PDTB Discourse Parsing as a Tagging Task: The Two Taggers Approach

Or Biran
Columbia University
orb@cs.columbia.edu

Kathleen McKeown
Columbia University
kathy@cs.columbia.edu

Abstract

Full discourse parsing in the PDTB framework is a task that has only recently been attempted. We present the Two Taggers approach, which reformulates the discourse parsing task as two simpler tagging tasks: identifying the relation within each sentence, and identifying the relation between each pair of adjacent sentences. We then describe a system that uses two CRFs to achieve an F1 score of 39.33, higher than the only previously existing system, at the full discourse parsing task. Our results show that sequential information is important for discourse relations, especially cross-sentence relations, and that a simple approach to argument span identification is enough to achieve state of the art results. We make our easy to use, easy to extend parser publicly available.

1 Introduction

Discourse structure is an important part of what makes a text coherent. Parts of the text are connected to one another by what is known as *discourse relations*, such as causality, contrast, and specification. Discourse parsing is the task of automatically determining the discourse structure of a text according to a particular theory of discourse. The ability to parse an entire document is crucial for understanding its linguistic structure and the intentions of its authors.

Discourse parsing is a difficult task. While some discourse relations have explicit lexical cues called discourse *connectives* or *markers*, such as "because" and "but", these are often ambiguous: they may apply to more than one relation category, or they may be used in a way that has nothing to do with discourse at all. In addition, many relations are not marked by connectives in text, and disambiguating these *implicit* relations is difficult even when it is known a relation exists. Adding to the difficulty is the fact that the arguments of the relation (there are usually two, although some frameworks allow more for certain relations) do not necessarily correspond to sentences or clauses, and may not even be contiguous under some theories.

Over the years, multiple theories of discourse have been proposed. Most recently, the Penn Discourse Treebank (PDTB) (Prasad et al., 2008) has been introduced, featuring hierarchical relation categories which generalize over other theories such as Rhetorical Structure Theory (RST) (Mann and Thompson, 1987) and SDRT (Asher and Lascarides, 2003), as well as a relatively large annotated corpus aligned with the WSJ section of the Penn Treebank (PTB) (Marcus et al., 1993). While the relation *categories* in PDTB are hierarchical, unlike RST and other frameworks, the discourse *structure* of a PDTB document is not fully hierarchical so that documents in general do not have a tree-like discourse structure. This is a crucial detail which allows our proposed method to work on PDTB documents.

While there has been much work recently on disambiguating discourse relations in the PDTB, most have not been full parsing systems. That is, they operate in an experimental environment where some information is given (for example, some systems disambiguate only implicit relations, where it is assumed that the arguments of the relation have been identified and that the relation is known to be implicit (Pitler and Nenkova, 2009; Park and Cardie, 2012)). Full systems, in contrast, operate on unannotated text documents producing the full discourse structure of the text, including both implicit and explicit relations, and so can be realistically used in NLP applications. Although not strictly parsing in the case of PTDB, such systems perform what has been called the *end-to-end*

Proceedings of the SIGDIAL 2015 Conference, pages 96–104,
Prague, Czech Republic, 2-4 September 2015.

discourse parsing task. Interest in full discourse parsing in the PDTB has been increasing, and it is this year's CoNLL shared task.

The only work, to our knowledge, which provides end-to-end PDTB discourse parsing is (Lin et al., 2014); they use a four-stage architecture where each stage carries out one subtask in identifying discourse relations (e.g., explicit or implicit). The parser is evaluated in terms of *exact match* and *partial match*. Unlike exact match results, which are considered correct only if both the relation type and the exact span of its arguments are identified correctly, partial match results are correct as long as the relation type is correctly identified and each proposed argument shares at least one noun and verb with the true argument. We believe that partial match results are best to focus on at this point in time, since current performance on exact match results is too low to be useful. Many current NLP applications (such as summarization and question answering) focus on sentences or clauses anyway and would find this formulation natural.

In this paper, we present a simple yet powerful sequential approach to PDTB discourse parsing, utilizing two CRFs and features that are designed to discriminate both explicit and implicit relations. We surpass state-of-the-art performance with a simpler structure, less hand-crafted rules for special scenarios and with an approach that makes adding new features extremely easy.

2 Related Work

Early data-driven work on discourse has focused on frameworks such as RST, using the small RST Discourse Treebank (Carlson et al., 2001). Marcu (1997) and later Soricut and Marcu (2003) developed methods for parsing documents into the RST discourse representation. There has also been more recent work on end-to-end RST-style parsing (LeThanh et al., 2004; duVerle and Prendinger, 2009).

Recently, there has been more focus on the PDTB (Prasad et al., 2008), the largest annotated discourse corpus currently in existence. Most work so far has focused on solving specific subtasks of the overall parsing task. Pitler and Nenkova (2009) focused on explicit relations and found that they are relatively easy to disambiguate using syntactic features. Wellner (2009) used both lexical and syntactic features to identify the arguments of a relation. Identifying and disambiguating implicit relations has been the hardest task to achieve good performance at, and is an active area of research. Pitler et al. (2009) were the first to identify implicit relations in the PDTB in a realistic setting, and later work has improved on their methods as well as introducing new ideas (Lin et al., 2009; Zhou et al., 2010; Park and Cardie, 2012; Biran and McKeown, 2013; Li and Nenkova, 2014a).

Most recently, Lin et al. (2014) have introduced and evaluated the first system which provides end-to-end discourse parsing over PDTB (the *Lin parser*). In their work, they have combined much of the earlier work on specific subtasks, utilizing a connective disambiguation component and an explicit relation disambiguation component inspired by Pitler and Nenkova (2009)'s method, as well as an implicit relation disambiguation component descending from their own previous work (Lin et al., 2009). Their approach is to decipher the document in a structured way, in four steps: first, identify explicit discourse connectives; second, identify the text spans of the arguments (in PDTB, there are always two arguments, arg1 and arg2) corresponding to the connective; third, identify the type of relation between the arguments (the third step completes the subtask of finding *explicit relations*); and fourth, for every adjacent pair of sentences, identify which type of *implicit relation* - relations where there is no connective - exists between them (or, if none does, identify the relation as EntRel - meaning the sentences share an entity but not a relation, or NoRel - meaning they share nothing at all). [1]

While the structured approach of the Lin parser has many advantages in that it attempts to solve the sub-tasks of discourse parsing in an organized, intuitive way, it has some disadvantages. One is that because of the pipeline structure, errors propagate from step to step. For example, if a (truly) implicit relation was incorrectly identified as an explicit relation because of a false connective, the features used by the implicit relation identifier that may correctly discriminate its type will not get a chance to be used.

Another disadvantage is the fact that in the

[1] There is also a fifth step, identifying spans that attribute a statement to a source, e.g. "B.P. explains that ...". Attribution span detection is a secondary task which is evaluated separately from the main discourse structure pipeline, and we are not concerned with it here.

structured approach, potential relations are considered individually, although adjacent relations can intuitively be indicators of the relation type.

Finally, building such a system requires significant design and engineering, and making changes that are not localized to a specific component can be difficult and time-consuming. At this point in time, when work on discourse parsing in PDTB is at its early stage, a more flexible and easily extensible approach would be beneficial to the community.

3 Method

PDTB discourse relations can be seen as a triple: relation type, argument 1 and argument 2. While in principle, the discourse structure theory of PDTB allows for the two arguments of a discourse relation to be located anywhere, in practice 92.9% of the relations annotated either a) are wholly contained in a single sentence, or b) span two adjacent sentences, with each argument contained in one of the sentences.[2]

Given this information, and the intuition that the structure of the document as a whole (in particular, the sequence of discourse relations) can be useful for determining the type of a relation, we reformulate the task of parsing the PDTB discourse relations as the combination of two tagging tasks. For each document, we separately tag the sequence of sentences for intra-sentence relations, and the sequence of adjacent sentence pairs for cross-sentence relations. While intra-sentence relations are always explicit, adjacent sentence relations may be explicit, implicit, or fall into the PDTB's AltLex or EntRel categories. Unlike previous work, we use a single method to disambiguate all adjacent sentence relations. We call this approach to discourse parsing the *Two Taggers* approach.

As a result, we have a sequence of sentences, each tagged with the relation that exists within it and each adjacent pair tagged with the relation that exists between them. In order to transform this structure to a full discourse parse, we must also identify the arguments and their spans. Since our goal is a simpler system and our focus is on partial match results, we avoid using a complicated syntactic rule system for each possible scenario in favor of a few simple rules. For adjacent sentence relations, we mark arg1 as being the entire first sentence and arg2 as being the entire second sentence (under partial match, this turns out to be correct in all but 0.002% of relations in the training set). For single-sentence relations, we distinguish among two cases: if the first word of the sentence is an intra-sentence initial connective[3] then we identify arg2 from the beginning of the sentence until the end of the first VP, and arg1 from there to the end of the sentence. Otherwise we identify arg1 from the beginning of the sentence to the middle connective (if there are more than one) and arg2 from there to the end of the sentence. While this approach ignores many complexities of the true argument structure of PDTB (for example, arguments may be nested, and a sentence may include text that is not inside an argument), it works well for partial match. In fact, as we show in our evaluation, it is also not too far behind the state of the art on a slightly more lenient version of exact match. We use Pitler and Nenkova (2009)'s high performing connective classifier (F1 above 95) to distinguish discourse connectives from their non-discourse counterparts.

The PDTB relation categories are hierarchical, and we are interested in finding the *type*, or second-level categories, of which there are 16 (plus EntRel and NoRel, for a total of 18). The first level (the *class*, of which there are 4) is too coarse to be useful for many applications, and the third level (the *subtype*, of which there are 25) is too fine-grained and difficult to disambiguate. Table 1 shows the hierarchy of 4 classes and 16 types. The Lin parser also deals with type-level categories, but almost all other previous work has focused on the significantly easier class-level categories.

Treating discourse parsing as a tagging problem has many advantages. Tagging tasks have been widely explored in NLP and there are many off-the-shelf tools and methods for tackling them. Many generic taggers that can be applied to this task with minimal effort are available to researchers, while generic parsers do no exist. Tagging is a simpler and often more tractable task than parsing, and it can be done using sequential classifiers, which are both fast and powerful.

There are also some limitations to the tagging

[2] It should be noted that by the definition given in the annotation manual, all implicit relations in PDTB exist between arguments contained within two adjacent sentences.

[3] *After, although, as, because, before, except, if, since, though, unless, until, when, whereas,* and *while* (as well as variations such as *if and when*).

approach. As mentioned earlier, some rare relations span more than two sentences, or sentences that are not adjacent. In addition, there are (also rare) situations where there are multiple relations in a single sentence, and with our approach we can at most tag one correctly. Because of these two limitations, we have an upper bound on F-measure performance of 89.4 in the PDTB corpus. Since current state-of-the-art performance is far below this level, we do not view this as an urgent problem. At any rate, additional specialized approaches can be added to correctly handle those rare cases.

In this paper, we use Conditional Random Fields (CRFs) for both taggers. CRFs were first introduced by Lafferty et al. (2001) and have been successfully used for many NLP tagging tasks such as named entity recognition (McCallum and Li, 2003) and shallow parsing (Sha and Pereira, 2003). We use simple linear-chain CRFs for both taggers. In the linear-chain CRF model, the posterior probabilities for an ordered sequence input $\mathbf{x} = \{x_1, \ldots, x_{|x|}\}$ of tag labels $\mathbf{y} = \{y_1, \ldots, y_{|x|}\}$ are defined as

$$P(\mathbf{y}|\mathbf{x}) \propto \prod_{i=1}^{|x|} exp\left(\sum_{k=1}^{K} \theta_k \Phi_k(y_{i-1}, \mathbf{x})\right)$$

where θ_k are weights corresponding to the features Φ_k. The feature values at index i of the sequence may be computed based on the previous tag in the sequence y_{i-1} and the entire sequence $\mathbf{x}$. The weights θ_k are estimated using gradient descent to maximize the likelihood of the input.

In our formulation, each $\mathbf{x}$ is a PDTB document, consisting of a sequence of sentences (for the intra-sentence relation tagger) or a sequence of sentence pairs (for the adjacent sentence relation tagger). $\mathbf{y}$ consists of all type-level discourse relation categories.

In our experiments, we used a maximum likelihood prior and limited the gradient descent to a maximum of 200 epochs instead of waiting for it to converge.

While CRFs have been used in the past for subtasks of RST discourse parsing (Feng and Hirst, 2014) and for finding the arguments of explicit relations in PDTB (Ghosh et al., 2011), no sequential approaches have ever been used in a way that models the sequential dependency between PDTB relations. Previous work (Pitler et al., 2009; Zhou et al., 2010) has utilized features that consider adjacent lexical information in relation type classification, but true sequential or joint classifications have not been attempted.

Class (Level 1)	Type (Level 2)
Comparison	Concession
	Contrast
	Pragmatic Concession
	Pragmatic Contrast
Contingency	Cause
	Condition
	Pragmatic Cause
	Pragmatic Condition
Expansion	Alternative
	Conjunction
	Exception
	Instantiation
	List
	Restatement
Temporal	Asynchronous
	Synchrony

Table 1: The PTDB relation category hierarchy, with level 1 classes and level 2 types. The level 3 subtypes are not shown

4 Features

4.1 Intra-sentence tagger

The intra-sentence tagger deals only with explicit relations, and as such focuses on features related to discourse connectives. We use Pitler and Nenkova (2009)'s connective classifier to identify discourse connectives within the sentence, and for each connective generate the following binary features:

- Connective
- Previous word + connective
- Connective + next word
- Connective's syntactic category
- Parent's category
- Left sibling's category
- Right sibling's category
- Path to root
- Compressed path to root

All of which are features used in explicit relation detection by Pitler and Nenkova (2009) or by Lin et al. (2014).

4.2 Adjacent sentence tagger

The adjacent sentence tagger utilizes a larger variety of features, designed to disambiguate relations across sentences that may be explicit, implicit, AltLex or EntRel.

We divide the features into four thematic types: lexical, connective-related, syntactic and structural features.

4.2.1 Lexical features

Lexical features are based on the surface lexical terms of the sentence pair.

In addition to **unigrams** and **bigrams**, we make use of **word pair similarity features**, the set of features described in Biran and McKeown (2013), which utilize sets of word pairs that were mined from unannotated corpora around each discourse connective. The word pair scores within the set are given by TF*IDF and treated as a vector. The feature value is the cosine similarity of the connective's vector to the vector of word pairs extracted from the pair of adjacent sentences, where each pair contains one word from each sentence. It models the similarity of the sentence pair to a sentence where the connective is used directly, and is intended to help in identifying implicit relations. We also add a variation on these features: the **word pair similarity average for connective pair**, where we get the similarities of the adjacent sentence pair to the word pair sets of a couple of connectives (we use every possible combination of two connectives) and use the average as the feature value. The idea is that if two connectives are related to the same relation type, a high average similarity to both may be a stronger indicator for that relation.

We also utilize a simplistic form of topic centrality. **Centrality in document** is the cosine similarity of the sentence pair to the document as a whole. The intuition is that certain relations (e.g., argumentative relations such as causality and concession) would tend to be more common around the main topic of the document.

Finally, we include features for words that are shared by both sentences called **expanded shared words** - expanded because we use WordNet (Fellbaum, 1998) to expand the usual list of words in each sentence with all synonyms and immediate hypernyms of each word's most frequent sense.

4.2.2 Connective features

For each sentence separately, we find all connectives (using Pitler and Nenkova (2009)'s connective classifier), and use the **connective** itself as a feature, as well as the **previous word and the connective**, which includes cases where the previous word is the implicit [START] (when the connective is the first word of the sentence). These features are mainly useful for disambiguating cross-sentence explicit relations.

4.2.3 Syntactic features

Syntactic features are derived from the parse tree of the sentence. We use the Stanford Parser (Klein and Manning, 2003) to derive the trees. Unlike much previous work, we do not use the gold parse trees of the PTB.

Lin et al. (2009) introduced the *production rule* features, which are some of the strongest for implicit relation disambiguation. Production rules are all parent-children relations in the constituent parse of a sentence, e.g. [VP → NP PP NP]. The binary feature formulation includes the existence of each rule in arg1, in arg2, and in both. Li and Nenkova (2014b) hypothesized that production rules are too sparse, and found that using their *production stick* features achieved higher performance. Unlike a production rule, which relates to all children of a parent, a production stick is a parent-single child relation. We experimented with both feature sets, and found that we achieve the best performance with a novel middle-ground formulation. **Production angles** are a family of features indicating the appearance of syntactic triples: a parent and two adjacent children. In cases where a parent has only one child, as in the lexical leaf nodes of the tree, we produce a stick-like feature (e.g. [NP → resources]. The triples are formed using the label of each node and the descendant directionality. We use features for angles in each sentence separately, as well as for angles that are shared by both.

4.2.4 Structural features

Structural features are related to the structure of the document. One intuitively important feature is the **paragraph split** feature which indicates whether the pair is split across two paragraphs or not. We also use a binary feature that specifies

whether the sentence pair is in a **short document** (three sentences or less).

4.3 Sequential features

Sequential features are the *transitional* features that consider the previous tag in the sequence. The same sequential features are used in both taggers.

We use two basic pieces of information from the previous tag: the **previous tag type** is the type (second-level relation category) of the previous tag, while the **previous tag class** is the class (first-level relation category) of the previous tag.

5 Evaluation

Following Lin et al. (2014) and other previous work, we use sections 2-21 of the PDTB as the training set, section 22 as the development set, and section 23 as the test set. Since we use an automatic parser for our syntactic features, our results are equivalent to Lin et al.'s "Partial, Auto + EP" overall results for partial match, and to their "Exact, Auto + EP" results for exact match. We consider the results using gold standard parses to be less important for an end-to-end system, the main function of which is an out of the box document parsing tool. The evaluation metric in all experiments, following Lin et al., is the micro-averaged F1 score.

We show our final partial match results on the test set in Table 2, compared with the Lin Parser performance. We also compare our approach with the results achieved by using the exact same formulation and features (other than the sequential features, of course) in two Logistic Regression classifiers, to show that the sequential approach is in fact helpful. To illustrate the effect of our simplistic argument span identification rules, we also show results without span matching, where argument spans are presumed to always partially match if the sentence/sentences and relation type are correctly identified.

The results of each tagger individually are shown in Table 3. Note that the overall results are compared against all true relations in the document, including those that our method inherently cannot identify (hence the upper bound), while the individual tagger results are only in the context of the individual tagging task. This is why the recall of the end-to-end results is smaller than the recall of either of the individual taggers.

While we are focused on partial match results, we also show exact match results in Table 4. In error analysis we noticed that many of our errors on exact match arise because we include in the span another discourse connective, or an initial word like "Eventually" or "Admittedly" in a non-discourse usage. We therefore include another set of results we call "almost-exact match" which allows a match if there is at most one word at the beginning or the end of the span that does not match. Using this less strict definition, we reach a performance that comes close to the Lin parser exact match results.

To emphasize how much harder it is to identify the level 2 relation types than it is to identify the level 1 classes, we also provide results on the class-level discourse parsing task in Table 5.

	Prec.	Recall	F1
Two classifiers	46.12	31.68	37.56
Lin Parser			38.18
Two Taggers	48.52	33.06	**39.33**
No span matching	48.72	33.32	39.57
Upper bound	100	80.82	89.40

Table 2: Partial match results on all relations in the PDTB. The Lin parser paper does not report precision and recall

	Prec.	Recall	F1
Intra-sent. tagger	66.36	49.82	**56.91**
Intra-sent. classifier	66.19	48.77	56.16
Adj. sent. tagger	40.31	36.53	**38.33**
Adj. sent. classifier	37.13	34.21	35.61

Table 3: Results for each of the two taggers separately

5.1 Discussion

As seen in Table 2, we achieve higher performance than the Lin parser on partial match results. This is despite the fact that we use fewer manually-crafted

	Prec.	Recall	F1
2T exact match	14.47	5.93	8.41
2T almost-exact match	29.61	14.75	19.69
Lin Parser			20.64

Table 4: Exact match results on all relations in the PDTB. The Lin parser paper does not report precision and recall

	Prec.	Recall	F1
Two Taggers	62.56	44.3	**51.87**
Upper bound	100	80.82	89.40

Table 5: Results for the same task when using the level 1 classes instead of the level 2 type relation categories

rules and do not rely on a complex argument span identification component. Moreover, the two taggers are clearly stronger than two classifiers with identical features, especially for the adjacent sentence task, which shows that there is value to the sequential approach.

It is clear from Table 3 that identifying relations in adjacent sentence pairs is a more difficult task than identifying them inside a single sentence. This makes sense because single sentence relations are always explicit in the PDTB while most adjacent sentence relations are implicit. It is well established that implicit relations are much harder to disambiguate than explicit ones. While we cannot provide an evaluation for implicit relations only - it is not clear how to fairly define false positives since we tag the entire document without differentiating between explicit and implicit relations - we can provide a lower bound for our performance by using only implicit relations to collect the true positives and false negatives, and all tagged relations to collect false positives. Our lower bound F-measure for implicit relations is 28.32.[4] In the Lin parser, the F-measure performance of the implicit relation classifier is 25.46, while the explicit relation classifier has an F-measure over 80. These numbers imply that our method is especially advantageous for implicit relations, while explicit relations may be harder to disambiguate without the specialized argument location/span identification step taken by the Lin parser. In addition, the relations that our approach inherently cannot handle are all explicit.

It is interesting to note that the difference between the taggers and the classifiers is much larger for the adjacent sentence pairs, meaning that the sequential features are very strong in the adjacent sentences tagger. This may indicate that intra-sentence relations are more "stand-alone" in nature while inter-sentence relations are more connected with the rest of the document. This result, and the fact that our performance on intra-sentence relations are not as high as previous results on explicit relations, suggest that one promising path for future work is the combination of a more structured intra-sentence explicit relation approach (one that would, among other advantages, allow finding multiple relations within the same sentence) with a sequential adjacent-sentence approach. Our performance suggests that this separation (intra-sentence and adjacent sentence) in methodology, which allows a sequential view, may in some cases be more useful than the traditional explicit vs. implicit separation.

Our approach beats state-of-the-art performance using partial match, which is the natural evaluation to use at this point in time given exact match performance (this view has been expressed by Lin et al. (2014) as well). While we do not achieve the same results on exact match, which is to be expected given our very simple approach to argument span identification, Table 4 shows that we come very close if a slightly less restrictive evaluation is used. This reaffirms the conclusion that exact match is a very difficult task: even with complex hand-crafted syntactic rules, correctly identified spans are relatively simple cases which can also be identified (if a single word error is allowed) by a much simpler method.

Table 5 illustrates how much harder the type-level parsing task is than the class-level parsing task. While it is possible that the class-level parsing can be useful for some downstream applications, we believe that the more granular type-level parsing is a better choice for properly understanding a document's discourse structure.

6 Conclusion and Future Work

We presented a reformulation of the PTDB discourse parsing task as two simple tagging tasks. This formulation makes it easier to approach the task and can be used as a convenient way to evaluate new ideas and features as they arise. Using chain-CRFs to implement this approach, we surpass state-of-the-art performance at the overall parsing task. While we used some of the strongest features that have shown up in the literature in this evaluation, there are many immediate candidate methods for improving the results, such as adding more specific features for the various grammatical classes of explicit connectives described in the PDTB.

[4] Precision is 28.02 and recall is 28.63.

Our results show that treating the task as sequential is useful. One interesting direction for continuing this research is to transform the two tagging tasks into two joint prediction tasks, and perhaps eventually into one joint prediction task.

While we build on previous work in defining our features, we also introduced some novel variations. We have defined the *production angles* family of features, which are related to the *production rules* of Lin et al. (2009) and the *production sticks* of Li and Nenkova (2014b). We also contribute to the *word pair features* line of research, which started with Marcu and Echihabi (2002) and has been part of most work on implicit relation disambiguation since, with our variations on the dense word pair similarity features introduced by Biran and McKeown (2013). Our *expanded shared words* features are also novel.

Our main aim in this paper was to show that experiments with discourse parsing can be done fairly easily using one of the many freely available sequential models. We hope that this method will make the task more accessible to researchers and help in moving towards a fully statistical and holistic approach to discourse parsing. The parser described in this paper is publicly available at `www.cs.columbia.edu/~orb`.

References

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

Or Biran and Kathleen McKeown. 2013. Aggregated word pair features for implicit discourse relation disambiguation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 2: Short Papers*, pages 69–73. The Association for Computer Linguistics.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2001. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Proceedings of the Second SIGdial Workshop on Discourse and Dialogue - Volume 16*, SIGDIAL '01, pages 1–10, Stroudsburg, PA, USA. Association for Computational Linguistics.

David A. duVerle and Helmut Prendinger. 2009. A novel discourse parser based on support vector machine classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, pages 665–673, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

Vanessa Wei Feng and Graeme Hirst. 2014. A Linear-Time Bottom-Up Discourse Parser with Constraints and Post-Editing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 511–521, Baltimore, Maryland, June. Association for Computational Linguistics.

Sucheta Ghosh, Richard Johansson, Giuseppe Riccardi, and Sara Tonelli. 2011. Shallow discourse parsing with conditional random fields. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1071–1079.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 423–430, Stroudsburg, PA, USA. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Huong LeThanh, Geetha Abeysinghe, and Christian Huyck. 2004. Generating discourse structures for written texts. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jessy Junyi Li and Ani Nenkova. 2014a. Addressing class imbalance for improved recognition of implicit discourse relations. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 142–150. Association for Computational Linguistics.

Jessy Junyi Li and Ani Nenkova. 2014b. Reducing sparsity improves the recognition of implicit discourse relations. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 199–207. Association for Computational Linguistics.

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.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2014. A pdtb-styled end-to-end discourse parser. *Natural Language Engineering*, 20(2):151–184.

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.

Daniel Marcu. 1997. The rhetorical parsing, summarization, and generation of natural language texts. Technical report.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.*, 19(2):313–330, June.

Andrew McCallum and Wei Li. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 188–191, Stroudsburg, PA, USA. Association for Computational Linguistics.

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*.

Fei Sha and Fernando Pereira. 2003. Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 134–141, Stroudsburg, PA, USA. Association for Computational Linguistics.

Radu Soricut and Daniel Marcu. 2003. Sentence level discourse parsing using syntactic and lexical information. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 149–156, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ben Wellner. 2009. *Sequence Models and Ranking Methods for Discourse Parsing*. Ph.D. thesis, Waltham, MA, USA. AAI3339383.

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*.