Learning Discourse Relations with Active Data Selection

Tadashi Nomoto*
National Institute of Japanese Literature
1-16-10 Yutaka Shinagawa
Tokyo 142-8585 Japan
nomoto@nijl.ac.jp

Yuji Matsumoto
Nara Institute of Science and Technology
8916-5 Takayama Ikoma Nara
630-0101 Japan
matsu@is.aist-nara.ac.jp

Abstract

The paper presents a new approach to identifying discourse relations, which makes use of a particular sampling method called *committee-based sampling* (CBS). In the committee-based sampling, multiple learning models are generated to measure the utility of an input example in classification; if it is judged as not useful, then the example will be ignored. The method has the effect of reducing the amount of data required for training. In the paper, we extend CBS for decision tree classifiers. With an additional extension called *error feedback*, it is found that the method achieves an increased accuracy as well as a substantial reduction in the amount of data for training classifiers.

1 Introduction

The success of corpus-based approaches to discourse ultimately depends on whether one is able to acquire a large volume of data annotated for discourse-level information. However, to acquire merely a few hundred texts annotated for discourse information is often impossible due to the enormity of the human labor required.

This paper presents a novel method for reducing the amount of data for training a decision tree classifier, while not compromising the accuracy. While there has been some work exploring the use of machine leaning techniques for discourse and dialogue (Marcu, 1997; Samuel et al., 1998), to our knowledge, no computational research on discourse or dialogue so far has addressed the problem of reducing or minimizing the amount of data for training a learning algorithm.

A particular method proposed here is built on the *committee-based sampling*, initially proposed for probabilistic classifiers by Dagan and Engelson (1995), where an example is selected from the corpus according to its utility in improving statistics. We extend the method for decision tree classifiers using a statistical technique called *bootstrapping* (Cohen, 1995). With an additional extension, which we call *error feedback*, it is found that the method achieves an increased accuracy as well as a significant reduction of training data. The method proposed here should be of use in domains other than discourse, where a decision tree strategy is found applicable.

2 Tagging a corpus with discourse relations

In tagging a corpus, we adopted Ichikawa (1990)'s scheme for organizing discourse relations (Table 1). The advantage of Ichikawa (1990)'s scheme is that it directly associates discourse relations with explicit surface cues (eg. sentential connectives), so it is possible for the coder to determine a discourse relation by figuring a most natural cue that goes with a sentence he/she is working on. Another feature is that, unlike Rhetorical Structure Theory (Mann and Thompson, 1987), the scheme assumes a discourse relation to be a local one, which is defined strictly over two consecutive sentences.[1] We expected that these features would make a tagging task less laborious for a human coder than it would be with RST. Further, our earlier study indicated a very low agreement rate with

* The work reported here was conducted while the first author was with Advanced Research Lab., Hitachi Ltd, 2520 Hatoyama Saitama 350-0395 Japan.

[1] This does not mean to say that *all* of the discourse relations are local. There could be some relations that involve sentences separated far apart. However we did not consider non-local relations, as our preliminary study found that they are rarely agreed upon by coders.

Table 1: Ichikawa (1990)'s taxonomy of discourse relations. The first column indicates major classes and the second subclasses. The third column lists some examples associated with each subclass. Note that the EXPANDING subclass has no examples in it. This is because no explicit cue is used to mark the relationship.

LOGICAL	CONSEQUENTIAL	dakara *therefore*, shitagatte *thus*
	ANTITHESIS	shikashi *but*, daga *but*
SEQUENCE	ADDITIVE	soshite *and*, tsuigi-ni *next*
	CONTRAST	ippô *in contrast*, soretomo *or*
	INITIATION	tokorode *to change the subject*, sonouchi *in the meantime*
ELABORATION	APPOSITIVE	tatoeba *for example*, yôsuruni *in other words*
	COMPLEMENTARY	nazenara *because*, chinamini *incidentally*
	EXPANDING	NONE

RST ($\kappa = 0.43$; three coders); especially for a casual coder, RST turned out to be a quite difficult guideline to follow.

In Ichikawa (1990), discourse relations are organized into three major classes: the first class includes logical (or strongly semantic) relationships where one sentence is a logical consequence or contradiction of another; the second class consists of sequential relationships where two semantically independent sentences are juxtaposed; the third class includes elaboration-type relationships where one of the sentences is semantically subordinate to the other.

In constructing a tagged corpus, we asked coders not to identify abstract discourse relations such as LOGICAL, SEQUENCE and ELABORATION, but to choose from a list of predetermined connective expressions. We expected that the coder would be able to identify a discourse relation with far less effort when working with explicit cues than when working with abstract concepts of discourse relations. Moreover, since 93% of sentences considered for labeling in the corpus did not contain predetermined relation cues, the annotation task was in effect one of guessing a possible connective cue that may go with a sentence. The advantage of using explicit cues to identify discourse relations is that even if one has little or no background in linguistics, he or she may be able to assign a discourse relation to a sentence by just asking him/herself whether the associated cue fits well with the sentence. In addition, in order to make the usage of cues clear and unambiguous, the annotation instruction carried a set of examples for each of the cues. Further, we developed an emacs-based software aid which guides the coder to work through a corpus and also is capable of prohibiting the coder from making moves inconsistent with the tagging instruction.

As it turned out, however, Ichikawa's scheme, using subclass relation types, did not improve agreement ($\kappa = 0.33$, three coders). So, we modified the relation taxonomy so that it contains just two major classes, SEQUENCE and ELABORATION, (LOGICAL relationships being subsumed under the SEQUENCE class) and assumed that a lexical cue marks a major class to which it belongs. The modification successfully raised the κ score to 0.70. Collapsing LOGICAL and SEQUENCE classes may be justified by noting that both types of relationships have to do with relating two semantically independent sentences, a property not shared by relations of the elaboration type.

3 Learning with Active Data Selection

3.1 Committee-based Sampling

In the committee-based sampling method (CBS, henceforth) (Dagan and Engelson, 1995; Engelson and Dagan, 1996), a training example is selected from a corpus according to its usefulness; a preferred example is one whose addition to the training corpus improves the current estimate of a model parameter which is relevant to classification and also affects a large proportion of examples. CBS tries to identify such an example by randomly generating multiple models (*committee members*) based on posterior dis-

tributions of model parameters and measuring how much the member models disagree in classifying the example. The rationale for this is: disagreement among models over the class of an example would suggest that the example affects some parameters sensitive to classification, and furthermore estimates of affected parameters are far from their true values. Since models are generated randomly from posterior distributions of model parameters, their disagreement on an example's class implies a large variance in estimates of parameters, which in turn indicates that the statistics of parameters involved are insufficient and hence its inclusion in the training corpus (so as to improve the statistics of relevant parameters).

For each example it encounters, CBS goes through the following steps to decide whether to select the example for labeling.

1. Draw k models (*committee members*) randomly from the probability distribution $P(M \mid S)$ of models M given the statistics S of a training corpus.
2. Classify an input example by each of the committee members and measure how much they disagree on classification.
3. Make a biased random decision as to whether or not to select the example for labeling. This would make a highly disagreed-upon example more likely to be selected.

As an illustration of how this might work, consider a problem of tagging words with parts of speech, using a Hidden Markov Model (HMM). A (bigram) HMM tagger is typically given as:

$$T(w_1 \ldots w_n) = \underset{t_1 \ldots t_n}{\operatorname{argmax}} \prod_{i=1}^{n} P(w_i \mid t_i) P(t_{i+1} \mid t_i)$$

where $w_1 \ldots w_n$ is a sequence of input words, and $t_1 \ldots t_n$ is a sequence of tags. For a sequence of input words $w_1 \ldots w_n$, a sequence of corresponding tags $T(w_1 \ldots w_n)$ is one that maximizes the probability of reaching t_n from t_1 via t_i $(1 < i < n)$ and generating $w_1 \ldots w_n$ along with it. Probabilities $P(w_i \mid t_i)$ and $P(t_{i+1} \mid t_i)$ are called *model parameters* of an HMM tagger. In Dagan and Engelson (1995), $P(M \mid S)$ is given as the posterior multinomial distribution $P(\alpha_1 = a_1, \ldots, \alpha_n = a_n \mid S)$, where α_i is a model parameter and a_i represents one of the possible values. $P(\alpha_1 = a_1, \ldots, \alpha_n = a_n \mid S)$ represents the proportion of the times that each parameter α_i takes a_i, given the statistics S derived from a corpus. (Note that $\sum_i^n P(\alpha_i = a_i \mid S) = 1$.) For instance, consider a task of randomly drawing a word with replacement from a corpus consisting of 100 different words $(w_1, \ldots, w_{100})$. After 10 trials, you might have outcomes like $w_1 = 3, w_2 = 1, \ldots, w_{55} = 2, \ldots, w_{71} = 3, \ldots, w_{76} = 1, \ldots, w_{100} = 0$: *i.e.*, w_1 was drawn three times, w_2 was drawn once, w_{55} was drawn twice, etc. If you try another 10 times, you might get different results. A multinomial distribution tells you how likely you get a particular sequence of word occurrences. Dagan and Engelson (1995)'s idea is to assume the distribution $P(\alpha_1 = a_1, \ldots, \alpha_n = a_n \mid S)$ as a set of binomial distributions, each corresponding to one of its parameters. An arbitrary HMM model is then constructed by randomly drawing a value a_i from a binomial distribution for a parameter α_i, which is approximated by a normal distribution. Given k such models (*committee members*) from the multinomial distribution, we ask each of them to classify an input example. We decide whether to select the example for labeling based on how much the committee members disagree in classifying that example. Dagan and Engelson (1995) introduces the notion of *vote entropy* to quantify disagreements among members. Though one could use the *kappa* statistic (Siegel and Castellan, 1988) or other disagreement measures such as the α statistic (Krippendorff, 1980) instead of the vote entropy, in our implementation of CBS, we decided to use the vote entropy, for the lack of reason to choose one statistic over another. A precise formulation of the vote entropy is as follows:

$$V(e) = -\sum_{c} \frac{V(c,e)}{k} \log \frac{V(c,e)}{k}$$

Here e is an input example and c denotes a class. $V(c,e)$ is the number of votes for c. k is the number of committee members. A selection function is given in probabilistic terms,

based on $V(e)$.

$$P_{select}(e) = \frac{g}{\log k} V(e)$$

g here is called the *entropy gain* and is used to determine the number of times an example is selected; a greater g would increase the number of examples selected for tagging. Engelson and Dagan (1996) investigated several plausible approaches to the selection function but were unable to find significant differences among them.

At the beginning of the section, we mentioned some properties of 'useful' examples. A useful example is one which contributes to reducing variance in parameter values and also affects classification. By randomly generating multiple models and measuring a disagreement among them, one would be able to tell whether an example is useful in the sense above; if there were a large disagreement, then one would know that the example is relevant to classification and also is associated with parameters with a large variance and thus with insufficient statistics.

In the following section, we investigate how we might extend CBS for use in decision tree classifiers.

3.2 Decision Tree Classifiers

Since it is difficult, if not impossible, to express the model distribution of decision tree classifiers in terms of the multinomial distribution, we turn to the bootstrap sampling method to obtain $P(M \mid S)$. The bootstrap sampling method provides a way for artificially establishing a sampling distribution for a statistic, when the distribution is not known (Cohen, 1995). For us, a relevant statistic would be the posterior probability that a given decision tree may occur, given the training corpus.

Bootstrap Sampling Procedure
Repeat $i = 1 \ldots K$ times:

1. Draw a bootstrap pseudosample S_i^* of size N from S by sampling with replacement as follows:
   Repeat N times: select a member of S at random and add it to S_i^*.
2. Build a decision tree model M from S_i^*. Add M to S_B.

S is a small set of samples drawn from the tagged corpus. Repeating the procedure 100 times would give 100 decision tree models, each corresponding to some S_i^* derived from the sample set S. Note that the bootstrap procedure allows a datum in the original sample to be selected more than once.

Given a sampling distribution of decision tree models, a committee can be formed by randomly selecting k models from S_B. Of course, there are some other approaches to constructing a committee for decision tree classifiers (Dietterich, 1998). One such, known as *randomization*, is to use a single decision tree and randomly choose a path at each attribute test. Repeating the process k times for each input example produces k models.

3.2.1 Features

In the following, we describe a set of features used to characterize a sentence. As a convention, we refer to a current sentence as 'B' and the preceding sentence as 'A'.

<LocSen> defines the location of a sentence by:

$$\frac{\#S(X)}{\#S(Last\_Sentence)}$$

'$\#S(X)$' denotes an ordinal number indicating the position of a sentence X in a text, *i.e.*, $\#S(k\text{th\_sentence}) = k$, $(k \geq 0)$. 'Last_Sentence' refers to the last sentence in a text. **LocSen** takes a continuous value between 0 and 1. A text-initial sentence takes 0, and a text-final sentence 1.

<LocPar> is defined similarly to **DistPar**. It records information on the location of a paragraph in which a sentence X occurs.

$$\frac{\#Par(X)}{\#Last\_Paragraph}$$

'$\#Par(X)$' denotes an ordinal number indicating the position of a paragraph containing X. '#Last_Paragraph' is the position of the last paragraph in a text, represented by the ordinal number.

<LocWithinPar> records information on the location of a sentence X within a paragraph in which it appears.

$$\frac{\#S(X) - \#S(Par\_Init\_Sen)}{Length(Par(X))}$$

'Par_Init_Sen' refers to the initial sentence of a paragraph in which X occurs, 'Length(Par(X))' denotes the number of sentences that occur in that paragraph. `LocWithinPar` takes continuous values ranging from 0 to 1. A paragraph initial sentence would have 0 and a paragraph final sentence 1.

`<LenText>` the length of a text, measured in Japanese characters.

`<LenSenA>` the length of A in Japanese characters.

`<LenSenB>` the length of B in Japanese characters.

`<Sim>` encodes the lexical similarity between A and B, based on an information-retrieval measure known as $tf \cdot idf$ (Salton and McGill, 1983).[2] One important feature here is that we defined similarity based on (Japanese) characters rather than on words: in practice, we broke up nominals from relevant sentences into simple alphabetical characters (including graphemes) and used them to measure similarity between the sentences. (Thus in our setup x_i in footnote 2 corresponds to one character, and not to one whole word.) We did this to deal with abbreviations and rewordings, which we found quite frequent in the corpus.

`<Cue>` takes a discrete value 'y' or 'n'. The cue feature is intended to exploit surface cues most relevant for distinguishing between the SEQUENCE and ELABORATION relations. The feature takes 'y' if a sentence contains one or more cues relevant to distinguishing between the two relation types. We considered up to 5 word n-grams found in the training corpus. Out of these, those whose $INFO_X$ values are below a particular threshold are included in the set of cues.[3] And if a sentence contains one of the cues in the set, it is marked 'y', and 'n' otherwise. The cutoff is determined in such a way as to minimize $INFO_{cue}(T)$, where T is a set of sentences (represented with features) in the training corpus. We had the total of 90 cue expressions. Note that using a single binary feature for cues alleviates the data sparseness problem; though some of the cues may have low frequencies, they will be aggregated to form a single cue category with a sufficient number of instances. In the training corpus, which contained 5221 sentences, 1914 sentences are marked 'y' and 3307 are marked 'n' with the cutoff at 0.85, which is found to minimize the entropy of the distribution of relation types. It is interesting to note that the entropy strategy was able to pick up cues which could be linguistically motivated (Table 2). In contrast to Samuel et al. (1998), we did not consider relation cues reported in the linguistics literature, since they would be useless unless they contribute to reducing the cue entropy. They may be linguistically 'right' cues, but their utility in the machine learning context is not known.

`<PrevRel>` makes available information about a relation type of the preceding sentence. It has two values, ELA for the elaboration relation, and SEQ for the sequence relation.

In the Japanese linguistics literature, there is a popular theory that sentence endings are relevant for identifying semantic relations among

[2] For a word j in a sentence S_i ($j \in S_i$), its *weight* w_{ij} is defined by:

$$w_{ij} = tf_{ij} \cdot \log \frac{N}{df_j}$$

df_j is the number of sentences in the text which have an occurrence of a word j. N is the total number of sentences in the text. The $tf \cdot idf$ metric has the property of favoring high frequency words with local distribution. For a pair of sentences $\vec{X} = (x_1, \ldots)$ and $\vec{Y} = (y_1, \ldots)$, where x and y are words, we define the lexical similarity between $\vec{X}$ and $\vec{Y}$ by:

$$SIM(\vec{X}, \vec{Y}) = \frac{2 \sum_{i=1}^{t} w(x_i) w(y_i)}{\sum_{i=1}^{t} w(x_i) + \sum_{i=1}^{t} w(y_i)}$$

where $w(x_i)$ represents a *tf·idf* weight assigned to the term x_i. The measure is known as the *Dice coefficient* (Salton and McGill, 1983)

[3] $INFO_X(T)$ measures the entropy of the distribution of classes in a set T with respect to a feature X. We define $INFO_X$ just as given in Quinlan (1993):

$$INFO_X(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \times INFO(T_i)$$

T_i represents a partition of T corresponding to one of the values for X. $INFO(T)$ is defined as follows:

$$INFO(T) = -\sum_{j=1}^{k} \frac{freq(C_j, T)}{|T|} \times \log_2 \frac{freq(C_j, T)}{|T|}$$

$freq(C, T)$ is the number of cases from class C in a set T of cases.

Table 2: Some of the 'linguistically interesting' cues identified by the entropy strategy.

mata *on the other hand,* dôjini *at the same time,* ippou *in contrast,* sarani *in addition,* mo *topic marker,* ni-tsuite-wa *regarding,* tameda *the reason is that,* kekka *as the result* ga-nerai *the goal is that*

sentences. Some of the sentence endings are inflectional categories of verbs such as PAST/NON-PAST, INTERROGATIVE, and also morphological categories like nouns and particles (eg. question-markers). Based on Ichikawa (1990), we defined six types of sentence-ending cues and marked a sentence according to whether it contains a particular type of cue. Included in the set are inflectional forms of the verb and the verbal adjective, PAST/NON-PAST, morphological categories such as COPULA, and NOUN, parentheses (quotation markers), and sentence-final particles such as -ka. We use the following two attributes to encode information about sentence-ending cues.

<EndCueA> records information about a sentence-ending form of the preceding sentence. It takes a discrete value from 0 to 6, with 0 indicating the absence in the sentence of relevant cues.

<EndCueB> Same as above except that this feature is concerned with a sentence-ending form of the current sentence, *i.e.* the 'B' sentence.

Finally, we have two classes, ELABORATION and SEQUENCE.

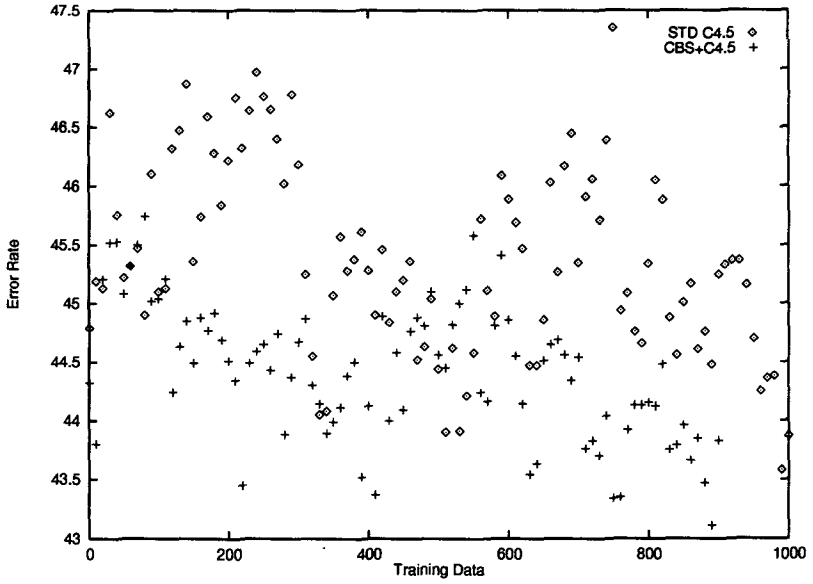
4 Evaluation

To evaluate our method, we carried out experiments, using a corpus of news articles from a Japanese economics daily (Nihon-Keizai-Shimbun-Sha, 1995). The corpus had 477 articles, randomly selected from issues that were published during the year. Each sentence in the articles was tagged with one of the discourse relations at the subclass level (*i.e.* CONSEQUENTIAL, ANTITHESIS, etc.). However, in evaluation experiments, we translated a subclass relation into a corresponding major class relation (SEQUENCE/ELABORATION) for reasons discussed earlier. Furthermore, we explicitly asked coders not to tag a paragraph initial sentence for a discourse relation, for we found that coders rarely agree on their classifications. Paragraph-initial sentences were dropped from the evaluation corpus. This had left us with 5221 sentences, of which 56% are labeled as SEQUENCE and 44% ELABORATION.

To find out effects of the committee-based sampling method (CBS), we ran the C4.5 (Release 5) decision tree algorithm with CBS turned on and off (Quinlan, 1993) and measured the performance by the 10-fold cross validation, in which the corpus is divided evenly into 10 blocks of data and 9 blocks are used for training and the remaining one block is held out for testing. On each validation fold, CBS starts with a set of about 512 samples from the set of training blocks and sequentially examines samples from the rest of the training set for possible labeling. If a sample is selected, then a decision tree will be trained on the sample together with the data acquired so far, and tested on the held-out data. Performance scores (error rates) are averaged over 10 folds to give a summary figure for a particular learning strategy. Throughout the experiments, we assume that $k = 10$ and $g = 1$, *i.e.*, 10 committee members and the entropy gain of 1. Figure 1 shows the result of using CBS for a decision tree. Though the performance fluctuates erratically, we see a general tendency that the CBS method fares better than a decision tree classifier alone. In fact differences between C4.5/CBS and C4.5 alone proved statistically significant ($t = 7.06$, $df = 90$, $p < .01$).

While there seems to be a tendency for performance to improve with an increase in the amount of training data, either with or without CBS, it is apparent that an increase in the training data has non-linear effects on performance, which makes an interesting contrast with probabilistic classifiers like HMM, whose performance improves linearly as the training data grow. The reason has to do with the structural complexity of the decision tree model: it is possible that small changes in the *INFO* value lead to

Figure 1: Effects of CBS on the decision tree learning. Each point in the scatterplots represents a summary figure, *i.e.* the average of figures obtained for a given x in 10-fold cross validation trials. The x-axis represents the amount of training data, and the y-axis the error rate. The error rate is the proportion of the misclassified instances to the total number of instances.



a drastic restructuring of a decision tree. In the face of this, we made a small change to the way CBS works. The idea, which we call a *sampling with error feedback*, is to remove harmful examples from the training data and only use those with positive effects on performance. It forces the sampling mechanism to return to *status quo ante* when it finds that an example selected degrades performance. More precisely, this would be put as follows:

$$S_{t+1} = \begin{cases} S_t \cup \{e\}, & \text{if } E(C^{S \cup \{e\}}) < E(C^{S_t}) \\ S_t & \text{otherwise} \end{cases}$$

S_t is a training set at time t. $C^{\mathcal{S}}$ denotes a classifier built from the training set $\mathcal{S}$. $E(C^{\mathcal{S}})$ is an error rate of a classifier $C^{\mathcal{S}}$. Thus if there is an increase or no reduction in the error rate after adding an example e to the training set, a classifier goes back to the state before the change.

As Figure 2 shows, the error feedback produced a drastic reduction in the error rate. At 900, the committee-based method with the error feedback reduced the error rate by as much as 23%. Figure 3 compares performance of three sampling methods, random sampling, the committee-based sampling with 100 bootstrap replicates (*i.e.*, $K = 100$) and that with 500 bootstrap replicates. In the random sampling method, a sample is selected randomly from the data and added to the training data. Figure 4 compares a random sampling approach with CBS with 500 bootstrap replicates. Both used the error feedback mechanism. Differences, though they seem small, turned out to be statistically significant ($t = 4.51$, $df = 90$, $p < .01$), which demonstrates the significance of C4.5/CBS approach. Furthermore, Figure 5 demonstrates that the number of bootstrap replicates affects performance ($t = 8.87$, $df = 90$, $p < .01$). CBS with 500 bootstraps performs consistently better than that with 100 bootstrap replicates. This might mean that in the current setup, 100 replicates are not enough to simulate the true distribution of $P(M \mid S)$. Note that CBS with 500 replicates achieves the error rate of 33.40 with only 1008 training samples, which amount to one fourth of the training data C4.5 alone required to reach 44.64. While a direct comparison with other learning schemes in discourse such as a transformation method (Samuel et al., 1998) is not feasible, if Samuel et al. (1998)'s approach is indeed comparable to C5.0, as discussed in Samuel et al. (1998), then the present method might be able to reduce the

Figure 2: The committee-based method (with 100 bootstraps) with the error feedback as compared against one without. The error rate decreases rapidly with the growth of the training data (the lower scatterplot). The upper scatterplot represents CBS with the error feedback disabled.

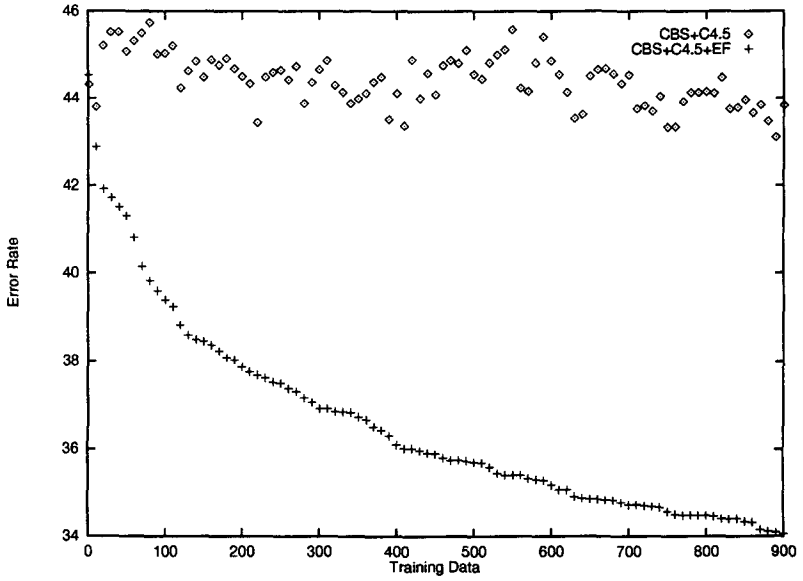
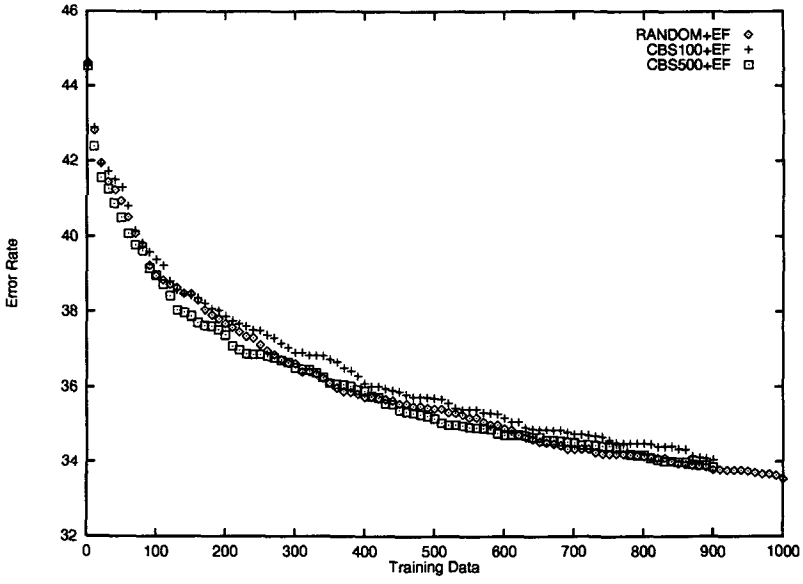


Figure 3: Comparing performance of three approaches, random sampling (RANDOM-EF), bootstrapped CBS with 100 replicates (CBS100-EF), and bootstrapped CBS with 500 replicates (CBS500-EF), all with the error feedback on.



amount of training data without hurting performance.

5 Conclusions

We presented a new approach for identifying discourse relations, built upon the committee-based sampling method, in which useful examples are selected for training and those not useful are discarded. Since the committee-based sampling method was originally developed for probabilistic classifiers, we extended the method for a decision tree classifier, us-

Figure 4: Differences in performance of RANDOM-EF (random sampling with the error feedback) and CBS500-EF (CBS with the error feedback, with 500 bootstrap replicates).

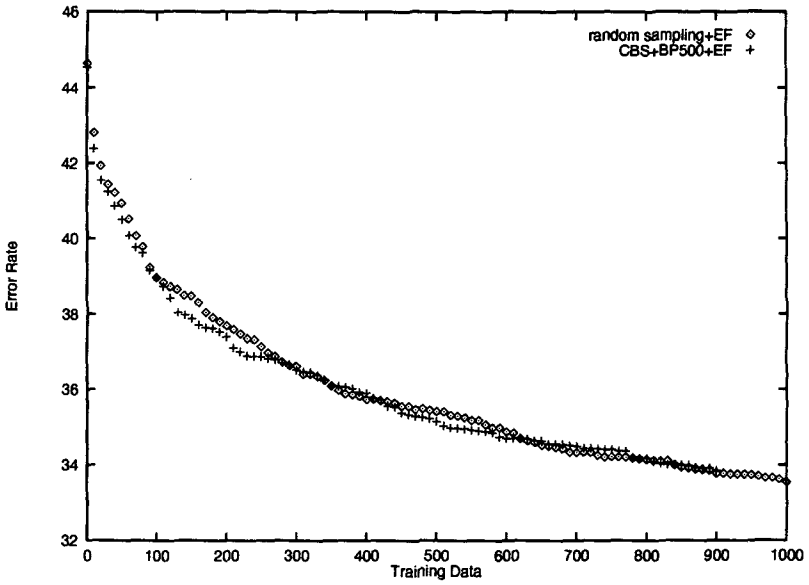
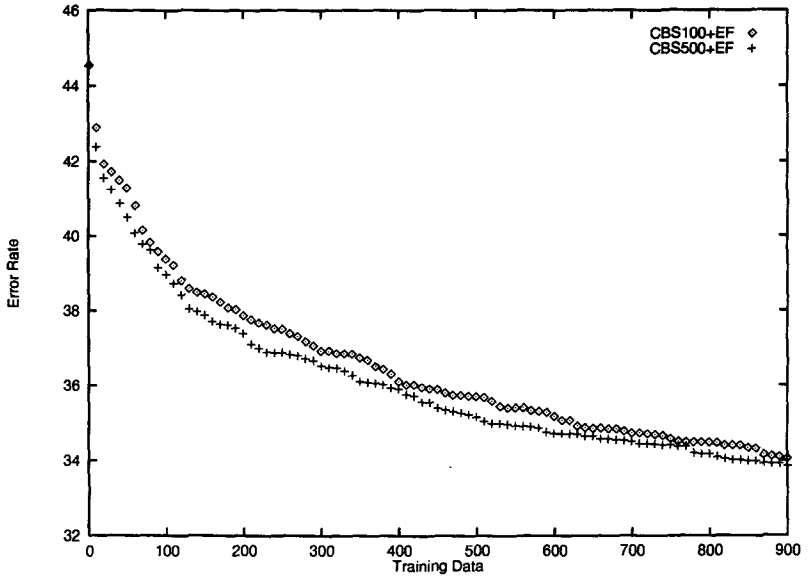


Figure 5: Differences in performance of CBS500-EF and CBS100-EF.



ing a statistical technique called *bootstrapping*. The use of the method for learning discourse relations resulted in a drastic reduction in the amount of data required and also an increased accuracy. Further, we found that the number of bootstraps has substantial effects on performance; CBS with 500 bootstraps performed better than that with 100 bootstraps

References

Paul R. Cohen. 1995. *Empirical Methods in Artificial Intelligence*. The MIT Press.

Ido Dagan and Sean Engelson. 1995. Committee-based sampling for training probabilistic classifiers. In *Proceedings of International Conference on Machine Learning*, pages 150–157, July.

Thomas G. Dietterich. 1998. An experimental

comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *submitted to Machine Learning.*

Sean P. Engelson and Ido Dagan. 1996. Minimizing manual annotation cost in supervised training from corpora. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 319–326. ACL, June. University of California,Santa Cruz.

Takashi Ichikawa. 1990. *Bunshôron-gaisetsu.* Kyôiku-Shuppan, Tokyo.

Klaus Krippendorff. 1980. *Content Analysis: An Introduction to Its Methodology*, volume 5 of *The Sage* COMMTEXT *series.* The Sage Publications, Inc.

W. C. Mann and S. A. Thompson. 1987. Rhetorical Structure Theory. In L. Polyani, editor, *The Structure of Discourse.* Ablex Publishing Corp., Norwood, NJ.

Daniel Marcu. 1997. The Rhetorical Parsing of Natural Language Texts. In *Proceedings of the 35th Annual Meetings of the Association for Computational Linguistics and the 8th European Chapter of the Association for Computational Linguistics*, pages 96–102, Madrid, Spain, July.

Nihon-Keizai-Shimbun-Sha. 1995. Nihon Keizai Shimbun 95 nen CD-ROM ban. CD-ROM. Nihon Keizai Shimbun, Inc., Tokyo.

J. Ross Quinlan. 1993. *C4.5: Programs for Machine Learning.* Morgan Kaufmann.

Gerald Salton and Michael J. McGill. 1983. *Introduction to Modern Information Retreival.* McGraw-Hill Computer Science Series. McGraw-Hill Publishing Co.

Ken Samuel, Sandra Carberry, and K. Vijay-Shanker. 1998. Diaglogue act tagging with transformation-based learning. In *Proceedings of the 36th Annual Meeting of the Association of Computational Linguistics and the 17th International Conference on Computational Linguistics*, pages 1150–1156, August 10-14. Montreal, Canada.

Sidney Siegel and N. John Castellan. 1988. *Nonparametric Statistics for the Behavioral Sciences.* McGraw-Hill, Second edition.