

Combining Labeled and Unlabeled Data for Learning Cross-document Structural Relationships

Zhu Zhang and Dragomir Radev

School of Information

and Department of Electrical Engineering and Computer Science

University of Michigan

Ann Arbor, MI 48109

{zhuzhang, radev}@umich.edu

Abstract

Multi-document discourse analysis has emerged with the potential of improving various NLP applications. Based on the newly proposed Cross-document Structure Theory (CST), this paper describes an empirical study that classifies CST relationships between sentence pairs extracted from topically related documents, exploiting both labeled and unlabeled data. We investigate a binary classifier for determining existence of structural relationships and a full classifier using the full taxonomy of relationships. We show that in both cases the exploitation of unlabeled data helps improve the performance of learned classifiers.

1 Introduction

With the proliferation of web-based information resources and related applications, such as information extraction, text summarization, and question answering, computational models for natural language discourse structures have gained increasing attention. Recently, the study of multi-document discourse has emerged, which is different from traditional discourse analysis in that multiple related documents may be written in different styles, use different vocabulary, and reflect inconsistent perspectives.

Inspired by Rhetorical Structure Theory (RST) (Mann and Thompson, 1988), the notion of Cross-document Structure Theory (CST) was

proposed by (Radev, 2000). The central idea is to posit a set of rhetorical relationships that hold between sentences across topically-related documents. It has been shown that the availability of such information can help multi-document text summarization (Zhang et al., 2002). It is also conceivable that other NLP- or IR-related applications, such as semantic entity and relation extraction (where semantic relations may instantiate cross document boundaries), and non-factoid question answering (where answer generation may demand “fusing” information from multiple documents), can potentially benefit from the understanding of multi-document discourse structure.

However, as far as we are aware of, not much work has been done to show whether the relationships posited in the CST framework can be automatically identified from free text. In (Zhang et al., 2003), we explored the possibility of classifying CST relationships by using a strictly supervised machine learning approach. In this paper, we try to improve the classification model by leveraging both labeled and unlabeled data.

2 Related work

Effort has been made to identify RST relationships from text. Marcu (1997) proposed a first-order formalization of the high-level rhetorical structure of text, and provided a theoretical analysis and an empirical comparison of four algorithms for automatic derivation of text structures. Marcu’s knowledge-based approach relied on “cue phrases” in implementing algorithms to discover the valid RST trees for a single doc-

ument. This is reasonable because of the conventions of writing and the valid assumption that authors tend to write documents using certain rhetorical techniques. However, in the case of multiple documents and cross-document relationships, we cannot expect to encounter a reliable analog to the cue phrase. This is because separate documents, even when they are related to a common topic, are generally not written with an overarching structure in mind. Therefore, when identifying CST relationships, it may pay off to look for deeper-level cues and to pursue statistical approaches instead.

(Marcu and Echihabi, 2002) presented a machine learning approach to classifying RST relationships. Only lexical features are used in the naive Bayes classifier, and part-of-speech information is only used for feature selection for comparison purposes. Worth noting is that the authors take advantage of the available linguistic knowledge and exploit various cues in obtaining training data. Doing so is, again, much harder in the cross-document context.

In our most recent work (Zhang et al., 2003), we used the strictly supervised AdaBoost algorithm (Freund and Schapire, 1997) to classify sentence-level CST relationships from free text, and achieved promising results on both binary classification and full classification.

Within the realm of machine learning, there has been growing interest in a family of learning techniques that aim at inducing classifiers by leveraging a small amount of labeled data and a large amount of unlabeled data. Among them is bootstrapping, or weakly supervised learning. It chooses the unlabeled instances with the highest probability of being correctly labeled and uses them to augment labeled training data. Two of the most influential bootstrapping algorithms are the co-training algorithm (Blum and Mitchell, 1998) and Yarowsky algorithm (Yarowsky, 1995). Recently (Abney, 2002) provided a very cogent comparison of the two with some nice extensions.

Bagging (bootstrap aggregation) predictors (Breiman, 1996) is a method for generating multiple versions of a predictor and using them to get an aggregated predictor. The aggregation averages over the versions when predicting a numerical outcome and does a plurality vote

when predicting a class. The multiple versions are formed by making bootstrap replicates of the learning set (specifically, the replicates are of the same size as the original data and are generated by sampling with replacement) and using these as new learning sets. Variations of the idea have been explored by (Banko and Brill, 2001) and (Ng and Cardie, 2003).

3 Problem definition

3.1 Cross-document Structure Theory

Cross-document Structure Theory (CST) is a functional theory for multi-document discourse structure. It is used to describe cross-document semantic connections, such as “elaboration”, “contradiction”, “attribution”, and “historical background”, among text units of related documents. Instead of assuming deliberate writing, as RST does, CST views topically-related documents as generated by a “collective authorship”. While the graph-like conceptual representation looks like “semantic hyperlinks” (Salton et al., 1997), the relationships are all linguistically motivated. We focus on sentence-level CST relationships in this study.

A total of 18 CST relationships are defined in (Zhang et al., 2003). All relationships are domain-independent. Some of them, such as the *paraphrase* example shown below, are symmetric (*multinuclear*, in RST terms).

```
Derek Bell is experiencing a  
resurgence in his career.
```

```
Derek Bell is having a  
"comeback year."
```

Some other ones, such as the *subsumption* example shown below, do have directionality, i.e., they have *nucleus* and *satellite*.

```
With 3 wins this year, Green  
Bay has the best record in the  
NFL.
```

```
Green Bay has 3 wins this year.
```

3.2 Formulation of the classification problem

As a first approximation, we cast the CST relationship identification problem in a standard classification framework. Conceptually, given an unordered sentence pair $P(S_1, S_2)$, where sentences

S_1 and S_2 are from two different but topically related documents, we are interested in determining the type(s) of cross-document relationships between them.

In this paper, we investigate the following two scenarios:

Binary classification Here we are interested in the existence of cross-document relationships regardless of type. If the two sentences are related in any types, the pair is assigned a label “1”, otherwise a “0”.

Full classification In this case we do care about the type(s) of cross-document relationships between the sentence pair. Moreover, it is possible for a single pair to have multiple labels (see section 4 and 5.3 for more details). The class labels are adapted from the CST taxonomy. Therefore, there are 19 possible labels in total (18 CST types plus a special type “no relationship”).

4 Experimental setup and data collection

Due to nonexistence of readily available data set, we had to actively collect data and have human judges annotate the CST relationships.

As our first attempt, we collected six clusters of related news articles from various sources. The clusters were chosen to be diverse with respect to their topics, the time span across the documents, the cluster size, and the publishers. Table 1 shows the characteristics of the clusters. The cluster names reflect the source from which the cluster of documents was obtained.

The *Milan9* cluster was used strictly for corpus development and judge training purposes. It was carefully annotated for CST relationships by the authors in developing the markup scheme and the guidelines to be used by the independent judges to be hired. The other five clusters were annotated by the human judges.

Human annotation is expensive, and it does not always yield ideal results. Human judges often do not agree, due to the inherently ambiguous nature of natural language. The large search space makes the situation even worse. In a ten-document cluster with 20 sentences on average in each document, for example, a human judge will

have to examine roughly 18,000 sentence pairs if he or she needs to exhaust all possibilities. This is an incredibly tedious job in any sense, and consequently, it is very difficult for multiple judges to reach reasonable agreement on the annotation.

One possible way to alleviate the problem is to exploit the observation that CST relationships are unlikely to exist between sentences that are *lexically* very dissimilar to each other. In other words, certain similarity measures might behave as a useful proxy for finding CST-related sentence pairs. We experimented with a few lexical-level similarity metrics, including Cosine (Salton and Lesk, 1968), word overlap, longest common subsequence and BLEU (Papineni et al., 2002), and then measured their correlation with CST-relatedness. Using the very carefully annotated *Milan9* as “training” data, we found that word overlap rate 0.12 is the “best” cutoff criterion for selecting sentence pairs, in the sense that it helps minimize selected number of sentence pairs without losing too many CST-related pairs (the recall is 87.5%). We then applied this measure on the other five clusters and selected, from a huge number of possible sentence pairs, a total of 4,931 potentially “interesting” ones for human judges to annotate. This way the judges’ workload is significantly reduced. Eight judges were hired; each judge annotated at least one cluster; each cluster was annotated by two judges. The judges were allowed to assign multiple CST types to a single sentence pair, given the inherently ambiguous nature of the problem and the fact that the CST types are sometimes not mutually exclusive.

The more interesting part of the story is that we can get unlabeled data from any other sources virtually for free. The purpose of this study is exactly to explore the possibility of improving CST classifiers by exploiting both labeled and unlabeled data. Specifically, we use a set of 6,000 sentence pairs of arbitrary similarity from another news cluster (*Shuttle10*, which is about the Columbia Shuttle accident in 2003) as unlabeled data in our experiments.

5 Classification exploiting both labeled and unlabeled data

In this section, we discuss the micro-level supervised learning component and the macro-level

Cluster	Topic	Articles	Time span	Ave. length (sent.)	No. sources	Clustering method
Milan9	Milan plane crash	9	2 days	30	5	manual
DUC	John Lennon biography	4	4 years	46	4	manual
Gulfair11	Bahrain plane crash	11	4 days	27	6	manual
HKNews	Air and water quality	8	2.5 years	32	1	manual
NIE	N. Korea nuclear weapons	5	18 days	14	3	automatic
Novelty	Cancer and power lines	4	4 years	21	2	manual

Table 1: Characteristics of the document clusters

bootstrapping procedure respectively.

5.1 Underlying supervised learning: algorithm and features

In (Zhang et al., 2003), we choose to use boosting, specifically AdaBoost (Freund and Schapire, 1997), for our task. In this study, we build upon the same supervised learning component and investigate the effect of unlabeled data.

The basic idea of the boosting algorithm is to find a “strong” hypothesis by combining many “weak” or “base” hypotheses. Moreover, Boostexter (Schapire and Singer, 2000), the off-the-shelf implementation of boosting, explicitly supports multi-label classification, which is very convenient for the multi-label full classification scenario in our problem.

Lexical features by themselves are apparently not sufficient for identifying CST relationships between sentences. Therefore, we considered various features at three linguistic levels, the details of which follow. The general idea is to quantify the similarity or distance between two sentences at different levels.

For most sentence pairs, the procedures for computing all features, such as tokenization, part-of-speech (POS) extraction, and head guessing, can directly or indirectly take advantage of the parse trees produced by the Charniak parser (Charniak, 1999). For the very few sentences on which the parser fails, we used heuristic backoff procedures.

Lexical features

At this level, we are only interested in the surface tokens. No stemming or stop-word deletion is done. Three features are considered:

- Number of tokens in S_1
- Number of tokens in S_2

- Number of tokens in common

Shallow syntactic-level features

We try to capture the overlap between two sentences with regard to 6 parts of speech: regular noun, proper noun, verb, adjective, adverb, and (cardinal) number, which are considered to convey relatively more substantial information than others.

For each x in the 6 POS types above, we compute the following counts, which gives us a total of 18 features:

- Number of tokens having POS x in S_1
- Number of tokens having POS x in S_2
- Number of common tokens having POS x

Deeper syntactic-level features

The idea here is to find the most prominent concepts discussed in each sentence pair (by taking advantage of the syntactic structure) and compute their lexical semantic distance by using Wordnet (Fellbaum, 1998). More specifically, this is done through the following steps:

1. Find the top level NP (noun phrase) and VP (verb phrase) in S_1 and S_2 .
2. Find the head tokens of both NP and VP by using the head rules in (Collins, 1999).
3. Align the heads correspondingly (i.e., NP vs. NP, VP vs. VP).
4. For each head pair, compute lexical semantic distance measures (*lch*, *jcn*, *res*, *lin*, and *hso*) by using the semantic distance toolkit (Patwardhan and Pedersen, 2002).

For each sentence pair, we have two pairs of heads (heads of NPs and heads of VPs). For each

head pair, we compute the five semantic distance measures above. Therefore we have a total of 10 features in this group. For example, given the following sentence pair, five distance measures will be computed for word pairs {Bush, President} and {visit, arrive} respectively.

```
(S1(S(NP(NNP Bush))(VP(VBZ
visits)(NP(NNP China))(. .)))
(S1(S(NP(DT The)(NNP
President))(VP(VBZ arrives)(PP
(IN in)(NP(NNP Beijing)))(.
.)))
```

More discussion of the Wordnet-based features can be found in (Zhang et al., 2003).

5.2 Committee-based bootstrapping using bagging

The central idea of weakly supervised learning is to take advantage of the information hidden in large amount of unlabeled data. In this case specifically, we want to choose the unlabeled data points that have the highest probability of being correctly labeled (by a classifier or a set of classifiers trained on currently available labeled data) and include them into the labeled data set.

We propose a committee-based data augmentation procedure (Algorithm 1) using bagging, which is in spirit very similar to those used in (Banko and Brill, 2001) and (Ng and Cardie, 2003) but with an extra step size parameter. There are two reasons why we added this parameter:

- It is conceivable that the classifier performance will not keep improving infinitely with additional unlabeled data. We are interested in finding the best cutoff point.
- A more important intuition is to maintain inertia for the augmented data set and the corresponding classifier, instead of introducing too much perturbation at one time. In other words, the number of extra data points added into the labeled set in each iteration should not be very large compared to the size of the seed set.

Once the labeled data set is augmented by Algorithm 1, there are two options:

- Train a single classifier on the augmented data set L , and use it as the final predictor (referred to as the *FinalSingle* strategy).

Algorithm 1 Committee-based data augmentation

Require: labeled data set L

Require: unlabeled data set U

Require: augmentation step size S

repeat

 Generate n bootstrap datasets from L using bagging

 Train a committee of n classifiers on the n bootstrap data sets respectively

 Run the committee on U

 Add at most S agreed-upon (by all committee members) data points into L

until No data points can be added into L

- Train a set of bagging predictors on the augmented L , and let the committee vote on new data points (referred to as the *FinalBagging* strategy).

We experiment with both strategies and examine the difference in performance.

5.3 Data treatment

Proper treatment of the labeled data is crucial, as they are used as “seeds” for the bootstrapping process.

As mentioned in section 4, each document cluster is annotated by two judges, and the judges are allowed to assign multiple labels to a single pair. The judges don’t always agree: they may either disagree on whether two sentences are CST-related at all or disagree on the types of CST relationships between them. Instead of asking the judges to resolve the disagreements, we decided to only include the data points on which the two judges at least agree on the existence of CST relationships (regardless of type). 3,942 out of 4,931 sentence pairs satisfy this condition ($kappa = 0.53$). This is an important decision based on our understanding of the underlying linguistic phenomenon, instead of technical inability of dealing with noisy data. Since the ability to determine the existence of any CST relationships is important for many potential applications, we want the model to be as clean as possible. On the other hand, once the CST-relatedness is known, it is reasonable for multiple CST relationships to exist between two sentences.

Given the constraint above, labels can be assigned to data points in the binary and full classification scenarios respectively:

- In the binary classification case, a label “1” is assigned to a pair if it is unanimously believed to be CST-related, and a label “0” if it is unanimously believed to be not related. (These are the only two cases possible due to the constraint above.)
- In the full classification case, each sentence pair is assigned the union of the labels given by the two judges if they agree that the two sentences are CST-related, or a label “0” if they agree that the two sentences are not related. (Again, these are the only two cases possible due to the constraint above.)

The whole labeled data set is then split into a (seed) training set, a validation set, and a test set by uniform random sampling without replacement in the proportion of 6:2:2.

The treatment of unlabeled data is less tricky. We only need to compute feature vectors for them and assign blank labels correspondingly.

5.4 Evaluation metrics

For binary classification, besides the standard classification accuracy, we also measure precision, recall, and F-measure as defined in the information retrieval literature (Salton and Lesk, 1968).

For full classification, we compute the following aggregate metrics suggested by (Schapire and Singer, 2000):

- One-accuracy (whether the top-ranked label is among the correct ones)
- Coverage (how far do we have to go down the ranked label list to find all the correct ones)
- Average precision (analogous to non-interpolated average precision for evaluating document ranking performance)

We also measure precision, recall, and F-measure for each individual class label.

6 Experiments and results

In presenting the experimental results, we are especially interested in whether the exploitation of unlabeled data can help improve the performance of resultant classifiers. Notice that in both cases, the baseline strategy is to assign the label “0” (i.e., no CST relationship) to all data points, which achieves an accuracy of 75.13% on the test set.

In the experiments, we used the following parameter values:

- Rounds of boosting: 400
- Number of bags: 10
- Step size for data augmentation: 200

The same supervised learning component is used across all experiments, for the purpose of fair comparison.

6.1 Binary classification

By observing the classifier behavior on the validation set, we notice that the performance maximizes with roughly 400 additional unlabeled data points. The actual performance on the final test set is presented in Table 2. As we can see, using unlabeled data does result in significantly improved classifier. A minor note is that the *FinalBagging* strategy gives slightly worse results than *FinalSingle*, although still better than the strictly supervised learner.

6.2 Full classification

In this scenario, we again find that adding roughly 400 additional unlabeled data points gives the best results on the validation set. The improvement in aggregate performance metrics still consistently holds, except that the *FinalBagging* strategy works better than *FinalSingle* this time (Table 3).

Class-specific results are summarized in Table 4 (Only CST types that occur more than 20 times in the test data are shown; the F-measures of the strictly supervised learner are also presented for comparison purposes). We see mixed results here. With extra unlabeled data, the bootstrapped classifier wins on some CST types and loses on some others. A possible explanation is the sparseness of the seed training data relative to the large number of classes.

	Accuracy	Precision	Recall	F-measure
Labeled data only	0.8789	0.8278	0.6477	0.7267
Labeled & unlabeled data (<i>FinalSingle</i>)	0.8905	0.8699	0.6580	0.7493
Labeled & unlabeled data (<i>FinalBagging</i>)	0.8853	0.8562	0.6477	0.7375

Table 2: Performance of binary classifier

	One-Accuracy	Coverage	Average precision
Labeled data only	0.8093	1.1070	0.8729
Labeled & unlabeled data (<i>FinalSingle</i>)	0.8157	1.0773	0.8768
Labeled & unlabeled data (<i>FinalBagging</i>)	0.8376	1.0721	0.8839

Table 3: Performance of full classifier: aggregate metrics

6.3 Lessons learned

Experiments in both classification scenarios justified the hypothesis that there is an upper bound on performance of classifiers that incorporate unlabeled data. The improvement of performance is always followed by a drop. (Banko and Brill, 2001) contends that this is because the gains from additional training data are eventually offset by the sample bias in mining the unlabeled data points. On the other hand, this indicates the necessity of further, more focused, human annotation, which gives rise to potential combination of bootstrapping and active learning (Muslea, 2002).

Secondly, the conservative stepwise strategy of augmenting labeled data is very appropriate; aggressive greedy augmentation hurts more than it helps. In the binary classification scenario, for example, the latter strategy would have added over 5,000 data points into the labeled set within 3 iterations, and results in a accuracy of 0.8489 and a F-measure of 0.7073, which is even worse than the strictly supervised learner.

7 Conclusion and future work

This paper describes an empirical study that leverages both labeled and unlabeled data to train machine-learned classifiers for cross-document structural relationships. We show that both the binary classifier and the full classifier can be improved by exploiting unlabeled data.

From a machine learning theoretical point of view, we were able to show that bagging and boosting, traditionally viewed as competitors, can be combined to generate strong performance.

Looking into the future, on one hand, there is plenty of room for improving the underlying su-

pervised learning performance by designing more sophisticated feature vectors; on the other hand, it is worthwhile investigating meaningful ways of keeping humans in the loop and overcoming the performance bottleneck by focused annotation.

The classification problem studied in this paper is still a somewhat simplified version of the full CST relationship identification problem. Some relationships have directionality, e.g., *A* following-up *B* is different from *B* following-up *A*. To be able to address issues like this, more intelligence needs to be built into the CST identifier. Another caveat is that the classifiers in the current experiments only look at “local” information within each sentence pair. In some cases, the “global” context plays an important role in determining the CST relationship(s).

In this paper, we have made further attempt to show that automatic identification of CST relationships is feasible. Various NLP- and IR-related applications may expect to see improvements by exploiting cross-document structure.

Acknowledgments

This material is based upon work supported by the National Science Foundation (Washington, DC) under Grant No. 0082884. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

Steven Abney. 2002. Bootstrapping. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 360–367.

CST type	Precision	Recall	F-measure	F-measure (supervised)
No relationship	0.8875	0.9605	0.9226	0.9186
Equivalence	0.5000	0.3200	0.3902	0.3429
Subsumption	0.1000	0.0417	0.0588	0.0513
Follow-up	0.4727	0.2889	0.3586	0.3946
Elaboration	0.3125	0.1282	0.1818	0.2478
Description	0.3333	0.1071	0.1622	0.2353
Overlap	0.5263	0.2941	0.3773	0.4324

Table 4: Full classifier performance on individual CST types

- Michele Banko and Eric Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Meeting of the Association for Computational Linguistics*, pages 26–33.
- Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *COLT: Proceedings of the Workshop on Computational Learning Theory*, Morgan Kaufmann Publishers.
- Leo Breiman. 1996. Bagging predictors. *Machine Learning*, 24(2):123–140.
- Eugene Charniak. 1999. A maximum-entropy-inspired parser. Technical Report CS-99-12, Computer Science Department, Brown University.
- Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.
- Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.
- Yoav Freund and Robert E. Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139.
- William C. Mann and Sandra A. Thompson. 1988. Rhetorical Structure Theory: towards a functional theory of text organization. *Text*, 8(3):243–281.
- Daniel Marcu and Abdessamad Echihabi. 2002. An unsupervised approach to recognizing discourse relations. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 368–375.
- Daniel Marcu. 1997. *The Rhetorical Parsing, Summarization, and Generation of Natural Language Texts*. Ph.D. thesis, Department of Computer Science, University of Toronto, December.
- Ion Muslea. 2002. *Active learning with multiple views*. Ph.D. thesis, USC.
- Vincent Ng and Claire Cardie. 2003. Weakly supervised natural language learning without redundant views.
- 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 Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318.
- S. Patwardhan and T. Pedersen. 2002. distance.pl: Perl program that measures the semantic relatedness of words (version 0.11). <http://www.d.umn.edu/~tpederse/distance.html>.
- Dragomir Radev. 2000. A common theory of information fusion from multiple text sources, step one: Cross-document structure. In *Proceedings, 1st ACL SIGDIAL Workshop on Discourse and Dialogue*, Hong Kong, October.
- G. Salton and M. E. Lesk. 1968. Computer evaluation of indexing and text processing. *Journal of the ACM (JACM)*, 15(1):8–36.
- Genrald Salton, Amit Singhal, Mandar Mitra, and Chris Buckley. 1997. Automatic text structuring and summarization. *Information Processing & Management*, 33:193–207.
- Robert E. Schapire and Yoram Singer. 2000. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168.
- David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 189–196.
- Zhu Zhang, Sasha Blair-Goldensohn, and Dragomir Radev. 2002. Towards CST-enhanced summarization. In *Proceedings of the 18th National Conference on Artificial Intelligence*, Edmonton, Alberta, August.
- Zhu Zhang, Jahna Otterbacher, and Dragomir Radev. 2003. Learning cross-document structural relationships using boosting. In *Proceedings of the 12th International Conference on Information and Knowledge Management CIKM 2003*, New Orleans, LA, November.