/f) is a translation pair.

A tree for our example sentence under this model is shown in Figure 1(center). The tree's probability is again the product of rule probabilities:

$$\begin{aligned} P(S \rightarrow A(see/vois)) & \\ & \cdot P(A(see/vois) \rightarrow [C\ B(see/vois)]) \\ & \cdot P(C \rightarrow C(I/Je)) \\ & \cdot P(B(see/vois) \rightarrow \langle C(see/vois)\ C \rangle) \\ & \cdot P(C \rightarrow C(them/les)) \end{aligned}$$

2.2 Head-Modifier Bilexicalization

One disadvantage of the model above is that it is not capable of modeling bilexical dependencies on the right hand side of the rules. Thus, while the probability of a production being straight or inverted depends on a bilingual word pair, it does not take head-modifier relations in either language into account. However, modeling complete bilingual bilexical dependencies as theorized in Melamed (2003) implies a huge parameter space of $O(|V|^2|T|^2)$, where $|V|$ and $|T|$ are the vocabulary sizes of the two languages. So, instead of modeling cross-language word translations and within-language word dependencies in

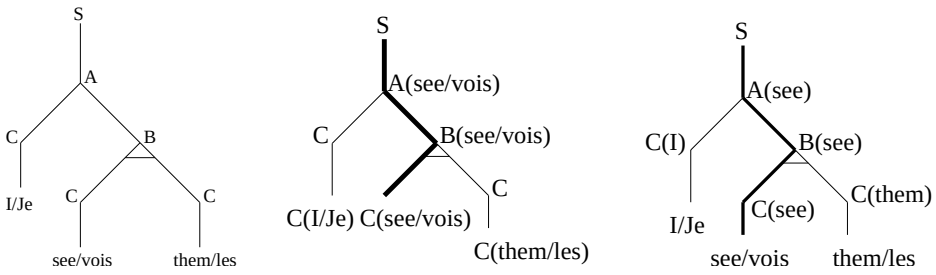


Figure 1: Parses for an example sentence pair under unlexicalized ITG (left), cross-language bilexicalization (center), and head-modifier bilexicaliztion (right). Thick lines indicate head child; crossbar indicates inverted production.

a joint fashion, we factor them apart. We lexicalize the dependencies in the synchronous tree using words from only one language and translate the words into their counterparts in the other language only at the bottom of the tree. Formally, we have the following patterns of binary dependency rules:

$$X(\mathbf{e}) \rightarrow [Y(\mathbf{e})\ Z(e')]$$

$$X(\mathbf{e}) \rightarrow [Y(e')\ Z(\mathbf{e})]$$

$$X(\mathbf{e}) \rightarrow \langle Y(\mathbf{e})\ Z(e')\rangle$$

$$X(\mathbf{e}) \rightarrow \langle Y(e')\ Z(\mathbf{e})\rangle$$

where $\mathbf{e}$ is an English head and e' is an English modifier.

Equally importantly, we have the unary lexical rules that generate foreign words:

$$X(e) \rightarrow e/f$$

To make the generative story complete, we also have a top rule that goes from the unlexicalized start symbol to the highest lexicalized nonterminal in the tree:

$$S \rightarrow X(e)$$

Figure 1(right), shows our example sentence's tree under the new model. The probability of a bilexical synchronous tree between the two sentences is:

$$\begin{aligned} P(S \rightarrow A(see)) \\ &\cdot P(A(\mathbf{see}) \rightarrow [C(I)\, B(\mathbf{see})]) \\ &\cdot P(C(I) \rightarrow I/Je) \\ &\cdot P(B(\mathbf{see}) \rightarrow \langle C(\mathbf{see})\, C(them)\rangle) \\ &\cdot P(C(see) \rightarrow see/vois) \\ &\cdot P(C(them) \rightarrow them/les) \end{aligned}$$

Interestingly, the lexicalized $B(\mathbf{see})$ predicts not only the existence of $C(them)$, but also that there is an inversion involved going from $C(\mathbf{see})$ to $C(them)$. This reflects the fact that direct object pronouns come after the verb in English, but before the verb in French. Thus, despite conditioning on information about words from only one language, the model captures syntactic reordering information about the specific language pair it is trained on. We are able to discriminate between the straight and inverted binary nodes in our example tree in a way that cross-language bilexicalization could not.

In terms of inferencing within the framework, we do the usual Viterbi inference to find the best bilexical synchronous tree and treat the dependencies and the alignment given by the Viterbi parse as the best ones, though mathematically the best alignment should have the highest probability marginalized over all dependencies constrained by the alignment. We do unsupervised training to obtain the parameters using EM. Both EM and Viterbi inference can be done using the dynamic programming framework of synchronous parsing.

3 Inside-Outside Parsing with the Hook Trick

ITG parsing algorithm is a CYK-style chart parsing algorithm extended to bitext. Instead of building up constituents over spans on a string, an ITG chart parser builds up constituents over subcells within a cell defined by two strings. We use $\beta(X(e), s, t, u, v)$ to denote the inside probability of $X(e)$ which is over the cell of (s, t, u, v) where (s, t) are indices into the source language string and (u, v) are indices into the target language string. We use $\alpha(X(e), s, t, u, v)$ to denote its outside probability. Figure 2 shows how smaller cells adjacent along diagonals can be combined to create a large cell. We number the subcells counterclockwise. To analyze the complexity of the algorithm with respect to input string

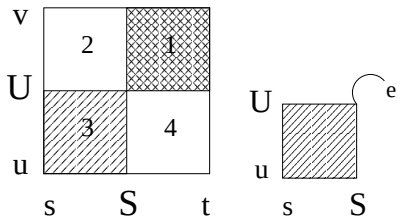


Figure 2: Left: Chart parsing over the bitext cell of (s, t, u, v). Right: One of the four hooks built for four corners for more efficient parsing.

length, without loss of generality, we ignore the nonterminal symbols X, Y, and Z to simplify the derivation.

The inside algorithm in the context of bilexical ITG is based on the following dynamic programming equation:

$$\begin{aligned}&\beta\,(e, s, t, u, v)\\ &= \sum_{S,U,e'} \begin{pmatrix} \beta_1(e) \cdot \beta_3(e') \cdot P([e'e] \mid e) \\ +\beta_2(e) \cdot \beta_4(e') \cdot P(\langle ee' \rangle \mid e) \\ +\beta_3(e) \cdot \beta_1(e') \cdot P([ee'] \mid e) \\ +\beta_4(e) \cdot \beta_2(e') \cdot P(\langle e'e \rangle \mid e) \end{pmatrix}\end{aligned}$$

So, on the right hand side, we sum up all possible ways (S, U) of splitting the left hand side cell and all possible head words (e') for the non-head subcell. e, e', s, t, u, v, S, and U all eight variables take $O(n)$ values given that the lengths of the source string and the target string are $O(n)$. Thus the entire DP algorithm takes $O(n^8)$ steps.

Fortunately, we can reduce the maximum number of interacting variables by factorizing the expression.

Let us keep the results of the summations over e' as:

$$\begin{aligned}\beta_1^+(e) &= \sum_{e'} \beta_1(e') \cdot P([ee'] \mid e)\\ \beta_2^+(e) &= \sum_{e'} \beta_2(e') \cdot P(\langle e'e \rangle \mid e)\\ \beta_3^+(e) &= \sum_{e'} \beta_3(e') \cdot P([e'e] \mid e)\\ \beta_4^+(e) &= \sum_{e'} \beta_4(e') \cdot P(\langle ee' \rangle \mid e)\end{aligned}$$

The computation of each β^+ involves four boundary indices and two head words. So, we can rely on DP to compute them in $O(n^6)$. Based on these intermediate results, we have the equivalent DP expression for computing inside probabilities:

$$\begin{aligned}&\beta\,(e, s, t, u, v)\\ &= \sum_{S,U} \begin{pmatrix} \beta_1(e) \cdot \beta_3^+(e) \\ +\beta_2(e) \cdot \beta_4^+(e) \\ +\beta_3(e) \cdot \beta_1^+(e) \\ +\beta_4(e) \cdot \beta_2^+(e) \end{pmatrix}\end{aligned}$$

We reduced one variable from the original expression. The maximum number of interacting variables throughout the algorithm is 7. So the improved inside algorithm has a time complexity of $O(n^7)$.

The trick of reducing interacting variables in DP for bilexical parsing has been pointed out by Eisner and Satta (1999). Melamed (2003) discussed the applicability of the so-called hook trick for parsing bilexical multitext grammars. The name hook is based on the observation that we combine the non-head constituent with the bilexical rule to create a special constituent that matches the head like a hook as demonstrated in Figure 2. However, for EM, it is not clear from their discussions how we can do the hook trick in the outside pass. The bilexical rules in all four directions are analogous. To simplify the derivation for the outside algorithm, we just focus on the first case: straight rule with right head word.

The outside probability of the constituent (e, S, t, U, v) in cell 1 being a head of such rules is:

$$\begin{aligned}&\sum_{s,u,e'} \left(\alpha(e) \cdot \beta_3(e') \cdot P([e'e] \mid e)\right)\\ &= \sum_{s,u} \left(\alpha(e) \cdot \left(\sum_{e'} \beta_3(e') \cdot P([e'e] \mid e)\right)\right)\\ &= \sum_{s,u} \Big(\alpha(e) \cdot \beta_3^+(e)\Big)\end{aligned}$$

which indicates we can reuse β^+ of the lower left neighbors of the head to make the computation feasible in $O(n^7)$.

On the other hand, the outside probability for (e', s, S, u, U) in cell 3 acting as a modifier of such

a rule is:

$$\sum_{t,v,e} \left(\alpha(e) \cdot \beta_1(e) \cdot P([e'e] \mid e)\right)$$
$$= \sum_{e} \left(P([e'e] \mid e) \cdot \left(\sum_{t,v} \alpha(e) \cdot \beta_1(e) \right) \right)$$
$$= \sum_{e} \Big(P([e', e] \mid e) \cdot \alpha_3^+(e) \Big)$$

in which we memorize another kind of intermediate sum to make the computation no more complex than $O(n^7)$.

We can think of α_3^+ as the outside probability of the hook on cell 3 which matches cell 1. Generally, we need outside probabilities for hooks in all four directions.

$$\alpha_1^+(e) = \sum_{s,u} \alpha(e) \cdot \beta_3(e)$$
$$\alpha_2^+(e) = \sum_{t,u} \alpha(e) \cdot \beta_4(e)$$
$$\alpha_3^+(e) = \sum_{t,v} \alpha(e) \cdot \beta_1(e)$$
$$\alpha_4^+(e) = \sum_{s,v} \alpha(e) \cdot \beta_2(e)$$

Based on them, we can add up the outside probabilities of a constituent acting as one of the two children of each applicable rule on top of it to get the total outside probability.

We finalize the derivation by simplifying the expression of the expected count of $(e \rightarrow [e'e])$.

$$EC(e \rightarrow [e'e])$$
$$= \sum_{s,t,u,v,S,U} \left(P([e'e] \mid e) \cdot \beta_3(e') \cdot \alpha(e) \cdot \beta_1(e)\right)$$
$$= \sum_{s,S,u,U} \left(P([e'e] \mid e) \cdot \beta_3(e') \cdot \left(\sum_{t,v} \alpha \cdot \beta_1 \right) \right)$$
$$= \sum_{s,S,u,U} \Big(P([e'e] \mid e) \cdot \beta_3(e') \cdot \alpha_3^+(e) \Big)$$

which can be computed in $O(n^6)$ as long as we have α_3^+ ready in a table. Overall we can do the inside-outside algorithm for the bilexical ITG in $O(n^7)$, by reducing a factor of n through intermediate DP.

The entire trick can be understood very clearly if we imagine the bilexical rules are unary rules that are applied on top of the non-head constituents to reduce it to a virtual lexical constituent (a hook) covering the same subcell while sharing the head word with the head constituent. However, if we build hooks looking for all words in a sentence whenever a complete constituent is added to the chart, we will build many hooks that are never used, considering that the words outside of larger cells are fewer and pruning might further reduce the possible outside words. Blind guessing of what might appear outside of the current cell will offset the saving we can achieve. Instead of actively building hooks, which are intermediate results, we can build them only when we need them and then cache them for future use. So the construction of the hooks will be invoked by the heads when the heads need to combine with adjacent cells.

3.1 Pruning and Smoothing

We apply one of the pruning techniques used in Zhang and Gildea (2005). The technique is general enough to be applicable to any parsing algorithm over bitext cells. It is called tic-tac-toe pruning since it involves an estimate of both the inside probability of the cell (how likely the words within the box in both dimensions are to align) and the outside probability (how likely the words outside the box in both dimensions are to align). By scoring the bitext cells and throwing away the bad cells that fall out of a beam, it can reduce over 70% of $O(n^4)$ cells using 10^{-5} as the beam ratio for sentences up to 25 words in the experiments, without harming alignment error rate, at least for the unlexicalized ITG.

The hook trick reduces the complexity of bilexical ITG from $O(n^8)$ to $O(n^7)$. With the tic-tac-toe pruning reducing the number of bitext cells to work with, also due to the reason that the grammar constant is very small for ITG. the parsing algorithm runs with an acceptable speed,

The probabilistic model has lots of parameters of word pairs. Namely, there are $O(|V|^2)$ dependency probabilities and $O(|V||T|)$ translation probabilities, where $|V|$ is the size of English vocabulary and $|T|$ is the size of the foreign language vocabulary. The translation probabilities of $P(f|X(e))$ are backed off to a uniform distribution. We let the bilexical dependency probabilities back off to uni-lexical dependencies in the following forms:

$$P([Y(*)\, Z(e')] \mid X(*))$$
$$P([Y(e')\, Z(*)] \mid X(*))$$
$$P(\langle Y(*)\, Z(e') \rangle \mid X(*))$$
$$P(\langle Y(e')\, Z(*) \rangle \mid X(*))$$

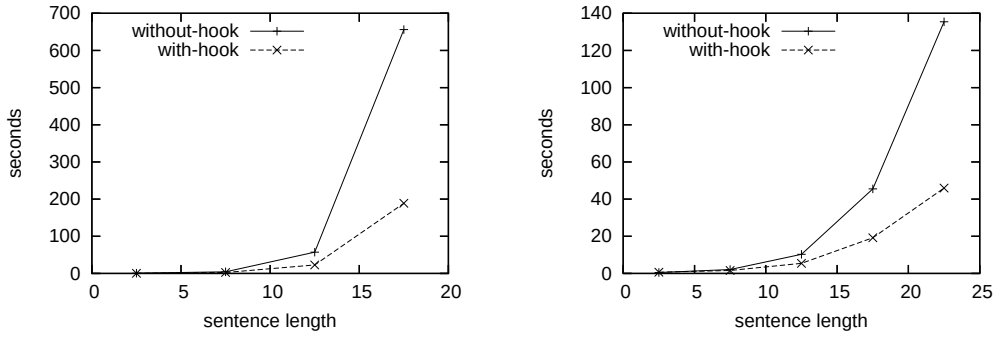


Figure 3: Speedup for EM by the Hook Trick. (a) is without pruning. In (b), we apply pruning on the bitext cells before parsing begins.

The two levels of distributions are interpolated using a technique inspired by Witten-Bell smoothing (Chen and Goodman, 1996). We use the expected count of the left hand side lexical nonterminal to adjust the weight for the EM-trained bilexical probability. For example,

$$P([Y(e)\ Z(e')] \mid X(e)) = (1-\lambda)P_{EM}([Y(e)\ Z(e')] \mid X(e)) + \lambda P([Y(*)\ Z(e')] \mid X(*))$$

where

$$\lambda = 1/(1 + Expected\_Counts(X(e)))$$

4 Experiments

First of all, we are interested in finding out how much speedup can be achieved by doing the hook trick for EM. We implemented both versions in C++ and turned off pruning for both. We ran the two inside-outside parsing algorithms on a small test set of 46 sentence pairs that are no longer than 25 words in both languages. Then we put the results into buckets of $(1-4)$, $(5-9)$, $(10-14)$, $(15-19)$, and $(20-24)$ according to the maximum length of two sentences in each pair and took averages of these timing results. Figure 3 (a) shows clearly that as the sentences get longer the hook trick is helping more and more. We also tried to turn on pruning for both, which is the normal condition for the parsers. Both are much faster due to the effectiveness of pruning. The speedup ratio is lower because the hooks will less often be used again since many cells are pruned away. Figure 3 (b) shows the speedup curve in this situation.

We trained both the unlexicalized and the lexicalized ITGs on a parallel corpus of Chinese-English newswire text. The Chinese data were automatically segmented into tokens, and English capitalization was retained. We replaced words occurring only once with an unknown word token, resulting in a Chinese vocabulary of 23,783 words and an English vocabulary of 27,075 words.

We did two types of comparisons. In the first comparison, we measured the performance of five word aligners, including IBM models, ITG, the lexical ITG (LITG) of Zhang and Gildea (2005), and our bilexical ITG (BLITG), on a hand-aligned bilingual corpus. All the models were trained using the same amount of data. We ran the experiments on sentences up to 25 words long in both languages. The resulting training corpus had 18,773 sentence pairs with a total of 276,113 Chinese words and 315,415 English words.

For scoring the Viterbi alignments of each system against gold-standard annotated alignments, we use the alignment error rate (AER) of Och and Ney (2000), which measures agreement at the level of pairs of words:

$$AER = 1 - \frac{|A \cap G_P| + |A \cap G_S|}{|A| + |G_S|}$$

where A is the set of word pairs aligned by the automatic system, G_S is the set marked in the gold standard as "sure", and G_P is the set marked as "possible" (including the "sure" pairs). In our Chinese-English data, only one type of alignment was marked, meaning that $G_P = G_S$.

In our hand-aligned data, 47 sentence pairs are no longer than 25 words in either language and were used to evaluate the aligners.

A separate development set of hand-aligned sentence pairs was used to control overfitting. The subset of up to 25 words in both languages was used. We chose the number of iterations for EM

	Precision	*Recall*	*Alignment Error Rate*
IBM-1	.56	.42	.52
IBM-4	.67	.43	.47
ITG	.68	.52	.41
LITG	.69	.51	.41
BLITG	.68	.51	.42

	Precision	*Recall*	*Dependency Error Rate*
ITG-lh	.11	.11	.89
ITG-rh	.22	.22	.78
LITG	.13	.12	.88
BLITG	.24	.22	.77

Table 1: Bilingual alignment and English dependency results on Chinese-English corpus ($\leq$ 25 words on both sides). LITG stands for the cross-language Lexicalized ITG. BLITG is the within-English Bilexical ITG. ITG-lh is ITG with left-head assumption on English. ITG-rh is with right-head assumption.

	Precision	*Recall*	*AER*
ITG	.59	.60	.41
LITG	.60	.57	.41
BLITG	.58	.55	.44

	Precision	*Recall*	*DER*
ITG-rh	.23	.23	.77
LITG	.11	.11	.89
BLITG	.24	.24	.76

Table 2: Alignment and dependency results on a larger Chinese-English corpus.

training as the turning point of AER on the development data set. The unlexicalized ITG was trained for 3 iterations. LITG was trained for only 1 iteration, partly because it was initialized with fully trained ITG parameters. BLITG was trained for 3 iterations.

For comparison, we also included the results from IBM Model 1 and Model 4. The numbers of iterations for the training of the IBM models were also chosen to be the turning points of AER changing on the development data set.

We also want to know whether or not BLITG can model dependencies better than LITG. For this purpose, we also used the AER measurement, since the goal is still getting higher precision/recall for a set of recovered word links, although the dependency word links are within one language. For this reason, we rename AER to Dependency Error Rate. Table 1(right) is the dependency results on English side of the test data set. The dependency results on Chinese are similar.

The gold standard dependencies were extracted from Collins' parser output on the sentences. The LITG and BLITG dependencies were extracted from the Viterbi synchronous trees by following the head words.

For comparison, we also included two base-line results. ITG-lh is unlexicalized ITG with left-head assumption, meaning the head words always come from the left branches. ITG-rh is ITG with right-head assumption.

To make more confident conclusions, we also did tests on a larger hand-aligned data set used in Liu et al. (2005). We used 165 sentence pairs that are up to 25 words in length on both sides.

5 Discussion

The BLITG model has two components, namely the dependency model on the upper levels of the tree structure and the word-level translation model at the bottom. We hope that the two components will mutually improve one another. The current experiments indicate clearly that the word level alignment does help inducing dependency structures on both sides. The precision and recall on the dependency retrieval sub-task are almost doubled for both languages from LITG which only has a kind of uni-lexical dependency in each language. Although 20% is a low number, given the fact that the dependencies are learned basically through contrasting sentences in two languages, the result is encouraging. The results slightly improve over ITG with right-head assumption for English, which is based on linguistic insight. Our results also echo the findings of Kuhn (2004). They found that based on the guidance of word alignment between English and multiple other languages, a modified EM training for PCFG on English can bootstrap a more accurate monolingual probabilistic parser. Figure 4 is an example of the dependency tree on the English side from the output of BLITG, comparing against the parser output.

We did not find that the feedback from the de-

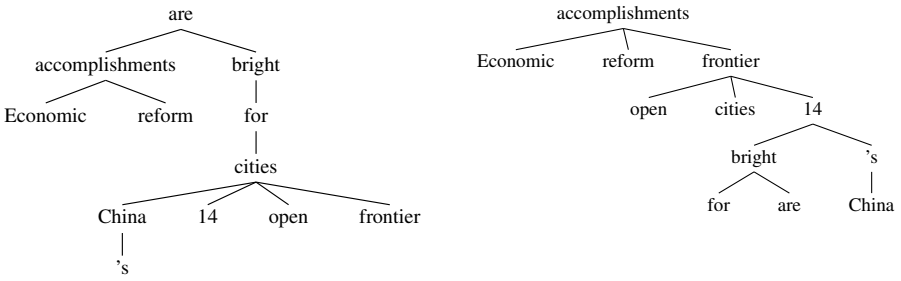


Figure 4: Dependency tree extracted from parser output vs. Viterbi dependency tree from BLITG

pendencies help alignment. To get the reasons, we need further and deeper analysis. One might guess that the dependencies are modeled but are not yet strong and good enough given the amount of training data. Since the training algorithm EM has the problem of local maxima, we might also need to adjust the training algorithm to obtain good parameters for the alignment task. Initializing the model with good dependency parameters is a possible adjustment. We would also like to point out that alignment task is simpler than decoding where a stronger component of reordering is required to produce a fluent English sentence. Investigating the impact of bilexical dependencies on decoding is our future work.

Acknowledgments This work was supported by NSF ITR IIS-09325646 and NSF ITR IIS-0428020.

References

Albert V. Aho and Jeffery D. Ullman. 1972. *The Theory of Parsing, Translation, and Compiling*, volume 1. Prentice-Hall, Englewood Cliffs, NJ.

Hiyan Alshawi, Srinivas Bangalore, and Shona Douglas. 2000. Learning dependency translation models as collections of finite state head transducers. *Computational Linguistics*, 26(1):45–60.

Stanley F. Chen and Joshua Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Conference of the Association for Computational Linguistics (ACL-96)*, pages 310–318, Santa Cruz, CA. ACL.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Conference of the Association for Computational Linguistics (ACL-05)*, pages 263–270, Ann Arbor, Michigan.

Jason Eisner and Giorgio Satta. 1999. Efficient parsing for bilexical context-free grammars and head automaton grammars. In *37th Annual Meeting of the Association for Computational Linguistics*.

Jonas Kuhn. 2004. Experiments in parallel-text based grammar induction. In *Proceedings of the 42nd Annual Conference of the Association for Computational Linguistics (ACL-04)*.

Yang Liu, Qun Liu, and Shouxun Lin. 2005. Log-linear models for word alignment. In *Proceedings of the 43rd Annual Conference of the Association for Computational Linguistics (ACL-05)*, Ann Arbor, Michigan.

I. Dan Melamed. 2003. Multitext grammars and synchronous parsers. In *Proceedings of the 2003 Meeting of the North American chapter of the Association for Computational Linguistics (NAACL-03)*, Edmonton.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Conference of the Association for Computational Linguistics (ACL-00)*, pages 440–447, Hong Kong, October.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Conference of the Association for Computational Linguistics (ACL-01)*, Toulouse, France.

Richard Zens and Hermann Ney. 2003. A comparative study on reordering constraints in statistical machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Sapporo, Japan.

Hao Zhang and Daniel Gildea. 2004. Syntax-based alignment: Supervised or unsupervised? In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-04)*, Geneva, Switzerland, August.

Hao Zhang and Daniel Gildea. 2005. Stochastic lexicalized inversion transduction grammar for alignment. In *Proceedings of the 43rd Annual Conference of the Association for Computational Linguistics (ACL-05)*, Ann Arbor, MI.

Subword-based Tagging for Confidence-dependent Chinese Word Segmentation

Ruiqiang Zhang[1,2] and **Genichiro Kikui**[*] and **Eiichiro Sumita**[1,2]
[1]National Institute of Information and Communications Technology
[2]ATR Spoken Language Communication Research Laboratories
2-2-2 Hikaridai, Seiika-cho, Soraku-gun, Kyoto, 619-0288, Japan
{ruiqiang.zhang,eiichiro.sumita}@atr.jp

Abstract

We proposed a subword-based tagging for Chinese word segmentation to improve the existing character-based tagging. The subword-based tagging was implemented using the maximum entropy (MaxEnt) and the conditional random fields (CRF) methods. We found that the proposed subword-based tagging outperformed the character-based tagging in all comparative experiments. In addition, we proposed a confidence measure approach to combine the results of a dictionary-based and a subword-tagging-based segmentation. This approach can produce an ideal tradeoff between the in-vocaulary rate and out-of-vocabulary rate. Our techniques were evaluated using the test data from Sighan Bakeoff 2005. We achieved higher F-scores than the best results in three of the four corpora: PKU(0.951), CITYU(0.950) and MSR(0.971).

1 Introduction

Many approaches have been proposed in Chinese word segmentation in the past decades. Segmentation performance has been improved significantly, from the earliest maximal match (dictionary-based) approaches to HMM-based (Zhang et al., 2003) approaches and recent state-of-the-art machine learning approaches such as maximum entropy (MaxEnt) (Xue and Shen, 2003), support vector machine (SVM) (Kudo and Matsumoto, 2001), conditional random fields (CRF) (Peng and McCallum, 2004), and minimum error rate training (Gao et al., 2004). By analyzing the top results in the first and second Bakeoffs, (Sproat and Emerson, 2003) and (Emerson, 2005), we found the top results were produced by direct or indirect use of so-called "IOB" tagging, which converts the problem of word segmentation into one of character tagging so that part-of-speech tagging approaches can be used for word segmentation. This approach was also called "LMR" (Xue and Shen, 2003) or "BIES" (Asahara et al., 2005) tagging. Under the scheme, each character of a word is labeled as "B" if it is the first character of a multiple-character word, or "I" otherwise, and "O" if the character functioned as an independent word. For example, "全(whole) 北京市(Beijing city)" is labeled as "全/O 北/B 京/I 市/I". Thus, the training data in word sequences are turned into IOB-labeled data in character sequences, which are then used as the training data for tagging. For new test data, word boundaries are determined based on the results of tagging.

While the IOB tagging approach has been widely used in Chinese word segmentation, we found that so far all the existing implementations were using character-based IOB tagging. In this work we propose a subword-based IOB tagging, which assigns tags to a pre-defined lexicon subset consisting of the most frequent multiple-character words in addition to single Chinese characters. If only Chinese characters are used, the subword-based IOB tagging is downgraded to a character-based one. Taking the same example mentioned above, "全北京市" is la-

[*] Now the second author is affiliated with NTT.

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 961–968,
Sydney, July 2006.

beled as "全/O 北京/B 市/I" in the subword-based tagging, where "北京/B" is labeled as one unit. We will give a detailed description of this approach in Section 2.

There exists a clear weakness with the IOB tagging approach: It yields a very low in-vocabulary rate (R-iv) in return for a higher out-of-vocabulary (OOV) rate (R-oov). In the results of the closed test in Bakeoff 2005 (Emerson, 2005), the work of (Tseng et al., 2005), using CRFs for the IOB tagging, yielded a very high R-oov in all of the four corpora used, but the R-iv rates were lower. While OOV recognition is very important in word segmentation, a higher IV rate is also desired. In this work we propose a confidence measure approach to lessen this weakness. By this approach we can change the R-oov and R-iv and find an optimal tradeoff. This approach will be described in Section 2.3.

In addition, we illustrate our word segmentation process in Section 2, where the subword-based tagging is described by the MaxEnt method. Section 3 presents our experimental results. The effects using the MaxEnts and CRFs are shown in this section. Section 4 describes current state-of-the-art methods with Chinese word segmentation, with which our results were compared. Section 5 provides the concluding remarks and outlines future goals.

2 Chinese word segmentation framework

Our word segmentation process is illustrated in Fig. 1. It is composed of three parts: a dictionary-based N-gram word segmentation for segmenting IV words, a maximum entropy subword-based tagger for recognizing OOVs, and a confidence-dependent word disambiguation used for merging the results of both the dictionary-based and the IOB-tagging-based. An example exhibiting each step's results is also given in the figure.

2.1 Dictionary-based N-gram word segmentation

This approach can achieve a very high R-iv, but no OOV detection. We combined with it the N-gram language model (LM) to solve segmentation ambiguities. For a given Chinese character sequence, $C = c_0 c_1 c_2 \dots c_N$, the problem of word segmentation can be formalized as finding a word sequence,

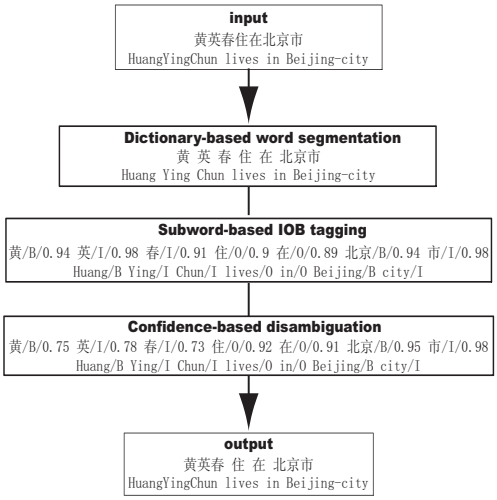


Figure 1: Outline of word segmentation process

$W = w_{t_0} w_{t_1} w_{t_2} \dots w_{t_M}$, which satisfies

$$w_{t_0} = c_0 \dots c_{t_0}, \quad w_{t_1} = c_{t_0+1} \dots c_{t_1}$$
$$w_{t_i} = c_{t_{i-1}+1} \dots c_{t_i}, \quad w_{t_M} = c_{t_{M-1}+1} \dots c_{t_M}$$
$$t_i > t_{i-1}, \quad 0 \le t_i \le N, \quad 0 \le i \le M$$

such that

$$\begin{aligned} W &= \arg\max_W P(W|C) = \arg\max_W P(W)P(C|W) \\ &= \arg\max_W P(w_{t_0} w_{t_1} \dots w_{t_M}) \delta(c_0 \dots c_{t_0}, w_{t_0}) \\ &\quad \delta(c_{t_0+1} \dots c_{t_1}, w_{t_1}) \dots \delta(c_{t_{M-1}+1} \dots c_M, w_{t_M}) \end{aligned} \tag{1}$$

We applied Bayes' law in the above derivation. Because the word sequence must keep consistent with the character sequence, $P(C|W)$ is expanded to be a multiplication of a Kronecker delta function series, $\delta(u, v)$, equal to 1 if both arguments are the same and 0 otherwise. $P(w_{t_0} w_{t_1} \dots w_{t_M})$ is a language model that can be expanded by the chain rule. If trigram LMs are used, we have

$$P(w_0)P(w_1|w_0)P(w_2|w_0 w_1) \cdots P(w_M|w_{M-2} w_{M-1})$$

where w_i is a shorthand for w_{t_i} .

Equation 1 indicates the process of dictionary-based word segmentation. We looked up the lexicon to find all the IVs, and evaluated the word sequences by the LMs. We used a beam search (Jelinek, 1998) instead of a viterbi search to decode the best word

sequence because we found that a beam search can speed up the decoding. N-gram LMs were used to score all the hypotheses, of which the one with the highest LM scores is the final output. The experimental results are presented in Section 3.1, where we show the comparative results as we changed the order of LMs.

2.2 Subword-based IOB tagging

There are several steps to train a subword-based IOB tagger. First, we extracted a word list from the training data sorted in decreasing order by their counts in the training data. We chose all the single characters and the top multi-character words as a lexicon subset for the IOB tagging. If the subset consists of Chinese characters only, it is a character-based IOB tagger. We regard the words in the subset as the subwords for the IOB tagging.

Second, we re-segmented the words in the training data into subwords of the subset, and assigned IOB tags to them. For the character-based IOB tagger, there is only one possibility for re-segmentation. However, there are multiple choices for the subword-based IOB tagger. For example, "北 京 市(Beijing-city)" can be segmented as "北 京 市(Beijing-city)/O," or "北 京(Beijing)/B 市(city)/I," or "北(north)/B 京(capital)/I 市(city)/I." In this work we used forward maximal match (FMM) for disambiguation. Because we carried out FMMs on each words in the manually segmented training data, the accuracy of FMM was much higher than applying it on whole sentences. Of course, backward maximal match (BMM) or other approaches are also applicable. We did not conduct comparative experiments due to trivial differences in the results of these approaches.

In the third step, we used the maximum entropy (MaxEnt) approach (the results of CRF are given in Section 3.4) to train the IOB tagger (Xue and Shen, 2003). The mathematical expression for the MaxEnt model is

$$P(t|h) = \exp\left(\sum_i \lambda_i f_i(h,t)\right)/Z, \quad Z = \sum_t P(t|h) \quad (2)$$

where t is a tag, "I,O,B," of the current word; h, the context surrounding the current word, including word and tag sequences; f_i, a binary feature equal to 1 if the i-th defined feature is activated and 0 otherwise; Z, a normalization coefficient; and λ_i, the weight of the i-th feature.

Many kinds of features can be defined for improving the tagging accuracy. However, to conform to the constraints of closed test in Bakeoff 2005, some features, such as syntactic information and character encodings for numbers and alphabetical characters, are not allowed. Therefore, we used the features available only from the provided training corpus.

- Contextual information:
  $w_0, t_{-1}, w_0t_{-1}, w_0t_{-1}w_1, t_{-1}w_1, t_{-1}t_{-2}, w_0t_{-1}t_{-2}$,
  $w_0w_1, w_0w_1w_2, w_{-1}, w_0w_{-1}, w_0w_{-1}w_1$,
  $w_{-1}w_1, w_{-1}w_{-2}, w_0w_{-1}w_{-2}, w_1, w_1w_2$
  where w stands for word and t, for IOB tag. The subscripts are position indicators, where 0 means the current word/tag; $-1, -2$, the first or second word/tag to the left; $1, 2$, the first or second word/tag to the right.

- Prefixes and suffixes. These are very useful features. Using the same approach as in (Tseng et al., 2005), we extracted the most frequent words tagged with "B", indicating a prefix, and the last words tagged with "I", denoting a suffix. Features containing prefixes and suffixes were used in the following combinations with other features, where p stands for prefix; s, suffix; p_0 means the current word is a prefix and s_1 denotes that the right first word is a suffix, and so on.
  $p_0, w_0p_{-1}, w_0p_1, s_0, w_0s_{-1}, w_0s_1$,
  $p_0w_{-1}, p_0w_1, s_0w_{-1}, s_0w_{-2}$

- Word length. This is defined as the number of characters in a word. The length of a Chinese word has discriminative roles for word composition. For example, single-character words are more apt to form new words than are multiple-character words. Features using word length are listed below, where l_0 means the word length of the current word. Others can be inferred similarly.
  $l_0, w_0l_{-1}, w_0l_1, w_0l_{-1}l_1, l_0l_{-1}, l_0l_1$

As to feature selection, we simply adopted the absolute count for each feature in the training data as

the metric, and defined a cutoff value for each feature type.

We used IIS to train the maximum entropy model. For details, refer to (Lafferty et al., 2001).

The tagging algorithm is based on the beam-search method (Jelinek, 1998). After the IOB tagging, each word is tagged with a B/I/O tag. The word segmentation is obtained immediately. The experimental effect of the word-based tagger and its comparison with the character-based tagger are made in section 3.2.

2.3 Confidence-dependent word segmentation

In the last two steps we produced two segmentation results: the one by the dictionary-based approach and the one by the IOB tagging. However, neither was perfect. The dictionary-based segmentation produced a result with a higher R-iv but lower R-oov while the IOB tagging yielded the contrary results. In this section we introduce a confidence measure approach to combine the two results. We define a confidence measure, $CM(t_{iob}|w)$, to measure the confidence of the results produced by the IOB tagging by using the results from the dictionary-based segmentation. The confidence measure comes from two sources: IOB tagging and dictionary-based word segmentation. Its calculation is defined as:

$$CM(t_{iob}|w) = \alpha CM_{iob}(t_{iob}|w) + (1-\alpha)\delta(t_w, t_{iob})_{ng} \quad (3)$$

where t_{iob} is the word w's IOB tag assigned by the IOB tagging; t_w, a prior IOB tag determined by the results of the dictionary-based segmentation. After the dictionary-based word segmentation, the words are re-segmented into subwords by FMM before being fed to IOB tagging. Each subword is given a prior IOB tag, t_w. $CM_{iob}(t|w)$, a confidence probability derived in the process of IOB tagging, which is defined as

$$CM_{iob}(t|w) = \frac{\sum_{h_i} P(t|w, h_i)}{\sum_t \sum_{h_i} P(t|w, h_i)}$$

where h_i is a hypothesis in the beam search. $\delta(t_w, t_{iob})_{ng}$ denotes the contribution of the dictionary-based segmentation.

$\delta(t_w, t_{iob})_{ng}$ is a Kronecker delta function defined as

$$\delta(t_w, t_{iob})_{ng} = \begin{cases} 1 & \texttt{if } t_w = t_{iob} \\ 0 & \texttt{otherwise} \end{cases}$$

In Eq. 3, α is a weighting between the IOB tagging and the dictionary-based word segmentation. We found an empirical value 0.8 for α.

By Eq. 3 the results of IOB tagging were re-evaluated. A confidence measure threshold, t, was defined for making a decision based on the value. If the value was lower than t, the IOB tag was rejected and the dictionary-based segmentation was used; otherwise, the IOB tagging segmentation was used. A new OOV was thus created. For the two extreme cases, $t = 0$ is the case of the IOB tagging while $t = 1$ is that of the dictionary-based approach. In Section 3.3 we will present the experimental segmentation results of the confidence measure approach. In a real application, we can actually change the confidence threshold to obtain a satisfactory balance between R-iv and R-oov.

An example is shown in Figure 1. In the stage of IOB tagging, a confidence is attached to each word. In the stage of confidence-based, a new confidence was made after merging with dictionary-based results where all single-character words are labeled as "O" by default except "Beijing-city" labeled as "Beijing/B" and "city/I".

3 Experiments

We used the data provided by Sighan Bakeoff 2005 to test our approaches described in the previous sections. The data contain four corpora from different sources: Academia sinica, City University of Hong Kong, Peking University and Microsoft Research (Beijing). The statistics concerning the corpora is listed in Table 3. The corpora provided both unicode coding and Big5/GB coding. We used the Big5 and CP936 encodings. Since the main purpose of this work is to evaluate the proposed subword-based IOB tagging, we carried out the closed test only. Five metrics were used to evaluate the segmentation results: recall (R), precision (P), F-score (F), OOV rate (R-oov) and IV rate (R-iv). For a detailed explanation of these metrics, refer to (Sproat and Emerson, 2003).

Corpus	Abbrev.	Encodings	Training size (words)	Test size (words)
Academia Sinica	AS	Big5/Unicode	5.45M	122K
Beijing University	PKU	CP936/Unicode	1.1M	104K
City University of Hong Kong	CITYU	Big5/Unicode	1.46M	41K
Microsoft Research (Beijing)	MSR	CP936/Unicode	2.37M	107K

Table 1: Corpus statistics in Sighan Bakeoff 2005

3.1 Effects of N-gram LMs

We obtained a word list from the training data as the vocabulary for dictionary-based segmentation. N-gram LMs were generated using the SRI LM toolkit. Table 2 shows the performance of N-gram segmentation by changing the order of N-grams.

We found that bigram LMs can improve segmentation over unigram, though we observed no effect from the trigram LMs. For the PKU corpus, there was a relatively strong improvement due to using bigrams rather than unigrams, posssibly because the PKU corpus' training size was smaller than the others. For a sufficiently large training corpus, the unigram LMs may be enough for segmentation. This experiment revealed that language models above bigrams do not improve word segmentation. Since there were some single-character words present in test data but not in the training data, the R-oov rates were not zero in this experiment. In fact, we did not use any OOV detection for the dictionary-based approach.

3.2 Comparisons of Character-based and Subword-based tagger

In Section 2.2 we described the character-based and subword-based IOB tagging methods. The main difference between the two is the lexicon subset used for re-segmentation. For the subword-based IOB tagging, we need to add some multiple-character words into the lexicon subset. Since it is hard to decide the optimal number of words to add, we test three different lexicon sizes, as shown in Table 3. The first one, s1, consisting of all the characters, is a character-based approach. The second, s2, added 2,500 top words from the training data to the lexicon of s1. The third, s3, added another 2,500 top words to the lexicon of s2. All the words were among the most frequent in the training corpora. After choosing the subwords, the training data were re-segmented using the subwords by FMM. The final lexicons were collected again, consisting of single-character words and multiple-character words. Table 3 shows the sizes of the final lexicons. Therefore, the minus of the lexicon size of s2 to s1 are not 2,500, exactly.

	AS	CITYU	MSR	PKU
s1	6,087	4,916	5,150	4,685
s2	8,332	7,338	7,464	7,014
s3	10,876	9,996	9,990	9,053

Table 3: Three different vocabulary sizes used in subword-based tagging. s1 contains all the characters. s2 and s3 contains some common words.

The segmentation results of using three lexicons are shown in Table 4. The numbers are separated by a "/" in the sequence of "s1/s2/s3." We found although the subword-based approach outperformed the character-based one significantly, there was no obvious difference between the two subword-based approaches, s2 and s3, adding respective 2,500 and 5,000 subwords to s1. The experiments show that we cannot find an optimal lexicon size from 2,500 to 5,000. However, there might be an optimal point less than 2,500. We did not take much effort to find the optimal point, and regarded 2,500 as an acceptable size for practical usages.

The F-scores of IOB tagging shown in Table 4 are better than that of N-gram word segmentation in Table 2, which proves that the IOB tagging is effective in recognizing OOV. However, we found there was a large decrease in the R-ivs, which shows the weakness of the IOB tagging approach. We use the confidence measure approach to deal with this problem in next section.

3.3 Effects of the confidence measure

Up to now we had two segmentation results by using the dictionary-based word segmentation and the IOB tagging. In Section 2.3, we proposed a confidence measure approach to re-evaluate the results of IOB tagging by combining the two results. The effects of

	R	P	F	R-oov	R-iv
AS	0.934/0.942/0.941	0.884/0.881/0.881	0.909/0.910/0.910	0.041/0.040/0.038	0.975/0.983/0.982
CITYU	0.924/0.929/0.928	0.851/0.851/0.851	0.886/0.888/0.888	0.162/0.162/0.164	0.984/0.990/0.989
PKU	0.938/0.949/0.948	0.909/0.912/0.912	0.924/0.930/0.930	0.407/0.403/0.408	0.971/0.982/0.981
MSR	0.965/0.969/0.968	0.927/0.927/0.927	0.946/0.947/0.947	0.036/0.036/0.048	0.991/0.994/0.993

Table 2: Segmentation results of dictionary-based segmentation in closed test of Bakeoff 2005. A "/" separates the results of unigram, bigram and trigram.

	R	P	F	R-oov	R-iv
AS	0.922/0.942/0.943	0.914/0.930/0.930	0.918/0.936/0.937	0.641/0.628/0.609	0.935/0.956/0.959
CITYU	0.906/0.933/0.934	0.905/0.929/0.927	0.906/0.931/0.930	0.668/0.671/0.671	0.925/0.954/0.955
PKU	0.913/0.934/0.936	0.922/0.938/0.940	0.918/0.936/0.938	0.744/0.724/0.713	0.924/0.946/0.949
MSR	0.929/0.953/0.953	0.934/0.955/0.952	0.932/0.954/0.952	0.656/0.684/0.665	0.936/0.961/0.961

Table 4: Segmentation results by the pure subword-based IOB tagging. The separator "/" divides the results by three lexicon sizes as illustrated in Table 3. The first is character-based (s1), while the other two are subword-based with different lexicons (s2/s3).

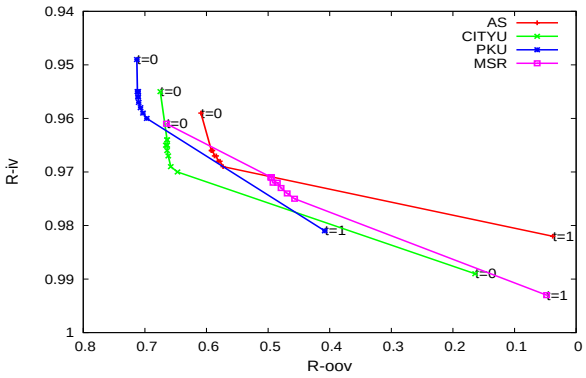


Figure 2: R-iv and R-oov varing as the confidence threshold, t.

the confidence measure are shown in Table 5, where we used $\alpha = 0.8$ and confidence threshold $t = 0.7$. These are empirical numbers. We obtained the optimal values by multiple trials on held-out data. The numbers in the slots of Table 5 are divided by a separator "/" and displayed as the sequence "s1/s2/s3", just as Table 4. We found that the results in Table 5 were better than those in Table 4 and Table 2, which proved that using the confidence measure approach yielded the best performance over the N-gram segmentation and the IOB tagging approaches.

Even with the use of the confidence measure, the subword-based IOB tagging still outperformed the character-based IOB tagging, proving that the proposed subword-based IOB tagging was very effective. Though the improvement under the confidence measure was decreasing, it was still significant.

We can change the R-oov and R-iv by changing the confidence threshold. The effect of R-oov and R-iv's varing as the threshold is shown in Fig. 2, where R-oovs and R-ivs are moving in different directions. When the confidence threshold $t = 0$, the case for the IOB tagging, R-oovs are maximal. When $t = 1$, representing the dictionary-based segmentation, R-oovs are the minimal. The R-oovs and R-ivs varied largely at the start and end point but little around the middle section.

3.4 Subword-based tagging by CRFs

Our proposed approaches were presented and evaluated using the MaxEnt method in the previous sections. When we turned to CRF-based tagging, we found a same effect as the MaxEnt method. Our subword-based tagging by CRFs was implemented by the package "CRF++" from the site "http://www.chasen.org/~taku/software."

We repeated the previous sections' experiments using the CRF approach except that we did one of the two subword-based tagging, the lexicon size s3. The same values of the confidence measure threshold and α were used. The results are shown in Table 6.

We found that the results using the CRFs were much better than those of the MaxEnts. However, the emphasis here was not to compare CRFs and MaxEnts but the effect of subword-based IOB tagging. In Table 6, the results before "/" are the character-based IOB tagging and after "/", the subword-based. It was clear that the subword-based approaches yielded better results than the character-based approach though the improvement was not as higher as that of the MaxEnt approaches. There was

	R	P	F	R-oov	R-iv
AS	0.938/0.950/0.953	0.945/0.946/0.951	0.941/0.948/0.948	0.674/0.641/0.606	0.950/0.964/0.969
CITYU	0.932/0.949/0.946	0.944/0.933/0.944	0.938/0.941/0.945	0.705/0.597/0.667	0.950/0.977/0.968
PKU	0.941/0.948/0.949	0.945/0.947/0.947	0.943/0.948/0.948	0.672/0.662/0.660	0.958/0.966/0.966
MSR	0.944/0.959/0.961	0.959/0.964/0.963	0.951/0.961/0.962	0.671/0.674/0.631	0.951/0.967/0.970

Table 5: Effects of combination using the confidence measure. Here we used $\alpha = 0.8$ and confidence threshold $t = 0.7$. The separator "/" divides the results of s1, s2, and s3.

no change on F-score for AS corpus, but a better recall rate was found. Our results are better than the best one of Bakeoff 2005 in PKU, CITYU and MSR corpora.

Detailed descriptions about subword tagging by CRF can be found in our paper (Zhang et al., 2006).

4 Discussion and Related works

The IOB tagging approach adopted in this work is not a new idea. It was first implemented in Chinese word segmentation by (Xue and Shen, 2003) using the maximum entropy methods. Later, (Peng and McCallum, 2004) implemented the idea using the CRF-based approach, which yielded better results than the maximum entropy approach because it could solve the label bias problem (Lafferty et al., 2001). However, as we mentioned before, this approach does not take advantage of the prior knowledge of in-vocabulary words; It produced a higher R-oov but a lower R-iv. This problem has been observed by some participants in the Bakeoff 2005 (Asahara et al., 2005), where they applied the IOB tagging to recognize OOVs, and added the OOVs to the lexicon used in the HMM-based or CRF-based approaches. (Nakagawa, 2004) used hybrid HMM models to integrate word level and character level information seamlessly. We used confidence measure to determine a better balance between R-oov and R-iv. The idea of using the confidence measure has appeared in (Peng and McCallum, 2004), where it was used to recognize the OOVs. In this work we used it more than that. By way of the confidence measure we combined results from the dictionary-based and the IOB-tagging-based and as a result, we could achieve the optimal performance.

Our main contribution is to extend the IOB tagging approach from being a character-based to a subword-based one. We proved that the new approach enhanced the word segmentation significantly in all the experiments, MaxEnts, CRFs and using confidence measure. We tested our approach using the standard Sighan Bakeoff 2005 data set in the closed test. In Table 7 we align our results with some top runners' in the Bakeoff 2005.

Our results were compared with the best performers' results in the Bakeoff 2005. Two participants' results were chosen as bases: No.15-b, ranked the first in the AS corpus, and No.14, the best performer in CITYU, MSR and PKU. . The No.14 used CRF-modeled IOB tagging while No.15-b used MaxEnt-modeled IOB tagging. Our results produced by the MaxEnt are denoted as "ours(ME)" while "ours(CRF)" for the CRF approaches. We achieved the highest F-scores in three corpora except the AS corpus. We think the proposed subword-based approach played the important role for the achieved good results.

A second advantage of the subword-based IOB tagging over the character-based is its speed. The subword-based approach is faster because fewer words than characters needed to be labeled. We observed a speed increase in both training and testing. In the training stage, the subword approach was almost two times faster than the character-based.

5 Conclusions

In this work, we proposed a subword-based IOB tagging method for Chinese word segmentation. The approach outperformed the character-based method using both the MaxEnt and CRF approaches. We also successfully employed the confidence measure to make a confidence-dependent word segmentation. By setting the confidence threshold, R-oov and R-iv can be changed accordingly. This approach is effective for performing desired segmentation based on users' requirements to R-oov and R-iv.

	R	P	F	R-oov	R-iv
AS	0.953/0.956	0.944/0.947	0.948/0.951	0.607/0.649	0.969/0.969
CITYU	0.943/0.952	0.948/0.949	0.946/0.951	0.682/0.741	0.964/0.969
PKU	0.942/0.947	0.957/0.955	0.949/0.951	0.775/0.748	0.952/0.959
MSR	0.960/0.972	0.966/0.969	0.963/0.971	0.674/0.712	0.967/0.976

Table 6: Effects of using CRF. The separator "/" divides the results of s1, and s3.

Participants	R	P	F	R-oov	R-iv
Hong Kong City University					
ours(CRF)	0.952	0.949	0.951	0.741	0.969
ours(ME)	0.946	0.944	0.945	0.667	0.968
14	0.941	0.946	0.943	0.698	0.961
15-b	0.937	0.946	0.941	0.736	0.953
Academia Sinica					
15-b	0.952	0.951	0.952	0.696	0.963
ours(CRF)	0.956	0.947	0.951	0.649	0.969
ours(ME)	0.953	0.943	0.948	0.608	0.969
14	0.95	0.943	0.947	0.718	0.960
Microsoft Research					
ours(CRF)	0.972	0.969	0.971	0.712	0.976
14	0.962	0.966	0.964	0.717	0.968
ours(ME)	0.961	0.963	0.962	0.631	0.970
15-b	0.952	0.964	0.958	0.718	0.958
Peking University					
ours(CRF)	0.947	0.955	0.951	0.748	0.959
14	0.946	0.954	0.950	0.787	0.956
ours(ME)	0.949	0.947	0.948	0.660	0.966
15-b	0.93	0.951	0.941	0.76	0.941

Table 7: List of results in Sighan Bakeoff 2005

Acknowledgements

The authors thank the reviewers for the comments and advice on the paper. Some related software for this work will be released very soon.

References

Masayuki Asahara, Kenta Fukuoka, Ai Azuma, Chooi-Ling Goh, Yotaro Watanabe, Yuji Matsumoto, and Takashi Tsuzuki. 2005. Combination of machine learning methods for optimum chinese word segmentation. In *Forth SIGHAN Workshop on Chinese Language Processing, Proceedings of the Workshop*, pages 134–137, Jeju, Korea.

Thomas Emerson. 2005. The second international chinese word segmentation bakeoff. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, Jeju, Korea.

Jianfeng Gao, Andi Wu, Mu Li, Chang-Ning Huang, Hongqiao Li, Xinsong Xia, and Haowei Qin. 2004. Adaptive chinese word segmentation. In *ACL-2004*, Barcelona, July.

Frederick Jelinek. 1998. *Statistical methods for speech recognition*. the MIT Press.

Taku Kudo and Yuji Matsumoto. 2001. Chunking with support vector machine. In *Proc. of NAACL-2001*, pages 192–199.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML-2001*, pages 591–598.

Tetsuji Nakagawa. 2004. Chinese and japanese word segmentation using word-level and character-level information. In *Proceedings of Coling 2004*, pages 466–472, Geneva, August.

Fuchun Peng and Andrew McCallum. 2004. Chinese segmentation and new word detection using conditional random fields. In *Proc. of Coling-2004*, pages 562–568, Geneva, Switzerland.

Richard Sproat and Tom Emerson. 2003. The first international chinese word segmentation bakeoff. In *Proceedings of the Second SIGHAN Workshop on Chinese Language Processing*, Sapporo, Japan, July.

Huihsin Tseng, Pichuan Chang, Galen Andrew, Daniel Jurafsky, and Christopher Manning. 2005. A conditional random field word segmenter for Sighan bakeoff 2005. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, Jeju, Korea.

Nianwen Xue and Libin Shen. 2003. Chinese word segmentation as LMR tagging. In *Proceedings of the Second SIGHAN Workshop on Chinese Language Processing*.

Huaping Zhang, HongKui Yu, Deyi xiong, and Qun Liu. 2003. HHMM-based Chinese lexical analyzer ICTCLAS. In *Proceedings of the Second SIGHAN Workshop on Chinese Language Processing*, pages 184–187.

Ruiqiang Zhang, Genichiro Kikui, and Eiichiro Sumita. 2006. Subword-based tagging by conditional random fields for chinese word segmentation. In *Proc. of HLT-NAACL*.

BiTAM: Bilingual Topic AdMixture Models for Word Alignment

Bing Zhao[†] and **Eric P. Xing**[†‡]
{bzhao,epxing}@cs.cmu.edu
Language Technologies Institute[†] and Machine Learning Department[‡]
School of Computer Science, Carnegie Mellon University

Abstract

We propose a novel bilingual topical admixture (BiTAM) formalism for word alignment in statistical machine translation. Under this formalism, the parallel sentence-pairs within a document-pair are assumed to constitute a mixture of hidden topics; each word-pair follows a topic-specific bilingual translation model. Three BiTAM models are proposed to capture topic sharing at different levels of linguistic granularity (i.e., at the sentence or word levels). These models enable word-alignment process to leverage topical contents of document-pairs. Efficient variational approximation algorithms are designed for inference and parameter estimation. With the inferred latent topics, BiTAM models facilitate coherent pairing of bilingual linguistic entities that share common topical aspects. Our preliminary experiments show that the proposed models improve word alignment accuracy, and lead to better translation quality.

1 Introduction

Parallel data has been treated as sets of unrelated sentence-pairs in state-of-the-art statistical machine translation (SMT) models. Most current approaches emphasize within-sentence dependencies such as the distortion in (Brown et al., 1993), the dependency of alignment in HMM (Vogel et al., 1996), and syntax mappings in (Yamada and Knight, 2001). Beyond the sentence-level, corpus-level word-correlation and contextual-level topical information may help to disambiguate translation candidates and word-alignment choices. For example, the most frequent source words (e.g., functional words) are likely to be translated into words which are also frequent on the target side; words of the same topic generally bear correlations and similar translations. Extended contextual information is especially useful when translation models are vague due to their reliance solely on word-pair co-occurrence statistics. For example, the word *shot* in "*It was a nice shot.*" should be translated differently depending on the context of the sentence: a *goal* in the context of sports, or a *photo* within the context of sightseeing. Nida (1964) stated that sentence-pairs are tied by the logic-flow in a document-pair; in other words, the document-pair should be word-aligned as one entity instead of being uncorrelated instances. In this paper, we propose a probabilistic admixture model to capture latent topics underlying the context of document-pairs. With such topical information, the translation models are expected to be sharper and the word-alignment process less ambiguous.

Previous works on topical translation models concern mainly explicit logical representations of semantics for machine translation. This include knowledge-based (Nyberg and Mitamura, 1992) and interlingua-based (Dorr and Habash, 2002) approaches. These approaches can be expensive, and they do not emphasize stochastic translation aspects. Recent investigations along this line includes using word-disambiguation schemes (Carpua and Wu, 2005) and non-overlapping bilingual word-clusters (Wang et al., 1996; Och, 1999; Zhao et al., 2005) with particular translation models, which showed various degrees of success. We propose a new statistical formalism: Bilingual Topic AdMixture model, or BiTAM, to facilitate topic-based word alignment in SMT.

Variants of admixture models have appeared in population genetics (Pritchard et al., 2000) and text modeling (Blei et al., 2003). Statistically, an object is said to be derived from an *admixture* if it consists of a bag of elements, each sampled independently or coupled in some way, from a mixture model. In a typical SMT setting, each document-pair corresponds to an object; depending on a chosen modeling granularity, all sentence-pairs or word-pairs in the document-pair correspond to the elements constituting the object. Correspondingly, a latent topic is sampled for each pair from a prior topic distribution to induce topic-specific translations; and the resulting sentence-pairs and word-pairs are marginally dependent. Generatively, this *admixture formalism* enables word translations to be instantiated by topic-specific bilingual models

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 969–976,
Sydney, July 2006.

and/or monolingual models, depending on their contexts. In this paper we investigate three instances of the BiTAM model, They are data-driven and do not need hand-crafted knowledge engineering.

The remainder of the paper is as follows: in section 2, we introduce notations and baselines; in section 3, we propose the topic admixture models; in section 4, we present the learning and inference algorithms; and in section 5 we show experiments of our models. We conclude with a brief discussion in section 6.

2 Notations and Baseline

In statistical machine translation, one typically uses parallel data to identify entities such as "word-pair", "sentence-pair", and "document-pair". Formally, we define the following terms[1]:

- A *word-pair* (f_j, e_i) is the basic unit for word alignment, where f_j is a French word and e_i is an English word; j and i are the *position indices* in the corresponding French sentence $\mathbf{f}$ and English sentence $\mathbf{e}$.
- A *sentence-pair* $(\mathbf{f}, \mathbf{e})$ contains the *source* sentence $\mathbf{f}$ of a *sentence length* of J; a *target* sentence $\mathbf{e}$ of *length* I. The two sentences $\mathbf{f}$ and $\mathbf{e}$ are translations of each other.
- A *document-pair* $(\mathbf{F}, \mathbf{E})$ refers to two documents which are translations of each other. Assuming sentences are one-to-one correspondent, a document-pair has a sequence of N parallel sentence-pairs $\{(\mathbf{f}_n, \mathbf{e}_n)\}$, where $(\mathbf{f}_n, \mathbf{e}_n)$ is the $n'th$ parallel sentence-pair.
- A *parallel corpus* $\mathbf{C}$ is a collection of M parallel document-pairs: $\{(\mathbf{F}_d, \mathbf{E}_d)\}$.

2.1 Baseline: IBM Model-1

The translation process can be viewed as operations of word substitutions, permutations, and insertions/deletions (Brown et al., 1993) in noisy-channel modeling scheme at parallel sentence-pair level. The translation lexicon $p(f|e)$ is the key component in this generative process. An efficient way to learn $p(f|e)$ is IBM-1:

$$p(\mathbf{f}|\mathbf{e}) = \prod_{j=1}^{J} \sum_{i=1}^{I} p(f_j|e_i) \cdot p(e_i|\mathbf{e}). \quad (1)$$

[1]We follow the notations in (Brown et al., 1993) for English-French, i.e., $\mathbf{e} \leftrightarrow \mathbf{f}$, although our models are tested, in this paper, for English-Chinese. We use the *end-user terminology* for *source* and *target* languages.

IBM-1 has global optimum; it is efficient and easily scalable to large training data; it is one of the most informative components for re-ranking translations (Och et al., 2004). We start from IBM-1 as our baseline model, while higher-order alignment models can be embedded similarly within the proposed framework.

3 Bilingual Topic AdMixture Model

Now we describe the BiTAM formalism that captures the latent topical structure and generalizes word alignments and translations beyond sentence-level via topic sharing across sentence-pairs:

$$\mathbf{E}^* = \arg\max_{\{\mathbf{E}\}} p(\mathbf{F}|\mathbf{E})p(\mathbf{E}), \quad (2)$$

where $p(\mathbf{F}|\mathbf{E})$ is a document-level translation model, generating the document $\mathbf{F}$ as one entity. In a BiTAM model, a document-pair $(\mathbf{F}, \mathbf{E})$ is treated as an admixture of topics, which is induced by random draws of a topic, from a pool of topics, for each sentence-pair. A unique normalized and real-valued vector θ, referred to as a *topic-weight vector*, which captures contributions of different topics, are instantiated for each document-pair, so that the sentence-pairs with their alignments are generated from topics mixed according to these common proportions. Marginally, a sentence-pair is word-aligned according to a unique bilingual model governed by the hidden topical assignments. Therefore, the sentence-level translations are coupled, rather than being independent as assumed in the IBM models and their extensions.

Because of this coupling of sentence-pairs (via topic sharing across sentence-pairs according to a common topic-weight vector), BiTAM is likely to improve the coherency of translations by treating the document as a whole entity, instead of uncorrelated segments that have to be independently aligned and then assembled. There are at least two levels at which the hidden topics can be sampled for a document-pair, namely: the *sentence-pair* and the *word-pair* levels. We propose three variants of the BiTAM model to capture the latent topics of bilingual documents at different levels.

3.1 BiTAM-1: The Frameworks

In the first BiTAM model, we assume that topics are sampled at the sentence-level. Each document-pair is represented as a random mixture of latent topics. Each topic, topic-k, is presented by a topic-specific word-translation table: B_k, which is

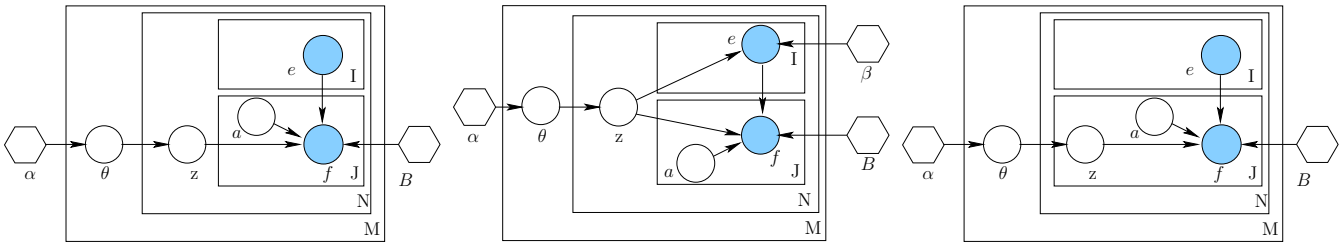


Figure 1: BiTAM models for Bilingual document- and sentence-pairs. A node in the graph represents a random variable, and a hexagon denotes a parameter. Un-shaded nodes are hidden variables. All the plates represent replicates. The outmost plate (M-plate) represents M bilingual document-pairs, while the inner N-plate represents the N repeated choice of topics for each sentence-pairs in the document; the inner J-plate represents J word-pairs within each sentence-pair. (a) BiTAM-1 samples one topic (denoted by z) per sentence-pair; (b) BiTAM-2 utilizes the sentence-level topics for both the translation model (i.e., $p(f|e,z)$) and the monolingual word distribution (i.e., $p(e|z)$); (c) BiTAM-3 samples one topic per word-pair.

a translation lexicon: $B_{i,j,k}$=$p(f$=$f_j|e$=e_i, z=$k)$, where z is an indicator variable to denote the choice of a topic. Given a specific *topic-weight vector* θ_d for a document-pair, each sentence-pair draws its conditionally independent topics from a mixture of topics. This generative process, for a document-pair $(\mathbf{F}_d, \mathbf{E}_d)$, is summarized as below:

1. Sample *sentence-number* N from a Poisson(γ).
2. Sample *topic-weight vector* θ_d from a Dirichlet(α).
3. For each sentence-pair $(\mathbf{f}_n, \mathbf{e}_n)$ in the $d'th$ doc-pair ,
   (a) Sample *sentence-length* J_n from Poisson(δ);
   (b) Sample a topic z_{dn} from a Multinomial(θ_d);
   (c) Sample e_j from a monolingual model $p(e_j)$;
   (d) Sample each word alignment link a_j from a uniform model $p(a_j)$ (or an HMM);
   (e) Sample each f_j according to a topic-specific translation lexicon $p(f_j|\mathbf{e}, a_j, z_n, \mathbf{B})$.

We assume that, in our model, there are K possible topics that a document-pair can bear. For each document-pair, a K-dimensional Dirichlet random variable θ_d, referred to as the topic-weight vector of the document, can take values in the $(K-1)$-simplex following a probability density:

$$p(\theta|\alpha) = \frac{\Gamma(\sum_{k=1}^{K}\alpha_k)}{\prod_{k=1}^{K}\Gamma(\alpha_k)}\theta_1^{\alpha_1-1}\cdots\theta_K^{\alpha_K-1}, \qquad (3)$$

where the hyperparameter α is a K-dimension vector with each component α_k>0, and $\Gamma(x)$ is the Gamma function. The alignment is represented by a J-dimension vector $\mathbf{a} = \{a_1, a_2, \cdots, a_J\}$; for each French word f_j at the position j, an position variable a_j maps it to an English word e_{a_j} at the position a_j in English sentence. The word level translation lexicon probabilities are topic-specific, and they are parameterized by the matrix $\mathbf{B} = \{B_k\}$.

For simplicity, in our current models we omit the modelings of the *sentence-number* N and the *sentence-length* J_n, and focus only on the bilingual translation model. Figure 1 (a) shows the graphical model representation for the BiTAM generative scheme discussed so far. Note that, the sentence-pairs are now connected by the node θ_d. Therefore, marginally, the sentence-pairs are *not* independent of each other as in traditional SMT models, instead they are *conditionally independent* given the topic-weight vector θ_d. Specifically, BiTAM-1 assumes that each sentence-pair has one single topic. Thus, the word-pairs within this sentence-pair are *conditionally independent* of each other given the hidden topic index z of the sentence-pair.

The last two sub-steps (3.d and 3.e) in the BiTam sampling scheme define a translation model, in which an alignment link a_j is proposed and an observation of f_j is generated according to the proposed distributions. We simplify *alignment model* of $\mathbf{a}$, as in *IBM-1*, by assuming that a_j is sampled uniformly at random. Given the parameters α, B, and the English part $\mathbf{E}$, the joint conditional distribution of the topic-weight vector θ, the topic indicators $\mathbf{z}$, the alignment vectors $\mathbf{A}$, and the document $\mathbf{F}$ can be written as:

$$p(\mathbf{F},\mathbf{A},\theta,\mathbf{z}|\mathbf{E},\alpha,\mathbf{B}) = p(\theta\,|\,\alpha)\prod_{n=1}^{N} p(z_n|\theta)p(\mathbf{f}_n,\mathbf{a}_n|\mathbf{e}_n,\ \alpha,\ B_{z_n}), \qquad (4)$$

where N is the number of the sentence-pair. Marginalizing out θ and $\mathbf{z}$, we can obtain the marginal conditional probability of generating $\mathbf{F}$ from $\mathbf{E}$ for each document-pair:

$$p(\mathbf{F},\mathbf{A}|\mathbf{E},\alpha,B_{z_n}) = \int p(\theta|\alpha)\Big(\prod_{n=1}^{N}\sum_{z_n} p(z_n|\theta)p(\mathbf{f}_n,\mathbf{a}_n|\mathbf{e}_n,B_{z_n})\Big)d\theta, \qquad (5)$$

where $p(\mathbf{f}_n,\mathbf{a}_n|\mathbf{e}_n,B_{z_n})$ is a topic-specific sentence-level translation model. For simplicity, we assume that the French words f_j's are conditionally independent of each other; the alignment

variables a_j's are independent of other variables and are uniformly distributed *a priori*. Therefore, the distribution for each sentence-pair is:

$$p(\mathbf{f}_n, \mathbf{a}_n|\mathbf{e}_n, B_{z_n}) = p(\mathbf{f}_n|\mathbf{e}_n, \mathbf{a}_n, B_{z_n})p(\mathbf{a}_n|\mathbf{e}_n, B_{z_n}) = \frac{1}{I_n^{J_n}} \prod_{j=1}^{J_n} p(f_{nj}|e_{a_{nj}}, B_{z_n}). \quad (6)$$

Thus, the conditional likelihood for the entire parallel corpus is given by taking the product of the marginal probabilities of each individual document-pair in Eqn. 5.

3.2 BiTAM-2: Monolingual Admixture

In general, the monolingual model for English can also be a rich topic-mixture. This is realized by using the same topic-weight vector θ_d and the same topic indicator z_{dn} sampled according to θ_d, as described in §3.1, to introduce not only topic-dependent translation lexicon, but also topic-dependent monolingual model of the source language, English in this case, for generating each sentence-pair (Figure 1 (b)). Now $\mathbf{e}$ is generated from a topic-based language model β, instead of a uniform distribution in BiTAM-1. We refer to this model as BiTAM-2.

Unlike BiTAM-1, where the information observed in e_i is indirectly passed to z via the node of f_j and the hidden variable a_j, in BiTAM-2, the topics of corresponding English and French sentences are also strictly aligned so that the information observed in e_i can be directly passed to z, in the hope of finding more accurate topics. The topics are inferred more directly from the observed bilingual data, and as a result, improve alignment.

3.3 BiTAM-3: Word-level Admixture

It is straightforward to extend the sentence-level BiTAM-1 to a word-level admixture model, by sampling topic indicator $z_{n,j}$ for each word-pair (f_j, e_{a_j}) in the $n'th$ sentence-pair, rather than once for all (words) in the sentence (Figure 1 (c)). This gives rise to our BiTAM-3. The conditional likelihood functions can be obtained by extending the formulas in §3.1 to move the variable $z_{n,j}$ inside the same loop over each of the $f_{n,j}$.

3.4 Incorporation of Word "Null"

Similar to IBM models, "Null" word is used for the source words which have no translation counterparts in the target language. For example, Chinese words "de" (的) , "ba" (把) and "bei" (被) generally do not have translations in English. "Null" is attached to every target sentence to align the source words which miss their translations. Specifically, the latent Dirichlet allocation (LDA) in (Blei et al., 2003) can be viewed as a special case of the BiTAM-3, in which the target sentence contains only one word: "Null", and the alignment link a is no longer a hidden variable.

4 Learning and Inference

Due to the hybrid nature of the BiTAM models, exact posterior inference of the hidden variables $\mathbf{A}, \mathbf{z}$ and θ is intractable. A variational inference is used to approximate the true posteriors of these hidden variables. The inference scheme is presented for BiTAM-1; the algorithms for BiTAM-2 and BiTAM-3 are straight forward extensions and are omitted.

4.1 Variational Approximation

To approximate: $p(\theta, \mathbf{z}, \mathbf{A}|\mathbf{E}, \mathbf{F}, \alpha, B)$, the joint posterior, we use the fully factorized distribution over the same set of hidden variables:

$$q(\theta, \mathbf{z}, \mathbf{A}) \propto q(\theta|\gamma, \alpha) \cdot \prod_{n=1}^{N} q(z_n|\phi_n) \prod_{j=1}^{J_n} q(a_{nj}, f_{nj}|\varphi_{nj}, e_n, \mathbf{B}), \quad (7)$$

where the Dirichlet parameter γ, the multinomial parameters $(\phi_1, \cdots, \phi_n)$, and the parameters $(\varphi_{n1}, \cdots, \varphi_{nJ_n})$ are known as variational parameters, and can be optimized with respect to the Kullback-Leibler divergence from $q(\cdot)$ to the original $p(\cdot)$ via an iterative fixed-point algorithm. It can be shown that the fixed-point equations for the variational parameters in BiTAM-1 are as follows:

$$\gamma_k = \alpha_k + \sum_{n=1}^{N_d} \phi_{dnk} \quad (8)$$

$$\phi_{dnk} \propto \exp\Big(\Psi(\gamma_k) - \Psi(\sum_{k'=1}^{K} \gamma_{k'})\Big) \cdot \exp\Big(\sum_{j=1}^{J_{dn}} \sum_{i=1}^{I_{dn}} \varphi_{dnji} \log B_{f_j, e_i, k}\Big) \quad (9)$$

$$\varphi_{dnji} \propto \exp\Big(\sum_{k=1}^{K} \phi_{dnk} \log B_{f_j, e_i, k}\Big), \quad (10)$$

where $\Psi(\cdot)$ is a digamma function. Note that in the above formulas ϕ_{dnk} is the variational parameter underlying the topic indicator z_{dn} of the n-th sentence-pair in document d, and it can be used to predict the topic distribution of that sentence-pair.

Following a variational EM scheme (Beal and Ghahramani, 2002), we estimate the model parameters α and $\mathbf{B}$ in an unsupervised fashion. Essentially, Eqs. (8-10) above constitute the E-step,

where the posterior estimations of the latent variables are obtained. In the M-step, we update α and $\mathbf{B}$ so that they improve a lower bound of the log-likelihood defined bellow:

$$L(\gamma,\phi,\varphi;\alpha,\mathbf{B}) = E_q[\log p(\theta|\alpha)] + E_q[\log p(\mathbf{z}|\theta)] + E_q[\log p(\mathbf{a})] + E_q[\log p(\mathbf{f}|\mathbf{z},\mathbf{a},\mathbf{B})] - E_q[\log q(\theta)] - E_q[\log q(\mathbf{z})] - E_q[\log q(\mathbf{a})]. \quad (11)$$

The close-form iterative updating formula $\mathbf{B}$ is:

$$B_{f,e,k} \propto \sum_{d}^{M}\sum_{n=1}^{N_d}\sum_{j=1}^{J_{dn}}\sum_{i=1}^{I_{dn}} \delta(f,f_j)\delta(e,e_i)\phi_{dnk}\varphi_{dnji} \quad (12)$$

For α, close-form update is not available, and we resort to gradient accent as in (Sjölander et al., 1996) with re-starts to ensure each updated $\alpha_k > 0$.

4.2 Data Sparseness and Smoothing

The translation lexicons $B_{f,e,k}$ have a potential size of V^2K, assuming the vocabulary sizes for both languages are V. The data sparsity (i.e., lack of large volume of document-pairs) poses a more serious problem in estimating $B_{f,e,k}$ than the monolingual case, for instance, in (Blei et al., 2003). To reduce the data sparsity problem, we introduce two remedies in our models. First: *Laplace smoothing*. In this approach, the matrix set $\mathbf{B}$, whose columns correspond to parameters of conditional multinomial distributions, is treated as a collection of random vectors all under a symmetric Dirichlet prior; the posterior expectation of these multinomial parameter vectors can be estimated using Bayesian theory. Second: *interpolation smoothing*. Empirically, we can employ a linear interpolation with IBM-1 to avoid overfitting:

$$B^*_{f,e,k} = \lambda B_{f,e,k} + (1-\lambda)p(f|e). \quad (13)$$

As in Eqn. 1, $p(f|e)$ is learned via IBM-1; λ is estimated via EM on held out data.

4.3 Retrieving Word Alignments

Two word-alignment retrieval schemes are designed for BiTAMs: the *uni-direction* alignment (*UDA*) and the *bi-direction* alignment (*BDA*). Both use the posterior mean of the alignment indicators a_{dnji}, captured by what we call the *posterior alignment matrix* $\varphi \equiv \{\varphi_{dnji}\}$. UDA uses a French word f_{dnj} (at the $j'th$ position of $n'th$ sentence in the $d'th$ document) to query φ to get the best aligned English word (by taking the maximum point in a row of φ):

$$a_{dnj} = \arg\max_{i\in[1,I_{dn}]} \varphi_{dnji}. \quad (14)$$

BDA selects iteratively, for each f, the best aligned e, such that the word-pair (f,e) is the maximum of both row and column, or its neighbors have more aligned pairs than the other combpeting candidates.

A close check of $\{\varphi_{dnji}\}$ in Eqn. 10 reveals that it is essentially an exponential model: weighted log probabilities from individual topic-specific translation lexicons; or it can be viewed as weighted geometric mean of the individual lexicon's strength.

5 Experiments

We evaluate BiTAM models on the *word alignment accuracy* and the *translation quality*. For word alignment accuracy, *F-measure* is reported, i.e., the harmonic mean of precision and recall against a gold-standard reference set; for translation quality, *Bleu* (Papineni et al., 2002) and its variation of NIST scores are reported.

Table 1: Training and Test Data Statistics

Train	#Doc.	#Sent.	#Tokens	
			English	Chinese
Treebank	316	4172	133K	105K
FBIS.BJ	6,111	105K	4.18M	3.54M
Sinorama	2,373	103K	3.81M	3.60M
XinHua	19,140	115K	3.85M	3.93M
Test	95	627	25,500	19,726

We have two training data settings with different sizes (see Table 1). The small one consists of 316 document-pairs from Treebank (*LDC2002E17*). For the large training data setting, we collected additional document-pairs from FBIS (*LDC2003E14*, Beijing part), Sinorama (*LDC2002E58*), and Xinhua News (*LDC2002E18*, document boundaries are kept in our sentence-aligner (Zhao and Vogel, 2002)). There are 27,940 document-pairs, containing 327K sentence-pairs or 12 million (12M) English tokens and 11M Chinese tokens. To evaluate word alignment, we hand-labeled 627 sentence-pairs from 95 document-pairs sampled from TIDES'01 dryrun data. It contains 14,769 alignment-links. To evaluate translation quality, TIDES'02 Eval. test is used as development set, and TIDES'03 Eval. test is used as the unseen test data.

5.1 Model Settings

First, we explore the effects of Null word and smoothing strategies. Empirically, we find that adding "Null" word is always beneficial to all models regardless of number of topics selected.

Topics-Lexicons	Topic-1	Topic-2	Topic-3	Cooc.	IBM-1	HMM	IBM-4
p(ChaoXian (朝鲜)|Korean)	0.0612	0.2138	*0.2254*	38	0.2198	0.2157	0.2104
p(HanGuo (韩国)|Korean)	0.8379	0.6116	*0.0243*	46	0.5619	0.4723	0.4993

Table 2: Topic-specific translation lexicons are learned by a 3-topic BiTAM-1. The *third* lexicon (*Topic-3*) prefers to translate the word *Korean* into *ChaoXian* (朝鲜:North Korean). The co-occurrence (*Cooc*), IBM-1&4 and HMM only prefer to translate into *HanGuo* (韩国:South Korean). The two candidate translations may both fade out in the learned translation lexicons.

Unigram-rank	1	2	3	4	5	6	7	8	9	10
Topic A.	foreign	china	u.s.	development	trade	enterprises	technology	countries	year	economic
Topic B.	chongqing	companies	takeovers	company	city	billion	more	economic	reached	yuan
Topic C.	sports	disabled	team	people	cause	water	national	games	handicapped	members

Table 3: Three most distinctive topics are displayed. The English words for each topic are ranked according to $p(e|z)$ estimated from the topic-specific English sentences weighted by $\{\phi_{dnk}\}$. 33 functional words were removed to highlight the main content of each topic. Topic A is about Us-China economic relationships; Topic B relates to Chinese companies' merging; Topic C shows the sports of handicapped people.

The interpolation smoothing in §4.2 is effective, and it gives slightly better performance than Laplace smoothing over different number of topics for BiTAM-1. However, the interpolation leverages the competing baseline lexicon, and this can blur the evaluations of BiTAM's contributions. Laplace smoothing is chosen to emphasize more on BiTAM's strength. Without any smoothing, F-measure drops very quickly over two topics. In all our following experiments, we use both Null word and Laplace smoothing for the BiTAM models. We train, for comparison, IBM-1&4 and HMM models with 8 iterations of IBM-1, 7 for HMM and 3 for IBM-4 ($1^8h^74^3$) with Null word and a maximum fertility of 3 for Chinese-English.

Choosing the number of topics is a model selection problem. We performed a ten-fold cross-validation, and a setting of three-topic is chosen for both the small and the large training data sets. The overall computation complexity of the BiTAM is linear to the number of hidden topics.

5.2 Variational Inference

Under a non-symmetric Dirichlet prior, hyperparameter α is initialized randomly; $\mathbf{B}$ (K translation lexicons) are initialized uniformly as did in IBM-1. Better initialization of $\mathbf{B}$ can help to avoid local optimal as shown in § 5.5.

With the learned $\mathbf{B}$ and α fixed, the variational parameters to be computed in Eqn. (8-10) are initialized randomly; the fixed-point iterative updates stop when the change of the likelihood is smaller than 10^{-5}. The convergent variational parameters, corresponding to the highest likelihood from 20 random restarts, are used for retrieving the word alignment for unseen document-pairs. To estimate $\mathbf{B}$, β (for BiTAM-2) and α, at most eight variational EM iterations are run on the training data. Figure 2 shows absolute $2\sim3\%$ better F-measure over iterations of variational EM using two and three topics of BiTAM-1 comparing with IBM-1.

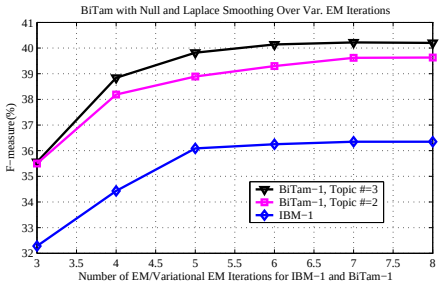


Figure 2: performances over eight Variational EM iterations of BiTAM-1 using both the "Null" word and the laplace smoothing; IBM-1 is shown over eight EM iterations for comparison.

5.3 Topic-Specific Translation Lexicons

The topic-specific lexicons B_k are smaller in size than IBM-1, and, typically, they contain topic trends. For example, in our training data, North *Korean* is usually related to *politics* and translated into "ChaoXian" (朝鲜); South *Korean* occurs more often with *economics* and is translated as "HanGuo"(韩国). BiTAMs discriminate the two by considering the topics of the context. Table 2 shows the lexicon entries for "*Korean*" learned by a 3-topic BiTAM-1. The values are relatively sharper, and each clearly favors one of the candidates. The co-occurrence count, however, only favors "HanGuo", and this can easily dominate the decisions of IBM and HMM models due to their ignorance of the topical context. Monolingual topics learned by BiTAMs are, roughly speaking, fuzzy especially when the number of topics is small. With proper filtering, we find that BiTAMs do capture some topics as illustrated in Table 3.

5.4 Evaluating Word Alignments

We evaluate word alignment accuracies in various settings. Notably, BiTAM allows to test alignments in two directions: English-to-Chinese (EC) and Chinese-to-English (CE). Additional *heuristics* are applied to further improve the accuracies. *Inter* takes the intersection of the two directions and generates high-precision alignments; the

SETTING	IBM-1	HMM	IBM-4	BITAM-1		BITAM-2		BITAM-3	
				UDA	BDA	UDA	BDA	UDA	BDA
CE (%)	36.27	43.00	45.00	40.13	48.26	40.26	48.63	40.47	49.02
EC (%)	32.94	44.26	45.96	36.52	46.61	37.35	46.30	37.54	46.62
REFINED (%)	**41.71**	44.40	48.42	**45.06**	**49.02**	**47.20**	47.61	**47.46**	48.18
UNION (%)	32.18	42.94	43.75	35.87	48.66	36.07	**48.99**	36.26	**49.35**
INTER (%)	39.86	**44.87**	**48.65**	43.65	43.85	44.91	45.18	45.13	45.48
NIST	6.458	6.822	6.926	6.937	6.954	6.904	6.976	6.967	6.962
BLEU	15.70	17.70	18.25	17.93	18.14	18.13	18.05	18.11	18.25

Table 4: Word Alignment Accuracy (F-measure) and Machine Translation Quality for BiTAM Models, comparing with IBM Models, and HMMs with a training scheme of $1^8h^74^3$ on the Treebank data listed in Table 1. For each column, the highlighted alignment (the best one under that model setting) is picked up to further evaluate the translation quality.

Union of two directions gives high-recall; *Refined* grows the intersection with the neighboring word-pairs seen in the union, and yields high-precision and high-recall alignments.

As shown in Table 4, the baseline IBM-1 gives its best performance of 36.27% in the CE direction; the UDA alignments from BiTAM-1∼3 give 40.13%, 40.26%, and 40.47%, respectively, which are significantly better than IBM-1. A close look at the three BiTAMs does not yield significant difference. BiTAM-3 is slightly better in most settings; BiTAM-1 is slightly worse than the other two, because the topics sampled at the sentence level are not very concentrated. The BDA alignments of BiTAM-1∼3 yield 48.26%, 48.63% and 49.02%, which are even better than HMM and IBM-4 — their best performances are at 44.26% and 45.96%, respectively. This is because BDA partially utilizes similar heuristics on the approximated posterior matrix $\{\varphi_{dnji}\}$ instead of direct operations on alignments of two directions in the heuristics of *Refined*. Practically, we also apply BDA together with heuristics for IBM-1, HMM and IBM-4, and the best achieved performances are at 40.56%, 46.52% and 49.18%, respectively. Overall, BiTAM models achieve performances close to or higher than HMM, using only a very simple IBM-1 style alignment model.

Similar improvements over IBM models and HMM are preserved after applying the three kinds of heuristics in the above. As expected, since BDA already encodes some heuristics, it is only slightly improved with the *Union* heuristic; UDA, similar to the viterbi style alignment in IBM and HMM, is improved better by the *Refined* heuristic.

We also test BiTAM-3 on large training data, and similar improvements are observed over those of the baseline models (see Table. 5).

5.5 Boosting BiTAM Models

The translation lexicons of $B_{f,e,k}$ are initialized uniformly in our previous experiments. Better initializations can potentially lead to better performances because it can help to avoid the undesirable local optima in variational EM iterations. We use the lexicons from IBM Model-4 to initialize $B_{f,e,k}$ to boost the BiTAM models. This is one way of applying the proposed BiTAM models into current state-of-the-art SMT systems for further improvement. The boosted alignments are denoted as BUDA and BBDA in Table. 5, corresponding to the uni-direction and bi-direction alignments, respectively. We see an improvement in alignment quality.

5.6 Evaluating Translations

To further evaluate our BiTAM models, word alignments are used in a phrase-based decoder for evaluating translation qualities. Similar to the Pharoah package (Koehn, 2004), we extract phrase-pairs directly from word alignment together with coherence constraints (Fox, 2002) to remove noisy ones. We use TIDES Eval'02 CE test set as development data to tune the decoder parameters; the Eval'03 data (919 sentences) is the unseen data. A trigram language model is built using 180 million English words. Across all the reported comparative settings, the key difference is the bilingual ngram-identity of the phrase-pair, which is collected directly from the underlying word alignment.

Shown in Table 4 are results for the small-data track; the large-data track results are in Table 5. For the small-data track, the baseline Bleu scores for IBM-1, HMM and IBM-4 are 15.70, 17.70 and 18.25, respectively. The UDA alignment of BiTAM-1 gives an improvement over the baseline IBM-1 from 15.70 to 17.93, and it is close to HMM's performance, even though BiTAM doesn't exploit any sequential structures of words. The proposed BiTAM-2 and BiTAM-3 are slightly better than BiTAM-1. Similar improvements are observed for the large-data track (see Table 5). Note that, the boosted BiTAM-3 us-

SETTING	IBM-1	HMM	IBM-4	BITAM-3			
				UDA	BDA	BUDA	BBDA
CE (%)	46.73	49.12	54.17	50.55	56.27	55.80	57.02
EC (%)	44.33	54.56	55.08	51.59	55.18	54.76	58.76
REFINED (%)	**54.64**	**56.39**	**58.47**	**56.45**	54.57	**58.26**	56.23
UNION (%)	42.47	51.59	52.67	50.23	**57.81**	56.19	**58.66**
INTER (%)	52.24	54.69	57.74	52.44	52.71	54.70	55.35
NIST	7.59	7.77	7.83	7.64	7.68	8.10	8.23
BLEU	19.19	21.99	23.18	21.20	21.43	22.97	24.07

Table 5: Evaluating Word Alignment Accuracies and Machine Translation Qualities for BiTAM Models, IBM Models, HMMs, and boosted BiTAMs using all the training data listed in Table. 1. Other experimental conditions are similar to Table. 4.

ing IBM-4 as the seed lexicon, outperform the *Refined* IBM-4: from 23.18 to 24.07 on Bleu score, and from 7.83 to 8.23 on NIST. This result suggests a straightforward way to leverage BiTAMs to improve statistical machine translations.

6 Conclusion

In this paper, we proposed novel formalism for statistical word alignment based on bilingual admixture (BiTAM) models. Three BiTAM models were proposed and evaluated on word alignment and translation qualities against state-of-the-art translation models. The proposed models significantly improve the alignment accuracy and lead to better translation qualities. Incorporation of within-sentence dependencies such as the alignment-jumps and distortions, and a better treatment of the source monolingual model worth further investigations.

References

M. J. Beal and Zoubin Ghahramani. 2002. The variational bayesian em algorithm for incomplete data: with application to scoring graphical model structures. In *Bayesian Statistics 7*.

David Blei, Andrew NG, and M.I. Jordan. 2003. Latent dirichlet allocation. In *Journal of Machine Learning Research*, volume 3, pages 1107–1135.

P.F. Brown, Stephen A. Della Pietra, Vincent. J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. In *Computational Linguistics*, volume 19(2), pages 263–331.

Marine Carpua and Dekai Wu. 2005. Evaluating the word sense disambiguation performance of statistical machine translation. In *Second International Joint Conference on Natural Language Processing (IJCNLP-2005)*.

Bonnie Dorr and Nizar Habash. 2002. Interlingua approximation: A generation-heavy approach. In *In Proceedings of Workshop on Interlingua Reliability, Fifth Conference of the Association for Machine Translation in the Americas, AMTA-2002*, Tiburon, CA.

Heidi J. Fox. 2002. Phrasal cohesion and statistical machine translation. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 304–311, Philadelphia, PA, July 6-7.

Philipp Koehn. 2004. Pharaoh: a beam search decoder for phrase-based smt. In *Proceedings of the Conference of the Association for Machine Translation in the Americans (AMTA)*.

Eugene A. Nida. 1964. *Toward a Science of Translating: With Special Reference to Principles Involved in Bible Translating*. Leiden, Netherlands: E.J. Brill.

Eric Nyberg and Truko Mitamura. 1992. The kant system: Fast, accurate, high-quality translation in practical domains. In *Proceedings of COLING-92*.

Franz J. Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, and Dragomir Radev. 2004. A smorgasbord of features for statistical machine translation. In *HLT/NAACL: Human Language Technology Conference*, volume 1:29, pages 161–168.

Franz J. Och. 1999. An efficient method for determining bilingal word classes. In *Ninth Conf. of the Europ. Chapter of the Association for Computational Linguistics (EACL'99)*, pages 71–76.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proc. of the 40th Annual Conf. of the Association for Computational Linguistics (ACL 02)*, pages 311–318, Philadelphia, PA, July.

J. Pritchard, M. Stephens, and P. Donnell. 2000. Inference of population structure using multilocus genotype data. In *Genetics*, volume 155, pages 945–959.

K. Sjölander, K. Karplus, M. Brown, R. Hughey, A. Krogh, I.S. Mian, and D. Haussler. 1996. Dirichlet mixtures: A method for improving detection of weak but significant protein sequence homology. *Computer Applications in the Biosciences*, 12.

S. Vogel, Hermann Ney, and C. Tillmann. 1996. Hmm based word alignment in statistical machine translation. In *Proc. The 16th Int. Conf. on Computational Lingustics, (Coling'96)*, pages 836–841, Copenhagen, Denmark.

Yeyi Wang, John Lafferty, and Alex Waibel. 1996. Word clustering with parallel spoken language corpora. In *proceedings of the 4th International Conference on Spoken Language Processing (ICSLP'96)*, pages 2364–2367.

K. Yamada and Kevin. Knight. 2001. Syntax-based statistical translation model. In *Proceedings of the Conference of the Association for Computational Linguistics (ACL-2001)*.

Bing Zhao and Stephan Vogel. 2002. Adaptive parallel sentences mining from web bilingual news collection. In *The 2002 IEEE International Conference on Data Mining*.

Bing Zhao, Eric P. Xing, and Alex Waibel. 2005. Bilingual word spectral clustering for statistical machine translation. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pages 25–32, Ann Arbor, Michigan, June. Association for Computational Linguistics.

An HMM-Based Approach to Automatic Phrasing for Mandarin Text-to-Speech Synthesis

Jing Zhu
Department of Electronic Engineering
Shanghai Jiao Tong University
zhujing@sjtu.edu.cn

Jian-Hua Li
Department of Electronic Engineering
Shanghai Jiao Tong University
lijh888@sjtu.edu.cn

Abstract

Automatic phrasing is essential to Mandarin text-to-speech synthesis. We select word format as target linguistic feature and propose an HMM-based approach to this issue. Then we define four states of prosodic positions for each word when employing a discrete hidden Markov model. The approach achieves high accuracy of roughly 82%, which is very close to that from manual labeling. Our experimental results also demonstrate that this approach has advantages over those part-of-speech-based ones.

1 Introduction

Owing to the limitation of vital capacity and contextual information, breaks or pauses are always an important ingredient of human speech. They play a great role in signaling structural boundaries. Similarly, in the area of text-to-speech (TTS) synthesis, assigning breaks is very crucial to naturalness and intelligibility, particularly in long sentences.

The challenge in achieving naturalness mainly results from prosody generation in TTS synthesis. Generally speaking, prosody deals with phrasing, loudness, duration and speech intonation. Among these prosodic features, phrasing divides utterances into meaningful chunks of information, called hierarchic breaks. However, there is no unique solution to prosodic phrasing in most cases. Different solution in phrasing can result in different meaning that a listener could perceive. Considering its importance, recent TTS research has focused on automatic prediction of prosodic phrase based on the part-of-speech (POS) feature or syntactic structure(Black and Taylor, 1994; Klatt, 1987; Wightman, 1992; Hirschberg 1996; Wang, 1995; Taylor and Black, 1998).

To our understanding, POS is a grammar-based structure that can be extracted from text. There is no explicit relationship between POS and the prosodic structure. At least, in Mandarin speech synthesis, we cannot derive the prosodic structure from POS sequence directly. By contrast, a word carries rich information related to phonetic feature. For example, in Mandarin, a word can reveal many phonetic features such as pronunciation, syllable number, stress pattern, tone, light tone (if available) and retroflexion (if available) etc. So we begin to explore the role of word in predicting prosodic phrase and propose a word-based statistical method for prosodic-phrase grouping. This method chooses Hidden Markov Model (HMM) as the training and predicting model.

2 Related Work

Automatic prediction of prosodic phrase is a complex task. There are two reasons for this conclusion. One is that there is no explicit relationship between text and phonetic features. The other lies in the ambiguity of word segmentation, POS tagging and parsing in the Chinese natural language processing. As a result, the input information for the prediction of prosodic phrase is quite “noisy”. We can find that most of published methods, including (Chen et al., 1996; Chen et al., 2000; Chou et al., 1996; Chou et al., 1997; Gu et al., 2000; Hu et al., 2000; Lv et al., 2001; Qian et al., 2001; Ying and Shi, 2001) do not make use of high-level syntactic features due to two reasons. Firstly, it is very challenging to parse Chinese sentence because no grammar is formal enough to be applied to Chinese parsing. In addition, lack of

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 977–982,
Sydney, July 2006.

morphologies also causes many problems in parsing. Secondly, the syntactic structure is not isomorphic to the prosodic phrase structure. Prosodic phrasing remains an open task in the Chinese speech generation. In summary, all the known methods depend on POS features more or less.

3 Word-based Prediction

As noted previously, the prosodic phrasing is associated with words to some extent in Mandarin TTS synthesis. We observe that some function words (such as "的") never occur in phrase-initial position. Some prepositions seldom act as phrase-finals. These observations lead to investigating the role of words in prediction of prosodic phrase. In addition, large-scale training data is readily available, which enables us to apply data-driven models more conveniently than before.

3.1 The Model

The sentence length in real text can vary significantly. A model with a fixed-dimension input does not fit the issue in prosodic breaking. Alternatively, the breaking prediction can be converted into an optimization problem that allows us to adopt the hidden Markov model (HMM).

An HMM for discrete symbol observations is characterized by the following:

- the state set $Q=\{q_i\}$, where $1 \le i \le N$, N is the number of states
- the number of distinct observation symbol per state M
- the state-transition probability distribution $A=\{a_{ij}\}$, where

$$a_{ij}=P[q_{t+1}=j|q_t=i],\ 1 \le i,j \le N$$

- the observation symbol probability distribution $B=\{b_j(k)\}$, where

$$b_j(k) = P[o_t = v_k \mid q_t = j],$$

$1 \le i,j \le N$

- the initial state distribution $\pi=\{\pi_i\}$, where $\pi_i = P[o_t=v_k|q_t=j]$, $1 \le i,j \le M$.

The complete parameter set of the model is denoted as a compact notation $\lambda=(A,B,\pi)$.

Here, we define our prosodic positions for a word to apply the HMM as follows.

0 phrase-initial
1 phrase-medial
2 phrase-final
3 separate

This means that Q can be represented as $Q=\{0,1,2,3\}$, corresponding to the four prosodic positions. The word itself is defined as a discrete symbol observation.

3.2 The Corpus

The text corpus is divided into two parts. One serves as training data. This part contains 17,535 sentences, among which, 9,535 sentences have corresponding utterances. The other is a test set, which includes 1,174 sentences selected from the Chinese *People's Daily*. The sentence length, namely the number of words in a sentence varies from 1 to 30. The distribution of word length, phrase length and sentence length(all in character number) is shown in Figure 1.

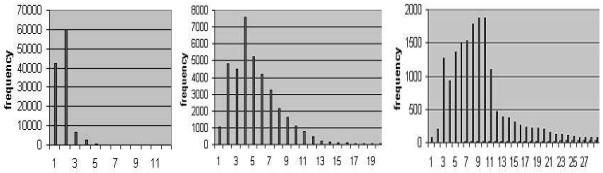


Figure 1. Statistical results from the corpus

In a real text, there may exist words that are difficult to enumerate in the system lexicon, called "non-standard" words (NSW). Examples of NSW are proper names, digit strings, derivative words by adding prefix or suffix.

Proper names include person name, place name, institution name and abbreviations, etc. Alternatively, some characters are usually viewed as prefix and suffix in Chinese text. For instance, the character 伪 (pseudo-) always serves as a prefix, while another character 般 (-like) serves as a suffix. There are 130 analogous Chinese characters have been collected roundly. A word segmentation module is designed to identify these non-standard words.

3.3 Parameter estimation

Parameter estimation of the model can be treated as an optimization problem. The parametric methods will be optimal if distribution derived from the training data is in the class of distributions being considered. But there is no

known way so far for maximizing the probability of the observation sequence in a closed form. In the present approach, a straightforward, reasonable yet, method to re-estimate parameters of the HMM is applied. Firstly, statistics for the occurring times of word, prosodic position, prosodic-position pair are conducted. Secondly, the simple ratio of occurring times is used to calculate the probability distribution. The following expressions are used to implement calculations,

State probability distribution

$$P[q_i] \approx \frac{F_i}{\sum_{j=1}^{N} F_j} \quad , \quad 1 \le i \le N$$

F_i is the occurring times of state q_i

the state-transition probability distribution $A = \{a_{ij}\}$,

$a_{ij} \approx \frac{F_{ij}}{F_i}$, $1 \le i, j \le N$, F_{ij} is the occurring times of state pair (q_i,q_j).

Observation probability *distribution* $B = \{b_j(k)\}$,

$$b_j(k) \approx \frac{F(q = j, o = v_k)}{F_j}$$

$$b_j(k) \propto \frac{F(q = j, o = v_k)}{P[q_j]}$$

where $F(q = j, o = v_k) = \sum_t F(q_t = j, o = v_k)$ is the concurring times of state q_j and observation v_k.

With respect to the proper names, all the person names are dealt with identically. This is based on an assumption that the proper names of individual category have the same usage.

3.4 Parameter adjustment

Note that the training corpus is discrete, finite set. The parameter set resulting from the limited samples cannot converge to the "true" values with probability. In particular, some words may not be included in the corpus. In this case, the above expressions for training may result in zero valued observation-probability. This, of course, is unexpected. The parameters should be adjusted after the automatic model training. The way is to use a sufficiently small positive constant ε to represent the zero valued observation-probabilities.

3.5 The search procedure

In this stage, an optimal state sequence that explains the given observations by the model is searched. That is to say, for the input sentence, an optimal prosodic-position sequence is predicted with the HHM. Instead of using the popular *Viterbi* algorithm, which is asymptotically optimal, we apply the Forward-Backward procedure to conduct searching.

Backward and forward search

All the definitions described in (Rabiner, 1999) are followed in the present approach.

The forward procedure

forward variable: $\alpha_t(i) = P(o_1 o_2 \cdots o_t, q_t = i \mid \lambda)$

initialization: $\alpha_1(i) = \pi_i b_i(o_1), \quad 1 \le i \le N.$

induction:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(o_{t+1}), \quad 1 \le t \le T-1, \quad 1 \le j \le N \ .$$

termination: $P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$

where T is the number of observations.

The backward procedure

backward variable:

$\beta_t(i) = P(o_{t+1} o_{t+2} \cdots o_T \mid q_t = i, \lambda)$

initialization $\beta_T(i) = 1, \quad 1 \le i \le N$

induction:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j) \quad t = T-1, T-2, \ldots 1, \quad 1 \le i \le N$$

The "optimal" state sequence

posteriori probability variable: $\gamma_t(i)$, this is the probability of being in state i at time t given the observation sequence O and the model λ. It can be expressed as follows:

$$\gamma_t(i) = P(q_t = i \mid O, \lambda) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)}$$

most likely state q_t^* at time t:

$$q_t^* = \arg \max_{1 \le i \le N} [\gamma_t(i)] \quad 1 \le t \le T \ .$$

Here comes a question. It is, whether the optimal state sequence means the optimal path.

Search based on dynamic programming

The preceding search procedure targets the optimal state sequence satisfying one criterion. But it does not reflect the probability of occurrence of sequences of states. This issue is explored based on a dynamic programming (DP) like approach, as described below.

For convenience, we illustrate the problem as shown in Figure 2.

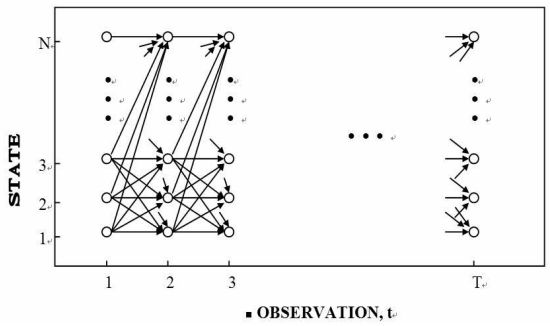


Figure 2. Illustration of search procedure in trellis (quoted from [Rabiner, 1999])

From Figure 2, it can be seen that the transition from state *i* to state *j* only occurs in the two consecutive stages, namely time synchronous. Totally, there are *T* stages, N^2T arcs. Therefore, the optimal-path issue is a multi-stage optimization problem, which is similar to the DP problem. The slight difference lies in that a node in the conventional DP problem does not contain any additional attribute, while a node in HMM carries the attribute of observation probability distribution. Considering this difference, we modify the conventional DP approach in the following way.

In the trellis above, we add a virtual node (state), where the start node q_s corresponding to time 0 before time 1. All the transitions from q_s to nodes in the first stage (time 1) equal to *1/N*. Furthermore, all the observation probability distributions equal to *1/M*. Denoting the optimal path from q_s to the node q_i of time *t* as *path(t,i)*, *path*(*t,i*) is a set of sequential states. Accordingly, we denote the score of *path(t,i)* as *s(t,i)*. Then, *s(t,i)* is associated with the state-transition probability distribution and observation probability distribution. We describe the induction process as follows.

initialization:

$$s(0,i) = \frac{1}{M \times N}, \quad 1 \le i \le N$$
$$path(0,i) = \{q_s\}.$$

induction:

given

$$j,\ s(t,j) = \max_{1 \le i \le N} [s(t-1,i) \times b_i(o_t) \times a_{ij}],\ 1 \le t \le T \quad ,$$

denotes

$$k = \arg \max_{1 \le i \le N} [s(t-1,i) \times b_i(o_t) \times a_{ij}] \quad ,$$ then

$path(t,j) = path(t-1,k) \cup \{k\}$.

termination:

at time T, $k = \arg \max_{1 \le i \le N} s(T,i)$.

then $path(T,k) - \{q_s\}$ *is the optimal path.*

Basically, the main idea of our approach lies in that if the final optimal path passes a node *j* at time *t*, it passes all the nodes in *path*(*t,j*) sequentially. This idea is similar to the forward procedure of DP. We can begin with the termination *T* and derive an alternative approach. As for time complexity, the above trellis can be viewed as a special DAG. The state transition from time *t* to time *t+1* requires $2N^2$ calculations, resulting in the time complexity $O(TN^2)$.

Intuitively, the optimal path differs from the optimal state sequence generated by the Forward-Backward procedure. The underlying idea of Forward-Backward procedure is that the target state sequence can explain the observations optimally. To support our claim, we can give a simple example ($T=2$, $N=2$, $\pi = [0.5, 0.5]^T$) as follows:

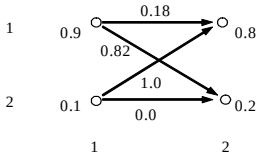


Figure 3. Optimal state sequence vs. optimal path

Apparently, the optimal state sequence is (1,1), while the optimal path is {1,2}.

4 Experimental Results

Before reporting the experimental results, we first define the criterion of evaluation and the related issues.

4.1 The evaluation method

After analyzing the existing evaluation methods, we feel that the method proposed in (Taylor and Black, 1998) is appropriate for our application. By employing this method, we can examine each word pair in the test set. If the algorithm generated break fully matches the manually labeled break, it marks correct. Similarly, if there is no labeled break and the algorithm does not place a break, it also marks correct. Otherwise, an error arises. To emphasize the effectiveness of break prediction, we define the adjusted score, S_a, as follows.

$$S_a = \frac{S - B}{1 - B}$$

where

S is the ratio of the number of correct word pairs to the total number of word pairs;

B is the ratio of non-breaks to the number of word-pairs.

4.2 The test corpora

From the perspective of perception, multiple predictions of prosodic phrasing may be acceptable in many cases. At the labeling stage, three experts (*E1, E2, E3*) were requested to label 1,174 sentences independently. Experts first read the sentences silently. Then, they marked the breaks in sentences independently. Table 1 and 2 show their labeling differences in terms of *S* and S_a, respectively.

	E1	*E2*	*E3*
E1	*1.00*	*0.87*	*0.87*
E2	*0.87*	*1.00*	*0.86*
E3	*0.87*	*0.86*	*1.00*

Table 1. Three experts' matching scores

	E1	*E2*	*E3*
E1	*1.00*	*0.74*	*0.67*
E2	*0.74*	*1.00*	*0.66*
E3	*0.72*	*0.72*	*1.00*

Table 2. Three experts' adjusted matching scores

Table 1 indicates that any two can achieve a consistency of roughly 87% among three experts.

4.3 The results

To evaluate the approaches mentioned above, we conducted a series of experiments. In all our experiments, we assume that no breaking is necessary for those sentences that are shorter than the average phrase length and remove them in the statistic computation. For the approaches based on HMM path, we further define that the initial and final words of a sentence can only assume two state values, namely, (*phrase initial, separate*) and (*phrase final, separate*), respectively. With this definition, we modify the approach *HMM-Path* to *HMM-Path-I*. Alternatively, to investigate acceptance, we also calculate the matching score between the approaches and *any* expert (We assume the prediction is acceptable if the predicted phrase sequence matches any of three phrase sequences labeled by the experts). By employing the preceding criterion, we achieve the results as shown in Table 3 and 4.

	E1	*E2*	*E3*	*Any*
HMM	*0.78*	*0.77*	*0.77*	*0.85*
HMM-path	*0.79*	*0.77*	*0.78*	*0.85*
HMM-path-I	*0.82*	*0.80*	*0.82*	*0.88*

Table 3. Matching scores of 3 approaches

	E1	*E2*	*E3*	*Any*
HMM	*0.55*	*0.53*	*0.44*	*0.66*
HMM-path	*0.52*	*0.54*	*0.44*	*0.67*
HMM-path-I	*0.62*	*0.60*	*0.55*	*0.74*

Table 4. Adjusted matching scores of 3 approaches

A sentence consumes less than 0.3 ms on average for all the evaluated methods. So they are all computationally efficient. Alternatively, we compared the HMM-based approach base on word format and some POS-based ones on the same training set and test set. Overall, *HMM-path*-I can achieve high accuracy by about 10%.

5 Conclusions/Discussions

We described an approach to automatic prosodic phrasing for Mandarin TTS synthesis based on word format and HMM and its variants. We also evaluated these methods through experiments and demonstrated promising results. According to the experimental results, we can conclude that word-based prediction is an effective approach and has advantages over the POS-based ones. It confirms that the syllable number of a word has substantial impact on prosodic phrasing.

References

Black, A.W., Taylor, P., 1994. "Assigning intonational elements and prosodic phrasing for

English speech synthesis from high level linguistic input", *Proc. ICSLIP*

Chen, S.H., Hwang, S.H., Wang, Y.R., 1998. "An RNN-based prosodic information synthesizer for Mandarin text-to-speech", *IEEE Trans. Speech Audio Processing*, 6: 226-239.

Chen, Y.Q., Gao, W., , Zhu, T.S., Ma, J.Y., 2000. "Multi-strategy data mining on Mandarin prosodic patterns", *Proc. ISCLIP*

Chou, F.C., Tseng, C.Y., Lee, L.S. 1996. "Automatic generation of prosodic structure for high quality Mandarin speech synthesis", *Proc. ICSLP*

Chou, F.C, Tseng, C.Y, Chen, K.J., Lee, L.S, 1997. "A Chinese text-to-speech system based on part-of-speech analysis, prosodic modeling and non-uniform units", ICASSP'97

Klatt, D.H., 1987, "Review of text-to-speech conversion for English", *J. Acoust. Soc. Am.*, 182: 737-79

Gu, Z.L, Mori, H., Kasuya, H. 2000. "Prosodic variation of focused syllables of disyllabic word in Mandarin Chinese", *Proc. ICSLP*,

Hirschberg, J., 1996. "Training intonational phrasing rules automatically for English and Spanish text-to-speech", *Speech Communication*, 18:281-290

Hu, Y., Liu, Q.F., Wang, R.H., 2000, "Prosody generation in Chinese synthesis using the template of quantified prosodic unit and base intonation contour", *Proc. ICSLIP*

Lu, S.N., He, L., Yang, Y.F., Cao, J.F., 2000, "Prosodic control in Chinese TTS system", *Proc. ICSLP*,

Lv, X., Zhao, T.J., Liu, Z.Y., Yang M.Y., 2001, "Automatic detection of prosody phrase boundaries for text-to-speech system", *Proc. IWPT*

Qian, Y., Chu, M., Peng, H., 2001, "Segmenting unrestricted Chinese text into prosodic words instead of lexical words", *Proc. ICASSP*.

Rabiner, L., 1999, *Fundamentals of Speech Recognition*, pp.336, Prentice-Hall and Tsinghua Univ. Press, Beijing

Taylor P., Black A.W., 1998, "Assigning phrase breaks from part-of-speech sequences", *Computer Speech and Language*, 12: 99-117,

Wang, M.Q., Hirschberg, J., 1995, "Automatic classification of intonational phrase boundaries", *Computer Speech and Language*, pp.175-196, Vol. 6,

Wightman, C.W., 1992, "Segmental durations in the vicinity of prosodic phrase boundaries", *J. Acoust. Soc. Am.*, 91:1707-1717

Ying, Z.W., Shi, X.H., 2001, "An RNN-based algorithm to detect prosodic phrase for Chinese TTS", *Proc. ICASSP*

Author Index