# Recognizing Implicit Discourse Relations in the Penn Discourse Treebank

**Ziheng Lin, Min-Yen Kan** and **Hwee Tou Ng**
Department of Computer Science
National University of Singapore
13 Computing Drive
Singapore 117417
{linzihen,kanmy,nght}@comp.nus.edu.sg

## Abstract

We present an implicit discourse relation classifier in the Penn Discourse Treebank (PDTB). Our classifier considers the context of the two arguments, word pair information, as well as the arguments' internal constituent and dependency parses. Our results on the PDTB yields a significant 14.1% improvement over the baseline. In our error analysis, we discuss four challenges in recognizing implicit relations in the PDTB.

## 1 Introduction

In the field of discourse modeling, it is widely agreed that text is not understood in isolation, but in relation to its context. One focus in the study of discourse is to identify and label the relations between textual units (clauses, sentences, or paragraphs). Such research can enable downstream natural language processing (NLP) such as summarization, question answering, and textual entailment. For example, recognizing causal relations can assist in answering *why* questions. Detecting contrast and restatements is useful for paraphrasing and summarization systems. While different discourse frameworks have been proposed from different perspectives (Mann and Thompson, 1988; Hobbs, 1990; Lascarides and Asher, 1993; Knott and Sanders, 1998; Webber, 2004), most admit these basic types of discourse relationships between textual units.

When there is a discourse connective (e.g., *because*) between two text spans, it is often easy to recognize the relation between the spans, as most connectives are unambiguous (Miltsakaki et al., 2005; Pitler et al., 2008). On the other hand, it is difficult to recognize the discourse relations when there are no explicit textual cues. We term these cases explicit and implicit relations, respectively. While the recognition of discourse structure has been studied in the context of explicit relations (Marcu, 1998) in the past, little published work has yet attempted to recognize implicit discourse relations between text spans.

Detecting implicit relations is a critical step in forming a discourse understanding of text, as many text spans do not mark their discourse relations with explicit cues. Recently, the Penn Discourse Treebank (PDTB) has been released, which features discourse level annotation on both explicit and implicit relations. It provides a valuable linguistic resource towards understanding discourse relations and a common platform for researchers to develop discourse-centric systems. With the recent release of the second version of this corpus (Prasad et al., 2008), which provides a cleaner and more thorough implicit relation annotation, there is an opportunity to address this area of work.

In this paper, we provide classification of implicit discourse relations on the second version of the PDTB. The features we used include contextual modeling of relation dependencies, features extracted from constituent parse trees and dependency parse trees, and word pair features. We show an accuracy of 40.2%, which is a significant improvement of 14.1% over the majority baseline.

After reviewing related work, we first give an overview of the Penn Discourse Treebank. We then describe our classification methodology, followed by experimental results. We give a detailed discussion on the difficulties of implicit relation classification in the PDTB, and then conclude the paper.

## 2 Related Work

One of the first works that use statistical methods to detect implicit discourse relations is that of Marcu and Echihabi (2002). They showed that word pairs extracted from two text spans provide clues for detecting the discourse relation between the text spans. They used a set of textual patterns to automatically construct a large corpus of text span pairs from the web. These text spans were assumed to be instances of specific discourse relations. They removed the discourse connectives from the pairs to form an implicit relation corpus. From this corpus, they collected word pair statistics, which were used in a Naïve Bayes framework to classify discourse relations.

Saito et al. (2006) extended this theme, to show that phrasal patterns extracted from a text span pair provide useful evidence in the relation classification. For example, the pattern "... should have done ..." usually signals a contrast. The authors combined word pairs with phrasal patterns, and conducted experiments with these two feature classes to recognize implicit relations between adjacent sentences in a Japanese corpus.

Both of these previous works have the shortcoming of downgrading explicit relations to implicit ones by removing the explicit discourse connectives. While this is a good approach to automatically create large corpora, natively implicit relations may be signaled in different ways. The fact that explicit relations are explicitly signaled indicates that such relations need a cue to be unambiguous to human readers. Thus, such an artificial implicit relation corpus may exhibit marked differences from a natively implicit one. We validate this claim later in this work.

Wellner et al. (2006) used multiple knowledge sources to produce syntactic and lexico-semantic features, which were then used to automatically identify and classify explicit and implicit discourse relations in the Discourse Graphbank (Wolf and Gibson, 2005). Their experiments show that discourse connectives and the distance between the two text spans have the most impact, and event-based features also contribute to the performance. However, their system may not work well for implicit relations alone, as the two most prominent features only apply to explicit relations: implicit relations do not have discourse connectives and the two text spans of an implicit relation are usually adjacent to each other.

The work that is most related to ours is the forthcoming paper of Pitler et al. (2009) on implicit relation classification on the second version of the PDTB. They performed classification of implicit discourse relations using several linguistically informed features, such as word polarity, verb classes, and word pairs, showing performance increases over a random classification baseline.

## 3 Overview of the Penn Discourse Treebank

The Penn Discourse Treebank (PDTB) is a discourse level annotation (Prasad et al., 2008) over the one million word *Wall Street Journal* corpus. The PDTB adopts the predicate-argument view of discourse relations, where a discourse connective (e.g., *because*) is treated as a predicate that takes two text spans as its arguments. The argument that the discourse connective structurally attaches to is called Arg2, and the other argument is called Arg1. The PDTB provides annotations for explicit and implicit discourse relations. By definition, an explicit relation contains an explicit discourse connective. In the PDTB, 100 explicit connectives are annotated. Example 1 shows an explicit Contrast relation signaled by the discourse connective *but*. The last line shows the relation type and the file in the PDTB from which the example is drawn.

(1) **Arg1:** In any case, the brokerage firms are clearly moving faster to create new ads than they did in the fall of 1987.
**Arg2:** **But** it remains to be seen whether their ads will be any more effective.
(Contrast - wsj_2201)

In the PDTB, implicit relations are constrained by *adjacency*: only pairs of adjacent sentences within paragraphs are examined for the existence of implicit relations. When an implicit relation was inferred by an annotator, he/she inserted an implicit connective that best reflects the relation. Example 2 shows an implicit relation, where the annotator inferred a Cause relation and inserted an implicit connective *so* (i.e., the original text does not include *so*). The text in the box (*he says*) shows the attribution, i.e., the agent that expresses the arguments. The PDTB provides annotation for the attributions and supplements of the arguments.

(2) **Arg1:** "A lot of investor confidence comes from the fact that they can speak to us," $\boxed{\text{he says}}$.
**Arg2:** [**so**] "To maintain that dialogue is absolutely crucial."
(Cause - wsj_2201)

The PDTB provides a three level hierarchy of relation tags for its annotation. The first level consists of four major relation *classes*: Temporal, Contingency, Comparison, and Expansion. For each class, a second level of *types* is defined to provide finer semantic distinctions. A third level of *subtypes* is defined for only some types to specify the semantic contribution of each argument. Relation classes and types in the PDTB are reproduced in the first two columns of Table 1.

We focus on implicit relation classification of the Level 2 types in the PDTB, as we feel that Level 1 classes are too general and coarse-grained for downstream applications, while Level 3 subtypes are too fine-grained and are only provided for some types. Table 1 shows the distribution of the 16 Level 2 relation types of the implicit relations from the training sections, i.e., Sections 2 – 21. As there are too few training instances for Condition, Pragmatic Condition, Pragmatic Contrast, Pragmatic Concession, and Exception, we removed these five types from further consideration. We thus use the remaining 11 Level 2 types in our work. The initial distribution and adjusted distribution are shown in the last two columns of the table. We see that the three predominant types are Cause (25.63%), Conjunction (22.25%), and Restatement (19.23%).

| Level 1 Class | Level 2 Type | Training instances | % | Adjusted % |
|---|---|---|---|---|
| Temporal | Asynchronous | 583 | 4.36 | 4.36 |
| | Synchrony | 213 | 1.59 | 1.59 |
| Contingency | Cause | 3426 | 25.61 | 25.63 |
| | Pragmatic Cause | 69 | 0.52 | 0.52 |
| | Condition | 1 | 0.01 | – |
| | Pragmatic Condition | 1 | 0.01 | – |
| Comparison | Contrast | 1656 | 12.38 | 12.39 |
| | Pragmatic Contrast | 4 | 0.03 | – |
| | Concession | 196 | 1.47 | 1.47 |
| | Pragmatic Concession | 1 | 0.01 | – |
| Expansion | Conjunction | 2974 | 22.24 | 22.25 |
| | Instantiation | 1176 | 8.79 | 8.80 |
| | Restatement | 2570 | 19.21 | 19.23 |
| | Alternative | 158 | 1.18 | 1.18 |
| | Exception | 2 | 0.01 | – |
| | List | 345 | 2.58 | 2.58 |
| Total | | 13375 | | |
| Adjusted total | | 13366 | | |

Table 1: Distribution of Level 2 relation types of implicit relations from the training sections (Sec. 2 – 21). The last two columns show the initial distribution and the distribution after removing the five types that have only a few training instances.

## 4 Methodology

Our implicit relation classifier is built using supervised learning on a maximum entropy classifier. As such, our approach processes the annotated argument pairs into binary feature vectors suitable for use in training a classifier. Attributions and supplements are ignored from the relations, as our system does not make use of them. We chose the following four classes of features as they represent a wide range of information – contextual, syntactic, and lexical – that have been shown to be helpful in previous works and tasks. We now discuss the four categories of features used in our framework.

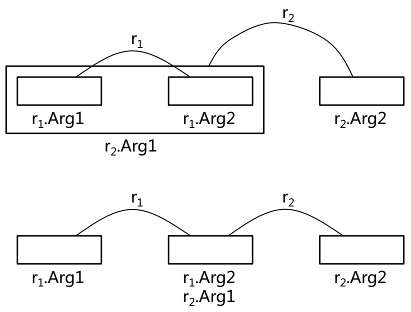

Figure 1: Two types of discourse dependency structures. Top: fully embedded argument, bottom: shared argument.

**Contextual Features.** Lee et al. (2006) showed that there are a variety of possible dependencies between pairs of discourse relations: independent, fully embedded argument, shared argument, properly contained argument, pure crossing, and partially overlapping argument. They argued that the last three cases – properly contained argument, pure crossing, and partially overlapping argument – can be factored out by appealing to discourse notions such as anaphora and attribution. Moreover, we also observed from the PDTB corpus that fully embedded argument and shared argument are the most common patterns, which are shown in Figure 1. The top portion of Figure 1 shows a case where relation $r_1$ is fully embedded in Arg1 of relation $r_2$, and the bottom portion shows $r_1$ and $r_2$ sharing an argument. We model these two patterns as contextual features. We believe that these discourse dependency patterns between a pair of adjacent relations are useful in identifying the relations. For example, if we have three items in a list, according to the PDTB binary predicate-argument definitions, there will be a List relation between the first item and the second item, and another List relation between the previous List relation and the third item, where the previous List relation is fully embedded in Arg1 of the current List relation. As we are using the gold standard argument segmentation from the PDTB, we can extract and leverage these dependency patterns. For each relation *curr*, we use the previous relation *prev* and the next relation *next* as evidence to fire six binary features, as defined in Table 2.

Note that while *curr* is an implicit relation to be classified, both *prev* and *next* can be implicit or explicit relations. Pitler et al. (2008) showed that the type of a relation sometimes correlates to the type of its adjacent relation. When the adjacent relation is explicit, its type may be suggested by its discourse connective. Thus we include another two groups of contextual features representing the connectives of *prev* and *next* when they are explicit relations.

| |
|---|
| **Fully embedded argument:** |
| prev embedded in curr.Arg1 |
| next embedded in curr.Arg2 |
| curr embedded in prev.Arg2 |
| curr embedded in next.Arg1 |
| **Shared argument:** |
| prev.Arg2 = curr.Arg1 |
| curr.Arg2 = next.Arg1 |

Table 2: Six contextual features derived from two discourse dependency patterns. *curr* is the relation we want to classify.

**Constituent Parse Features.** Research work from other NLP areas, such as semantic role labeling, has shown that features derived from syntactic trees are useful in semantic understanding. Such features include syntactic paths (Jiang and Ng, 2006) and tree fragments (Moschitti, 2004). From our observation of the PDTB relations, syntactic structure within one argument may constrain the relation type and the syntactic structure of the other argument. For example, the constituent parse structure in Figure 2(a) usually signals an Asynchronous relation when it appears in Arg2, as shown in Example 3, while the structure in Figure 2(b) usually acts as a clue for a Cause relation when it appears in Arg1, as shown in Example 4. In both examples, the lexicalized parts of the parse structure are bolded.

(3) **Arg1:** But the RTC also requires "working" capital to maintain the bad assets of thrifts that are sold
**Arg2:** [**subsequently**] That debt would be paid off **as** the assets are sold
(Asynchronous - wsj_2200)

(4) **Arg1:** It would **have been** too late to think about on Friday.
**Arg2:** [**so**] We had to think about it ahead of time.
(Cause - wsj_2201)

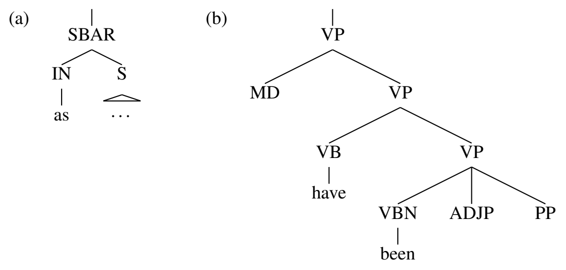

Figure 2: (a) constituent parse in Arg2 of Example 3, (b) constituent parse in Arg1 of Example 4.

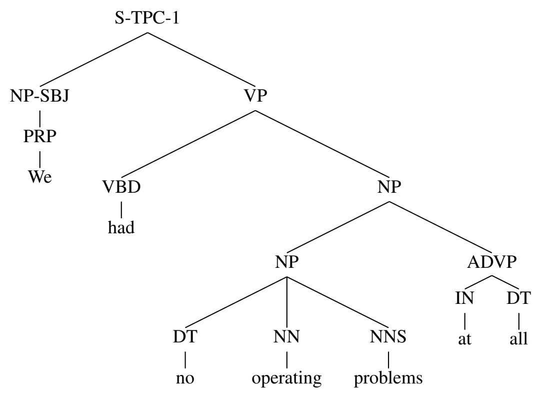

Figure 3: A gold standard subtree for Arg1 of an implicit discourse relation from wsj_2224.

For Arg1 and Arg2 of each relation, we extract the corresponding gold standard syntactic parse trees from the corpus. As an argument can be a single sentence, a clause, or multiple sentences, this results in a whole parse tree, parts of a parse tree, or multiple parse trees. From these parses, we extract all possible production rules. Although the structures shown in Figure 2 are tree fragments, tree fragments are not extracted as production rules act as generalization of tree fragments. As an example, Figure 3 shows the parse tree for Arg1 of an implicit discourse relation from the text wsj_2224. As Arg1 is a clause, the extracted tree is a subtree. We then collect all production rules from this subtree, with function tags (e.g., SBJ) removed from internal nodes. POS tag to word production rules are collected as well. The resulting production rules include ones such as: S $\rightarrow$ NP VP, NP $\rightarrow$ PRP, PRP $\rightarrow$ "We", etc. Each production rule is represented as three binary features to check whether this rule appears in Arg1, Arg2, and both arguments.

**Dependency Parse Features.** We also experimented with features extracted from dependency trees of the arguments. We used the Stanford dependency parser (de Marneffe et al., 2006), which takes in a constituent parse tree and produces a dependency tree. Again, for an argument, we may collect a whole dependency tree, parts of a tree, or multiple trees, depending on the span of the argument. The reason for using dependency trees is that they encode additional information at the word level that is not explicitly present in the constituent trees. From each tree, we collect all words with the dependency types from their dependents. Figure 4 shows the dependency subtree for the same example in Figure 3, from which we collect three dependency rules: "had" $\leftarrow$ nsubj dobj, "problems" $\leftarrow$ det nn advmod, "at" $\leftarrow$ dep.

Note that unlike the constituent parse features which are guaranteed to be accurate (as they are extracted from the gold parses of the corpus), the dependency parses occasionally contain errors. As with the constituent parse features, each dependency rule is represented as three binary features to check whether it appears in Arg1, Arg2, and both arguments.

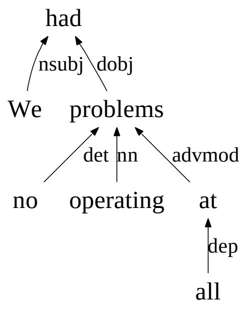

Figure 4: A dependency subtree for Arg1 of an implicit discourse relation from wsj_2224.

**Lexical Features.** Marcu and Echihabi (2002) demonstrated that word pairs extracted from the respective text spans are a good signal of the discourse relation between arguments. Thus we also consider word pairs as a feature class. We stemmed and collected all word pairs from Arg1 and Arg2, i.e., all $(w_i, w_j)$ where $w_i$ is a word from Arg1 and $w_j$ a word from Arg2. Unlike their study, we limit the collection of word pair statistics to occurrences only in the PDTB corpus.

### 4.1 Feature Selection

For the collection of production rules, dependency rules, and word pairs, we used a frequency cutoff of 5 to remove infrequent features. From the implicit relation dataset of the training sections (i.e., Sec. 2 – 21), we extracted 11,113 production rules, 5,031 dependency rules, and 105,783 word pairs in total. We applied mutual information (MI) to these three classes of features separately, resulting in three ranked lists. A feature $f$ has 11 MI values with all 11 types (for example, $MI(f, Cause)$ and $MI(f, Restatement)$), and we used the MI with the highest value for a feature to select features. In our experiments, the top features from the lists are used in the training and test phases.

## 5 Experiments

We experimented with a maximum entropy classifier from the OpenNLP MaxEnt package using various combinations of features to assess their efficacy. We used PDTB Sections 2 – 21 as our training set and Section 23 as the test set, and only used the implicit discourse relations.

In the PDTB, about 2.2% of the implicit relations are annotated with two types, as shown in Example 7 in Section 6. During training, a relation that is annotated with two types is considered as two training instances, each with one of the types. During testing, such a relation is considered one test instance, and if the classifier assigns either of the two types, we consider it as correct. Thus, the test accuracy is calculated as the number of correctly classified test instances divided by the total number of test instances.

In our work, we use the majority class as the baseline, where all instances are classified as Cause. This yields an accuracy of 26.1% on the test set. A random baseline yields an even lower accuracy of 9.1% on the test set.

### 5.1 Results and Analysis

To check the efficacy of the different feature classes, we trained individual classifiers on all features within a single feature class (Rows 1 to 4 in Table 3) as well as a single classifier trained with all features from all feature classes (Row 5). Among the four individual feature classes, production rules and word pairs yield significantly better performance over the baseline with $p < 0.01$ and $p < 0.05$ respectively, while context features perform slightly better than the baseline.

| | # Production rules | # Dependency rules | # Word pairs | Context | Acc. |
|---|---|---|---|---|---|
| R1 | 11,113 | – | – | No | 36.7% |
| R2 | – | 5,031 | – | No | 26.0% |
| R3 | – | – | 105,783 | No | 30.3% |
| R4 | – | – | – | Yes | 28.5% |
| R5 | 11,113 | 5,031 | 105,783 | Yes | 35.0% |

Table 3: Classification accuracy with all features from each feature class. Rows 1 to 4: individual feature class; Row 5: all feature classes.

Interestingly, we noted that the performance with all dependency rules is slightly lower than the baseline (Row 2), and applying all feature classes does not yield the highest accuracy (Row 5), which we suspected were due to noise. To confirm this, we employed MI to select the top 100 production rules and dependency rules, and the top 500 word pairs (as word pairs are more sparse). We then repeated the same set of experiments, as shown in Table 4 (Row 4 of this table is repeated from Table 3 for consistency). With only the top features, production rules, dependency rules, and word pairs all gave significant improvement over the baseline with $p < 0.01$. When we used all feature classes, as in the last row, we obtained the highest accuracy of 40.2%.

| | # Production rules | # Dependency rules | # Word pairs | Context | Acc. |
|---|---|---|---|---|---|
| R1 | 100 | – | – | No | 38.4% |
| R2 | – | 100 | – | No | 32.4% |
| R3 | – | – | 500 | No | 32.9% |
| R4 | – | – | – | Yes | 28.5% |
| R5 | 100 | 100 | 500 | Yes | 40.2% |

Table 4: Classification accuracy with top rules/word pairs for each feature class. Rows 1 to 4: individual feature class; Row 5: all feature classes.

Table 4 also validates the pattern of predictiveness of the feature classes: production rules contribute the most to the performance individually, followed by word pairs, dependency rules, and finally, context features. A natural question to ask is whether any of these feature classes can be omitted to achieve the same level of performance as the combined classifier. To answer this question, we conducted a final set of experiments, in which we gradually added in feature classes in the order of their predictiveness (i.e., production rules $\succ$ word pairs $\succ$ dependency rules $\succ$ context features), with results shown in Table 5. These results confirm that each additional feature class indeed contributes a marginal performance improvement, (although it is not significant) and that all feature classes are needed for optimal performance.

| | # Production rules | # Dependency rules | # Word pairs | Context | Acc. |
|---|---|---|---|---|---|
| R1 | 100 | – | – | No | 38.4% |
| R2 | 100 | – | 500 | No | 38.9% |
| R3 | 100 | 100 | 500 | No | 39.0% |
| R4 | 100 | 100 | 500 | Yes | 40.2% |

Table 5: Accuracy with feature classes gradually added in the order of their predictiveness.

Note that Row 3 of Table 3 corresponds to Marcu and Echihabi (2002)'s system which applies only word pair features. The difference is that they used a Naïve Bayes classifier while we used a maximum entropy classifier. As we did not implement their Naïve Bayes classifier, we compare their method's performance using the result from Table 3 Row 3 with ours from Table 5 Row 4, which shows that our system significantly ($p < 0.01$) outperforms theirs.

| Level 2 Type | Precision | Recall | $F_1$ | Count in test set |
|---|---|---|---|---|
| Asynchronous | 0.50 | 0.08 | 0.13 | 13 |
| Synchrony | – | – | – | 5 |
| Cause | 0.39 | 0.76 | 0.51 | 200 |
| Pragmatic Cause | – | – | – | 5 |
| Contrast | 0.61 | 0.09 | 0.15 | 127 |
| Concession | – | – | – | 5 |
| Conjunction | 0.30 | 0.51 | 0.38 | 118 |
| Instantiation | 0.67 | 0.39 | 0.49 | 72 |
| Restatement | 0.48 | 0.27 | 0.35 | 190 |
| Alternative | – | – | – | 15 |
| List | 0.80 | 0.13 | 0.23 | 30 |
| All (Micro Avg.) | 0.40 | 0.40 | 0.40 | 780 |

Table 6: Recall, precision, $F_1$, and counts for 11 Level 2 relation types. "–" indicates 0.00.

Table 6 shows the recall, precision, and $F_1$ measure for the 11 individual Level 2 relation types in the final experiment set up (Row 4 from Table 5). A point worth noting is that the classifier labels no instances of the Synchrony, Pragmatic Cause, Concession, and Alternative relation types. The reason is that the percentages for these four types are so small that the classifier is highly skewed towards the other types. From the distribution shown in Table 1, there are just 4.76% training data for these four types, but 95.24% for the remaining seven types. In fact, only 30 test instances are labeled with these four types, as shown in the last column of Table 6. As Cause is the most predominant type in the training data, the classifier tends to label uncertain relations as Cause, thus giving Cause high recall but low precision. We see that the F measures correlate well with the training data frequency, thus we hypothesize that accuracy may improve if more training data for low frequency relations can be provided.

Our work differs from that of (Pitler et al., 2009) in that our system performs classification at the more fine-grained Level 2 types, instead of the coarse-grained Level 1 classes. Their system applies a Naïve Bayes classifier whereas our system uses a maximum entropy classifier, and the sets of features used are also different. In addition, the data set of (Pitler et al., 2009) includes `EntRel` and `AltLex`, which are relations in which an implicit connective cannot be inserted between adjacent sentences, whereas ours excludes `EntRel` and `AltLex`.

## 6 Discussion: Why are implicit discourse relations difficult to recognize?

In the above experiments, we have shown that by using the four feature classes, we are able to increase the classification accuracy from 26.1% of the majority baseline to 40.2%. Although we feel a 14.1 absolute percentage improvement is a solid result, an accuracy of 40% does not allow downstream NLP applications to trust the output of such a classification system.

To understand the difficulties of the task more deeply, we analyzed individual training and validation data pairs, from which we were able to generalize four challenges to automated implicit discourse relation recognition. We hope that this discussion may motivate future work on implicit discourse relation recognition.

**Ambiguity.** There is ambiguity among the relations. For example, we notice that a lot of Contrast relations are mistakenly classified as Conjunction. When we analyzed these relations, we observed that Contrast and Conjunction in the PDTB annotation are very similar to each other in terms of words, syntax, and semantics, as Examples 5 and 6 show. In both examples, the same antonymous verb pair is used (*fell* and *rose*), different subjects are mentioned in Arg1 and Arg2 (*net* and *revenue* in the first example, and *net* and *sales* in the second), and these subjects are all compared to like items from the previous year. Moreover, the implicit discourse connective given by the annotators is *while* in both cases, which is an ambiguous connective as shown in (Miltsakaki et al., 2005).

(5) **Arg1:** In the third quarter, AMR said, net fell to $137 million, or $2.16 a share, from $150.3 million, or $2.50 a share.
**Arg2:** [**while**] Revenue rose 17% to $2.73 billion from $2.33 billion a year earlier.
(Contrast - wsj_1812)

(6) **Arg1:** Dow's third-quarter net fell to $589 million, or $3.29 a share, from $632 million, or $3.36 a share, a year ago.
**Arg2:** [**while**] Sales in the latest quarter rose 2% to $4.25 billion from $4.15 billion a year earlier.
(Conjunction - wsj_1926)

Relation ambiguity may be ameliorated if an instance is analyzed in context. However, according to the PDTB annotation guidelines, if the annotators could not disambiguate between two relation types, or if they felt both equally reflect their understanding of the relation between the arguments, they could annotate two types to the relation. In the whole PDTB corpus, about 5.4% of the explicit relations and 2.2% of the implicit relations are annotated with two relation types. Example 7 is such a case where the implicit connective *meanwhile* may be interpreted as expressing a Conjunction or Contrast relation.

(7) **Arg1:** Sales surged 40% to 250.17 billion yen from 178.61 billion.
**Arg2:** [**meanwhile**] Net income rose 11% to 29.62 billion yen from 26.68 billion.
(Conjunction; Contrast - wsj_2242)

**Inference.** Sometimes inference and a knowledge base are required to resolve the relation type. In Example 8, to understand that Arg2 is a restatement of Arg1, we need a semantic mechanism to show that either the semantics of Arg1 infers that of Arg2 or the other way around. In the below example, *I had calls all night long* infers *I was woken up every hour* semantically, as shown in: $receive\_call(I) \wedge duration(all\_night) \Rightarrow woken\_up(I) \wedge duration(every\_hour)$.

(8) **Arg1:** "I had calls all night long from the States," $\boxed{\text{he said.}}$
**Arg2:** "[**in fact**] I was woken up every hour – 1:30, 2:30, 3:30, 4:30."
(Restatement - wsj_2205)

In fact, most relation types can be represented using formal semantics (PDTB-Group, 2007), as shown in Table 7, where $|Arg1|$ and $|Arg2|$ represent the semantics extracted from Arg1 and Arg2, respectively. This kind of formal semantic reasoning requires a robust knowledge base, which is still beyond our current technology.

| Relation type | Semantic representation |
|---|---|
| Cause | $\lvert Arg1\rvert \prec \lvert Arg2\rvert \quad \vee \quad \lvert Arg2\rvert \prec \lvert Arg1\rvert$ |
| Concession | $A \prec C \quad \wedge \quad B \Rightarrow \neg C$ <br> where $A \in \lvert Arg1\rvert, B \in \lvert Arg2\rvert$ |
| Instantiation | $exemplify(\lvert Arg2\rvert, \lambda x.x \in E)$ <br> where $E = extract(\lvert Arg1\rvert)$ |
| Restatement | $\lvert Arg1\rvert \Rightarrow \lvert Arg2\rvert \quad \vee \quad \lvert Arg1\rvert \Leftarrow \lvert Arg2\rvert$ |
| Alternative | $\lvert Arg1\rvert \wedge \lvert Arg2\rvert \quad \vee \quad \lvert Arg1\rvert \oplus \lvert Arg2\rvert$ |

Table 7: Some examples of relation types with their semantic representations, as taken from (PDTB-Group, 2007).

**Context.** PDTB annotators adopted the *Minimality Principle* in argument selection, according to which they only included in the argument the minimal span of text that is sufficient for the interpretation of the relation. While the context is not necessary to interpret the relation, it is usually necessary to understand the meaning of the arguments. Without an analysis of the context, Arg1 and Arg2 may seem unconnected, as the following example shows, where the meaning of Arg1 is mostly derived from its previous context (i.e., *West German ... technical reactions*).

(9) **Prev. Context:** West German Economics Minister Helmut Haussmann said, "In my view, the stock market will stabilize relatively quickly. There may be one or other psychological or technical reactions,
**Arg1:** but they aren't based on fundamentals.
**Arg2:** [**in short**] The economy of West Germany and the EC European Community is highly stable."
(Conjunction - wsj_2210)

Sometimes the range of the context may easily extend to the whole text, which would require a system to possess a robust context modeling mechanism. In Example 10, in order to realize the causal relation between Arg2 and Arg1, we possibly need to read the whole article and understand what was happening: the machinist union was having a strike and the strike prevented most of its union members from working.

(10) **Arg1:** And at the company's Wichita, Kan., plant, about 2,400 of the 11,700 machinists still are working, $\boxed{\text{Boeing said.}}$
**Arg2:** [**because**] Under Kansas right-to-work laws, contracts cannot require workers to be union members.
(Cause - wsj_2208)

**World Knowledge.** Sometimes even context modeling is not enough. We may also need world knowledge to understand the arguments and hence to interpret the relation. In the following example, from the previous sentence of Arg1, it is reported that "the Senate voted to send a delegation of congressional staffers to Poland to assist its legislature", and this delegation is viewed as a "gift" in Arg1. It is suggested in Arg2 that the Poles might view the delegation as a "Trojan Horse". Here we need world knowledge to understand that "Trojan Horse" is usually applied as a metaphor for a person or thing that appears innocent but has harmful intent, and hence understand that Arg2 poses a contrasting view of the delegation as Arg1 does.

(11) **Arg1:** Senator Pete Domenici calls this effort "the first gift of democracy".
**Arg2:** [**but**] The Poles might do better to view it as a Trojan Horse.
(Contrast - wsj_2237)

These four classes of difficulties – ambiguity between relations, inference, contextual modeling, and world knowledge – show that implicit discourse relation classification needs deeper semantic representations, more robust system design, and access to more external knowledge. These obstacles may not be restricted to recognizing implicit relations, but are also applicable to other related discourse-centric tasks.

## 7 Conclusion

We implemented an implicit discourse relation classifier and showed initial results on the recently released Penn Discourse Treebank. The features we used include the modeling of the context of relations, features extracted from constituent parse trees and dependency parse trees, and word pair features. Our classifier achieves an accuracy of 40.2%, a 14.1% absolute improvement over the baseline. We also conducted a data analysis and discussed four challenges that need to be addressed in future to overcome the difficulties of implicit relation classification in the PDTB.

## References

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC 2006)*, pages 449–454.

Jerry R. Hobbs. 1990. Literature and cognition. In *CSLI Lecture Notes Number 21*. CSLI Publications.

Zheng Ping Jiang and Hwee Tou Ng. 2006. Semantic role labeling of NomBank: A maximum entropy approach. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006)*, pages 138–145, Sydney, Australia.

Alistair Knott and Ted Sanders. 1998. The classification of coherence relations and their linguistic markers: An exploration of two languages. *Journal of Pragmatics*, 30(2):135–175.

Alex Lascarides and Nicholas Asher. 1993. Temporal interpretation, discourse relations and commonsense entailment. *Linguistics and Philosophy*, 16(5):437–493.

Alan Lee, Rashmi Prasad, Aravind Joshi, Nikhil Dinesh, and Bonnie Webber. 2006. Complexity of dependencies in discourse: Are dependencies in discourse more complex than in syntax? In *Proceedings of the 5th International Workshop on Treebanks and Linguistic Theories*, Prague, Czech Republic, December.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical Structure Theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.

Daniel Marcu 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 2002)*, pages 368–375, Morristown, NJ, USA.

Daniel Marcu. 1998. A surface-based approach to identifying discourse markers and elementary textual units in unrestricted texts. In *Proceedings of the COLING-ACL 1998 Workshop on Discourse Relations and Discourse Markers*, pages 1–7, Montreal, Canada, August.

Eleni Miltsakaki, Nikhil Dinesh, Rashmi Prasad, Aravind Joshi, and Bonnie Webber. 2005. Experiments on sense annotations and sense disambiguation of discourse connectives. In *Proceedings of the Fourth Workshop on Treebanks and Linguistic Theories (TLT2005)*, Barcelona, Spain, December.

Alessandro Moschitti. 2004. A study on convolution kernels for shallow semantic parsing. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004)*, Barcelona, Spain.

PDTB-Group, 2007. *The Penn Discourse Treebank 2.0 Annotation Manual*. The PDTB Research Group, December.

Emily Pitler, Mridhula Raghupathy, Hena Mehta, Ani Nenkova, Alan Lee, and Aravind Joshi. 2008. Easily identifiable discourse relations. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, Manchester, UK, August.

Emily Pitler, Annie Louis, and Ani Nenkova. 2009. Automatic sense prediction for implicit discourse relations in text. To appear in *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP 2009)*.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The Penn Discourse Treebank 2.0. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*.

Manami Saito, Kazuhide Yamamoto, and Satoshi Sekine. 2006. Using phrasal patterns to identify discourse relations. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2006)*, pages 133–136, New York, USA, June.

Bonnie Webber. 2004. D-LTAG: Extending lexicalized TAG to discourse. *Cognitive Science*, 28(5):751–779, September.

Ben Wellner, James Pustejovsky, Catherine Havasi, Anna Rumshisky, and Roser Sauri. 2006. Classification of discourse coherence relations: An exploratory study using multiple knowledge sources. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, Sydney, Australia, July.

Florian Wolf and Edward Gibson. 2005. Representing discourse coherence: a corpus-based analysis. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, pages 134–140, Morristown, NJ, USA.