Aggregated Word Pair Features for Implicit Discourse Relation Disambiguation

Or Biran
Columbia University
Department of Computer Science
orb@cs.columbia.edu

Kathleen McKeown
Columbia University
Department of Computer Science
kathy@cs.columbia.edu

Abstract

We present a reformulation of the word pair features typically used for the task of disambiguating implicit relations in the Penn Discourse Treebank. Our word pair features achieve significantly higher performance than the previous formulation when evaluated without additional features. In addition, we present results for a full system using additional features which achieves close to state of the art performance without resorting to gold syntactic parses or to context outside the relation.

1 Introduction

Discourse relations such as *contrast* and *causality* are part of what makes a text coherent. Being able to automatically identify these relations is important for many NLP tasks such as generation, question answering and textual entailment. In some cases, discourse relations contain an explicit marker such as *but* or *because* which makes it easy to identify the relation. Prior work (Pitler and Nenkova, 2009) showed that where explicit markers exist, the class of the relation can be disambiguated with f-scores higher than 90%.

Predicting the class of *implicit* discourse relations, however, is much more difficult. Without an explicit marker to rely on, work on this task initially focused on using lexical cues in the form of *word pairs* mined from large corpora where they appear around an explicit marker (Marcu and Echihabi, 2002). The intuition is that these pairs will tend to represent semantic relationships which are related to the discourse marker (for example, word pairs often appearing around *but* may tend to be antonyms). While this approach showed some success and has been used extensively in later work, it has been pointed out by multiple authors that many of the most useful word pairs are pairs of very common functional words, which contradicts the original intuition, and it is hard to explain why these are useful.

In this work we focus on the task of identifying and disambiguating implicit discourse relations which have no explicit marker. In particular, we present a reformulation of the word pair features that have most often been used for this task in the past, replacing the sparse lexical features with dense aggregated score features. This is the main contribution of our paper. We show that our formulation outperforms the original one while requiring less features, and that using a stop list of functional words does not significantly affect performance, suggesting that these features indeed represent semantically related content word pairs.

In addition, we present a system which combines these word pairs with additional features to achieve near state of the art performance without the use of syntactic parse features and of context outside the arguments of the relation. Previous work has attributed much of the achieved performance to these features, which are easy to get in the experimental setting but would be less reliable or unavailable in other applications.[1]

2 Related Work

This line of research began with (Marcu and Echihabi, 2002), who used a small number of unambiguous explicit markers and patterns involving them, such as [Arg1, *but* Arg2] to collect sets of word pairs from a large corpus using the cross-product of the words in Arg1 and Arg2. The authors created a feature out of each pair and built a naive bayes model directly from the unannotated corpus, updating the priors and posteriors using maximum likelihood. While they demonstrated

[1] Reliable syntactic parses are not always available in domains other than newswire, and context (preceding relations, especially explicit relations) is not always available in some applications such as generation and question answering.

some success, their experiments were run on data that is unnatural in two ways. First, it is balanced. Second, it is constructed with the same unsupervised method they use to extract the word pairs - by assuming that the patterns correspond to a particular relation and collecting the arguments from an unannotated corpus. Even if the assumption is correct, these arguments are really taken from explicit relations with their markers removed, which as others have pointed out (Blair-Goldensohn et al., 2007; Pitler et al., 2009) may not look like true implicit relations.

More recently, implicit relation prediction has been evaluated on annotated implicit relations from the Penn Discourse Treebank (Prasad et al., 2008). PDTB uses hierarchical relation types which abstract over other theories of discourse such as RST (Mann and Thompson, 1987) and SDRT (Asher and Lascarides, 2003). It contains $40,600$ annotated relations from the WSJ corpus. Each relation has two arguments, Arg1 and Arg2, and the annotators decide whether it is explicit or implicit.

The first to evaluate directly on PDTB in a realistic setting were Pitler et al. (2009). They used word pairs as well as additional features to train four binary classifiers, each corresponding to one of the high-level PDTB relation classes. Although other features proved to be useful, word pairs were still the major contributor to most of these classifiers. In fact, their best system for *comparison* included only the word pair features, and for all other classes other than *expansion* the word pair features alone achieved an f-score within 2 points of the best system. Interestingly, they found that training the word pair features on PDTB itself was more useful than training them on an external corpus like Marcu and Echihabi (2002), although in some cases they resort to information gain in the external corpus for filtering the word pairs.

Zhou et al. (2010) used a similar method and added features that explicitly try to predict the *implicit marker* in the relation, increasing performance. Most recently to the best of our knowledge, Park and Cardie (2012) achieved the highest performance by optimizing the feature set. Another work evaluating on PDTB is (Lin et al., 2009), who are unique in evaluating on the more fine-grained second-level relation classes.

3 Word Pairs

3.1 The Problem: Sparsity

While Marcu and Echihabi (2002)'s approach of training a classifier from an unannotated corpus provides a relatively large amount of training data, this data does not consist of true implicit relations. However, the approach taken by Pitler et al. (2009) and repeated in more recent work (training directly on PDTB) is problematic as well: when training a model with so many sparse features on a dataset the size of PDTB (there are $22,141$ non-explicit relations overall), it is likely that many important word pairs will not be seen in training.

In fact, even the larger corpus of Marcu and Echihabi (2002) may not be quite large enough to solve the sparsity issue, given that the number of word pairs is quadratic in the vocabulary. Blair-Goldensohn et al. (2007) report that using even a very small stop list (25 words) significantly reduces performance, which is counter-intuitive. They attribute this finding to the sparsity of the feature space. An analysis in (Pitler et al., 2009) also shows that the top word pairs (ranked by information gain) all contain common functional words, and are not at all the semantically-related content words that were imagined. In the case of some reportedly useful word pairs (the-and; in-the; the-of...) it is hard to explain how they might affect performance except through overfitting.

3.2 The Solution: Aggregation

Representing each word pair as a single feature has the advantage of allowing the weights for each pair to be learned directly from the data. While powerful, this approach requires large amounts of data to be effective.

Another possible approach is to aggregate some of the pairs together and learn weights from the data only for the aggregated sets of words. For this approach to be effective, the pairs we choose to group together should have similar meaning with regard to predicting the relation.

Biran and Rambow (2011) is to our knowledge the only other work utilizing a similar approach. They used aggregated word pair set features to predict whether or not a sentence is argumentative. Their method is to group together word pairs that have been collected around the same explicit discourse marker: for every discourse marker such as *therefore* or *however*, they have a single feature whose value depends only on the word pairs

collected around that marker. This is reasonable given the intuition that the marker pattern is unambiguous and points at a particular relation. Using one feature per marker can be seen as analogous (yet complementary) to Zhou et al. (2010)'s approach of trying to predict the implicit connective by giving a score to each marker using a language model.

This work uses binary features which only indicate the appearance of one or more of the pairs. The original frequencies of the word pairs are not used anywhere. A more powerful approach is to use an informed function to weight the word pairs used inside each feature.

3.3 Our Approach

Our approach is similar in that we choose to aggregate word pairs that were collected around the same explicit marker. We first assembled a list of all 102 discourse markers used in PDTB, in both explicit and implicit relations.[2]

Next, we extract word pairs for each marker from the Gigaword corpus by taking the cross product of words that appear in a sentence around that marker. This is a simpler approach than using patterns - for example, the marker *because* can appear in two patterns: [Arg1 *because* Arg2] and [*because* Arg1, Arg2], and we only use the first. We leave the task of listing the possible patterns for each of the 102 markers to future work because of the significant manual effort required. Meanwhile, we rely on the fact that we use a very large corpus and hope that the simple pattern [Arg1 *marker* Arg2] is enough to make our features useful. There are, of course, markers for which this pattern does not normally apply, such as *by comparison* or *on one hand*. We expect these features to be down-weighted by the final classifier, as explained at the end of this section. When collecting the pairs, we stem the words and discard pairs which appear only once around the marker.

We can think of each discourse marker as having a corresponding unordered "document", where each word pair is a term with an associated frequency. We want to create a feature for each marker such that for each data instance (that is, for each potential relation in the PDTB data) the value for the feature is the relevance of the marker document to the data instance.

[2]in implicit relations, there is no marker in the text but the implicit marker is provided by the human annotators

Each data instance in PDTB consists of two arguments, and can therefore also be represented as a set of word pairs extracted from the cross-product of the two arguments. To represent the relevance of the instance to each marker, we set the value of the marker feature to the cosine similarity of the data instance and the marker's "document", where each word pair is a dimension.

While the terms (i.e. word pairs) of the data instance are weighted by simple occurence count, we weight the terms in each marker's document with tf-idf, where tf is defined in one of two ways: normalized term frequency ($\frac{count(t)}{\max\{count(s,d):s\in d\}}$) and pointwise mutual information ($\log \frac{count(t)}{count(w_1)*count(w_2)}$), where w_1 and w_2 are the member words of the pair. Idf is calculated normally given that the set of all documents is defined as the 102 marker documents.

We then train a binary classifier (logistic regression) using these 102 features for each of the four high-level relations in PDTB: *comparison*, *contingency*, *expansion* and *temporal*. To make sure our results are comparable to previous work, we treat *EntRel* relations as instances of *expansion* and use sections 2-20 for training and sections 21-22 for testing. We use a ten fold stratified cross-validation of the training set for development. Explicit relations are excluded from all data sets.

As mentioned earlier, there are markers that do not fit the simple pattern we use. In particular, some markers always or often appear as the first term of a sentence. For these, we expect the list of word pairs to be empty or almost empty, since in most sentences there are no words on the left (and recall that we discard pairs that appear only once). Since the features created for these markers will be uninformative, we expect them to be weighted down by the classifier and have no significant effect on prediction.

4 Evaluation of Word Pairs

For our main evaluation, we evaluate the performance of word pair features when used with no additional features. Results are shown in Table 1. Our word pair features outperform the previous formulation (represented by the results reported by (Pitler et al., 2009), but used by virtually all previous work on this task). For most relation classes, tf is significantly better than pmi. [3]

[3]Significance was verified for our own results in all experiments shown in this paper with a standard t-test

	Comparison	Contingency	Expansion	Temporal
Pitler et al., 2009	21.96 (56.59)	**45.6 (67.1)**	63.84 (60.28)	16.21 (61.98)
tf-idf, no stop list	23 (61.72)	44.03 (66.78)	**66.48 (60.93)**	**19.54 (68.09)**
pmi-idf, no stop list	**24.38 (61.72)**	38.96 (61.52)	62.22 (57.26)	16 (65.53)
tf-idf, with stop list	23.77	44.33	65.33	16.98

Table 1: Main evaluation. F-measure (accuracy) for various implementations of the word pairs features

	Comparison	Contingency	Expansion	Temporal
Best System	25.4 (63.36)	46.94 (68.09)	75.87 (62.84)	20.23 (68.35)
features used	pmi+1,2,3,6	tf+ALL	tf+8	tf+3,9
Pitler et al., 2009	21.96 (56.59)	47.13 (67.3)	76.42 (63.62)	16.76 (63.49)
Zhou et al., 2010	31.79 (58.22)	47.16 (48.96)	70.11 (54.54)	20.3 (55.48)
Park and Cardie, 2012	31.32 (74.66)	49.82 (72.09)	79.22 (69.14)	26.57 (79.32)

Table 2: Secondary evaluation. F-measure (accuracy) for the best systems. *tf* and *pmi* refer to the word pair features used (by tf implementation), and the numbers refer to the indeces of Table 3

		Comp.	Cont.	Exp.	Temp.
1	WordNet	20.07	34.07	52.96	11.58
2	Verb Class	14.24	24.84	49.6	10.04
3	MPN	23.84	38.58	49.97	13.16
4	Modality	17.49	28.92	13.84	10.72
5	Polarity	16.46	26.36	65.15	11.58
6	Affect	18.62	31.59	59.8	13.37
7	Similarity	20.68	34.5	43.16	12.1
8	Negation	8.28	22.47	75.87	11.1
9	Length	20.75	31.28	65.72	10.19

Table 3: F-measure for each feature category

We also show results using a stop list of 50 common functional words. The stop list has only a small effect on performance except in the *temporal* class. This may be because of functional words like *was* and *will* which have a temporal effect.

5 Other Features

For our secondary evaluation, we include additional features to complement the word pairs. Previous work has relied on features based on the gold parse trees of the Penn Treebank (which overlaps with PDTB) and on contextual information from relations preceding the one being disambiguated. We intentionally limit ourselves to features that do not require either so that our system can be readily used on arbitrary argument pairs.

WordNet Features: We define four features based on WordNet (Fellbaum, 1998) - *Synonyms*, *Antonyms*, *Hypernyms* and *Hyponyms*. The values are the counts of word pairs in the cross-product of the words in the arguments that have the particular relation (synonymy, antonymy etc) between them.

Verb Class: This is the count of pairs of verbs from Arg1 and Arg2 that share the same class, defined as the highest level Levin verb class (Levin, 1993) from the LCS database (Dorr, 2001).

Money, Percentages and Numbers (MPN): The counts of currency symbols/abbreviations, percentage signs or cues ("percent", "BPS"...) and numbers in each argument.

Modality: Presence or absence of each English modal in each argument.

Polarity: Based on MPQA (Wilson et al., 2005). We include the counts of positive and negative words according to the MPQA subjectivity lexicon for both arguments. Unlike Pitler et al. (2009), we do not use neutral polarity features. We also do not explicitly group negation with polarity (although we do have separate negation features).

Affect: Based on the Dictionary of Affect in Language (Whissell, 1989). Each word in the DAL gets a score for three dimensions - *pleasantness* (pleasant - unpleasant), *activation* (passive - active) and *imagery* (hard to imagine - easy to imagine). We use the average score for each dimension in each argument as a feature.

Content Similarity: We use the cosine similarity and word overlap of the arguments as features.

Negation: Presence or absence of negation terms in each of the arguments.

Length: The ratio between the lengths (counts of words) of the arguments.

6 Evaluation of Additional Features

For our secondary evaluation, we present results for each feature category on its own in Table 3 and for our best system for each of the relation classes in Table 2. We show results for the best systems from (Pitler et al., 2009), (Zhou et al., 2010) and

(Park and Cardie, 2012) for comparison.

7 Conclusion

We presented an aggregated approach to word pair features and showed that it outperforms the previous formulation for all relation types but *contingency*. This is our main contribution. With this approach, using a stop list does not have a major effect on results for most relation classes, which suggests most of the word pairs affecting performance are content word pairs which may truly be semantically related to the discourse structure.

In addition, we introduced the new and useful *WordNet*, *Affect*, *Length* and *Negation* feature categories. Our final system outperformed the best system from Pitler et al. (2009), who used mostly similar features, for *comparison* and *temporal* and is competitive with the most recent state of the art systems for *contingency* and *expansion* without using any syntactic or context features.

Acknowledgments

This research is supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center (DoI/NBC) contract number D11PC20153. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

References

Nicholas Asher and Alex Lascarides. 2003. *Logics of Conversation.* Studies in Natural Language Processing Series. Cambridge University Press.

Or Biran and Owen Rambow. 2011. Identifying justifications in written dialog by classifying text as argumentative. *International Journal of Semantic Computing*, 5(4):363–381, December.

Sasha Blair-Goldensohn, Kathleen McKeown, and Owen Rambow. 2007. Building and refining rhetorical-semantic relation models. In *HLT-NAACL*, pages 428–435. The Association for Computational Linguistics.

Bonnie J. Dorr. 2001. *LCS Verb Database, Online Software Database of Lexical Conceptual Structures and Documentation*. University Of Maryland College Park.

Christiane Fellbaum, editor. 1998. *WordNet An Electronic Lexical Database*. The MIT Press.

Beth Levin. 1993. *English Verb Classes and Alternations: A Preliminary Investigation*. University Of Chicago Press.

Ziheng Lin, Min-Yen Kan, and Hwee Tou Ng. 2009. Recognizing implicit discourse relations in the penn discourse treebank. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 343–351.

William C. Mann and Sandra A. Thompson. 1987. Rhetorical Structure Theory: A theory of text organization. Technical Report ISI/RS-87-190, ISI.

Daniel Marcu and Abdessamad Echihabi. 2002. An unsupervised approach to recognizing discourse relations. In *ACL*, pages 368–375. ACL.

Joonsuk Park and Claire Cardie. 2012. Improving implicit discourse relation recognition through feature set optimization. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 108–112.

Emily Pitler and Ani Nenkova. 2009. Using syntax to disambiguate explicit discourse connectives in text. In *ACL/IJCNLP (Short Papers)*, pages 13–16. The Association for Computer Linguistics.

Emily Pitler, Annie Louis, and Ani Nenkova. 2009. Automatic sense prediction for implicit discourse relations in text. In *ACL/IJCNLP*, pages 683–691. The Association for Computer Linguistics.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The penn discourse treebank 2.0. In *In Proceedings of LREC*.

Cynthia M. Whissell. 1989. The dictionary of affect in language.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 347–354.

Zhi-Min Zhou, Yu Xu, Zheng-Yu Niu, Man Lan, Jian Su, and Chew Lim Tan. 2010. Predicting discourse connectives for implicit discourse relation recognition. In *Proceedings of the 23rd International Conference on Computational Linguistics*.