DISCOURSE PARSING: INFERRING DISCOURSE STRUCTURE, MODELING COHERENCE, AND ITS APPLICATIONS

ZIHENG LIN

(B. Comp. (Hons.), NUS)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

SCHOOL OF COMPUTING

NATIONAL UNIVERSITY OF SINGAPORE

2011

Acknowledgments

First of all, I would like to express my gratitude to my supervisors, Prof. Min-Yen Kan and Prof. Hwee Tou Ng, for their continuous help and guidance throughout my graduate years. Without them, the work in this thesis would not have been possible, and I would not have been able to complete my Ph.D. studies.

During the third and final years of my undergraduate studies, I have had the great opportunities to work with Prof. Kan on two research projects in natural language processing. Since then I have found my interest and curiosity in this research field, and these have led me to my graduate studies. Prof. Kan has always been keen and patient to discuss with me problems that I have encountered in my research and to lead me to the correct directions every time when I was off-track. His positive attitude towards study, career, and life has a great influence on me.

I am also grateful to Prof. Ng, for always providing helpful insights and reminding me of the big picture in my research. His careful attitude towards formulation, modeling, and experiments of research problems has deeply shaped my understanding of doing research. He has inspired me to explore so much in the early stage of my graduate studies, and has also unreservedly shared with me his vast experience.

I would like to express my gratitude to my thesis committee members, Prof. Chew Lim Tan and Prof. Wee Sun Lee, for their careful reviewing of my graduate research paper, thesis proposal, and this thesis. Their critical questions helped me iron out the second half of this work in the early stage of my research. I am also indebted to Prof. Lee for his supervision in my final year project of my undergraduate studies.

I would also like to thank my external thesis examiner, Prof. Bonnie Webber, for giving me many valuable comments and suggestions on my work and the PDTB when we met in EMNLP and ACL.

My heartfelt thanks also go to my friends and colleagues from the Computational Linguistics lab and the Web Information Retrieval / Natural Language Processing Group

(WING), for the constructive discussions and wonderful gatherings: Praveen Bysani, Tao Chen, Anqi Cui, Daniel Dahlmeier, Jesse Prabawa Gozali, Cong Duy Vu Hoang, Wei Lu, Minh Thang Luong, Jun Ping Ng, Emma Thuy Dung Nguyen, Long Qiu, Hendra Setiawan, Kazunari Sugiyama, Yee Fan Tan, Pidong Wang, Aobo Wang, Liner Yang, Jin Zhao, Shanheng Zhao, Zhi Zhong.

I am grateful for the insightful comments from the anonymous reviewers of the papers that I have submitted. I was financially supported by the NUS Research Scholarship for the first four years and the NUS-Tsinghua Extreme Search Centre for the last half year.

Finally, but foremost, I would like to thank my parents and my wife, Yanru, for their understanding and encouragement in the past five years. I would not be able to finish my studies without their unwavering support.

Contents

Abstract

Discourse Parsing: Inferring Discourse Structure,
Modeling Coherence, and its Applications

Ziheng Lin

In this thesis, we investigate a natural language problem of parsing a free text into its discourse structure. Specifically, we look at how to parse free texts in the Penn Discourse Treebank representation in a fully data-driven approach. A difficult component of the parser is to recognize Implicit discourse relations. We first propose a classifier to tackle this with the use of contextual features, word-pairs, and constituent and dependency parse features. We then design a parsing algorithm and implement it into a full parser in a pipeline. We present a comprehensive evaluation on the parser from both component-wise and error-cascading perspectives. To the best of our knowledge, this is the first parser that performs end-to-end discourse parsing in the PDTB style.

Textual coherence is strongly connected to a text's discourse structure. We present a novel model to represent and assess the discourse coherence of a text with the use of our discourse parser. Our model assumes that coherent text implicitly favors certain types of discourse relation transitions. We implement this model and apply it towards the text ordering ranking task, which aims to discern an original text from a permuted ordering of its sentences. To the best our knowledge, this is also the first study to show that output from an automatic discourse parser helps in coherence modeling.

Besides modeling coherence, discourse parsing can also improve downstream applications in natural language processing (NLP). In this thesis, we demonstrate that incorporating discourse features can significantly improve two NLP tasks – argumentative zoning and summarization – in the scholarly domain. We also show that output from these two tasks can improve each other in an iterative model.

List of Tables

List of Figures

To my parents, Weiqun Lin and Lieqin Lin.

To my beloved wife, Yanru Lian.

Chapter 1

Introduction

Language is not simply formed by isolated and unrelated sentences, but instead by collocated, structured, and coherent texts of sentences. A piece of text is often not to be understood individually, but understood by joining it with other text units from its context. These units can be surrounding clauses, sentences, or even paragraphs. A text becomes semantically well-structured and understandable when its text units are analyzed with respect to each other and the context, and are joined interstructurally to derive high level structure and information. Most of the time, analyzing a text as a whole gives more semantic information to the user than summing up the information extracted from the individual units of this text. Such a coherent text segment of sentences is referred to as a **discourse** (Jurafsky and Martin, 2009).

1.1 Computational Discourse

The process of text-level or discourse-level analysis may lead to a number of natural language processing (NLP) tasks. One of them is anaphora resolution, which is to locate the referring expressions in the text and resolve them to the exact entities. For instance, in Example 1.1, the pronoun "They" in the second sentence refers to "These three countries" in the first sentence. To resolve what these three countries are, we may need to look back into the previous context.

(1.1) *These three countries* aren't completely off the hook, though.
They will remain on a lower-priority list that includes 17 other countries.

If we analyze the second sentence in isolation without performing anaphora resolution, it is difficult to understand what entities remain on a lower-priority list. And this may hinder the progress of downstream applications such as information extraction and question answering. In the case of question answering, it becomes problematic if the question is to find "all countries on the lower-priority list".

Another NLP task for discourse processing is to draw the connections between its text units. From a discourse point of view, these connections are usually referred to as the rhetorical or discourse relations.[1] Such connections may appear between any spans of text, where the spans can be clauses, sentences, or multiple sentences. As an example, an analysis of Example 1.1 shows that there lies a Contrast relation between these two sentences. We may illustrate this relation as follows: these three countries are not out of danger; rather, they will still remain on the lower-priority list. In fact, if we add the discourse connective "rather" at the beginning of the second sentence, it illustrates this relation explicitly without modifying its original meaning.

Discourse relations can be formed between any pair of text spans. When discourse relations in a text are identified, this will produce a representation of the discourse structure for the text. Figure 1.1 shows an excerpt taken from an article with ID wsj_2402 from the Penn Treebank corpus (Marcus et al., 1993). This text is segmented into clauses and sentences, and all discourse relations in the text are annotated in the Penn Discourse Treebank (Prasad et al., 2008). The discourse representation for this text is illustrated by Figure 1.2. This structure provides very useful information for readers or machines to understand the text from a "bird's eye view". There is a Conjunction relation between

[1]Throughout this thesis, the term *rhetorical relation* and *discourse relation* are used interchangeably.

[If you can swallow the premise that the rewards for such ineptitude are six-figure salaries, $]_a$ [you still are left puzzled, $]_b$ [because few of the yuppies consume very conspicuously. $]_c$ [In fact, few consume much of anything. $]_d$ [Two share a house almost devoid of furniture. $]_e$ [Michelle lives in a hotel room, $]_f$ [and although she drives a canary-colored Porsche, $]_g$ [she hasn't time to clean $]_h$ [or repair it; $]_i$ [the beat-up vehicle can be started only with a huge pair of pliers $]_j$ [because the ignition key has broken off in the lock. $]_k$ [And it takes Declan, the obligatory ladies' man of the cast, until the third episode to get past first base with any of his prey. $]_l$

Figure 1.1: An excerpt taken from a *Wall Street Journal* article wsj_2402. The text is segmented and each segment is subscripted with a letter. The discourse relations in this text are illustrated in the graph in Figure 1.2.

spans $fghi$ and l, and a causal relation between $fghi$ and jk. Within $fghi$, there is another Conjunction between f and ghi. g and hi are contrastive, and h and i elaborate alternative meaning. As a sentence, $fghijk$ also has a List relation with the previous sentence e.

Note that the structure in Figure 1.2 is not a tree but a graph structure. Nodes (*i.e.*, text spans or argument spans) can be shared by more than one relation. For example, d is an argument span of two relations Specification and Instantiation. Furthermore, relations may connect two text spans that are not consecutive, such as the Conjunction relation between spans $fghi$ and l. Another point worth mentioning here is that some of the relations are signaled by discourse connectives, which are underlined in Figure 1.1. For example, the causal relation between b and c is signaled by "because", and "in fact" hints at the Specification relation between c and d. Other relations, such as Instantiation between d and e and List between e and $fghijk$, are not *explicitly* signaled by discourse connectives, but are inferred by humans. These *implicit* discourse relations are comparatively more difficult to deduce than those with discourse connectives.

Discourse segmentation, or text segmentation, is another task in discourse processing that aims to segment a text into a *linear discourse structure*, based on the notion

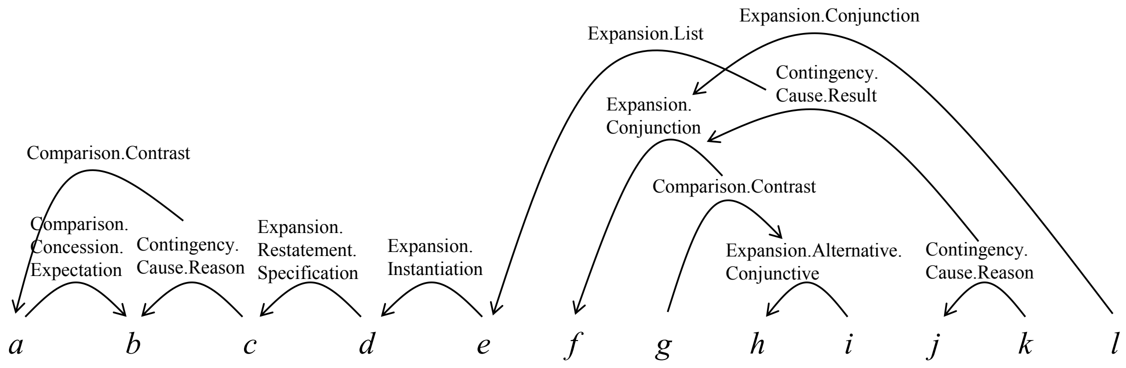


Figure 1.2: Discourse relations for the text in Figure 1.1. The relation annotation is taken from the Penn Discourse Treebank. For notational convenience, I denote discourse relations with an arrow, although there is no directionality distinction. I denote Arg2 as the origin of the arrow and Arg1 as the destination of the arrow.

of *subtopic shift*. A subtopic usually consists of multiple paragraphs. In the domain of scientific articles, subtopic structure is normally explicitly marked by section/subsection titles which group cohesive paragraphs together. Brown and Yule (1983) have shown that this is one of the most basic divisions in discourse. Many expository texts (for example, news articles) consist of long sequences of paragraphs without explicit structural demarcation. A subtopical segmentation system will be very useful in such texts. Figure 1.3 shows a subtopic structure for a 21-paragraph news article called *Stargazers*, taken from Hearst (Hearst, 1997).

Discourse segmentation is useful for other tasks and applications. For example, in information retrieval, it can automatically segment a TV news broadcast or a long web article into a sequence of video or text units so that we can index and search such finer-grained information units. For text summarization, given an article's subtopics, the system can summarize each subtopic and then aggregate the results into a final summary.

While all of these three tasks – anaphora resolution, discourse parsing, and discourse segmentation – are very important in analyzing and understanding the discourse of a text, in this thesis, we focus solely on the problem of discourse parsing, in which

1–3	Intro – the search for life in space
4–5	The moon's chemical composition
6–8	How early earth-moon proximity shaped the moon
9–12	How the moon helped life evolve on earth
13	Improbability of the earth-moon system
14–16	Binary/trinary star systems make life unlikely
17–18	The low probability of nonbinary/trinary systems
19–20	Properties of earth's sun that facilitate life
21	Summary

Figure 1.3: Subtopic structure for a 21-paragraph science news article called *Stargazers*, taken from Hearst (Hearst, 1997).

we infer the discourse relations and structure for a text. In particular, we will first look at the harder problem of classifying Implicit discourse relations. This class of discourse relations occupies a similar percentage as that for Explicit discourse connectives in the news domain as shown in (PDTB-Group, 2007) [2]. Although in the past, researchers paid less attention to the Implicit discourse relations, they are as important as their Explicit counterparts. We will design and implement a discourse parser that is capable of identifying text spans and classifying relation types for both Explicit and Implicit discourse relations.

Recently Prasad et al. (Prasad et al., 2008) released the Penn Discourse Treebank, or PDTB for short, which is a discourse-level annotation on top of the Penn Treebank (PTB) (Marcus et al., 1993). This corpus provides annotations for both Explicit and Implicit discourse relations. In this thesis, we conduct experiments for discourse parsing in this corpus.

[2]The percentages of Explicit and Implicit relations are likely to vary in other domains such as fiction, dialogue, and legal texts.

1.2 Motivations for Discourse Parsing

There are generally two motivations for finding the discourse relations in a text and constructing the corresponding discourse structure. One motivation is that such structure can be used in understanding the coherence of the text. Given two texts and their respective discourse structures, one can compare these two structures. Discourse patterns extracted from the structures may suggest which text is more coherent than the other. For example, Contrast-followed-by-Cause is one of the common patterns that can be found in discourse structures. This is illustrated by the relations among a, b, and c in Figure 1.2. Knowing which text is more coherent could be very useful in other tasks, such as automatic student essay grading.

Another motivation is the use of marking discourse relations and argument spans in downstream applications in natural language processing. Discourse parsing has been used in automatic text summarization (Marcu, 1997), as the relation types can provide indication of importance. For example, in Rhetorical Structure Theory, or RST (Mann and Thompson, 1988), the two text spans of a rhetorical relation are labeled nucleus and satellite. In this theory, the nucleus span provides central information, while the satellite span provides supportive information to the nucleus. Thus, to locate important spans in the text in order to construct a summary, one can concentrate on the nucleus spans. Other discourse framework, which may not have the similar focus of nuclearity but provide representation of relation types, can also be utilized in a summarization system. As identifying redundancy is very important in summarization, relations such as Conjunction, Instantiation, Restatement, and Alternative can provide clues to locate redundant information. Furthermore, one can also utilize Contrast to identify updating information in the task of update summarization, which aims to produce a summary with an assumption that user has some prior knowledge of the topic. Thus in the summarization task, discourse parsing can provide information on the relations between text spans and the corresponding roles of the text spans in the relations. In Chapter 6, we will demonstrate how an automatic discourse parser can improve a text summarization system by utilizing its discourse relation types.

Other NLP tasks, such as question answering (QA) and textual entailment, can also benefit from discourse parsing (Higashinaka and Isozaki, 2008). As an example, why-QA, which is a special type in QA, is to find answers to the question "Why X?". Here is an example for a "Why X?" question: Why are pandas on the verge of extinction? If one is able to resolve causal relations between pairs of text spans, one can leverage such information to locate the answers for a *why* question. For textual entailment, one can use a discourse relation classifier to check whether there exists a discourse relation that repeats or expands the semantics between the text T and the hypothesis H. Specifically, such relation can be a Conjunction, Instantiation, Restatement, Alternative, or List.

1.2.1 Problem Statement

In this thesis, we hypothesize that we can build a discourse parser to infer discourse structures, which can be utilized to model the textual coherence of a text and improve downstream NLP tasks. Specifically, we argue that one can build a classifier to tackle the harder problem of classifying Implicit discourse relations and integrate this into a full parser. We also argue that one can train a model by examining the discourse patterns of coherent and incoherent texts, and use the trained model to differentiate a coherent text from an incoherent one. We also show that features extracted from the discourse structure can improve the performance of two NLP tasks – text summarization and argumentative zoning – in a supervised approach. Argumentative zoning is a task defined by Teufel (1999) to label sentences in a scientific paper into one of the seven rhetorical labels. The purpose of doing this is to provide a high level view on the rhetorical moves and arguments of the paper.

Hypothesis: **A discourse parser with a component to tackle Implicit discourse relations can provide information to model textual coherence and improve user**

tasks in natural language processing.

1.3 Contributions

This thesis makes four major contributions in the area of discourse parsing, coherence modeling, text summarization, and argumentative zoning. They are summarized as follows:

- **Implicit discourse relation classification.** We develop a classifier to recognize Implicit discourse relations in the Penn Discourse Treebank. We propose the use of four feature classes: contextual features to check surrounding discourse relations, production rules extracted from constituent parse trees, dependency rules extracted from dependency parse trees, and word-pair features. We show that a classifier trained on these features yields statistically significant improvements over a suitable baseline. We perform data analysis on the PDTB and identify four challenges to this task: relation ambiguity, semantic inference, deeper context modeling, and world knowledge.
- **A PDTB-styled end-to-end discourse parser.** We design a parsing algorithm that performs discourse parsing in the PDTB representation. We implement this algorithm into a full parser that takes as input a free text, and returns a discourse structure. The parser is configured as a pipeline: joining together an Explict relation classifier, the aforementioned Implicit relation classifier, and an attribution span labeler. In the first step of classifying Explicit discourse relations, we implement a connective classifier that improves upon the previous work, and we propose a novel component to locate and label the two text spans for a relation. We also propose and present a comprehensive evaluation on the parser from both component-wise and error-cascading perspectives. To the best of our knowledge, this is the first parser that performs end-to-end discourse parsing in the PDTB style.

- **Evaluating textual coherence using discourse structures and relations.** We demonstrate that simply using the patterns of discourse relations is not enough to assess the coherence of a text, as this leads to sparse features. We propose a discourse role matrix that converts a discourse structure into a representation which schematically represents term occurrences in text units and associates each occurrence with its discourse roles in the text units. We show that statistics extracted from such discourse model can be used to distinguish coherent text from incoherent one. To the best of our knowledge, this is also the first study to show that output from an automatic discourse parser helps in coherence modeling.

- **Improving summarization and argumentative zoning using discourse relations.** We apply information extracted from the discourse structure for a text in both scientific paper summarization and argumentative zoning, and demonstrate significant improvement on these two tasks. We also propose an iterative model of these two tasks and show that their output can improve each other.

1.3.1 Research Publications

The work in this thesis has been published in (Lin et al., 2009; Lin et al., 2010; Lin et al., 2011):

- 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 (EMNLP 2009), pages 343–351, Singapore.

- Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2010. A PDTB-Styled End-to-End Discourse Parser. Technical Report TRB8/10, School of Computing, National University of Singapore, August.

- Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2011. Automatically Evaluating Text Coherence Using Discourse Relations. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011), pages 997–1006, Portland, Oregon, USA, June.

1.4 Overview of This Thesis

This thesis is structured into seven chapters.

- **Chapter 2** discusses background and related work for this thesis. Previous work that is related to this thesis basically belongs to five areas: Implicit discourse relation classification, automatic discourse parsing, textual coherence modeling, automatic text summarization (specifically in the scientific domain), and argumentative zoning. Furthermore, we give an overview of the Penn Discourse Treebank (PDTB), which is a discourse-level annotation atop the Penn Treebank and will be used as our working data set.

- In **Chapter 3**, we design and implement a system to recognize Implicit discourse relations in the PDTB. Features used in this classifier include the modeling of the context of relations, features extracted from constituent parse trees and dependency parse trees, and word pair features. We also conduct a data analysis in the PDTB and discuss four challenges for designing an Implicit relation classifier.

- In **Chapter 4**, we design an algorithm that performs discourse parsing in the PDTB representation, and implement it into an end-to-end system in a fully data-driven approach. This is the first end-to-end discourse parser that can parse any unrestricted text into its discourse structure in the PDTB style. The Implicit relation classifier is used in this pipeline as one component. In addition to this, we specifically develop other components to classify Explicit discourse relations and attributions. The demo and source code of the parser have been released online[3].

- In **Chapter 5**, with this discourse parser, we propose a coherence model that leverages the observation that coherent texts preferentially follow certain discourse patterns. We posit that such patterns can be represented and captured by discourse relation transitions, and demonstrate this using a matrix representation of the discourse patterns for a text. We also demonstrate that this coherence model is capable of differentiating a coherent text from an incoherent one.

- **Chapter 6** applies discourse parsing in two NLP tasks: summarization on scientific papers and argumentative zoning. We extract features from the output of the discourse parser, and demonstrate that such features can improve the performance on both tasks. In addition to this, we construct an iterative model of these two tasks, and show that their results can be used as features to improve each other.

- Lastly, **Chapter 7** summarizes the work in this thesis and outlines a number of future directions.

[3]`http://wing.comp.nus.edu.sg/~linzihen/parser/`

Chapter 2

Background and Related Work

In this chapter, we briefly describe previous work that is related to this thesis. We first give an overview of the Penn Discourse Treebank, followed by describing related work in Implicit discourse relation classification and discourse parsing. We also list recent research work that is conducted in the Penn Discourse Treebank. We will then describe previous work in the areas of coherence modeling, text summarization, and argumentative zoning.

2.1 Overview of the Penn Discourse Treebank

The Penn Discourse Treebank (PDTB) (Prasad et al., 2008) covers the set of one million word *Wall Street Journal* (WSJ) articles in the Penn Treebank (PTB) (Marcus et al., 1993), which is much larger than the previous existing discourse annotations, such as the RST Discourse Treebank (RST-DT) corpus (Carlson et al., 2001). The PDTB adopts a binary predicate-argument view on discourse relations, where the connective acts as a predicate that takes two text spans as its arguments. The span to which the connective is syntactically attached is called Arg2, while the other is called Arg1. All connectives annotated in the PDTB have exactly two arguments, which is unlike the predicate-argument structures of verbs in the PropBank (Palmer et al., 2005), where verbs can take any number of arguments.

The PDTB provides annotation for each discourse connective and its two arguments. **Explicit** relations are defined to be discourse relations that are explicitly signaled by discourse connectives. The PDTB defines a set of 100 discourse connectives. Example 2.1 shows an Explicit relation where the connective "because" is <u>underlined</u>, Arg1 span is *italicized*, and Arg2 span is **bolded**. The last line of the example shows the relation type and the file in the PDTB from which the example is drawn.

(2.1) *The federal government suspended sales of U.S. savings bonds* <u>because</u> **Congress hasn't lifted the ceiling on government debt.**

(Contingency.Cause.Reason - wsj_0008)

The PDTB also examined pairs of adjacent sentences within paragraphs for discourse relations other than Explicit relations. Example 2.2 shows such an **Implicit** relation where the annotator inferred an Implicit connective "for example" that most intuitively connects Arg1 and Arg2 spans. Some relations are alternatively lexicalized by non-connective expressions, *i.e.*, expressions that are not in the set of 100 discourse connectives in the PDTB. These relations are termed **AltLex** relations. Example 2.3 is such a relation with the non-connective expression "Another concern". Implicit connectives are represented by the annotation "<u>Implicit</u> = ...", and AltLex expressions are represented by "<u>AltLex</u> [...]".

(2.2) *"I believe in the law of averages,"* $\boxed{\text{declared San Francisco batting coach Dusty Baker after game two.}}$ <u>Implicit</u> = ACCORDINGLY **"I'd rather see a so-so hitter who's hot come up for the other side than a good hitter who's cold."**

(Contingency.Cause.Result - wsj_2202)

(2.3) *Political and currency gyrations can whipsaw the funds.* <u>AltLex</u> [**Another concern**]: **The funds' share prices tend to swing more than the broader**

market.

(Expansion.Conjunction - wsj_0034)

If no Implicit or AltLex relation exists between a sentence pair, annotators then checked whether an entity transition (*i.e.*, **EntRel** relation) holds, otherwise no relation (**NoRel**) was concluded. EntRel captures the cases in which the same entity is realized or repeated in both sentences. Example 2.4 shows an EntRel relation where the person "Pierre Vinken" in the first sentence is repeated as "Mr. Vinken" in the second sentence. Explicit, Implicit and AltLex relations are *discourse* relations, whereas EntRel and NoRel are *non-discourse* relations.

(2.4) *Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29.* **Mr. Vinken is chairman of Elsevier N.V., the Dutch publishing group.**

(EntRel - wsj_0001)

For each discourse relation, the PDTB also provides annotation for the attribution (*i.e.*, the agent that expresses the argument) for Arg1, Arg2, and the relation as a whole. For example, the text span in the box in Example 2.2 – "declared San Francisco batting coach Dusty Baker after game two" – is the attribution span for the whole relation.

Aside from annotating discourse relations, the PDTB also provides a three-level hierarchy of relation types. The first level consists of four major relation *classes*: Temporal, Contingency, Comparison, and Expansion. Temporal is used when the events or situations in Arg1 and Arg2 are related temporally. A discourse relation belongs to the Contingency relation when one argument causally influences the other. When the events in Arg1 and Arg2 are compared to highlight the difference, it is labeled as a Comparison relation. Otherwise it is called Expansion if one argument expands the semantics or discourse in the other argument.

For each class, a second level of *types* is defined to provide finer semantic distinctions which are listed in Table 2.1. For example, there are six types defined under the

Temporal	Comparison
Synchronous	Contrast
Asynchronous	Juxtaposition
Precedence	Opposition
Succession	Pragmatic Contrast
Contingency	Concession
Cause	Expectation
Reason	Contra-expectation
Result	Pragmatic Concession
Pragmatic Cause	Expansion
Justification	Conjunction
Condition	Instantiation
Hypothetical	Restatement
General	Specification
Unreal Present	Equivalence
Unreal Past	Generalization
Factual Present	Alternative
Factual Past	Conjunctive
Pragmatic Condition	Disjunctive
Relevance	Chosen Alternative
Implicit Assertion	Exception
	List

Table 2.1: Discourse relations in (Prasad et al., 2008): a hierarchy of semantic classes, types and subtypes.

Expansion class: Conjunction, Instantiation, Restatement, Alternative, Exception, and List. Restatement describes a situation where one argument restates the semantics in the other. In contrast, Conjunction is used when Arg2 provides additional, discourse-new information that is related to that in Arg1. Instantiation – which is sometimes signaled by the connective "for example" – exemplify or instantiate Arg1's event in the Arg2 span.

A third level of *subtypes* is defined for some types to specify the semantic contribution of each argument. For example, the relation type labeled for Example 2.2 is Expansion.Restatement.Specification, meaning that there is a Restatement relation between Arg1 and Arg2, and Arg2 (instead of Arg1) is the argument that provides specific details. Other types, such as Instantiation and List, are not further provided with subtypes.

2.2 Implicit Discourse Relations

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. For instance, the word pair (good, fails) in the following example provides a clue that a contrast relation holds between the two sentences.

(2.5) John is *good* in math and sciences.
Paul *fails* almost every class he takes.

As they did not have human annotated data for Implicit discourse relations, 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 a corpus of Implicit relations. For example, one of the patterns is

[BOS ... EOS] [BOS But ... EOS]

which matches two consecutive sentences where the second begins with “But” (BOS and EOS mean begin of sentence and end of sentence, respectively). “But” is then removed and the two sentences form a pair of Implicit discourse relation. They assumed “but” always indicates a Contrast relation, and assigned Contrast to this pair. However, this assumption is not always true, as we will show in Chapter 4. Not all occurrences of connectives exhibit discourse functions (*e.g.*, not every “but” is a discourse connective), and some discourse connectives are ambiguous with regard to the relation types (*e.g.*, not every “but” indicates Contrast).

From this corpus, they collected word pair statistics, which were used in a Naïve Bayes framework to classify discourse relations. They determined the most likely discourse relation that holds between a pair of sentences S_1 and S_2 by finding the discourse relation r that maximizes $P(r|S_1, S_2)$, which is equivalent to $logP(S_1, S_2|r) + logP(r)$. The first component $logP(S_1, S_2|r)$ can be calculated by

$$\sum_{(w_1,w_2)\in S_1,S_2} logP((w_1, w_2)|r)$$

where (w_1, w_2) is a word pair and w_1 and w_2 are extracted from S_1 and S_2, respectively.

Saito et al. (2006) extended this theme, to show that phrasal patterns extracted from a text span pair provide useful evidence in relation classification. For example, if the pattern “X should have done Y” is found in the first sentence and the second sentence is “A did B”, there is most likely a Contrast relation between these two sentences. Another example is the patterns “There is . . . ” and “Those are . . . ”, which we can usually conclude an Instantiation relation, even if there is no other word pair clues found inside “. . . ”. 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. They restricted the phrasal patterns to those that match the following regular expression in order to filter out less informative phrases:

“(noun-x | verb | adjective)? (particle | auxiliary verb | period)+$” or “adverb$”

They demonstrated that phrasal patterns are able to significantly improve the task in addition to the word pair features.

Both of these previous works have the shortcoming of transforming Explicit relations to Implicit ones by removing the Explicit discourse connectives, which was previously discussed in (Sporleder and Lascarides, 2008). 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 artificially Implicit relation corpus may exhibit marked differences from a natively Implicit one.

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). The features include: words at the beginning and end of the span to capture discourse connectives, distance between the two spans, semantic path between non-function words, word pair similarity, dependency relations between the two spans, temporal links between the two spans, and event-based features such as event head words and types. 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. As they did not separate the experimental results for Explicit and Implicit relations, it is not able to draw a conclusion on the performance on classifying Implicit relations.

Pitler et al. (2009) performed classification of Implicit discourse relations in the PDTB using several linguistically informed features, which include: polarity of the span, unigram and bigram language models, verb classes, first and last words of the span, modality of the span, and word pairs. The classification task is performed on the four Level-1 types, *i.e.*, Temporal, Contingency, Comparison, and Expansion. Their intuition for using word polarity is that words from contrastive sentences may possess opposite polarities. For example, “good” in Example 2.5 has positive polarity and “fails” negative polarity. Surprisingly, this feature class did not contribute to the performance when they conducted a binary classification of Comparison vs. Other. Their analysis showed that this is most likely due to the low coverage of the positive-negative pairs in the data. The idea behind using verb classes is that, if the verbs from the two text spans are from the same or close classes, the relation between them are very likely to be Expansion. They conducted four sets of binary classifications (*i.e.*, Relation vs. Other) with a Naïve Bayes classifier, and showed performance increases over a random classification baseline.

2.3 Discourse Parsing

Many discourse frameworks have been proposed in the literature of discourse modeling. Among them, there are the cohesive devices described by Halliday and Hasan (1976), Hobbs’ inventory of coherence relations based on abductive reasoning (Hobbs, 1985), the Rhetorical Structure Theory (RST) proposed by Mann and Thompson (1988), Grosz and Sidner (1986)’s models which aim to associate speakers’ intentions with their focus of attention in discourse, the Linguistic Discourse Model (LDM) proposed by (Scha and Polanyi, 1988; Polanyi and Scha, 1984), the Lexicalized Tree Adjoining Grammar for Discourse (D-LTAG) by (Webber and Joshi, 1998; Webber, 2004; Forbes et al., 2003), and the discourse model that associates discourse relations in a graph structure (Wolf and Gibson, 2005). A number of discourse parsing systems following the RST framework has been proposed, due to the availability of the RST Discourse Treebank (RST-DT) (Carlson et al., 2001). Thus, we will review RST and the automatic discourse parsers that follow this framework. Furthermore, as the corpus of interest of this thesis – the PDTB – is

[Farmington police had to help control traffic recently $]_a$ [when hundreds of people lined up to be among the first applying for jobs at the yet-to-open Marriott Hotel. $]_b$ [The hotel's help-wanted announcement – for 300 openings – was a rare opportunity for many unemployed. $]_c$ [The people waiting in line carried a message, a refutation, of claims that the jobless could be employed if only they showed enough moxie. $]_d$ [Every rule has exceptions, $]_e$ [but the tragic and too-common tableaux of hundreds or even thousands of people snake-lining up for any task with a paycheck illustrates a lack of jobs, $]_f$ [not laziness. $]_g$

Figure 2.1: A text taken from (Mann and Thompson, 1988), which originates from an editorial in *The Hartford Courant*. The text is segmented and each segment is subscripted with a number. The RST tree for this text is shown in Figure 2.2.

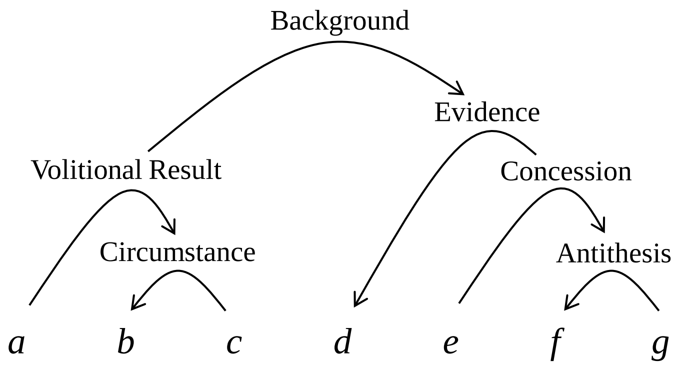


Figure 2.2: RST tree for the text in Figure 2.1.

developed atop the D-LTAG framework, we will also review D-LTAG and its rule-based parser.

Rhetorical Structure Theory, or RST, is a discourse theory that associates rhetorical relations with text structures. Mann and Thompson (1988) proposed the Rhetorical Structure Theory which takes a nucleus-satellite view on rhetorical relations. RST defines a set of rhetorical relations as well as discourse schemas for the structural constituency arrangements of text. As the RST schemas are recursive, they enable relation embedding that lead to a tree structure of a text. Figure 2.2 shows an RST tree for the excerpt in Figure 2.1. An arrow connects a satellite span to a nucleus span. Comparing Figure 2.2

with Figure 1.2, one can see that the PDTB relations are more freely interconnected – they construct a graph instead of a tree. However, there are no advantages of one representation over the other. The preference depends on the aims and system design. The framework of PDTB focuses more on local discourse coherence, , *i.e.*, how two adjacent texts (clause or sentence) are connected to each other. This enables reader to verify how text is glued together in a local context. However, there is also no restriction on long distance relation – readers are free to draw a relation between two text spans which are a few sentences away from each other, if they infer a discourse relation between these two spans. On the other hand, the recursive function of RST constructs a text from a global perspective but at the same time put more restriction on the way that a text can be handled by a machine.

Marcu (1997) formalized an algorithm to automatically parse an unrestricted text into its rhetorical tree using the RST framework. He made use of cue phrases to split a sentence into *elementary discourse units* (*edus*), designed algorithms that are able to recognize rhetorical relations with or without the signals of cue phrases, and proposed four rule-based algorithms for determining the valid discourse tree given the relations of adjacent *edus*. Take the previous example, the input to Marcu's parser will be the free text in Figure 2.1, and the output will be the RST tree shown in Figure 2.2.

Continuing this vein, Soricut and Marcu (2003) introduced probabilistic models to segment a sentence into *edus*, and to derive their corresponding sentence-level discourse structure, using lexical and syntactic features. A probability for each word is calculated to check whether to insert an *edu* boundary after this word. They proposed the notion of a dominance set. In order to construct the RST tree for the sentence, a dominance set is used to check that which *edu* from a pair of *edus* dominate the other one. They experimented with their models using the RST Discourse Treebank (RST-DT) corpus (Carlson et al., 2001), which is annotated in the RST framework and covers a small subset of 385 documents in the PTB. Figure 2.3 shows an RST tree for a sentence that is segmented into three *edus*. Note that there is a relation named Attribution, which in the

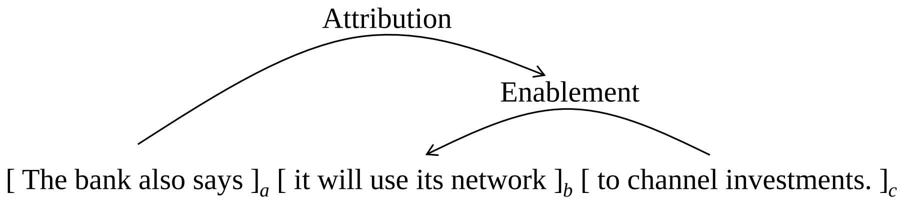

Attribution
Enablement
[The bank also says]$_a$ [it will use its network]$_b$ [to channel investments.]$_c$

Figure 2.3: RST structure of a sentence, borrowed from (Soricut and Marcu, 2003).

PDTB framework is not defined as a discourse relation. In Chapter 4, we will introduce an attribution span labeler to deal with attributions.

Huong et al. (2004) divided the discourse parsing process into two steps: first using syntactic information and cue phrases to segment sentences into *edus* and to generate discourse structures at sentence-level, and then generating text-level structure from the sentence-level ones in a constrained, bottom-up manner. They experimented in the RST-DT corpus and showed promising system performance on the following dimensions:

- *edu* segmentation at sentence level,
- Relation connection of text spans at both sentence and text level,
- Relation orientation (*i.e.*, nucleus vs. satellite) at both sentence and text level, and
- Relation type classification at both sentence and text level.

In our system experiments of our developed discourse parser in Chapter 4, we also perform component-wise evaluations in a similar fashion.

Recently, duVerle and Prendinger (2009) made use of a support vector machines (SVM) approach, using a rich set of shallow lexical, syntactic, and structural features to train two separate classifiers to identify the rhetorical structures and label the rhetorical roles drawn from the RST-DT. Their tree building algorithm is applied to subtrees in a recursive and bottom-up manner, starting from the *edus*: the binary relation identifier is applied to select the best match between adjacent subtrees, and the multi-class relation classifier is applied to label the newly created subtree. The process continues recursively until there is only one subtree left. As an example, to build the RST tree in Figure 2.3, their system may check the existence of relations for two pairs of subtrees, (a, b) and (b, c), select (b, c) and label it with Enablement. Since there is now only one subtree (a, bc) left, it may label this as Attribution.

Apart from the RST framework and systems built upon it, Webber and Joshi (1998) and Webber (2004) have developed a Lexicalized Tree Adjoining Grammar for Discourse (D-LTAG) which associates *local* discourse relations with lexical elements. In D-LTAG, discourse relations are triggered by lexical elements (*i.e.*, explicit or implicit discourse connectives), focusing on low-level discourse structures and semantics of monologic discourse. Taking the relation between spans b and c in Figure 1.2 as example, there is a Cause relation between these two spans that is triggered by the connective "because". This is illustrated by Figure 2.4, where (a) is a D-LTAG initial tree for subordinate substitution. Another essential feature of D-LTAG is its use of auxiliary trees (mainly for discourse adverbials), which is not shown in Figure 2.4. Auxiliary trees provide a recursive way to **adjoin** a discourse clause to a discourse tree with coordinating or null connective, or adjoin a discourse clause with adverbial to a discourse tree with **anaphoric link** pointing to previous discourse. The explanatory power of D-LTAG comes from both initial and auxiliary trees. In Chapter 4, we will propose a parsing algorithm that locates *local* discourse relations in a similar fashion: it checks whether a connective occurrence is a discourse connective, and if yes, it triggers the argument labeler to locate and label the two connected text spans.

Forbes et al. (2003) implemented a rule-based discourse parsing system following the D-LTAG framework with an integration of sentence-level and text-level parsing. Discourse structure is derived in two passes of parsing: the first parses each single sentence into its tree derivation, after which the second pass constructs a single discourse

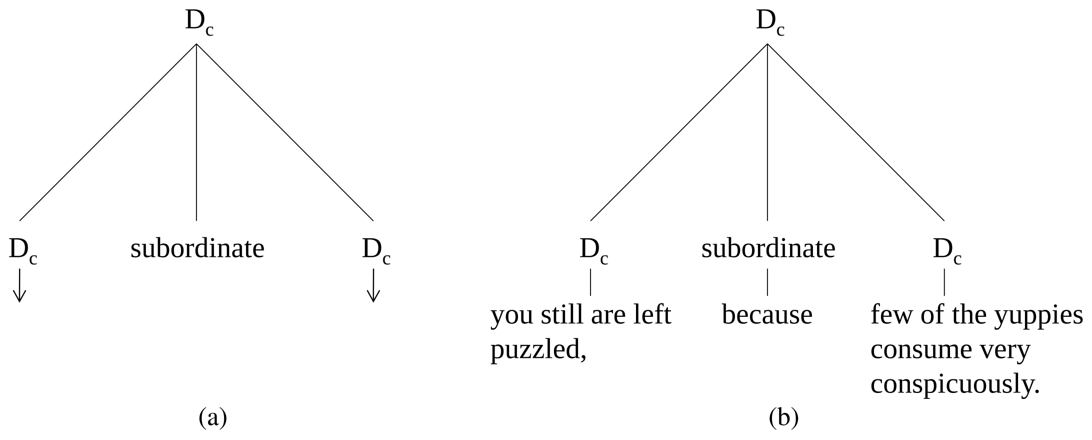

D_c
subordinate
D_c
D_c
↓
↓
(a)
D_c
D_c
subordinate
D_c
you still are left puzzled,
because
few of the yuppies consume very conspicuously.
(b)

Figure 2.4: (a) A D-LTAG initial tree for subordinate substitution. D_c stands for discourse clause, $\downarrow$ indicates a substitution point, and *subordinate* represents a subordinate conjunction. (b) The tree after applying (a) on to the span bc in Figure 1.1.

[The pilots could play hardball by noting they are crucial to any sale or restructuring <u>because</u> they can refuse to fly the airplanes. $]_a$ <u>If</u> [they were to insist on a low bid of, say \$200 a share, $]_b$ [the board mightn't be able to obtain a higher offer from other bidders $]_c$ <u>because</u> [banks might hesitate to finance a transaction the pilots oppose. $]_d$ <u>Also</u>, <u>because</u> [UAL chairman Stephen Wolf and other UAL executives have joined the pilots' bid, $]_e$ [the board might be able to exclude him from its deliberations in order to be fair to other bidders. $]_f$

Figure 2.5: A text excerpt taken from a WSJ article wsj_2172. Its discourse tree that is parsed by Forbes et al.'s rule-based parser is shown in Figure 2.6. Clauses are subscripted with letters.

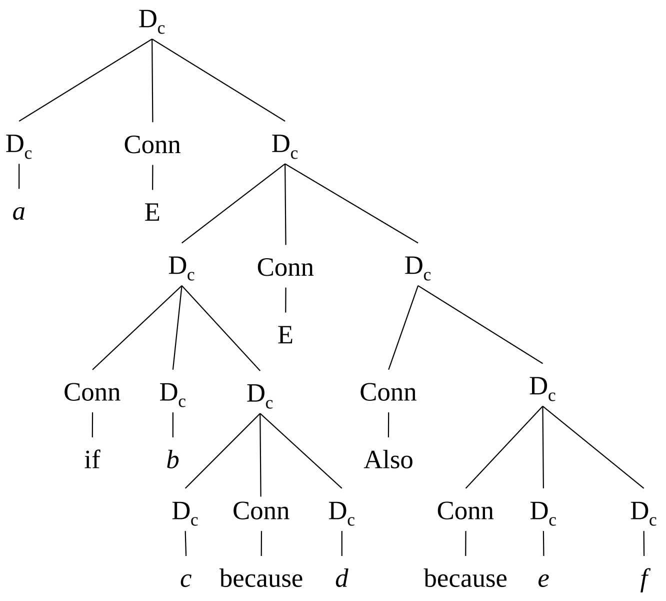


Figure 2.6: Discourse tree derived by Forbes et al.'s parser for the text in Figure 2.5. Null anchors are labeled with E.

tree from the derivation. Figure 2.5 gives a text excerpt taken from a WSJ article wsj_2172. When given this text as an input, Forbes et al.'s parser derives the tree shown in Figure 2.6. Note that the first discourse connective "because" is ignored by the parser, as the parser does not consider sententially embedded connectives. There is a number of shortcomings in this parsing algorithm:

- This parser only builds the syntactic structure for a discourse, without providing the semantics of this discourse structure, *i.e.*, the relation types such as Contrast and Cause.
- Although D-LTAG defines auxiliary trees and substitution rules, it does not enforce the text to be represented as a single discourse tree. In fact, due to the notion of locality, the discourse structures in the PDTB (which is constructed based on the D-LTAG framework) are mostly graphs.

- Sententially embedded discourse connectives are not taken into consideration, which may lead to a large portion of connectives being left out of the parsing process.

2.3.1 Recent Work in the PDTB

The PDTB is the first annotation that follows the lexically grounded, predicate-argument approach in the D-LTAG. With the advent of the PDTB, some recent work has attempted to recognize discourse connectives, relation types and arguments in this newer corpus. Using syntactic features extracted from the parse trees, Pitler and Nenkova (2009) introduced a model that is able to disambiguate the discourse usage of connectives (*i.e.*, discourse vs. non-discourse) and recognize Explicit relations. They extracted syntactic features from the constituent parses with regard to the connectives, and showed that it significantly outperforms the baselines on a 10-fold cross validation.

Wellner and Pustejovsky (2007) and Wellner (2009) proposed machine learning approaches to identify the *head words* of the two arguments for discourse connectives in the PDTB. They applied constituent features, dependency features, lexico-syntactic features, as well as the connective and its contextual features. Elwell and Baldridge (2008) followed this work with the use of general and connective specific rankers and their combinations. Although their method is capable of locating the positions of the arguments, it is not able to label the span of these arguments.

Besides (Pitler et al., 2009)'s work on classifying Implicit relations, as briefly introduced in the previous section, Wang et al. (2010) used a tree kernel and temporal ordering information to automatically recognize and classify both Explicit and Implicit discourse relations. The motivation is that temporal ordering information, such as verb tense, aspectual, and event orders between the two argument spans, usually constrain the discourse relation type.

All of these research efforts in the PDTB can be viewed as isolated components of a full parser. In Chapter 4 we propose a system that differs from these prior efforts in that it connects all subtasks into a single pipeline, and we implement this pipeline into an end-to-end parser in the PDTB style. The parser attempts to: recognize explicit discourse connectives, identify relation types and argument spans for both Explicit and Non-Explicit relations, and recognize attribution spans for these relations. Component-wise, we introduce two novel approaches to accurately locate and label arguments, and to label attribution spans. We also significantly improve on the current state-of-the-art connective classifier with newly introduced features.

2.4 Coherence Modeling

Coherence is a property that makes a text semantically meaningful and easy to read. Coherence is related to the discourse structure of a text, and in Chapter 5, we will apply discourse parsing in modeling coherence. In this section, we review related work in coherence modeling.

Barzilay and Lapata (2005; 2008) proposed a framework for representing and assessing local text coherence. The model is motivated by Centering Theory (Grosz et al., 1995), which states that subsequent sentences in a locally coherent text are likely to continue to focus on the same entities as in previous sentences. Their assumption was that the distribution of discourse entities (classes of coreference noun phrases) in locally coherent texts at the level of sentence-to-sentence transitions exhibits certain regularities. Local coherence is represented as an *entity grid*, in which the vertical axis corresponds to entities and the horizontal axis corresponds to a sequence of sentences. Each grid cell $r_{i,j}$ represents the role (subject (s), object (o), other (x), or absent (-)) of entity e_j in sentence S_i, and each grid column r_j represents the transitions of entity e_j from a sequence of sentences. Table 2.2 illustrates a fragment of an entity grid. They adopted a machine learning approach and demonstrated their model's ability to discern coherent texts from incoherent ones. Elsner and Charniak (2008) improved the entity-based model by adding

	Department	Trial	Microsoft	Evidence	Competitors	Markets	Products	Brands	Case	Netscape	Software	Tactics	Government	Suit	Earnings
S_1	s	o	s	x	o	-	-	-	-	-	-	-	-	-	-
S_2	-	-	o	-	-	x	s	o	-	-	-	-	-	-	-
S_3	-	-	s	o	-	-	-	-	s	o	o	-	-	-	-
S_4	-	-	s	-	-	-	-	-	-	-	-	s	-	-	-
S_5	-	-	-	-	-	-	-	-	-	-	-	-	s	o	-
S_6	-	x	s	-	-	-	-	-	-	-	-	-	-	-	o

Table 2.2: A fragment of the entity grid. Noun phrases are represented by their head nouns.

a model of discourse-new entity identification, and they subsequently presented a better discourse-new model in (Elsner and Charniak, 2011) by adding additional features that better capture the likelihood that an entity will be subsequently mentioned.

Another line of modeling local, sentential coherence is by constructing word co-occurrence coherence models. Lapata (2003) proposed a sentence ordering model, where she made the assumption that in an adjacent sentence pair (S_i, S_{i+1}), a word $w_{i+1} \in S_{i+1}$ is generated independently by words $w_i \in S_i$, and the coherence of S_i and S_{i+1} is defined by the product of all word pair probabilities $P(w_{i+1}|w_i)$. The coherence of a text is then determined by aggregating the sentential coherence:

$$\begin{aligned} P(T) &= \prod_i P(S_{i+1}|S_i) \\ &= \prod_i \prod_{(w_i,w_{i+1}) \in S_i \times S_{i+1}} P(w_{i+1}|w_i) \end{aligned} \tag{2.6}$$

In a similar vein, Soricut and Marcu (Soricut and Marcu, 2006) introduced a coherence model based on IBM Model 1 (Brown et al., 1993), as shown here:

$$P(T) = \prod_i \prod_{w_{i+1} \in S_{i+1}} \frac{\epsilon}{|S_i|+1} \sum_{w_i \in S_i} P(w_{i+1}|w_i) \tag{2.7}$$

This model is trained with an Expectation-Maximization (EM) procedure over pairs of consecutive sentences, in order to deduce hidden variables' values. The difference between this model and Lapata's model is that, with the hidden variables, we learn which w_i is responsible for generating w_{i+1}.

Newswire reports usually follow certain patterns of topic shift in reporting events. For example, articles reporting earthquakes typically contain information about quake strength, location, and casualties, and descriptions of casualties usually precede those of rescue efforts. Based on this domain-specific observation, a well-written earthquake report must be globally more coherent than a poorly-written one. Barzilay and Lee (2004) proposed a domain-dependent HMM model to capture topic shift in a text, where topics are represented by hidden states and sentences are observations. The global coherence of the text can then be summarized by the overall probability of topic shift from the first sentence to the last.

Following the two directions on local and global coherence, Soricut and Marcu (2006) and Elsner et al. (2007) combined the entity-based and HMM-based models and demonstrated that these two models are complementary to each other in coherence assessment. In Elsner et al. (2007), the entity-based model is combined into the HMM model in a way that, the observations (*i.e.*, sentences) generated by the HMM hidden states (*i.e.*, topics) are further categorized into known entities, new entities, and non-entities, when entity-based probabilities are defined.

The above approaches all deal with entities and terms that appear in the text, and do not address other phenomena that are also indicative of discourse coherence. The coherence model in Chapter 5 differs from these models in that it introduces and operationalizes another indicator of discourse coherence, by modeling a text's discourse relation transitions. Karamanis (2007) has tried to integrate local discourse relations into the Centering-based coherence metrics for the task of information ordering, but was not able to obtain improvement over the baseline method. This may have been partly due to the much smaller data set and the way the discourse relation information was utilized in their heuristic constraints and rules.

Realizing that adjacent sentences exhibit syntactic regularities and sentences with similar syntax are likely to have the same communicative goal, Louis and Nenkova (2012) recently proposed a coherence model based on syntactic patterns to capture intentional structure. They showed that such patterns correlate with the intentional structures of academic conference articles and the proposed model is able to predict the coherence of abstract, introduction, and related work sections of these articles.

2.5 Summarization and Argumentative Zoning

Discourse relations can provide crucial information for downstream NLP applications. For example, a text summarization system may utilize the contrast and restatement relations to recognize updating information and redundancy. In this section, we discuss related work for two NLP applications – summarization and argumentative zoning – upon which we will later apply our discourse parser on. For summarization, we specifically focus on summarizing scientific articles.

Automatic text summarization for scientific articles has been studied for more than half a century (Sparck-Jones, 1998; Mani and Maybury, 1999; Mani, 2001). One of the early work that produces abstracts for scientific articles is that of (Luhn, 1958). The intuition is that the salience factor of a sentence is derived from an analysis of its words, whereas the frequency of word occurrence in the article suggests a useful measure of the word salience. Luhn used word frequency statistics to compute a list of frequent words for an article, and measured the salience of a sentence by a normalized sum of the frequent words. The most salient sentences are output as the abstract of the article. This *key word* method was later on utilized in Edmundson (1969)'s automatic sentence extraction system. Besides this, Edmundson also proposed three new components: cue words (*i.e.*, pragmatic words such as “significant” and “impossible”), title words (namely, words collected from the titles, subtitles, and headings for that article), and sentence location. In the system, the relative weights for the four components were parameterized in a linear function so that the system can be tuned.

Many researchers have proposed methods to measure the salience of sentences in extractive summarization (Radev et al., 2004; Erkan and Radev, 2004a; Ye et al., 2005; Lin et al., 2007). Erkan and Radev (2004a) proposed LexRank to define sentence importance based on a graph algorithm to compute the centrality of sentences. A lexical graph is constructed by treating sentences as nodes and connecting them by their lexical similarities. After construction, they then apply a random walk on the graph until it converges. The weights of the sentence nodes are the sentence importance scores.

Current work on scientific article summarization includes summarizing citations and related work. The citation text of a target article A is defined as the set of sentences in other articles that explicitly cite A. Qazvinian and Radev (2008) argued that citation texts are useful in creating a summary of the important contributions of a scientific article. They constructed a lexical graph for the citation sentences and applied a network-based clustering algorithm to select representative sentences to form a citation summary for A. This notion was extended by Mohammad et al. (2009) to automatically create a survey paper given a research topic. They compared a textual graph algorithm and a clustering approach in summarizing multiple scientific papers in order to generate a survey. Hoang and Kan (2010) proposed the task of summarizing related work for a research topic. Given a set of research articles and a topic tree that defined by the user, their system automatically generates the related work section.

Ono et al. (1994) developed an automatic abstract generation system for Japanese based on the RST structure of the text. RST takes a nucleus-satellite view on rhetorical relations, which they leverage as an indication of the importance of the relation spans. They suggested a penalty score for every relation span based on their nucleus-satellite orientation: a span receives a penalty if it is the satellite in a relation. For example, in Figure 2.3, the spans a and b receive a penalty as they are the satellite spans for the Attribution and Enablement relations, respectively. The abstract can then be generated by selecting sentences with smallest penalty scores.

Recently, Louis et al. (2010) investigated the predictive power of two specific classes of information in discourse:

- The structure information provided by the RST structures, and
- The semantic information provided by the types of the relations in the PDTB.

They used as structural features the penalty scores from (Ono et al., 1994) and the promotion set scores from (Marcu, 1998). The motivation is similar as we discussed in the previous paragraph. To use as the semantic features, they applied the gold standard relation types from the PDTB. For each sentence, they checked whether it contains a single-sentential or multi-sentential relation, the Arg1 or Arg2 tag (when multi-sentential), the relation type of the discourse relation, whether it is an Explicit or Implicit relation, and the number of relations that a sentence is involved in. For example, the sentence in Example 2.1 will generate the semantic feature "expresses Explicit Contingency relation", and the first sentence in Example 2.2 will generate the feature "contains Arg1 of Implicit Contingency relation". They showed that structural features are more predictive than semantic features, and both are complementary to standard non-discourse features for summarization. Louis et al. experimented in the gold standard PDTB data and showed discourse information is complementary to non-discourse features. Although they applied discourse information on news domain, we believe that such information is also useful when we summarize scholarly papers. In Chapter 6, we will illustrate this by applying an automatic discourse parser in summarizing scholarly papers.

Teufel (Teufel, 1999) proposed *argumentative zoning* (AZ) as a task to annotate rhetorical functions of sentences in scientific articles. She defined a scheme with seven

Category	Abbr.	Description
Aim	AIM	Specific research aims
Basis	BAS	Other work that provides basis for own work
Background	BKG	Generally accepted background knowledge
Contrast	CTR	Contrast, comparison, weaknesses of other work
Other	OTH	Statements that describe other work
Own	OWN	Statements that describe authors' own work
Textual	TXT	Textual organization of the article

Table 2.3: Argumentative zones defined in (Teufel, 1999)

CTR	The previously proposed semantic-head-driven generation methods run into problems if none of the daughter constituents in the syntacto-semantic rule schemata of a grammar fits the definition of a semantic head given in (Shieber et al. 1990).
OTH	This is the case for the semantic analysis rules of certain constraint-based semantic representations, e.g. Underspecified Discourse Representation Structures (UDRSs) (Frank and Reyle 1992).
AIM	Since head-driven generation in general has its merits, we simply return to a syntactic definition of 'head' and demonstrate the feasibility of syntactic-head-driven generation.
OWN	In addition to its generality, a syntactic-head-driven algorithm provides a basis for a logically well-defined treatment of the movement of (syntactic) heads, for which only ad-hoc solutions existed, so far.

Figure 2.7: An abstract taken from a paper published in COLING 1994 (König, 1994). Sentences are labeled by their rhetorical functions.

categories of rhetorical functions, namely, aim (AIM), basis (BAS), background (BKG), contrast (CTR), other (OTH), own (OWN), and textual (TXT) (cf. Table 2.3). Among these seven categories, background, other, and own provide the basic intellectual ownership of the article, and the other four provide finer-grained description of the rhetorical moves of the article. As examples, AIM sentences are sentences that describe the specific research aims, contributions, and conclusions of the current work, and CTR sentences are statements of contrast, comparison, and weaknesses of other work. Figure 2.7 shows an abstract taken from a paper. The four sentences from this abstract are labeled with the corresponding AZ categories: CTR, OTH, AIM, and OWN. Automatic classification approaches have been proposed in (Teufel, 1999; Teufel and Moens, 2002; Merity et al., 2009; Teufel and Kan, 2011), in order to automatically assign AZ categories to the sentences in scientific articles. Teufel and Moens (2002) used a naïve Bayesian classifier with a rich set of features to perform AZ classification. These features include content features, verb syntax, citation features, locations, and a rich set of hand-crafted rules. The input to their classifier is a scientific article annotated with AZ categories, citation information, and structure information (*i.e.*, sections, titles, *etc.*). Teufel and Kan (2011) proposed a robust argumentative zoning classifier that is capable of taking plain texts from scientific articles as input. They used a maximum entropy classifier. They included n-gram features into (Teufel and Moens, 2002)'s feature pool, without the hand-crafted rules. We hypothesize that discourse information will be useful in recognizing certain AZ categories. For example, a Contrast relation can provide clue to judge whether a sentence belongs to a CTR category. We will illustrate how to apply discourse parsing in argumentative zoning in Chapter 6.

2.6 Conclusion

In this chapter, we gave an overview of the PDTB, and reviewed related work in Implicit discourse relation classification, discourse parsing, recent work in the PDTB, coherence modeling, and two NLP tasks - summarization and argumentative zoning. In the upcoming chapters, we will detail the building of a full discourse parser in the PDTB style and how we can make use of the discourse parser's output to model coherence and improve downstream NLP tasks.

Chapter 3

Classifying Implicit Discourse Relations

In this chapter, 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 yield a significant 14.1% improvement over the baseline. In our error analysis, we discuss four challenges in recognizing Implicit relations in the PDTB.

3.1 Introduction

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. As in the recent release of the second version of the PDTB (Prasad et al., 2008), among all relations annotated, about 45% of them are Explicit relations, 40% are Implicit relations, and the remaining $\sim$15% belongs to AltLex, EntRel, and NoRel relations. PDTB does not examine sentence pairs across paragraphs for Implicit relations. It is crucial that a discourse parser is capable of identifying Implicit relations. With the PDTB, which provides a larger, cleaner, and more thorough Implicit relation annotation than the previous discourse corpora such as RST-DT (cf. Chapter 2), there is an opportunity to address this area of work.

In this chapter, we provide classification of Implicit discourse relations on the second version of the PDTB. One thing to note is that a system that handles Implicit relations needs to identify that there exists an Implicit relation (but not Explicit, AltLex, EntRel, or NoRel) between two text spans before classifying the type of the Implicit relation. We focus on the second step in this chapter, *i.e.*, we assume the input to our classifier consists of all Implicit relations and we only classify their relation types. In the next Chapter, we will describe a separate module which is capable of both identifying the existence of an Implicit relation and classifying its relation type. The features we use 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.

We first explain the Implicit relation types that we are going to work on. 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 chapter.

3.2 Implicit Relation Types in PDTB

The PDTB provides a three level hierarchy of relation tags for its annotation. 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 3.1 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-

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 3.1: 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.

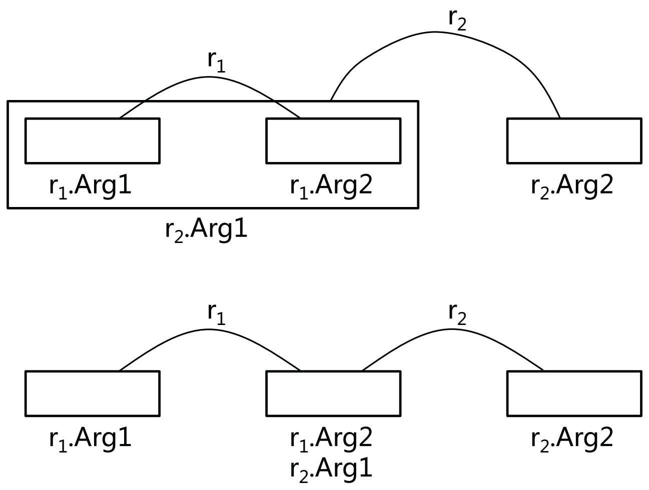


Figure 3.1: Two types of discourse dependency structures. Top: fully embedded argument, bottom: shared argument.

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%).

3.3 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.

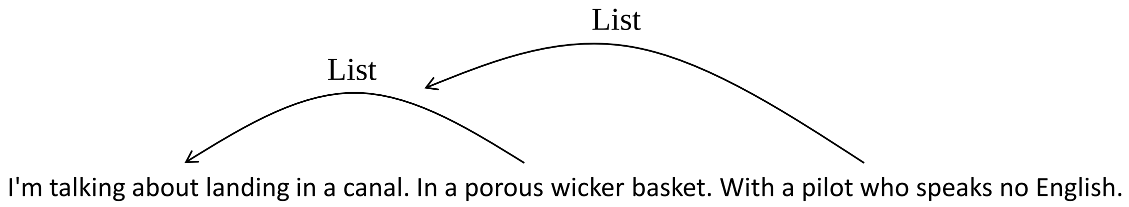

List
List
I'm talking about landing in a canal. In a porous wicker basket. With a pilot who speaks no English.

Figure 3.2: Two List relations. Similar to other figures in this thesis, I denote discourse relations with an arrow for notational convenience, although there is no directionality distinction. I denote Arg2 as the origin of the arrow and Arg1 as the destination of the arrow.

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 arguments and shared arguments are the most common patterns, which are shown in Figure 3.1. The top portion of Figure 3.1 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. Figure 3.2 shows such an example with two List relations[1] As we are using the

[1]The PDTB also contains some exceptions where Arg1 of the last List relation covers more than the

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

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 3.2. For the two List relations in Figure 3.2, the contextual features are "*curr* embedded in *next*.Arg1" for the first List relation, and "*prev* embedded in *curr*.Arg1" for the second. In real data where we do not have gold standard discourse relations, we may run the classifier first without contextual feature, and run it in a second iteration where we use the result from the first iteration to compute the contextual feature.

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.

previous List relation or partially covers the previous List relation.

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). Thus, we expect that syntactic features will also help in our task. 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 3.3(a) usually signals an Asynchronous relation when it appears in Arg2, as shown in Example 3.1, while the structure in Figure 3.3(b) usually acts as a clue for a Cause relation when it appears in Arg1, as shown in Example 3.2. In both examples, the lexicalized parts of the parse structure are highlighted with wavy underline.

(3.1) *But the RTC also requires "working" capital to maintain the bad assets of thrifts that are sold*

Implicit = SUBSEQUENTLY **That debt would be paid off as the assets are sold**

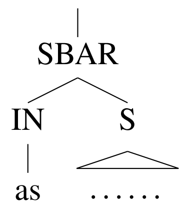
(Asynchronous - wsj_2200)

(3.2) *It would have been too late to think about on Friday.* Implicit = SO **We had to think about it ahead of time.**

(Cause - wsj_2201)

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 3.3 are tree fragments, tree fragments are not extracted since production rules act as generalization of tree fragments. As an example, Figure 3.4 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

(a)



(b)

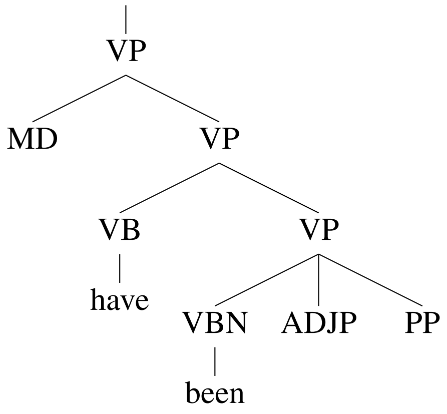


Figure 3.3: (a) constituent parse in Arg2 of Example 3.1, (b) constituent parse in Arg1 of Example 3.2.

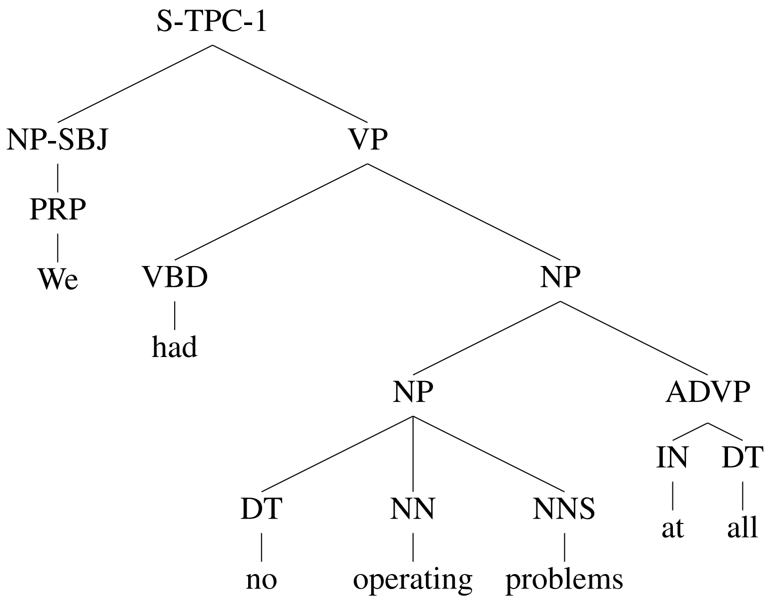


Figure 3.4: A gold standard subtree for Arg1 of an Implicit discourse relation from wsj_2224.

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. In the example in Figure 3.4, "... had/have no/not ... at all" usually signals a restatement relation with inferred connective of "indeed" or "in fact", and this pattern is captured by the set of production rules.

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 3.5 shows the dependency subtree for the same example in Figure 3.4, from which we collect three dependency rules: "had" $\leftarrow$ nsubj dobj, "problems" $\leftarrow$ det nn advmod, "at" $\leftarrow$ dep. Similarly as for Figure 3.4 and Figure 3.5, we hope to capture the same pattern with the set of dependency rules.

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. When we use real data, both constituent parse and dependency parse need to be done automatically, and we expect errors from both parsers.

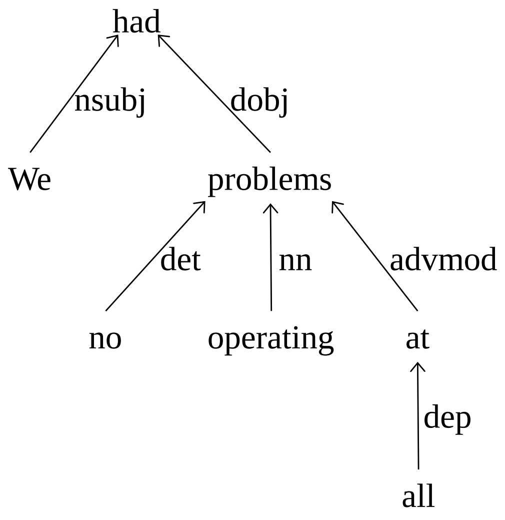


Figure 3.5: A dependency subtree for Arg1 of an Implicit discourse relation from wsj_2224.

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. At the beginning of Section 2.2, we have shown a word pair example of (good, fails) which helps to signal a Contrast relation between the adjacent sentence pair. The example is repeated in Example 3.3 for reference.

(3.3) John is *good* in math and sciences.
Paul *fails* almost every class he takes.

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. In Example 3.3, both Arg1 and Arg2 contain 7 stemmed words, which gives rise to $7 \times 7 = 49$ word pairs. Unlike Marcu and Echihabi's study, we limit the collection of word pair statistics to occurrences only in the training set of the PDTB corpus.

3.3.1 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., Sections 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.

3.4 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[2], 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 3.4, which is labeled as Conjunction and Contrast. 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.

(3.4) *Sales surged 40% to 250.17 billion yen from 178.61 billion.*

[2](PDTB-Group, 2007) suggests to use Sections 2 – 21 for training, 22 for development, 23 for testing, and 0, 1, and 24 for additional development.

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

Implicit = MEANWHILE **Net income rose 11% to 29.62 billion yen from 26.68 billion.**

(Conjunction; Contrast - wsj_2242)

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.

3.4.1 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 3.3), 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.

Interestingly, we note 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

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

pairs (as word pairs are sparser). We then repeated the same set of experiments, as shown in Table 3.4 (Row 4 of this table is repeated from Table 3.3 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%. Comparing Table 3.3 and 3.4, feature selection helps to improve the performance for production rules, dependency rules, and word pairs by 1.7%, 6.4%, and 2.6%, respectively. The main reason is that the feature set before feature selection contains too much noise. As in Example 3.3, only one out of the 49 word pairs – namely, (good, fails) – contributes to signal the Contrast relation. Such noise is more pronounced in relations with longer sentences.

Table 3.4 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 3.5. These results confirm that each additional feature class indeed contributes a marginal performance improvement (although not statistically significant), and that all

	# 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 3.5: Accuracy with feature classes gradually added in the order of their predictiveness.

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 3.6: Recall, precision, F_1, and counts for 11 Level-2 relation types. "–" indicates 0.00.

feature classes are needed for optimal performance.

Note that Row 3 of Table 3.3 corresponds to a system similar to (Marcu and Echihabi, 2002) which applies only word pair features. The differences are that they used a Naïve Bayes classifier while we used a maximum entropy classifier and they trained on a much larger data set of artificially constructed relations while we trained on actual relations. As we did not implement their Naïve Bayes classifier, we compare their method's performance using the result from Table 3.3 Row 3 with ours from Table 3.5 Row 4, which shows that our system significantly ($p < 0.01$) outperforms theirs.

Table 3.6 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 3.5). 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 3.1, 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 3.6. 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 our data set excludes EntRel and AltLex.

3.5 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.

Pitler et al. (2009) performed a 6-way classification on the four Level-1 types with EntRel and NoRel, and they achieved an accuracy of 44.58%. This also suggests that classification of Implicit relations is a tough task even if we back off to the coarse-grained Level-1 types.

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 relations without sufficient contextual and domain knowledge. For example, we notice that a lot of Contrast relations are mistakenly classified as Conjunction (in Table 4.8 where we train a classifier with Implicit, AltLex, EntRel, and NoRel relations, 26 Contrast relations have been mistakenly labeled 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 lexical choice, syntax, and semantics, as Examples 3.5 and 3.6 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. Without sufficient domain knowledge, it is very difficult to disambiguate the relation types for these two examples. It is worth noting that the implicit discourse connective given by the annotators is "while" in both examples, which is an ambiguous connective as shown in (Miltsakaki et al., 2005). As the annotation process of the implicit connectives and the relation types are separated (*i.e.*, the implicit connectives were ignored when relation type annotation was done), it shows that the annotators were aware of the similarity between these two examples and thus inferred the same implicit connective.

(3.5) *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.* Implicit = WHILE **Revenue rose 17% to $2.73 billion from $2.33 billion a year earlier.**

(Contrast - wsj_1812)

(3.6) *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.* Implicit = 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 assign 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 3.4 in Section 3.4 is such a case where the implicit connective "meanwhile" may be interpreted as expressing a Conjunction or Contrast relation.

Inference

Sometimes inference and a knowledge base are required to resolve the relation type. In Example 3.7, to understand that Arg2 is a restatement of Arg1, we need a semantic mechanism to show that either the semantics of Arg1 implies that of Arg2 or the other way around. In the below example, "I had calls all night long" implies "I was woken up every hour" semantically, as shown in:

$$receive\_call(I) \wedge duration(all\_night) \Rightarrow woken\_up(I) \wedge duration(every\_hour)$$

Relation type	Semantic representation
Cause	$\lvert Arg1\rvert \prec \lvert Arg2\rvert \quad \vee \quad \lvert Arg2\rvert \prec \lvert Arg1\rvert$
Concession	$A \prec C \quad \wedge \quad B \Rightarrow \neg C$ where $A \in \lvert Arg1\rvert$, $B \in \lvert Arg2\rvert$
Instantiation	$exemplify(\lvert Arg2\rvert, \lambda x.x \in E)$ 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 3.7: Some examples of relation types with their semantic representations, as taken from (PDTB-Group, 2007).

(3.7) *"I had calls all night long from the States,"* $\boxed{\textit{he said.}}$ "Implicit = 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 3.7, 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.

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*).

(3.8) **Previous 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,
but they aren't based on fundamentals. Implicit = 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 3.9, 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 all its union members from working, so the 2,400 machinists that are still working must not be union members, which is given the reason in the Arg2 span.

(3.9) *And at the company's Wichita, Kan., plant, about 2,400 of the 11,700 machinists still are working,* $\boxed{\textit{Boeing said.}}$ Implicit = 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 as a gift but actually has harmful intent, and hence understand that Arg2 poses a contrasting view of the delegation as Arg1 does.

(3.10) *Senator Pete Domenici calls this effort "the first gift of democracy".*

Implicit = BUT **The Poles might do better to view it as a Trojan Horse.**

(Contrast - wsj_2237)

Context and world knowledge are considered supplementary material. This material is not necessary for recognizing the existence of a relation and the relation type, but can improve the classifier when it is added to the system. The last row of Table 3.5 has shown that integrating a gold standard contextual feature marginally improves the performance.

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. We did not calculate the explicit percentage for these four types, as sometimes it is vague to point out whether a certain type appears or does not appear in a relation. For example, understanding the context will always help better understand the relation, but it is difficult to tell whether a relation needs contextual modeling or estimate how much contextual information is needed.

3.6 Conclusion

We implemented an Implicit discourse relation classifier and showed initial results on the PDTB. The features 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.

Chapter 4

An End-to-End Discourse Parser

In this chapter, we build on our Implicit classifier to design and develop an end-to-end discourse parser to parse free texts in the PDTB style, in a fully data-driven approach. The parser consists of multiple components joined in a sequential pipeline architecture, which includes a connective classifier, argument labeler, Explicit relation classifier, Non-Explicit relation classifier, and attribution span labeler. Non-Explicit relations include Implicit, AltLex, EntRel, and NoRel relations. Our trained parser first identifies all relations, locates and labels their arguments, and then classifies their relation types. When appropriate, the parser also determines the attribution spans to these relations. We introduce novel approaches to locate and label arguments, and to identify attribution spans. We also significantly improve on the current state-of-the-art connective classifier. We propose and present a comprehensive evaluation from both component-wise and error-cascading perspectives, in which we illustrate how each component performs in isolation from the rest of the pipeline, as well as how the whole pipeline performs with the components joined together and error propagated down the pipeline. The parser gives an overall system F_1 score of 46.80% for partial matching with gold standard parses, and 38.18% with full automation.

This system includes novel components to locate and label arguments as well as improved components from previous work. We also propose and present a comprehensive evaluation on the parser from both component-wise and error-cascading perspectives. To the best of our knowledge, this is the first parser that performs end-to-end discourse parsing in the PDTB style.

4.1 System Overview

We design our parsing algorithm as a sequential pipeline to mimic the annotation procedure performed by the PDTB annotators. Figure 4.1 shows the pseudocode. The input to the parser is a free text T, whereas the output is the discourse structure of T in the PDTB style. The algorithm consists of three steps which sequentially label Explicit relations, Non-Explicit relations, and attribution spans. Non-Explicit relations include all relations that are not Explicit – *i.e.*, Implicit, AltLex, EntRel, and NoRel.

The first step is to deal with Explicit relations. This step is further divided into the following sub-steps:

1. Identifying discourse connectives,
2. Labeling their Arg1 and Arg2 spans, and
3. Recognizing their Explicit relation types.

First, the parser identifies all connective occurrences in T (Line 2 in Figure 4.1), and labels them as to whether they function as discourse connectives or not (Lines 3–4). If a connective occurrence C is determined to be a discourse connective, its Arg1 and Arg2 spans are then identified. The system examines whether the Arg1 and Arg2 spans appear in the same sentence or different sentences. In this work, we focus on the first case where we extract the Arg1 and Arg2 spans from the same sentence. The parser next classifies the tuple (C, Arg1, Arg2) into one of the Explicit relation types (Lines 5–7).

```
Input: a text $T$
Output: a discourse structure of $T$
// Step 1: label Explicit relations
Identify all connective occurrences in $T$
for each connective occurrence $C$ do
    Label $C$ as disc-conn or non-disc-conn
    if $C$ is disc-conn then
        Label Arg1 span and Arg2 span of $C$
        Label ($C$, Arg1, Arg2) as one of the Explicit relations
    end if
end for

// Step 2: label Non-Explicit: Implicit, AltLex, EntRel, and NoRel relations
for each paragraph $P$ in $T$ do
    for each adjacent sentence pair ($S_i$, $S_j$) in $P$ do
        if ($S_i$, $S_j$) is not labeled as an Explicit relation in Step 1 then
            Label ($S_i$, $S_j$) as EntRel, NoRel, or one of the Implicit/AltLex relations
        end if
    end for
end for

// Step 3: label attribution spans
Split $T$ into clauses
for each clause $U$ do
    if $U$ is in some Explicit/Implicit/AltLex relation from Step 1 or 2 then
        Label $U$ as attr-span or non-attr-span
    end if
end for
```

Figure 4.1: Pseudocode for the discourse parsing algorithm.

The second step then examines all adjacent sentence pairs within each paragraph. For each pair (S_i, S_j) that is not identified in any Explicit relation from Step 1, the parser then classifies the pair into EntRel, NoRel, or one of the Implicit/AltLex relation types (Lines 12–15). Following the PDTB annotation convention, our parser also ignores inter-paragraph relations, *i.e.*, it ignores the adjacent sentence pair in between two paragraphs.

In Step 3, the parser first splits the text into clauses (Line 21), and for each clause U that appears in any discourse relations (*i.e.*, Explicit, Implicit, and AltLex relations; we define EntRel and NoRel as non-discourse relations), it checks whether U is an attribution span (Lines 22–24). In this step, the parser also follows the PDTB representation to only identify attribution spans appearing in discourse relations.

In our work, we adopt a sequential pipeline to parse a text, instead of following a top-down or bottom-up approach common to parsing in general and to parsing RST discourse trees (Marcu, 1997) in specific. The reason is twofold. First, the algorithm is designed to mimic the annotation procedure performed by the PDTB annotators. Second, as the PDTB makes no commitments to what kinds of high-level structures may be built up from the low-level units, it does not make sense to presume a tree-like structure and adopt a tree parsing algorithm. Note that this design allows a relation to be embedded within the argument of another relation, as well as an argument to be shared between two adjacent relations (cf. the two structures in Figure 3.1). Therefore, the structure that constructed by the relations is more like a graph, as illustrated by Figure 1.2 in Chapter 1.

The pipeline of the parser is shown in Figure 4.2. The first three components correspond to Step 1 in Figure 4.1, while the last two correspond to Steps 2 and 3, respectively. There are two sub-components in the argument labeler: an argument position classifier and an argument extractor. A detailed description of these components follows in the next section.

To illustrate the parsing algorithm, we look at how an ideal parser parses the excerpt of three sentences in Figure 4.3. In the first step, after comparing the text against

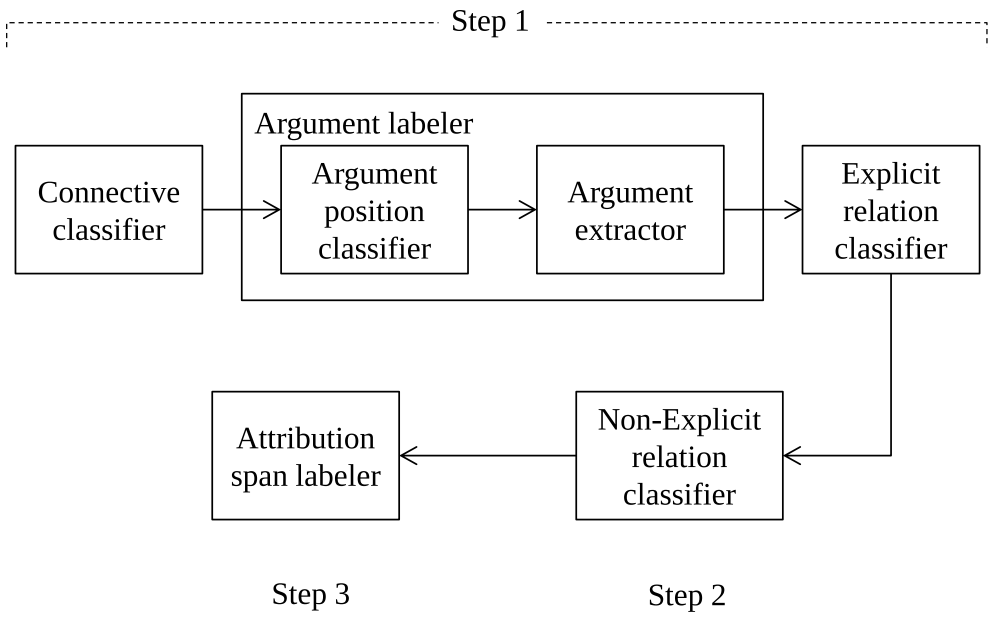


Figure 4.2: System pipeline for the discourse parser.

a list of 100 discourse connectives defined in the PDTB, three connective occurrences are identified, which are "and" in S_1, "and" in S_2, and "But" in S_3. The connective classifier then checks these occurrences and labels the second and third as discourse connectives, as underlined in Figure 4.3. The first connective "and" in "spot and futures prices" does not serve as a discourse connector. The argument labeler follows by labeling the Arg1 and Arg2 spans for each discourse connective. In S_2, the Arg1 and Arg2 spans for the connective "and" labeled out by the argument labeler are Span b ("He adds that ... a mutual funds,") and Span c ("volatility ... fund."). The tuple ("and", b, c) is next propagated to the Explicit relation classifier, which classifies the relation type as Conjunction. Similarly, the argument labeler marks S_2 and S_3 as the Arg1 and Arg2 spans for the connective "But", and the Explicit relation classifier labels the tuple ("But", S_2, S_3) as Concession.

In the second step, the parser examines all adjacent sentence pairs within this excerpt, namely, (S_1, S_2) and (S_2, S_3). Since an Explicit relation is already assigned to the

S_1	[Arbitraging on differences between spot and futures prices is an important part of many financial markets, he says. $]_a$
S_2	[He adds that his shares in a company savings plan are invested in a mutual fund, $]_b$ [and volatility, on a given day, may hurt the fund. $]_c$
S_3	[But "I'm a long-term investor," he says. $]_d$

Figure 4.3: An excerpt taken from a *Wall Street Journal* article wsj_0121. The text consists of three sentences. Relations arguments are subscripted with letters. The discourse relations in this text are illustrated in the discourse structure in Figure 4.4.

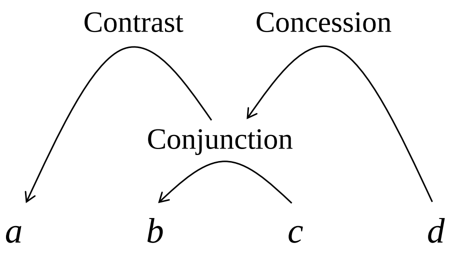


Figure 4.4: Discourse relations for the text in Figure 4.3. Arrows are pointing from Arg2 span to Arg1 span, and labeled with the respective relation types, but do not represent any ordering between the argument spans.

pair (S_2, S_3), it is exempt from further classification. The Non-Explicit relation classifier then classifies the remaining pair (S_1, S_2) as Contrast.

In the last step, the attribution span labeler will examine all discourse relations to label their attribution spans, when applicable. We make an assumption that all attribution spans appear in clauses and within the discourse relations. This will be further illustrated in Section 4.2.5. In Figure 4.3, three clauses are labeled as attribution spans, which are "he says", "He adds", and "he says". These three relations are shown in Example 4.1, 4.2, and 4.3. The results are also illustrated in the discourse structure in Figure 4.4, which is built by joining the relations together. Note that although there is no notion of direction in a PDTB relation, we use arrows to show the orientation of the arguments: an arrow is pointing from the Arg2 span to the Arg1 span. Figure 1.1 and Figure 1.2 show a more complex discourse structure in the PDTB representation.

(4.1) *Arbitraging on differences between spot and futures prices is an important part of many financial markets,* $\boxed{\textit{he says.}}$ $\boxed{\textbf{He adds}}$ **that his shares in a company savings plan are invested in a mutual fund, and volatility, on a given day, may hurt the fund.**

(Comparison.Contrast - wsj_0121)

(4.2) $\boxed{\textit{He adds}}$ *that his shares in a company savings plan are invested in a mutual fund,* <u>and</u> **volatility, on a given day, may hurt the fund.**

(Expansion.Conjunction - wsj_0121)

(4.3) $\boxed{\textit{He adds}}$ *that his shares in a company savings plan are invested in a mutual fund, and volatility, on a given day, may hurt the fund.* <u>But</u> **"I'm a long-term investor,"** $\boxed{\textbf{he says.}}$

(Comparison.Concession - wsj_0121)

We now refine this overview by detailing the individual components' structure and processing workflow.

4.2 Components

Our parser takes a fully-data driven, supervised learning approach. As such, the annotated data is processed into binary feature vectors that are suitable to be learnt by a maximum entropy model. In the following descriptions, we elaborate the component designs as well as the derived feature classes.

4.2.1 Connective Classifier

There are 100 types of discourse connectives defined in the PDTB. Given a connective occurrence such as "when", the parser needs to decide whether it is functioning as a discourse connective. To illustrate, compare the two "and" occurrences in Figure 4.3.

The first "and" in S_1 is used to conjoin "spot" and "futures prices" in a noun phrase, which does not have a discourse usage. On the other hand, "and" in S_2 is functioning as a discourse connective to join two discourse events in Arg1 and Arg2, thus the annotators labeled the type as Conjunction. Among the whole PDTB corpus, there are 29.65$\%$ of connective occurrences annotated as discourse connectives; in other words, more than 70$\%$ are non-discourse connectives. Thus, it is crucial to disambiguate the connectives before sending them down the pipeline to label their argument spans and relation types.

Pitler and Nenkova (2009) showed that syntactic features extracted from constituent parse trees are very useful in disambiguating discourse connectives. Beside the connective itself as a feature, they applied other syntactic features: the highest node in the tree that covers only the connective words (which they termed *self category*), the parent, left and right siblings of the self category, and two binary features that check whether the right sibling contains a VP and/or a trace. The best feature set they demonstrated also included pairwise interaction features between the connective and each syntactic feature (*i.e.*, each connective is paired with each syntactic feature to form a new feature), and the interaction features between pairs of syntactic features (*i.e.*, each syntactic feature is paired with another syntactic feature to form a new feature).

In addition to the above, we observed that a connective's context and part-of-speech (POS) give a very strong indication of its discourse usage. For example, the connective "after" is usually functioning as a discourse connective when it is followed by a present participle, as in "after rising 3.9$\%$". The syntactic parse path from the connective to the root of the tree models how it is syntactically connected to the sentence as a whole, reflecting its functionality within the sentence. Based on these observations, we propose a set of lexico-syntactic and path feature classes for a connective C with its previous word $prev$ and next word $next$:

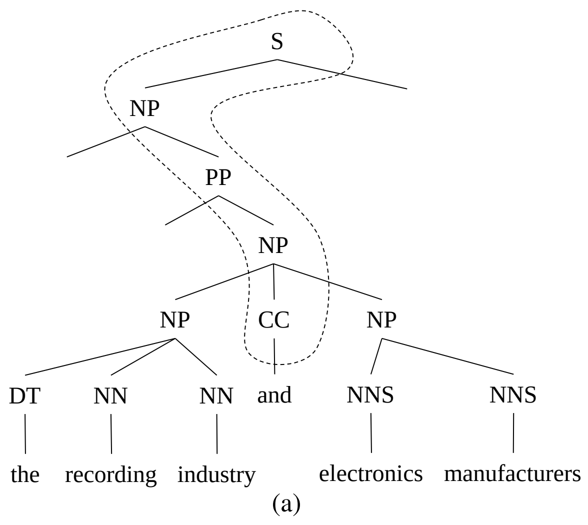
– C POS

– $prev + C$

- $prev$ POS
- $prev$ POS + C POS
- C + $next$
- $next$ POS
- C POS + $next$ POS
- path of C's parent $\rightarrow$ root
- compressed path of C's parent $\rightarrow$ root

Each of the above represents a feature class. The first seven feature classes model the connective's context and POS, while the last two are the path from C to the root and the compressed path where adjacent identical tags are combined (*e.g.*, -VP-VP- is combined into -VP-). The path feature class is novel in the way how it models the syntactic relation between the connective under consideration and the syntactic root. As an example to illustrate the path feature, Figure 4.5 shows the partial parse trees for "and" functioning as a non-discourse connective and a discourse connective. The path feature "path of C's parent $\rightarrow$ root" is circled in both figures. Figure 4.5(b) shows that "and" is directly under a sentence node which is in the verb phrase of the main sentence, while in the non-discourse case in Figure 4.5(a), "and" is embedded in a noun phrase which is part of a prepositional phrase. When analyzing the PDTB, we observe that the path feature for discourse connective "and" usually consists of CC, S, SBAR, and VP, which is captured by the path feature "CC $\uparrow$ S $\uparrow$ VP $\uparrow$ S" in Figure 4.5(b). However, in the non-discourse case of Figure 4.5(a), the same pattern is not observed.

Appendix A.1.1 lists out all features for the connective classifier to disambiguate the connective "after" in Example A.1. The constituent parse tree for the example is shown in Figure A.1. Appendix A.1 uses the Explicit relation in Example A.1 to illustrate the features extracted for the classifiers in Step 1. Features for the classifiers in Step 1 and 3 are shown in Appendix A for illustration and comparison.



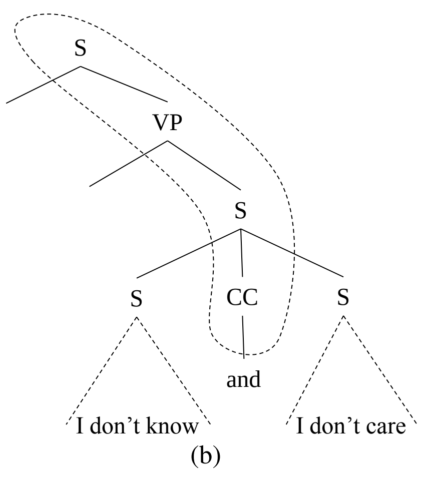


Figure 4.5: (a) Non discourse connective "and". (b) Discourse connective "and". The feature "path of C's parent $\rightarrow$ root" is circled out on both figures.

4.2.2 Argument Labeler

The parser now labels the Arg1 and Arg2 spans of every connective labeled in the previous step as a discourse connective, in two steps:

1. Identifying the locations of Arg1 and Arg2, and
2. Labeling the spans.

We note that Arg2 is the argument with which the connective is syntactically associated, and thus its position is fixed once we locate the connective. The remaining problem of the first step is in identifying the location of Arg1. We implement this as a classification task to recognize the *relative position* of Arg1, with respect to the connective (Line 2 in Figure 4.6). According to the different relative positions of Arg1, the argument extractor then attempts to extract the Arg1 and Arg2 spans in the second step (Lines 5–11 in Figure 4.6). Figure 4.6 gives the pseudocode for the argument labeler, which corresponds to Line 6 in Figure 4.1 and is further discussed in the following.

```
Input: a discourse connective $C$ and the text $T$
Output: Arg1 and Arg2 spans of $C$
 1: // Argument position classifier
 2: Classify the relative position of Arg1 as SS or PS
 3:
 4: // Argument extractor
 5: if the relative position of Arg1 is SS then
 6:     Identify the Arg1 and Arg2 subtree nodes within the sentence parse tree
 7:     Apply tree subtraction to extract the Arg1 and Arg2 spans
 8: else // the relative position of Arg1 is PS
 9:     Label the sentence containing $C$ as Arg2
10:     Identify and label the Arg1 sentence from all previous sentences of Arg2
11: end if
```

Figure 4.6: Pseudocode for the argument labeler, which corresponds to Line 6 in Figure 4.1.

4.2.2.1 Argument Position Classifier

Prasad et al. (2008) described the demographic breakdown of the positions of Arg1 in their study of the PDTB annotations. They showed that Arg1 can be located within the same sentence as the connective (SS), in some previous sentence of the connective (PS), or in some sentence following the sentence containing the connective (FS). When Arg1 is located in some previous sentence, it can either be in the immediately previous sentence of the connective (IPS), or in some non-adjacent previous sentence of the connective (NAPS). Example 4.2 is a relation where the arguments and connective appearing in the same sentence, while Example 4.3 shows a case in which Arg2 immediately follows Arg1. The distribution from their paper shows that 60.9% of the Explicit relations are SS, 39.1% are PS, and less than 0.1% are FS (only 8 instances in the whole PDTB corpus).

Motivated by this observation, we design an argument position classifier to identify the relative position of Arg1 as SS or PS. We ignore FS since there are too few training instances. We notice that the connective string itself is a very good feature. For example, when the connective token is “But” (*i.e.*, “but” with its first letter capitalized, as in Example 4.3), it is a comparison with the previous sentence and thus Arg1 is likely in PS; whereas when the connective token is lowercase (*e.g.*, “and” in Example 4.2), Arg1 is likely the clause at the left hand side of the connective and thus it is in SS. Furthermore, some connectives always take a particular position. For example, “when” always indicates an SS case, whereas “additionally” always indicates PS.

Aside from the connective string, we also use the contextual feature classes in the classifier for the connective C with its first and second previous words $prev_1$ and $prev_2$, as shown here ($prev_1$ is the word immediately adjacent to C and $prev_2$ the word before $prev_1$):

- C string
- position of C in the sentence: start, middle, or end
- C POS
- $prev_1$
- $prev_1$ POS
- $prev_1$ + C
- $prev_1$ POS + C POS
- $prev_2$
- $prev_2$ POS
- $prev_2$ + C
- $prev_2$ POS + C POS

Here $prev_1$ is the same as $prev$ in the feature list for the connective classifier. We use subscripted numbers to explicitly represent their positions to the connective. After the relative position of Arg1 is identified, the result is propagated to the argument extractor, which employs different strategies to extract the Arg1 and Arg2 spans depending on whether the result is SS or PS.

4.2.2.2 Argument Extractor

When the relative position of Arg1 is classified as PS from the previous stage, Arg1 is located in one of the previous sentences of the connective. A majority classifier labels the immediately previous sentence as Arg1, which already gives an F_1 of 76.90% under the gold standard setting on the entire PDTB. Since the focus of this work is not on identifying the Arg1 sentences for the PS case but on extracting the argument spans for the SS case, we employ the majority classifier as our classifier for the PS case. In this work we make the assumption that for the PS case, Arg1 and Arg2 appear in full sentences, while in the PDTB, Arg1 or Arg2 may be shorter or longer than a full sentence. Wellner and Pustejovsky (2007) made a stronger assumption that the argument can be represented by the head verb of the sentence. Next we describe our approach to extract the arguments for the SS case in details.

When Arg1 is classified as in the same sentence (SS), this means that Arg1, Arg2 and the connective itself are in the same sentence. This can be further divided into four situations depending on the overlapping and positioning of the two arguments in the sentence:

1. Arg1 precedes Arg2,
2. Arg2 precedes Arg1,
3. Arg2 is embedded within Arg1, and
4. Arg1 is embedded within Arg2.

These four situations are illustrated by Example 4.2, 4.4, 4.5, and 4.6, respectively. One possible approach is to split the sentence into clauses before deciding which clause is Arg1 or Arg2. The problem with this approach is that it is not able to recognize the last two cases, where one argument divides the other into two parts. Another challenge is to exclude the text spans that are not in the relation, such as the span "It's the ... American" in Example 4.5.

(4.4) Although **preliminary findings were reported more than a year ago,** *the latest results appear in today's New England Journal of Medicine, a forum likely to bring new attention to the problem.*

(Comparison.Contrast - wsj_0003)

(4.5) It's the petulant complaint of an impudent American *whom Sony hosted for a year* while **he was on a Luce Fellowship in Tokyo** – *to the regret of both parties.*

(Temporal.Synchrony - wsj_0037)

(4.6) **The prime minister,** *whose hair is thinning and gray and whose face has a perpetual pallor,* nonetheless **continues to display an energy, a precision of thought and a willingness to say publicly what most other Asian leaders dare say only privately.**

(Comparison.Concession.Contra-expectation - wsj_0296)

Dinesh et al. (2005) showed that Arg1 and Arg2 in the same sentence for subordinating connectives are always syntactically related as shown in Figure 4.7(a), where Arg1 and Arg2 nodes are the lowest nodes that cover the respective spans. They demonstrated that a rule-based algorithm is capable of extracting Arg1 and Arg2 in such cases for subordinating connectives. By using tree subtraction, the third case mentioned above can be easily recognized and the text spans that are not in the relation can also be excluded. In Figure 4.7(a), span 3 is labeled as Arg2 that divides Arg1 into two non-continuous spans 2 and 4, and the out-of-relation spans (Spans 1 and 5) are also excluded.

However, dealing with only the subordinating connectives is not enough, because they only occupy 40.93% of all SS cases; the percentages of coordinating connectives and discourse adverbials for SS cases occupy up to 37.50% and 21.57%, respectively, with respect to the whole PDTB. We observe that coordinating connectives ("and", "or", "but", etc.) usually constrain Arg1 and Arg2 to be syntactically related in one of two ways as shown in Figure 4.7(b)-(c), where CC is the connective POS. Discourse adverbials do

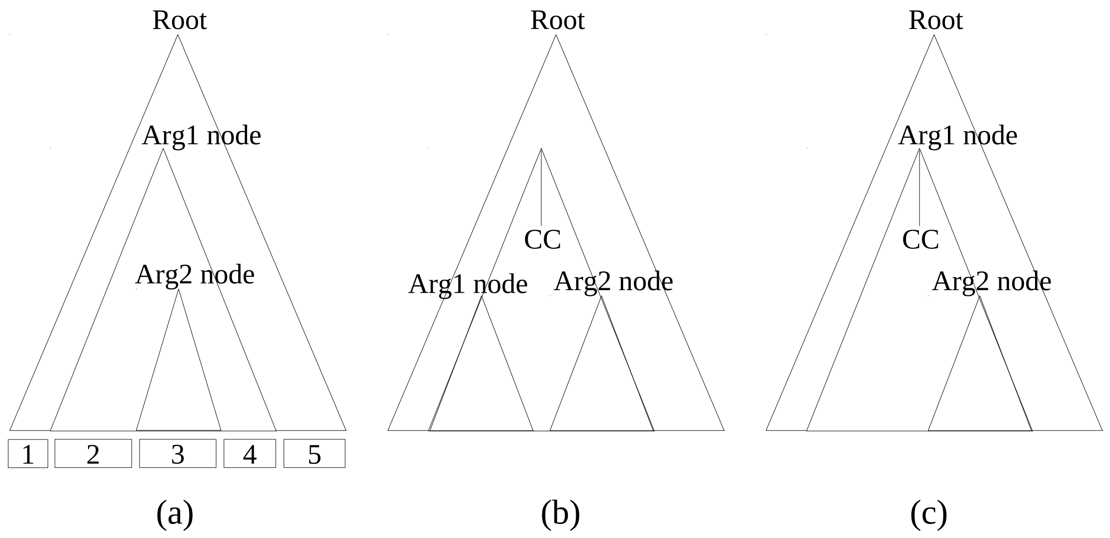


Figure 4.7: Syntactic relations of Arg1 and Arg2 subtree nodes in the parse tree. (a): Arg2 contains span 3 that divides Arg1 into two spans 2 and 4. (b)-(c): two syntactic relations of Arg1 and Arg2 for coordinating connectives.

not demonstrate such syntactic constraints as strongly as subordinating and coordinating connectives do, but their Arg1 and Arg2 are also syntactically bound to some extent. For example, Figure 4.8 shows the syntactic relation of Arg1 and Arg2 nodes for the discourse adverbial "still" in Example 4.7. Furthermore, the rule-based algorithm in (Dinesh et al., 2005) does not recognize the fourth case where Arg1 span is embedded within Arg2. The ratio of occurrences of the third case to the fourth case in the entire PDTB corpus is approximately 1:1. Thus we believe that the fourth case also needs to be taken care of.

(4.7) The ultimate result came in Hymowitz v. Lilly, where the highest New York court expanded the market-share approach for the first time $\boxed{\text{to say}}$ *that drug makers that could prove Mindy Hymowitz's mother didn't use their pill* **must** <u>still</u> **pay their share of any damages**.

(Comparison.Concession.Contra-expectation - wsj_0130)

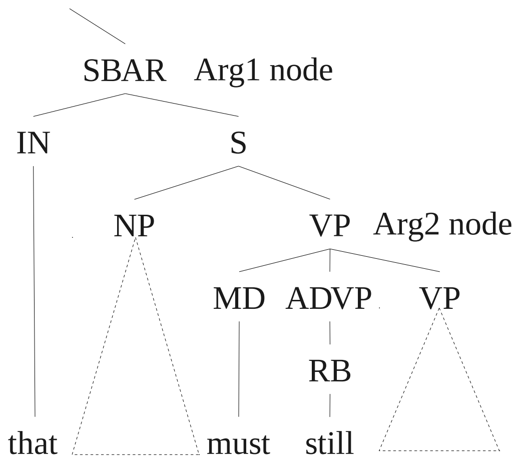


Figure 4.8: Part of the parse tree for Example 4.7 with Arg1 and Arg2 nodes labeled.

Given these observations, we design an automatic argument node identifier to first identify the Arg1 and Arg2 subtree nodes within the sentence parse tree for all subordinating connectives, coordinating connectives and discourse adverbials, then apply tree subtraction to extract the Arg1 and Arg2 spans. The argument node identifier labels each internal node (except POS nodes, which are parent nodes of leaf nodes) of the tree with three probabilities: functioning as Arg1-node, Arg2-node, and None. The internal node with the highest Arg1-node probability is chosen as the Arg1 node, and likewise for the Arg2 node. If the Arg1 node is the ancestor of the Arg2 node, the subtree under the Arg2 node is then subtracted from the Arg1 subtree to obtain the Arg1 span, and respectively when the Arg2 node is the ancestor of the Arg1 node. Motivated by the syntactic properties observed, we propose the following feature classes for the node N under consideration with regard to the connective C:

– C string
– C's syntactic category: subordinating, coordinating, or discourse adverbial
– number of left siblings of C
– number of right siblings of C

- the path P of C's parent $\rightarrow N$
- the path P and whether the number of C's left sibling is greater than one
- the relative position of N to C: left, middle, or right

The syntactic category (subordinating, coordinating, or discourse adverbial) of the connective is a useful clue of the locations of the Arg1 and Arg2 nodes. We obtain the corresponding categories for the connectives from the list provided in (Knott, 1996). The path from the C's parent node to the node N under consideration is also an informative feature, as it reflects how N is related to C syntactically. The following are two paths for the actual Arg2 node and the MD node in Figure 4.8:

$$\text{RB} \uparrow \text{ADVP} \uparrow \text{VP}$$
$$\text{RB} \uparrow \text{ADVP} \uparrow \text{VP} \downarrow \text{MD}$$

The relative position of N to C is medial when N is on the path of C to root. The position can also be left or right depending on whether it is located on the left- or right-hand side of this path. This feature also models the syntactic relation of C and N to some extent. To label each internal node with these three probabilities, we adopt a maximum entropy classifier, as it is capable of estimating class probabilities.

To illustrate how the argument position classifier and argument extractor work together to label the arguments, let us look at Example 4.5. After examining the features for the connective "while", the argument position classifier will decide that Arg1 and Arg2 are in the same sentence (SS), and pass it to the argument extractor. Since the class is SS, the argument extractor fires the argument node identifier to locate the internal nodes that covers the Arg1 span (*i.e.*, "whom Sony ... both parties") and that covers Arg2 and the connective (*i.e.*, "while he ... in Tokyo"). Finally, tree subtraction is applied to clean up and remove the Arg2 span from the Arg1 span. The results are: connective = "while", Arg1 = "whom Sony hosted for a year . . . to the regret of both parties", and Arg2 = "he was on a Luce Fellowship in Tokyo",

4.2.3 Explicit Relation Classifier

After identifying a discourse connective and its two arguments, the next step is to decide what Explicit relation it conveys. It is important to disambiguate the relation type of the connective, as the same connective may carry different semantics under different contexts. For example, the connective "and" has relation types of Expansion.Conjunction and Expansion.List in Example 4.2 and 4.8, respectively.

(4.8) *Microsoft added 2 1/8 to 81 3/4* <u>and</u> **Oracle Systems rose 1 1/2 to 23 1/4**.

(Expansion.List - wsj_0327)

Prasad et al. (2008) reported a human agreement of 94% on Level-1 classes and 84% on Level-2 types for Explicit relations over the whole PDTB corpus. The connective itself is a very good feature, as only a few connectives are ambiguous as pointed out in (Pitler et al., 2008), and the distribution of most ambiguous connectives are highly skewed toward certain types. We train an Explicit relation classifier using three types of feature classes of the connective C and its previous word $prev$:

- C string
- C's POS
- C + $prev$

We follow Chapter 3 to train and test on the Level-2 types. Different from Chapter 3, we do not remove the five relation types and use all 16 Level-2 types in the Explicit relation classifier. The reason for this is that we have a larger number of training instances for these five relations compared to that for the training data for Non-Explicit relation classifier: 1135 instances for Condition and 98 for the other four.

4.2.4 Non-Explicit Relation Classifier

The PDTB also provides annotation for Implicit relations, AltLex relations, entity transition (EntRel), and otherwise no relation (NoRel), which are lumped together as Non-

Explicit relations. The Non-Explicit relations are annotated for all adjacent sentence pairs within paragraphs. If there is already an Explicit relation from the previous step between two adjacent sentences, they are exempt from further examination.

This Non-Explicit relation classifier is adapted and re-trained from the classifier in the last chapter. Similarly, we adapt the Level-2 types for the Implicit and AltLex relations. As pointed out in Chapter 3, there are too few training instances for Condition, Pragmatic Condition, Pragmatic Contrast, Pragmatic Concession, and Exception relations (in total, only 9 training instances). Thus these five types are removed, resulting in 11 Level-2 types. Our Non-Explicit relation classifier assigns candidate sentence pairs to one of 13 types (11 Level-2 types plus EntRel and NoRel). We apply the three feature classes from Chapter 3:

– constituent parse features
– dependency parse features
– word-pair features

Besides including AltLex, EntRel, and NoRel, another difference between this classifier and the classifier developed in Chapter 3 is that no context features about the surrounding relations are used. The reason is that such features need gold standard relation types from the previous and next relations.

AltLex relations are very similar to their counterpart Explicit relations, except that they are *alternatively lexicalized* by some non-connective expressions, instead of by pre-defined discourse connectives. Such non-connective expressions are usually attached to the beginning of Arg2 (*e.g.*, such as "Another concern" in Arg2 of Example 2.3). To distinguish AltLex relations, we use three feature classes that represent the first three stemmed terms of Arg2. For the example above, the features that are turned on will be $term_1$=*another*, $term_2$=*concern*, and $term_3$=*the*. No other feature classes are designed to specifically handle both EntRel and NoRel relations. However, in the experimental section below, we will show that the current feature classes are capable of handling EntRel relations well.

4.2.5 Attribution Span Labeler

For each discourse relation (*i.e.*, Explicit, Implicit, or AltLex relation), the PDTB annotators labeled the attribution spans and annotated four dimensions for Arg1, Arg2, and the relation: their sources, types, scopal polarities, and determinacy. For the current parser, we develop a component to label the attribution spans, without labeling the four attribution dimensions and direction (Arg1, Arg2, or the relation) it is associated with. Our focus is on the attribution location and span; recognizing these additional four dimensions and attribution direction fall beyond our scope of study. We label attribution spans that appear within discourse relations.

In our work, we make the assumption that attribution spans always appear in clauses, and we only examine clauses within the identified relations from the previous steps. Note that in the PDTB not all attribution spans appear in exact clauses and some appear outside the texts of discourse relations. 7 out of 559 of the attribution spans appear in less than a clause (*i.e.*, the length of the attribution span is shorter than a clause), based on a calculation using our lightweight clause splitter. Attribution span and argument spans usually do not overlap in the PDTB, but attribution spans always appear in the same sentence of an argument span. 3.09% (60 of 1943) of all relations in the test data contain attributions appearing outside the argument sentences. Our attribution span labeler consists of the following two steps:

1. Splitting the text into clauses, and
2. Deciding which clauses are attribution spans.

In the first step we employ a lightweight clause splitter that we have developed which uses syntactically motivated approach similar to (Skadhauge and Hardt, 2005). This clause splitter makes use of punctuation symbols and syntactic structures of SBAR complements.

The attribution span labeler then classifies each clause into attribution span (attr-span) or non-attribution span (non-attr-span). Words (especially verbs) in the clause are a very good clue in deciding whether the clause is an attribution. For example, the verbs "declared" and "say" in Example 2.2 and 4.7. Another useful clue is by looking at the end of the previous clause and the start of the next one. In Example 2.2, which is partially replicated in the following, the previous clause of the current clause ends with a comma and a closing quotation mark, and the next clause starts with an opening quotation mark, which suggest that the previous and next clauses are in the same speech act and the current clause is probably the attribution of the speech.

... averages," $\boxed{\text{declared San Francisco batting coach Dusty Baker after game two.}}$ "I'd ...

Based on the observations, we propose the following feature classes extracted from the current, previous, and next clauses ($curr$, $prev$, and $next$):

- lowercased and lemmatized verbs in $curr$
- the first and last terms of $curr$
- the last term of $prev$
- the first term of $next$
- the last term of $prev$ + the first term of $curr$
- the last term of $curr$ + the first term of $next$
- the position of $curr$ in the sentence: start, middle, end, or whole sentence
- production rules extracted from $curr$

Appendix A.2 shows features extracted for the above example. Some clauses that belong to single attribution spans may be incorrectly split into more than one clause by the clause splitter. For example, "said C. Bruce Johnstone, who runs Fidelity Investments' $5 billion Equity-Income Fund." is annotated as a single attribution span in the PDTB, as shown in the following. It is (mistakenly) split into two clauses "said C. Bruce Johnstone," and “who runs Fidelity Investments’ $5 billion Equity-Income Fund.” by our clause splitter, and then both classified as attr-span. To correct such mistakes, adjacent attribution clauses within a sentence are combined to form a single attribution span after classification:

around,” $\boxed{\text{said C. Bruce Johnstone, who runs Fidelity Investments' \$5 billion Equity-Income Fund.}}$ “This

4.3 Evaluation

Identical to our setup in Chapter 3, we use Section 02–21 in the PDTB for training, Section 22 for development, and Section 23 for testing. All classifiers are trained with the OpenNLP maximum entropy package[1] without smoothing and with 100 iterations which is the default settings. However, further experiments can be done to evaluate how these two parameters affect the performance.

For each component, the experiments are carried out when there is no error propagated from the previous components (*i.e.*, using gold standard annotation for the previous components), and when there is error propagation. As the PDTB was annotated on top of the PTB, we can either directly use the gold standard parse trees and sentence boundaries from the PTB files, or we can apply an automatic parser and sentence splitter. The experiments are carried out under three settings for each component:

1. GS + no EP: using gold standard (GS) parses and sentence boundaries without error propagation (EP)
2. GS + EP: using GS with EP
3. Auto + EP: using both automatic parsing and sentence splitting (Auto) with EP.

Thus, GS + no EP corresponds to a clean, per component evaluation, whereas the Auto + EP setting assesses end-to-end fully automated performance (as would be

[1]`http://maxent.sourceforge.net/`

expected on new, unseen text input). We use the NIST's text segmenter[2] to insert sentence boundaries and the Charniak parser[3] to parse the sentences in the Auto setting. As there are no gold standard dependency parses for the PTB files, we employ the Stanford dependency parser[4] in both GS and Auto settings.

4.3.1 Results for Connective Classifier

On the connective classifier task, Pitler and Nenkova (2009) (P&N) reported an accuracy of 96.26% and F_1 of 94.19% with a 10-fold cross validation on Section 02–22. To compare with P&N, we also run a 10-fold CV on Section 02–22 using their features and obtain replicated accuracy of 96.09% and replicated F_1 of 93.57%. Adding in our lexico-syntactic and path features, the performance is increased to 97.25% accuracy and 95.36% F_1, yielding improvements of 0.99% and 1.17% over the reported results and 1.16% and 1.79% over the replicated results. A paired t-test shows that the improvements over our replication of P&N's results are significant with $p < 0.001$[5].

In Table 4.1, we report results from the connective classifiers trained on Section 02–21 and tested on Section 23. As there is no error propagated into the connective classifier as it is the first component, we report results for just the GS and Auto settings. The second and third columns show the accuracy and F_1 using the features of P&N, whereas the last two columns show the results when we add in the lexico-syntactic and path features (+new). Introducing the new features significantly (all with $p < 0.001$) increases the accuracy and F_1 by 2.04% and 3.01% under the GS setting, and 1.81% and 2.62% under Auto. This confirms the usefulness of integrating the contextual and syntactic information. As the connective classifier is the first component in the pipeline, good performance is crucial to mitigate the effect of cascaded errors downstream.

[2]`http://duc.nist.gov/duc2004/software/duc2003.breakSent.tar.gz`

[3]`ftp://ftp.cs.brown.edu/pub/nlparser/`

[4]`http://nlp.stanford.edu/software/lex-parser.shtml`

[5]It is not possible to conduct paired t-test on the reported results for P&N as we do not have the predictions.

	P&N		+new	
	Acc.	F_1	Acc.	F_1
GS	95.30	92.75	97.34	95.76
Auto	94.21	91.00	96.02	93.62

Table 4.1: Results for the connective classifier. No EP as this is the first component in the pipeline.

When we compare the performance under GS and Auto settings for our classifier, there are significant drops (with $p < 0.001$) of 1.32% and 2.14% for accuracy and F_1, respectively. This is because the lexico-syntactic and path features we introduced heavily depend on the syntactic structures of the sentences. When we use an automatic parser instead of gold standard parses, it dramatically affects the performance of these new features. When we look into the incorrectly labeled connectives, we find that the connective with the highest number of incorrect labels is “and” (8 false negatives and 4 false positives for the GS setting), which is not surprising, as “and” is always regarded as an ambiguous connective.

4.3.2 Results for Argument Labeler

We next perform evaluation on the argument position classifier and report the results in the contingency tables in Table 4.2 for the three settings. The last row of the tables show the numbers of errors propagated from the previous component. This does not apply to the setting of GS + no EP, as no error propagation is performed. The micro precision, recall and F_1, as well as the per class F_1, are calculated from Table 4.2 and reported in Table 4.3. The GS + no EP setting gives a high F_1 of 97.94%, which drops 3.59% and another 2.26% when error propagation and full automation are added in. The per class F_1 shows the performance degradation is mostly due to the SS (Arg 1 and 2 in the Same Sentence) class: the drops for SS are 5.36% and 3.35%, compared to 1.07% and 0.68%

Ground truth	Predictions					
	GS + no EP		GS + EP		Auto + EP	
	PS	SS	PS	SS	PS	SS
PS	369	8	364	5	360	5
SS	11	535	8	504	9	484
Error propagated	-	-	6	30	6	46

Table 4.2: Contingency tables for the argument position classifier for the three settings. The last row shows the numbers of errors propagated from the previous component, which does not apply to the first setting of GS + no EP.

				Per class F_1	
	Prec.	Recall	F_1	SS	PS
GS + no EP	97.94	97.94	97.94	98.26	97.49
GS + EP	94.66	94.04	94.35	92.90	96.42
Auto + EP	92.75	91.44	92.09	89.55	95.74

Table 4.3: Results for the argument position classifier.

for PS. When we look into the contingency table for the GS + EP setting, we notice that out of the 36 false positives propagated from the connective classifier, 30 of them are classified as SS; for the Auto + EP setting there are 46 out of 52 classified as SS. This shows that the difference in the performance drops for SS and PS is largely due to the error propagation from the connective classifier, and not the classes themselves.

We next evaluate the performance of the argument extractor. Table 4.4 illustrates the results of identifying the Arg1 and Arg2 subtree nodes for the SS case for the three connective categories. The last column shows the Arg1&Arg2 F_1 which requires both Arg1 and Arg2 nodes to be identified correctly. We only show the results for the GS + no EP setting. As expected, Arg1 and Arg2 nodes for subordinating connectives are the easiest ones to identify and give a high Arg2 F_1 of 97.93% and a Arg1&Arg2 level F_1 of 86.98%. We note that the Arg1 F_1 and Arg2 F_1 for coordinating connectives are the same, which is strange, as we expect Arg2 nodes to be easier to classify as Arg2 and the connective are syntactically associated. Error analysis shows that Arg2 spans for coordinating con-

	Arg1 F_1	Arg2 F_1	Arg1&Arg2 F_1
Subordinating	88.46	97.93	86.98
Coordinating	90.34	90.34	82.39
Discourse adverbial	46.88	62.50	37.50
All	86.63	93.41	82.60

Table 4.4: Results for identifying the Arg1 and Arg2 subtree nodes for the SS case under the GS + no EP setting for the three categories.

nectives tend to include extra text that cause the Arg2 nodes to move lower down in the parse tree. For example, "... and Mr. Simpson said he resigned in 1988" contains the extra span "Mr. Simpson said" which causes the Arg2 node (which covers "he resigned in 1988") moving two levels down the tree. The system erroneously labels "Mr. Simpson ... 1988" as Arg2. One solution to this is to feedback from the attribution span label to the argument extractor, so that we can remove this extra text of attribution span.

Also as we discussed in Section 4.2.2.2, discourse adverbials are difficult to identify as their Arg1 and Arg2 nodes are not strongly bound in the parse trees. However, as they do not take up a large percentage in the test data (only 5.38% of the test data is for identifying Arg1 and Arg2 nodes for discourse adverbials under the GS + no EP setting), they do not lead to a large degradation as shown in the last row of the overall performance of the three categories.

Miltsakaki et al. (2004) reported human agreements on both exact and partial matches to be 90.2% and 94.5%, respectively. They found that most of the disagreements for exact match come from partial overlaps which do not show significant semantic difference. We follow such work and report both exact and partial matches. When checking exact match, we require two spans to match identically, excluding any leading and ending punctuation symbols. A partial match is credited if there is any overlap between the verbs and nouns of the two spans. The results for the overall performance for both SS and PS cases are shown in Table 4.5. The GS + no EP setting gives a satisfactory

		Arg1 F_1	Arg2 F_1	Arg1&Arg2 F_1
Partial	GS + no EP	86.67	99.13	86.24
	GS + EP	83.62	94.98	83.52
	Auto + EP	81.72	92.64	80.96
Exact	GS + no EP	59.15	82.23	53.85
	GS + EP	57.64	79.80	52.29
	Auto + EP	47.68	70.27	40.37

Table 4.5: Overall results for the argument extractor.

F_1 of 86.24% for partial matching on Arg1&Arg2 F_1. On the other hand, the results for exact matching are much lower than the human agreement. We observe that most misses are due to small portions of text being deleted from or added to the spans by the annotators to follow the *minimality principle*, which is in accordance with the explanation in (Miltsakaki et al., 2004). The minimality principle states that the annotation should define the argument to only include the minimal span of text that is sufficient for the interpretation of the relation. This ability requires deep semantic analysis and poses difficulties for machines to follow.

4.3.3 Results for Explicit Classifier

Following the pipeline, we then evaluate the Explicit relation classifier, with its performance shown in Table 4.6. Recall that human agreement on Level-2 types is 84.00% and a baseline classifier that uses only the connectives as features yields an F_1 of 86.00% under the GS + no EP setting on Section 23. Adding our new features improves F_1 to 86.77%. With full automation and error propagation, we obtain an F_1 of 80.61%. Pitler and Nenkova (2009) show that using the same syntactic features as their connective classifier is able to improve the classifier on a 10-fold cross validation on Section 02-22. We have trained the classifier on Section 02-21 using their features and tested on Section 23, but it actually performs worse than the baseline. Therefore we do not include their features in

	Precision	Recall	F_1
GS + no EP	86.77	86.77	86.77
GS + EP	83.19	82.65	82.92
Auto + EP	81.19	80.04	80.61

Table 4.6: Results for the Explicit relation classifier.

	Precision	Recall	F_1	Baseline F_1
GS + no EP	39.63	39.63	39.63	21.34
GS + EP	26.21	27.63	26.90	20.30
Auto + EP	24.54	26.45	25.46	19.31

Table 4.7: Results for the Non-Explicit relation classifier.

the Explicit relation classifier.

4.3.4 Results for Non-Explicit Classifier

For the Non-Explicit relation classifier, a majority class baseline that labels all instances as EntRel yields an F_1 in the low 20s, as shown in the last column of Table 4.7. The percentage of EntRel is slightly higher than the most frequent Implicit Cause relations (21.34% vs. 21.24% in the Implicit relations). A single component evaluation (GS + no EP) shows a micro F_1 of 39.63%. This is slightly lower than the result of 40.2% reported in Chapter 3. The reason is that in the previous chapter, we trained and tested only on the 11 Level-2 types; in contrast, here we added two additional classes – EntRel and NoRel. Although the F_1 scores for the GS + EP and Auto + EP settings are unsatisfactory, they still significantly outperform the majority class baseline by about 6%. This performance is in line with the difficulties of classifying Implicit relations as discussed in detail in the previous chapter.

Table 4.8 shows the contingency table for 11 Level-2 types, EntRel, and NoRel under the GS + no EP setting. Unlike in the previous chapter, we added EntRel and

Ground truth	Predictions												
	Asynchronous	Synchrony	Cause	Pragmatic Cause	Contrast	Concession	Conjunction	Instantiation	Restatement	Alternative	List	EntRel	NoRel
Asynchronous	0	0	7	0	0	0	0	0	0	0	0	7.5	0
Synchrony	0	0	0	0	0	0	2	0	0	0	0	1	0
Cause	1	0	152	0	2	0	15.5	0	15	0	0	29.5	0
Pragmatic Cause	0	0	4	0	0	0	1	0	0	0	0	0	0
Contrast	1	0	46.5	0	16	1	26	1	7	0	0	28	0
Concession	0	0	3	0	0	0	2	0	0	0	0	1	0
Conjunction	0	0	35.5	0	5	0	33	1	9	0	0	40.5	0
Instantiation	1	0	7	0	0	0	3.5	22	12	0	0	23.5	0
Restatement	1	0	81	0	3	0	26	6	26	0	0	47	0
Alternative	0	0	12	0	0	0	1	0	0	0	0	2	0
List	0	0	5	0	2	0	9	1	2	0	4	5	0
EntRel	0	0	36	0	4	0	16	4	7	0	0	150	0
NoRel	0	0	0	0	0	0	0	0	0	0	0	4	0

Table 4.8: Contingency table for Non-Explicit relation classification for 11 Level-2 relation types, EntRel, and NoRel under the GS + no EP setting. As some instances were annotated with two types, the instance is considered correct if one of these two is predicted. This is why we can have .5 in the table.

Relation Type	Precision	Recall	F$_1$	% in test set	% in training set
Asynchronous	–	–	–	1.47	3.53
Synchrony	–	–	–	0.49	1.26
Cause	39.07	70.37	50.25	21.24	20.02
Pragmatic Cause	–	–	–	0.49	0.38
Contrast	50.00	12.50	20.00	12.59	9.26
Concession	–	–	–	0.59	1.10
Conjunction	24.44	26.40	25.38	12.29	16.84
Instantiation	62.86	30.14	40.74	7.18	6.64
Restatement	33.33	13.61	19.33	18.78	14.42
Alternative	–	–	–	1.47	0.87
List	100.00	12.90	22.86	3.05	1.89
EntRel	44.25	69.12	53.96	21.34	22.68
NoRel	–	–	–	0.39	1.12
All (Micro Avg.)	39.63	39.63	39.63		

Table 4.9: Precision, recall, and F_1 for 11 Level-2 relation types, EntRel, and NoRel under the GS + no EP setting. "–" indicates 0.00.

NoRel in the classification, and thus we wish to discuss their classification performance here. From the table, we can see that as both Cause and EntRel are the majority classes (the percentage of EntRel is slightly higher than that of Cause), most errors for the other classes are due to being mis-classified into these two types. Table 4.9 shows the precision, recall, and F_1 derived from the contingency table. We see that EntRel and NoRel have counts of 217 and 4 instances in the test set, and their percentages in the training data are 22.68% and 1.12%. The performance figures for EntRel are 44.25%, 69.12%, and 53.96% for precision, recall, and F_1, respectively. EntRel gives the best F_1 score compared to the other relations. This illustrates that although we use the same features from Chapter 3 and do not propose new features specifically for EntRel and NoRel, these three feature classes are able to capture EntRel relation. The classifier is not able to identify the NoRel cases. However, as there are only 4 NoRel instances in the test case, it does not affect the overall performance much. The performance figures for the other 11 Level-2 relations are similar to that reported and discussed in Table 3.6 and Section 3.4.1.

4.3.5 Results for Attribution Span Labeler

The final component, the attribution span labeler, is evaluated under both partial and exact match, in accordance with the argument extractor. As we only examine clauses that appear inside identified relations, the missing clauses are also included in the performance calculation. From Table 4.10, we see that the GS + no EP setting achieves F_1 scores of 79.68% and 65.95% for partial and exact match, respectively. The gap between partial and exact match is due to our assumption that all attribution spans appear in clauses and the fact that not all attribution spans in the PDTB are clauses. In the example below, "reportedly" is an attribution span but it is not in a clause form. The system was not able to locate the exact span for this example and label it as an attribution span.[6]

. . . , " Olivetti $\boxed{\text{reportedly}}$ began shipping these tools in 1984.

When error propagation is introduced, the degradation of F_1 is largely due to the drop in precision. This is not surprising as at this point, the test data contains a number of false positives propagated from the previous components. This has effect on the precision calculation but not recall (the recall scores do not change). When full automation is further added, the degradation is largely due to the drop in recall. This is because the automatic parser introduces noise that causes errors in the clause splitting step. We expect a better parser or clause splitter will improve the performance.

4.3.6 Overall Performance

To evaluate the whole pipeline, we look at the Explicit and Non-Explicit relations that are correctly identified. We define a relation as correct if its relation type is classified correctly,

[6]As we do not have the gold standard annotation of clauses in the PDTB, we are not able to calculate the percentage of errors that is due to the non-clausal attribution spans.

		Precision	Recall	F_1
Partial	GS + no EP	79.40	79.96	79.68
	GS + EP	65.93	79.96	72.27
	Auto + EP	64.40	51.68	57.34
Exact	GS + no EP	65.72	66.19	65.95
	GS + EP	54.57	66.19	59.82
	Auto + EP	47.83	38.39	42.59

Table 4.10: Results for the attribution span labeler.

		F_1
Partial	GS + EP	46.80
	Auto + EP	38.18
Exact	GS + EP	33.00
	Auto + EP	20.64

Table 4.11: Overall performance for both Explicit and Non-Explicit relations. GS + no EP setting is not included, as this is not a component-wise evaluation.

and both its Arg1 and Arg2 are partially or exactly matched. The overall performance is shown in Table 4.11. Under partial matching, the GS + EP setting gives an overall system F_1 of 46.80%, while under exact matching, it achieves an F_1 of 33.00%. Auto + EP gives 38.18% F_1 for partial match and 20.64% F_1 for exact match. A large portion of the misses come from the Non-Explicit relations, as they are more difficult to classify in comparison to the Explicit relations. The GS + EP results are close to the system F_1 of 44.3% of an RST parser reported in (duVerle and Prendinger, 2009).

4.3.7 Mapping Results to Level-1 Relations

We map Level-2 types to their corresponding Level-1 types (Comparison, Contingency, Expansion, and Temporal) and calculate the performance of the Explicit classifier on the four Level-1 types and the Non-Explicit relation classifier on the four Level-1 types, EntRel, and NoRel. The results are shown in Table 4.12 and 4.13.

	Precision	Recall	F_1
GS + no EP	95.23	95.23	95.23
GS + EP	91.59	91.00	92.29
Auto + EP	89.88	88.61	89.24

Table 4.12: Results for the Explicit relation classifier on the four Level-1 types.

For the Explicit classifier, 16 Level-2 types are mapped to the four Level-1 types, which gives rise to a 4-way classifier. Pitler and Nenkova (2009) reported 94.15% F_1 of the corresponding four Level-1 types on 10-fold cross validation over Section 2–22 gold standard data. Compared to their system, our Explicit classifier achieves a F_1 of 95.23% under the GS + no EP setting in Section 23. The F_1 drops 3% when error propagation is added and drops another 3% when we use automatic sentence splitting and parsing.

Our Non-Explicit classifier from the previous section is capable of handling the 11 Level-2 types for both Implicit and AltLex relations (recall that our Non-Explicit classifier also handles AltLex relations with new features to capture the non-connective expressions), as well as EntRel and NoRel. We map these 11 Level-2 types to the corresponding four Level-1 types. The Non-Explicit classifier gives 47.30% F_1 when it runs as a 6-way classifier (*i.e.*, four Level-1 types plus EntRel and NoRel) under the GS + no EP setting. Table 4.13 shows that the performance degradation is mostly due to error propagation: $\sim$15% drop when EP is introduced and only $\sim$1% drop when Auto is further introduced. Pitler et al. (2009) reported a F_1 of 44.58% for their 6-way classifier. However, the results are not strictly comparable, as they did not include AltLex relations and they trained and tested on different sections. Note that as both (Pitler and Nenkova, 2009) and (Pitler et al., 2009) tested their systems on gold standard data, we also use gold standard and report results on the GS + no EP setting.

Note that the Explicit and Non-Explicit classifiers are trained and tested on Level-2 types, and the results are mapped to the corresponding Level-1 types. The results are

	Precision	Recall	F_1
GS + no EP	47.30	47.30	47.30
GS + EP	31.81	33.53	32.65
Auto + EP	30.29	32.65	31.42

Table 4.13: Results for the Non-Explicit relation classifier on the four Level-1 types, EntRel, and NoRel.

unlikely to be the same if we train and test the classifiers directly on Level-1 types.

4.4 Discussion and Future Work

The overall performance of the whole pipeline shows that the Non-Explicit relation classifier generates a large portion of the errors, which suggests that there is still large room for improvement on the classifier. In Chapter 3 on classifying Implicit relations, we have shown that the difficulties of this task are mostly attributed to four types of challenges: the ambiguity among the relation types, the need for using inference and a knowledge base, the analysis of the contextual information in understanding the arguments, and the access to world knowledge. We plan to tackle some of these same challenges for the Non-Explicit relations in our future work. We used the three feature classes from the Implicit relation classifier in Chapter 3 in our Non-Explicit relation classifier, where we also included EntRel and NoRel. As no features are proposed specifically for EntRel and NoRel, we plan to investigate other features that can differentiate EntRel and NoRel from other Implicit relations.

In our Explicit relation classifier, although the tuple (C, Arg1, Arg2) is passed into the classifier, the current approach does not make use of information from Arg1 and Arg2. The features that we applied only involve the connective string, its POS, and its previous word. One future work is to extract informative features from the two arguments for the Explicit relation classifier. For AltLex relations, Prasad et al. (2010b) showed that the set of AltLex markers is open-ended with unconstrained syntactic possibilities. As AltLex are alternatively lexicalized by non-predefined and non-connective expressions, an identifier that solely attacks these issues may also improve the overall performance of the Non-Explicit relation classifier. As we know, Explicit relations are signaled by the explicit discourse connectives, while AltLex relations by non-connective expressions. When we move from the one-million word PTB to other domains or corpora, it is expected that we will encounter new discourse connectives and non-discourse expressions that do not appear in PTB. One possible future work is to look at the problem from a domain adaptation point of view.

The PDTB provides a three level hierarchy of relation tags for its annotation. In this work we focused on 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. As we discussed in Section 2.1, Level-3 provides the orientation of the relation. For example, Contingency.Cause.Reason and Contingency.Cause.Result show that the causal relation is pointing from Arg2 to Arg1 and from Arg1 to Arg2, respectively. Similar to Level-1 and -2, Level-3 relations are also unequally distributed. Our current work did not provide features to identify the orientation of the relations. We plan to extend our current system with capabilities to handle relation orientation for Level-3 types.

The current approach also does not deal with identifying Arg1 from all previous sentences for the PS case. Although about 77% of Arg1s can be located in the immediately previous sentence of Arg2s in this case, it is important to take the rest into consideration to make this component complete. Furthermore, this task will not be easy, as there is no restriction of the distance between Arg1 and Arg2. Example 4.9 shows a situation where Arg1 is located four sentences away from Arg2. Our next step is to design a full PS identifier and integrate it into the current pipeline. One possibility is to follow (Prasad et al., 2010a) to use a set of filters and heuristics to locate the positions of Arg1 spans, which also takes into account whether the connectives are in paragraph-initial sentences.

(4.9) *GOODY PRODUCTS Inc. cut its quarterly dividend to five cents a share from 11.5 cents a share.* The reduced dividend is payable Jan. 2 to stock of record Dec. 15. The Kearny, N.J.-based maker of hair accessories and other cosmetic products said it cut the dividend due to its third-quarter loss of $992,000, or 15 cents a share. In the year-ago quarter, the company reported net income of $1.9 million, or 29 cents a share. **The company** <u>also</u> **adopted an anti-takeover plan.**

(Expansion.Conjunction - wsj_0068)

The PDTB also provides annotations for the four dimensions for the attributions (their sources, types, scopal polarities, and determinacy) as well as the direction of the attributions to indicate whether an attribution is pointing to Arg1, Arg2, or the relation as a whole. Currently our parser does not consider these data as output targets for classification nor as input features. In Example 4.10, the attribution "traders said" has scope over Arg1 of "However", while the second attribution "Disney ... said" is taken to have scope over the connective "However" and hence over the entire coherence relation.

(4.10) *But then it shot upward 7 1/2 as Goldman, Sachs & Co. stepped in and bought*, $\boxed{\text{traders said}}$. <u>However</u>, $\boxed{\text{Disney specialist Robert Fagenson said}}$: **"I would be surprised if Goldman represented 4% of the opening volume."**

(Comparison - wsj_2232)

Such information can also be incorporated into the parser, as this provides finer grained information on the opinions and the opinion holders, which are useful for the downstream subjectivity analysis. The current attribution span labeler only considers clauses within the relation, which may result in missing out clauses that are attribution spans of the relation but reside outside the relation. For instance, in Example 4.11, the attribution span "said David ... Sunday's go" for the relation "*I'm for* ... **lost yesterday**" resides outside the relation itself, thus it will not be examined by our system. One possible approach is to use a window of sentences to check previous and following sentences for attributions that are pointing to this relation.

(4.11) "*I'm for the Giants today*, **but only** <u>because</u> **they lost yesterday**.
I love 'em both. The only thing I'm rooting for is for the Series to go seven games,"
$\boxed{\text{said David Williams, a Sacramento septuagenarian, at the Coliseum}}$
$\boxed{\text{before Sunday's go}}$.

(Contingency.Cause.Reason - wsj_2202)

Wellner (2009) pointed out that verbs from the attribution spans are useful features in identifying the argument head words. In his work, Wellner checked whether the argument verb (as only argument verbs, not argument spans, are identified) is a potentially attribution-denoting verb. This suggests that we can use the downstream results from the attribution span labeler as input to the earlier argument labeling stage. In fact, we can enlarge this feedback loop, feeding all results from the end of the pipeline into the start, to construct a joint learning model (imagine an arrow being drawn from the attribution span labeler back to the connective classifier in Figure 4.2). We believe that discourse parsing is very useful in downstream applications, such as text summarization and question answering (QA). For example, a text summarization system may utilize the contrast and restatement relations to recognize updates and redundancy, whereas causal relations can be used in a QA system to answer *why*-questions. The attribution spans from the parser are also very useful for applications on opinion mining and subjectivity analysis to locate the opinion holders. In the next two chapters, we will show the applicability of this discourse parser in coherence modeling, text summarization, and argumentative zoning.

4.5 Conclusion

In this chapter, we have designed a parsing algorithm that performs discourse parsing in the PDTB representation, and implemented it into an end-to-end system in a fully data-driven approach. We have proposed automatic approaches to locate the relative positions of Arg1 and label the exact spans of the arguments when they appear in the same sentence. The performance of the connective classifier is also significantly improved from previous work. We have implemented a component to label the attribution spans for the relations. We evaluated the system both component-wise as well as in an end-to-end fashion with cascaded errors. We reported overall system F_1 scores for partial matching of 46.80% with gold standard parses and 38.18% with full automation.

Chapter 5

Evaluating Text Coherence Using Discourse Relations

In this chapter, we present a novel model to represent and assess the discourse coherence of text. Our model assumes that coherent text implicitly favors certain types of discourse relation transitions. We implement this model and apply it towards the text ordering ranking task, which aims to discern an original text from a permuted ordering of its sentences. The experimental results demonstrate that our model is able to significantly outperform the coherence model by (Barzilay and Lapata, 2005), reducing the error rate of the previous approach by an average of 29% over three data sets against human upper bounds. We further show that our model is synergistic with the previous approach, demonstrating an error reduction of 73% when the features from both models are combined for the task.

5.1 Introduction

Figure 5.1 shows two texts taken from Knott's thesis on *A Data-Driven Methodology for Motivating a Set of Coherence Relations* (Knott, 1996). Text (a) on the left column is

The World in 1993 1993 will start with the world in a pessimistic frame of mind. That gloom should soon dispel itself. A clear economic recovery is under way. Though it will be hesitant at first, it will last the longer for being so. If you are sitting in one of the world's blackspots, this prediction will seem hopelessly optimistic. But next year's wealth won't return to yesteryear's winners; these middle-aged rich people need to look over their shoulders to the younger world that is closing in on them.	The World in 1993 1993 will start with the world in a pessimistic frame of mind. A clear economic recovery is under way. That gloom should soon dispel itself. These middle-aged rich people need to look over their shoulders to the younger world that is closing in on them. But next year's wealth won't return to yesteryear's winners; it will last the longer for being so if you are sitting in one of the world's blackspots. Though it will be hesitant at first, this prediction will seem hopelessly optimistic.
(a)	(b)

Figure 5.1: Coherent and incoherent texts, from Knott's thesis (Knott, 1996). Text (a) on the left column is taken from the editorial of an issue of *The Economist*, whilst Text (b) on the right column contains exactly the same sentences as (a), but in a randomized order.

an original editorial of an issue of *The Economist*, while Text (b) on the right column is a permutation of (a), namely, sentences from (a) are randomized to form (b). Knott discussed two types of incoherence relating to Text (b). The first one has to do with resolving the referents of anaphoric expressions. As the text is simply reordered without rewording, readers will find it difficult to sort out what entities the anaphoric expressions refer to. For instance, the phrase "these middle-aged rich people" cannot be interpreted, as the sentence that contains the entity that this anaphoric expression refers to is shifted. This type of incoherence has been tackled by the entity-based coherence model in (Barzilay and Lapata, 2005).

The second class of incoherence is related to the readers' inability to understand the discourse of the text: why two text spans are juxtaposed, and how they are connected to each other. In the following example, the connective "if" suggests a causal relation between the two text spans. However, the discourse of these two spans does not resolve to a causal relation, even when we put them in the context of the whole text.

(5.1) it will last the longer for being so
 if you are sitting in one of the world's blackspots.

As another example, look at the first two sentences of Text (b):

(5.2) 1993 will start with the world in a pessimistic frame of mind.
 A clear economic recovery is under way.

It does not make sense for these two sentences to be placed next to each other: why should the world be pessimistic, if an economic recovery is under way? In other words, *the discourse does not follow coherently* from the first sentence to the second. In this chapter, we propose a coherence model to tackle this type of incoherence by comparing discourse patterns extracted from a pair of source and permuted texts.

Intra- and Inter-Discourse Relation Orderings

The coherence of a text is usually reflected by its discourse structure and relations. In Rhetorical Structure Theory (RST), Mann and Thompson (1988) observed that certain RST relations tend to favor one of two possible canonical orderings. Some relations (*e.g.*, Concessive and Conditional) favor arranging their satellite span before the nucleus span. In contrast, other relations (*e.g.*, Elaboration and Evidence) usually order their nucleus before the satellite. If a text that uses non-canonical relation orderings is rewritten to use canonical orderings, it often improves text quality and coherence.

This notion of preferential ordering of discourse relations is observed in natural language in general, and generalizes to other discourse frameworks aside from RST. The following example shows a Contrast relation signaled by "but" between two sentences.

(5.3) Everyone agrees that most of the nation's old bridges need to be repaired or replaced.

But there's disagreement over how to do it.

Here the second sentence provides contrasting information to the first. If this order is swapped without rewording, it produces an incoherent text (Marcu, 1996).

In addition to the intra-relation ordering, such preferences also extend to inter-relation ordering:

(5.4) The Constitution does not expressly give the president such power.
However, the president does have a duty not to violate the Constitution.
The question is whether his only means of defense is the veto.

The second sentence above provides a contrast to the previous sentence and an explanation for the next one. This pattern of Contrast-followed-by-Cause is rather common in text (Pitler et al., 2008). Ordering the three sentences differently results in incoherent, cryptic text.

Thus coherent text exhibits measurable preferences for specific intra- and inter-discourse relation ordering. Our key idea is to use the converse of this phenomenon to assess the coherence of a text. In this paper, we detail our model to capture the coherence of a text based on the statistical distribution of the discourse structure and relations. Our method specifically focuses on the discourse relation transitions between adjacent sentences, modeling them in a discourse role matrix.

In this chapter, we apply the discourse parser developed in Chapter 4 to generate input to our coherence model. We implement and validate our model on three data sets, which show robust improvements over (Barzilay and Lapata, 2005)'s entity-based model for coherence assessment. We also provide the first assessment of the upper-bound of human performance on the standard task of distinguishing coherent from incoherent orderings. To the best our knowledge, this is also the first study in which we show output from an automatic discourse parser helps in coherence modeling. We are aware that (Barzilay and Lapata, 2005)'s entity-based model is not the state-of-the-art model, as

Elsner and Charniak (2008; 2011) have proposed an improved model with discourse-new entity identification (cf. Chapter 2). On top of Barzilay and Lapata's model which looks at where and how entities are mentioned in the text, Elsner and Charniak separated the discourse-new (*i.e.*, newly mentioned) entities from discourse-old entities and added new features extracted from named entity recognition and coreference resolution to improve the model. Compared to the entity-based approach which models entity transition, our work takes a different approach by modeling text coherence from the point of view of discourse structure. To examine how our model differs from the core entity-based model, we compare our discourse model directly with (Barzilay and Lapata, 2005)'s core entity-based model and do not compare with Elsner and Charniak's enhanced model.

5.2 Using Discourse Relations

To utilize discourse relations of a text, we apply the automatic discourse parsing developed in the previous chapter. In contrast to the previous chapter, we discard the Level-2 type information generated by the parser and use only the four PDTB Level-1 relation types – Temporal, Contingency, Comparison, and Expansion – plus the entity relation and no relation (EntRel and NoRel) types. We found Level-1 types to give better performance empirically, which may due to the fact that the parser performance on Level-1 types is better than that on Level-2. Recall that as reported in Chapter 4, the parser's performance under fully automatic setting is 89.24% (Explicit) and 31.42% (Non-Explicit) for Level-1 types, and 80.61% (Explicit) and 25.46% (Non-Explicit) for Level-2 types.

A simple approach to directly model the connections among discourse relations is to use the sequence of discourse relation transitions. Text 5.4 in Section 5.1 can be represented by $S_1 \xrightarrow{Comparison} S_2 \xrightarrow{Contingency} S_3$, for instance, when we use Level-1 types. In such a basic approach, we can compile a distribution of the n-gram discourse relation transition sequences in gold standard coherent text, and a similar one for incoherent text.

For example, Text 5.4 would generate the transition bigram Comparison$\rightarrow$Contingency. We can build a classifier to distinguish one from the other through learned examples or using a suitable distribution distance measure (*e.g.*, KL Divergence).

In our pilot work where we implemented such a basic model with n-gram features for relation transitions, the performance was very poor. In the task of discerning original text from its permutation in the Earthquakes and Accidents data sets (which will be detailed in our experimental section), Barzilay and Lapata's entity-based models gave 83.59% and 89.93% accuracies, our pilot model which uses bigram and trigram discourse relation transition gave 65.9% and 70.1%, while the random baseline is 50% for both. Our analysis revealed a serious shortcoming: as the number of discourse relation transitions in short texts is small, we have very little data to base the coherence judgment on. However, when faced with even short text excerpts, humans can distinguish coherent texts from incoherent ones, as exemplified in our example texts. The basic approach also does not model the intra-relation preference. In Text 5.3, a Comparison relation would be recorded between the two sentences, regardless of whether S_1 or S_2 comes first. However, it is clear that the ordering of ($S_1 \prec S_2$) is more coherent. Thus, the core problem of using the n-gram features is data sparseness, which is most pronounced in short texts.

5.3 A Refined Approach

The central problem with the basic approach is in its sparse modeling of discourse relations. In developing an improved model, we need to better exploit the discourse parser's output to provide more circumstantial evidence to support the system's coherence decision. We thus introduce the concept of a discourse role matrix. This matrix aims to capture an expanded set of discourse relation transition patterns. We describe how to represent the coherence of a text with its discourse relations and how to transform such information into a matrix representation. We then illustrate how we use the matrix to formulate a preference ranking problem.

5.3.1 Discourse Role Matrix

Figure 5.2 and Figure 5.3 show a text and its gold standard PDTB discourse relations. When a term appears in a discourse relation, the discourse role of this term is defined as the discourse relation type plus the argument span in which the term is located (*i.e.*, the argument tag). For instance, consider the term "cananea" in the first relation. Since the relation type is a Comparison and "cananea" is found in the Arg1 span, the discourse role of "cananea" is defined as Comparison.Arg1. When terms appear in different relations and/or argument spans, they obtain different discourse roles in the text. For instance, "cananea" plays a different discourse role of Temporal.Arg1 in the third relation in Figure 5.2. In the fourth relation, since "cananea" appears in both argument spans, it has two additional discourse roles, Expansion.Arg1 and Expansion.Arg2. The discourse role matrix thus represents the different discourse roles of the terms across the continuous text units. We use sentences as the text units, and define terms to be the stemmed forms of the open class words: nouns, verbs, adjectives, and adverbs. We formulate the discourse role matrix such that it encodes the discourse roles of the terms across adjacent sentences.

Figure 5.4 shows a fragment of the matrix representation of the text in Figure 5.2. Columns correspond to the extracted terms; rows, the contiguous sentences. A cell C_{T_i,S_j} contains the set of the discourse roles of the term T_i that appears in sentence S_j. For example, the term "cananea" from S_1 takes part in the first relation, so the cell $C_{cananea,S_1}$ contains the role Comparison.Arg1. A cell may be empty (*nil*, as in $C_{cananea,S_2}$) or contain multiple discourse roles (as in $C_{cananea,S_3}$, as "cananea" in S_3 participates in the second, third, and fourth relations). Given these discourse relations, building the matrix is straightforward: we note the relations that a term T_i from a sentence S_j participates in, and record its discourse roles in the respective cell. Note that discourse relations can appear within a sentence (*i.e.*, arguments are clauses) and these relations are also captured

S_1 Japan normally depends heavily on the Highland Valley and **Cananea** mines as well as the Bougainville mine in Papua New Guinea.

S_2 Recently, Japan has been buying copper elsewhere.

$S_{3.1}$ But as Highland Valley and **Cananea** begin operating,
$S_{3.2}$ they are expected to resume their roles as Japan's suppliers.

$S_{4.1}$ According to Fred Demler, metals economist for Drexel Burnham Lambert, New York,
$S_{4.2}$ "Highland Valley has already started operating
$S_{4.3}$ and **Cananea** is expected to do so soon."

Figure 5.2: An excerpt with four contiguous sentences from wsj_0437. The term "cananea" is highlighted for the purpose of illustration. $S_{i.j}$ means the jth clause in the ith sentence.

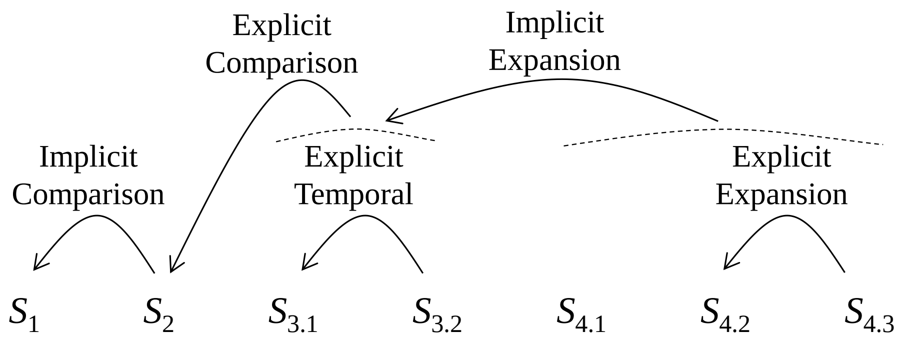


Figure 5.3: Five gold standard discourse relations on the excerpt in Figure 5.2. Arrows are pointing from Arg2 to Arg1.

$S\#$	Terms				
	copper	cananea	operat	depend	...
S_1	nil	Comparison.Arg1	nil	Comparison.Arg1	
S_2	Comparison.Arg2 Comparison.Arg1	nil	nil	nil	
S_3	nil	Comparison.Arg2 Temporal.Arg1 Expansion.Arg1	Comparison.Arg2 Temporal.Arg1 Expansion.Arg1	nil	
S_4	nil	Expansion.Arg2	Expansion.Arg1 Expansion.Arg2	nil	

Figure 5.4: Discourse role matrix fragment for Figure 5.2 and 5.3. Rows correspond to sentences, columns to stemmed terms, and cells contain extracted discourse roles.

in the matrix. This is not affected by the fact that we use sentences as processing units in the matrix.

We hypothesize that the sequence of discourse role transitions in a coherent text provides clues that distinguish it from an incoherent text. The discourse role matrix thus provides the foundation for computing such role transitions, on a per term basis. In fact, each column of the matrix corresponds to a lexical chain (Morris and Hirst, 1991) for a particular term across the whole text. The key differences from the traditional lexical chains are that our chain nodes' entities are simplified (they share the same stemmed form, instead being connected by WordNet relations), but are further enriched by being typed with discourse relations.

We compile the set of sub-sequences of discourse role transitions for every term in the matrix. These transitions tell us how the discourse role of a term varies through the progression of the text. For instance, "cananea" functions as Comparison.Arg1 in S_1 and Comparison.Arg2 in S_3, and plays the role of Expansion.Arg1 and Expansion.Arg2 in S_3 and S_4, respectively. As we have six relation types (Temporal, Contingency, Comparison, Expansion, EntRel and NoRel) and two argument tags (Arg1 and Arg2) for each type, we have a total of $6 \times 2 = 12$ possible discourse roles, plus a *nil* value. We define a *discourse role transition* as the sub-sequence of discourse roles for a term in multiple consecutive sentences. For example, the discourse role transition of "cananea" from S_1 to S_2 is Comparison.Arg1→nil. As a cell may contain multiple discourse roles, a transition may produce multiple sub-sequences. For example, the length 2 sub-sequences for "cananea" from S_3 to S_4, are Comparison.Arg2→Expansion.Arg2, Temporal.Arg1→Expansion.Arg2, and Expansion.Arg1→Expansion.Arg2.

Each sub-sequence has a probability that can be computed from the matrix. To illustrate the calculation, suppose the matrix fragment in Figure 5.4 is the entire discourse role matrix. Then since there are in total 25 length 2 sub-sequences and the sub-sequence Comparison.Arg2→Expansion.Arg2 has a count of two, its probability is $2/25 = 0.08$.

A key property of our approach is that, while discourse transitions are captured locally on a per-term basis, the probabilities of the discourse transitions are aggregated globally, across all terms. We believe that the overall distribution of discourse role transitions for a coherent text is distinguishable from that for an incoherent text. Our model captures the distributional differences of such sub-sequences in coherent and incoherent text in training to determine an unseen text's coherence. To evaluate the coherence of a text, we extract sub-sequences with various lengths from the discourse role matrix as features[1] and compute the sub-sequence probabilities as the feature values.

To further refine the computation of the sub-sequence distribution, we follow (Barzilay and Lapata, 2005) and divide the matrix into a salient matrix and a non-salient matrix. Terms (columns) with a frequency greater than a threshold form the salient matrix, while the rest form the non-salient matrix. The term frequency is calculated based on the text under consideration, *i.e.*, it will vary when we apply the model to a new text. The sub-sequence distributions are then calculated separately for these two matrices. The threshold can be set empirically, maximizing performance on a set task (described below).

5.3.2 Preference Ranking

While some texts can be said to be simply coherent or incoherent, often it is a matter of degree. A text can be less coherent when compared to one text, but more coherent when compared to another. As such, since the notion of coherence is relative, we feel that coherence assessment is better represented as a ranking problem rather than a classification problem. Given a pair of texts, the system ranks them based on how coherent they are. Applications of such a system include differentiating a text from its permutation (*i.e.*, the sentence ordering of the text is shuffled) and identifying a more well-written essay from a pair. Such a system can easily generalize from pair-wise ranking into list-wise, suitable for the ordinal ranking of a set of texts. Coherence scoring equations can also be

[1]Sub-sequences consisting of only *nil* values are not used as features.

deduced (Lapata and Barzilay, 2005) from such a model, yielding coherence scores. Most recently, we showed in (Lin et al., 2012) that this model is also capable of ranking a list of texts in the task of summarization evaluation.

To induce a model for preference ranking, we use the SVM^{light} package[2] by (Joachims, 1999) with the preference ranking configuration for training and testing. All parameters are set to their default values.

5.4 Experiments

We evaluate our coherence model on the task of *text ordering ranking*, a standard coherence evaluation task used in both (Barzilay and Lapata, 2005) and (Elsner et al., 2007). In this task, the system is asked to decide which of two texts is more coherent. The pair of texts consists of a source text and one of its permutations (*i.e.*, the text's sentence order is randomized). Assuming that the original text is always more discourse-coherent than its permutation, an ideal system will prefer the original to the permuted text. A system's accuracy is thus the number of times the system correctly chooses the original divided by the total number of test pairs.

In order to acquire a large data set for training and testing, we follow the approach in (Barzilay and Lapata, 2005) to create a collection of synthetic data from *Wall Street Journal* (WSJ) articles in the Penn Treebank. All of the WSJ articles are randomly split into a training and a testing set; 40 articles are held out from the training set for development. For each article, its sentences are permuted up to 20 times to create a set of permutations[3]. Each permutation is paired with its source text to form a pair.

We also evaluate on two other data collections (cf. Table 5.1), provided by (Barzilay and Lapata, 2005), for a direct comparison with their entity-based model. These two data sets consist of Associated Press articles about earthquakes from the North American News

[2]`http://svmlight.joachims.org/`

[3]Short articles may produce less than 20 permutations.

		WSJ	Earthquakes	Accidents
Train	# Articles	1040	97	100
	# Pairs	19120	1862	1996
	Avg. # Sents	22.0	10.4	11.5
Test	# Articles	1079	99	100
	# Pairs	19896	1956	1986

Table 5.1: Details of the WSJ, Earthquakes, and Accidents data sets, showing the number of training/testing articles, number of pairs of articles, and average length of an article (in sentences).

Corpus, and narratives from the National Transportation Safety Board. These collections are much smaller than the WSJ data, as each training/testing set contains only up to 100 source articles. Similar to the WSJ data, we construct pairs by permuting each source article up to 20 times.

Our model has two parameters:

1. The term frequency (TF) that is used as a threshold to identify salient terms, and
2. The lengths of the sub-sequences that are extracted as features.

These parameters are tuned on the development set, and the best ones that produce the optimal accuracy are TF $>=$ 2 and lengths of the sub-sequences $<=$ 3.

We must also be careful in using the automatic discourse parser. We note that the discourse parser employed is trained on the PDTB, which provides annotations on top of the whole WSJ data. As we also use the WSJ data for this evaluation, we must avoid parsing an article that has already been used in training the parser to prevent training on the test data. We re-train the parser with 24 WSJ sections and use the trained parser to parse the sentences in our WSJ collection from the remaining section. We repeat this re-training/parsing process for all 25 sections. Because the Earthquakes and Accidents data do not overlap with the WSJ training data, we use the original parser to parse these two remaining data sets. Since the discourse parser utilizes paragraph boundaries but a permuted text does not have such boundaries, we ignore paragraph boundaries and treat the source text as if it has only one paragraph. This is to make sure that we do not give the system extra information because of this difference between the source and permuted text.

5.4.1 Human Evaluation

While the text ordering ranking task has been used in previous studies, two key questions about this task have remained unaddressed in the previous work:

1. To what extent is the assumption that the source text is more coherent than its permutation correct?
2. How well do humans perform on this task?

The answer to the first is needed to validate the correctness of this synthetic task, while the second aims to obtain the upper bound for evaluation. We conduct a human evaluation to answer these questions.

We randomly select 50 source text/permutation pairs from each of the WSJ, Earthquakes, and Accidents training sets. We observe that some of the source texts have formulaic structures in their initial sentences that give away the correct ordering. Sources from the Earthquakes data always begin with a headline sentence and a location-newswire sentence, such as:

BC-Taiwan-Earthquake|Quake Jolts Taiwan
TAIPEI, Taiwan (AP) An earthquake with ...

and many sources from the Accidents data start with the following two sentences:

This is preliminary ... errors.
Any errors ... completed.

We remove these sentences from the source and permuted texts, to avoid the subjects judging based on these clues instead of textual coherence. For each set of 50 pairs, we

WSJ	Earthquakes	Accidents	Overall
90.0	90.0	94.0	91.3

Table 5.2: Inter-subject agreements on the three data sets.

assigned two human subjects (who was not the author of this thesis nor his supervisors) to perform the ranking. The subjects are told to identify the source text from the pair. When both subjects rank a source text higher than its permutation, we interpret it as the subjects agreeing that the source text is more coherent than the permutation. Table 5.2 shows the inter-subject agreements.

While our study is limited and only indicative, we conclude from these results that the task is tractable. Also, since our subjects' judgments correlate highly with the gold standard, the assumption that the original text is always more coherent than the permuted text is supported. Importantly though, human performance is not perfect, suggesting fair upper bound limits on system performance. We note that the Accidents data set is relatively easier to rank, as it has a higher upper bound than the other two. This is because articles in Accidents data set are accident reports which are more formulaic than the news articles in the other two data sets.

5.4.2 Baseline

Barzilay and Lapata (2005) showed that their entity-based model is able to distinguish a source text from its permutation accurately. Thus, it can serve as a good comparison point for our discourse relation-based model. Their full model is Coreference+Syntax+Salience, which uses coreference resolution to group entity classes, syntax to determine whether an entity is subject, object, or other, and salience to determine whether an entity appears frequently in the text. We compare against their Syntax+Salience setting, instead of the setting with coreference resolution. Since they did not automatically determine the coreferential information of a permuted text but obtained that from its corresponding source text, we do not perform automatic coreference resolution in our reimplementation of their system. For fair comparison, we follow their experiment settings as closely as possible. We re-use their Earthquakes and Accidents dataset as is, using their exact permutations and pre-processing. For the WSJ data, we need to perform our own pre-processing, thus we employed the Stanford parser[4] to perform sentence segmentation and constituent parsing, followed by entity extraction.

5.4.3 Results

We perform a series of experiments to answer the following four questions:

1. Does our model outperform the baseline?
2. How do the different features derived from using relation types, argument tags, and salience information affect performance?
3. Can the combination of the baseline and our model outperform the single models?
4. How does system performance of these models compare with human performance on the task?

Baseline results are shown in the first row of Table 5.3. The results on the Earthquakes and Accidents data are quite similar to those published in (Barzilay and Lapata, 2005) (they reported 83.4% on Earthquakes and 89.7% on Accidents), validating the correctness of our reimplementation of their method.

Row 2 in Table 5.3 shows the overall performance of the proposed refined model, answering Question 1. The model setting of Type+Arg+Sal means that the model makes use of the discourse roles consisting of (1) relation types and (2) argument tags (*e.g.*, the discourse role Comparison.Arg2 consists of the type Comparison and the tag Arg2), and

[4]`http://nlp.stanford.edu/software/lex-parser.shtml`

	WSJ	Earthquakes	Accidents
Baseline	85.71	83.59	89.93
Type+Arg+Sal	**88.06****	**86.50****	**89.38**
Arg+Sal	88.28**	85.89*	87.06
Type+Sal	87.06**	82.98	86.05
Type+Arg	85.98	82.67	87.87
Baseline & Type+Arg+Sal	89.25**	89.72**	91.64**

Table 5.3: Test set ranking accuracy. The first row shows the baseline performance, the next four show our model with different settings, and the last row is a combined model. Double (**) and single (*) asterisks indicate that the respective model significantly outperforms the baseline at $p < 0.01$ and $p < 0.05$, respectively. We follow (Barzilay and Lapata, 2008) and use the Fisher Sign test.

(3) two distinct feature sets from salient and non-salient terms. Recall that the features are sub-sequences of discourse roles with lengths $<= 3$ and values are their probabilities in the matrix. Comparing these accuracies to the baseline, our model significantly outperforms the baseline with $p < 0.01$ in the WSJ and Earthquakes data sets with accuracy increments of 2.35$\%$ and 2.91$\%$, respectively. In Accidents, our model's performance is slightly lower than the baseline, but the difference is not statistically significant. The reason is that the accident reports in this data set are more formulaic and contain many mentions of the pilots and airplane parts, which are easily captured by the entity-based model.

To answer Question 2, we perform feature ablation testing. We eliminate each of the information sources from the full model. In **Row 3**, we first delete relation types from the discourse roles, which causes discourse roles to only contain the argument tags. A discourse role such as Comparison.Arg2 becomes Arg2 after deleting the relation type. Comparing Row 3 to Row 2, we see performance reductions on the Earthquakes and Accidents data after eliminating type information. **Row 4** measures the effect of omitting argument tags (Type+Sal). In this setting, the discourse role Comparison.Arg2 reduces to Comparison. We see a large reduction in performance across all three data sets. This model is also most similar to the basic naïve model in Section 5.2. These results suggest that the argument tag information plays an important role in our discourse role transition model. **Row 5** omits the salience information (Type+Arg), which also markedly reduces performance. This result supports the use of salience, in line with the conclusion drawn in (Barzilay and Lapata, 2005).

To answer Question 3, we train and test a combined model using features from both the baseline and our model (shown as **Row 6** in Table 5.3). The entity-based model of Barzilay and Lapata (2005) connects the local entity transition with textual coherence, while our model looks at the patterns of discourse relation transitions. As these two models focus on different aspects of coherence, we expect that they are complementary to each other. The combined model in all three data sets gives the highest performance in comparison to all single models, and it significantly outperforms the baseline model with $p < 0.01$. This confirms that the combined model is linguistically richer than the single models as it integrates different information together, and the entity-based model and our model are synergistic.

To answer Question 4, when compared to the human upper bound (Table 5.2), the performance gaps for the baseline model are relatively large, while those for our full model are more acceptable in the WSJ and Earthquakes data. For the combined model, the error rates are significantly reduced in all three data sets. The average error rate reductions against 100% are 9.57% for the full model and 26.37% for the combined model. If we compute the average error rate reductions against the human upper bounds (rather than an oracular 100%), the average error rate reduction for the full model is 29% and that for the combined model is 73%. While these are only indicative results, they do highlight the significant gains that our model is making towards reaching human performance levels.

We further note that some of the permuted texts may read as coherently as the original text. This phenomenon has been observed in several natural language synthesis tasks such as generation and summarization, in which a single gold standard is inadequate to fully assess performance. As such, both automated systems and humans may actually perform better than our performance measures indicate. We leave it to future work to measure the impact of this phenomenon.

Given that we have shown the upper bound for the task in all three data sets, we also note that it would be good to show the performance of our system with gold standard discourse structures. However, we do not have gold standard discourse structures for the Earthquakes and Accidents data sets. Although we have that for the original articles in the WSJ data, we also do not have gold standard discourse parsing for the corresponding permutations. Thus it is not possible to conduct the same experiments under gold standard setting.

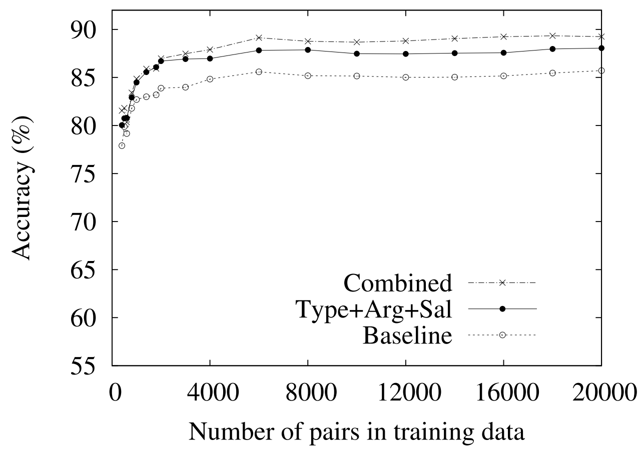
5.5 Analysis and Discussion

When we compare the accuracies of the full model in the three data sets (Row 2), the accuracy in the Accidents data is the highest (89.38%), followed by that in the WSJ (88.06%), with Earthquakes at the lowest (86.50%). To explain the variation, we examine the ratio between the number of the relations in the article and the article length (*i.e.*, number of sentences). This ratio is 1.22 for the Accidents source articles, 1.2 for the WSJ, and 1.08 for Earthquakes. The relation/length ratio gives us an idea of how often a sentence participates in discourse relations. A high ratio means that the article is densely interconnected by discourse relations, and may make distinguishing this article from its permutation easier compared to that for a loosely connected article.

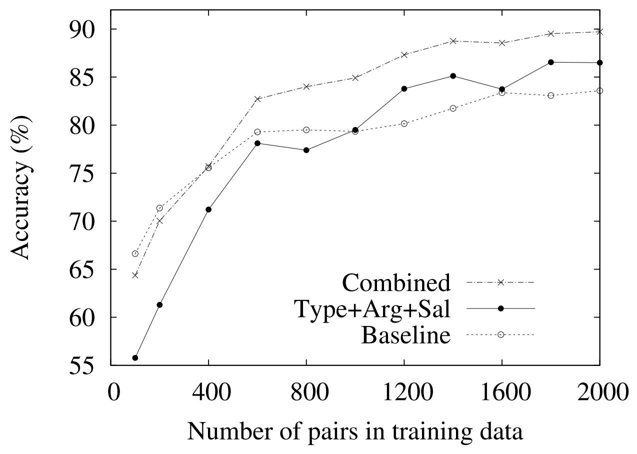
We expect that when a text contains more discourse relation types (*i.e.*, Temporal, Contingency, Comparison, Expansion), it is easier to compute how coherent this text is. This is because compared to articles with less discourse relations, these four discourse relations can combine to produce meaningful transitions, such as the example Text 5.4. To examine how this affects performance, we calculate the average ratio between the number of the four discourse relations in the permuted text and the length for the permuted text. The ratio is 0.58 for those that are correctly ranked by our system, and 0.48 for those that are incorrectly ranked, which supports our hypothesis.

We also examined the learning curves for our Type+Arg+Sal model, the baseline model, and the combined model on the data sets, as shown in Figures 5.5(a)–5.5(c). In the WSJ data, the accuracies for all three models increase rapidly as more pairs are added to the training set. After 2,000 pairs, the increase slows until 8,000 pairs, after which the curve is nearly flat. From the curves, our model consistently performs better than the baseline with a significant gap, and the combined model also consistently and significantly outperforms the other two. Only about half of the total training data is needed to reach optimal performance for all three models. The learning curves in the Earthquakes data show that the performance for all models is always increasing as more training pairs are utilized. The Type+Arg+Sal and combined models start with lower accuracies than the baseline, but catch up with it at 1,000 and 400 pairs, respectively, and consistently outperform the baseline beyond this point. On the other hand, the learning curves for the Type+Arg+Sal and baseline models in Accidents do not show any one curve consistently better than the other: our model outperforms in the middle segment but underperforms in the first and last segments. The curve for the combined model shows a consistently significant gap between it and the other two curves after the point at 400 pairs.

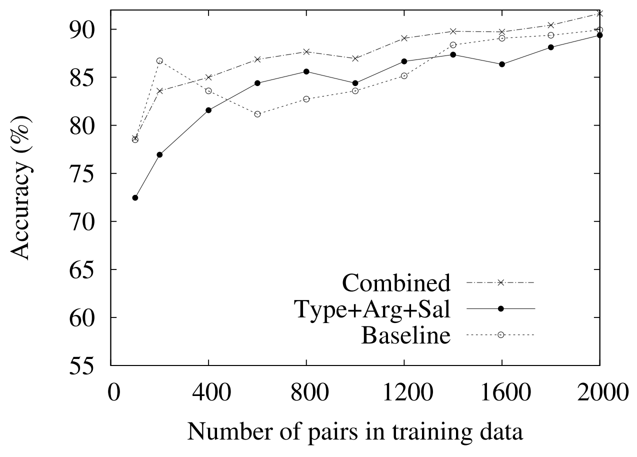
With the performance of the model as it is, how can future work improve upon it? We point out one weakness that we plan to explore. We use the full Type+Arg+Sal model trained on the WSJ training data to test Text (5.4) from the introduction, which is repeated in Figure 5.6. As (5.4) has 3 sentences, permuting it gives rise to 5 permutations, as shown in Figure 5.6. The model is able to correctly determine the original in four of these 5 pairs. The only permutation it fails on is ($S_3 \prec S_1 \prec S_2$), when the last sentence is moved to the beginning. A very good clue of coherence in Text 5.4 is the Explicit Comparison relation between S_1 and S_2 signaled by “However”. Since this clue



(a) WSJ



(b) Earthquakes



(c) Accidents

Figure 5.5: Learning curves for the Type+Arg+Sal, the baseline, and the combined models on the three data sets.

S_1 The Constitution does not expressly give the president such power.
S_2 However, the president does have a duty not to violate the Constitution.
S_3 The question is whether his only means of defense is the veto.

Original ordering:
$S_1 \prec S_2 \prec S_3$

Permuted orderings:
$S_1 \prec S_3 \prec S_2$
$S_2 \prec S_1 \prec S_3$
$S_2 \prec S_3 \prec S_1$
$S_3 \prec S_1 \prec S_2$
$S_3 \prec S_2 \prec S_1$

Figure 5.6: An exemplar text of three sentences and its five permutations.

is retained in ($S_3 \prec S_1 \prec S_2$), it is difficult for the system to distinguish this ordering from the source. In contrast, as "However" becomes less coherent with the movement of its Arg1, it is easier to distinguish them as incoherent. By modeling longer range discourse relation transitions, we may be able to discern these two cases.

5.6 Conclusion

In this chapter, we have proposed a new model for discourse coherence that leverages the observation that coherent texts preferentially follow certain discourse structures. We posit that these structures can be captured in and represented by the patterns of discourse relation transitions. We first demonstrate that simply using the sequence of discourse relation transition leads to sparse features and is insufficient to distinguish coherent from incoherent text. To address this, our method transforms the discourse relation transitions into a discourse role matrix. The matrix schematically represents term occurrences in text units and associates each occurrence with its discourse roles in the text units. In our approach, per-term transition sub-sequences constitute the evidence used in our model to distinguish coherence from incoherence.

When applied to distinguish a source text from a sentence-reordered permutation, our model significantly outperforms the previous the entity-based local coherence model. While the entity-based model captures repetitive mentions of entities, our discourse relation-based model gleans its evidence from the argumentative and discourse structure of the text. Our model is complementary to the entity-based model, as it tackles the same problem from a different perspective. Experiments validate our claim, with a combined model outperforming both single models.

The idea of modeling coherence with discourse relations and formulating it in a discourse role matrix can also be applied to other NLP tasks. We plan to apply our methodology to other tasks, such as summarization, text generation and essay scoring, which also need to produce and assess discourse coherence. Apart from applying our discourse parser in coherence modeling, in the next chapter, we will propose the applications of discourse parsing in both summarization and argumentative zoning.

Chapter 6

Applying Discourse Relations in Summarization and Argumentative Zoning of Scholarly Papers

In Chapter 5, we have shown that we can build a textual coherence model by leveraging discourse patterns from our fully automatic discourse parser and apply this model to distinguish coherent from incoherent texts. In this chapter, we show that discourse structures and relations can also improve downstream NLP applications. We demonstrate that incorporating discourse features can significantly improve two NLP tasks – argumentative zoning and summarization – in the scholarly domain. We also show that output from these two tasks can improve each other in an iterative model.

6.1 Introduction

Each segment in a text has a specific rhetorical function with regard to its context. Focusing on the scholarly domain, a scientific paper usually devotes parts of the body to explain the research goals of the work, describe others' work, compare their own work against others', and most importantly, illustrate the nuances of their own contributions. Each sentence can thus be supposed to have its own rhetorical function to guide the reader in understanding the paper's arguments. To this end, argumentative zoning (hereafter, "AZ" or "zoning") is a task proposed by Teufel (Teufel, 1999) to assign a rhetorical function to each sentence from a set of seven canonical rhetorical functions: AIM (aim), BAS (basis), BKG (background), CTR (contrast), OTH (other), OWN (own), and TXT (textual). The description of these labels was shown in Table 2.3 in Chapter 2.

Discourse parsing differs from argumentative zoning in that it constructs the discourse structure of a text in order to unveil the discourse relations between text spans. Despite this difference, both discourse parsing and zoning target the understanding of a text from its structure, context, and syntactic arguments. For instance, we have shown that discourse parsing provides a measure for how coherent a text is in the last chapter; argumentative zoning can also be said to provide an indication of a paper's coherence: a well-written research paper usually follows conventional rhetorical patterns in organizing and explaining its aims, background, own work, and comparison to other work. Thus, we argue that discourse parsing and argumentative zoning exhibit a strong connection, and we hypothesize that discourse parsing can assist in the task of argumentative zoning.

Text summarization has also been shown to have strong connection with discourse parsing and can benefit from an understanding of a text's discourse structure. Carlson et al. (2001)'s RST Discourse Treebank contains both nucleus-satellite and nucleus-nucleus relations of all informational relations, which shows that nuclearity is distinct from relation type. Marcu (Marcu, 1997) explained that the nucleus-satellite view on discourse relations provide an indication of the importance of the text units. This indication of importance can be applied in summarization to promote and select important text spans. The PDTB framework does not have the notion of nuclearity. However, its relation types and arguments together may provide hints at which argument spans contain more relevant information to specific application needs. For instance, in the Cause relation signaled by "because" in Example 2.1, Arg2 provides the reason to support the event in Arg1. Thus, we also hypothesize that automated text summarization can benefit from the output of discourse parsing. Related to this, Louis et al. (2010) have already demonstrated the use of PDTB relations in summarization. However, our work differs from theirs in that we utilize a fully automated discourse parser whereas they experimented on the gold standard data set.

We first explain how we incorporate the discourse features from our parser for both argumentative zoning and scientific paper summarization tasks in the next section. We then describe our experimental setup and results for each task. We then present an iterative model that accomplishes both tasks.

6.2 Methodology

We first apply discourse parsing on the input scientific papers with our automatic parser developed in Chapter 4. Similar to applying discourse parsing in coherence modeling in the chapter, we leverage the four PDTB Level-1 types (Temporal, Contingency, Comparison, and Expansion), EntRel, and NoRel. Both Explicit and Implicit relations are utilized.

6.2.1 Discourse Features for Argumentative Zoning

In the argumentative zoning task, each sentence is assigned one of the seven labels: AIM, BAS, BKG, CTR, OTH, OWN, and TXT. At sentence level, the discourse parser provides information on whether a particular sentence has certain types of discourse connections with its context (specifically, its previous and next sentences) and within itself. We hypothesize that such sentence-level features are related to AZ labels. For example, sentences that have a Comparison relation may be more likely to be labeled as CTR, as CTR (contrast) means to compare with others' work; whereas sentences with Expansion relations may relate more to the OWN sentences, as Expansion relations are used to exemplify, restate, and generalize the discussion, which are usually used by one in explaining one's OWN work. Thus, for each sentence, we use the following output of our discourse parser:

If both Arg1 and Arg2 spans of a relation appear in a sentence (*i.e.*, the SS case), the following feature is triggered:

SS + relation type

else if one argument span of a relation appears in a sentence (*i.e.*, the PS case), this feature is triggered instead:

PS + relation type + argument tag

where *argument tag* is Arg1 or Arg2. Here SS means same sentence and PS means the immediately previous sentence (cf. Chapter 4). We also include the number of relations that this sentence participates in as a feature.

For instance, in the following example (Example 6.1), there is an Implicit Comparison relation between the first and the second sentences (the last line of the example shows the article number that this example belongs to). The first sentence contains the Arg1 span of this relation and the second contains the Arg2 span. Thus, for the second sentence, we will extract the feature "PS_Comparison_Arg2". This indicates that the second sentence is contrastive to the first one and therefore its AZ class is CTR. The second sentence is also involved in another relation with the sentence following it, thus the number of relations for the second sentence is 2.

(6.1) In fact, it appears that gapping is felicitous in those constructions where VP-ellipsis requires a syntactic antecedent, whereas gapping is infelicitous in cases where VP-ellipsis requires only a suitable semantic antecedent.

Past approaches to VP-ellipsis that operate within a single module of language processing fail to make the distinctions necessary to account for these differences.

(CMP-LG 9405010)

6.2.2 Discourse Features for Summarization

Summarization techniques are usually divided into *extractive* and *abstractive* summarization. Simply put, one generates a summary by extracting important and representative sentences, whilst the other rewrites or generates a natural text as the summary. In this work, we follow a supervised extractive summarization approach, in which we train a system to score sentence importance and extract the most salient sentences to construct a summary.

Knowing the relation type and argument tag of a sentence in a discourse relation can provide clues for the importance of this sentence with regard to its context. The discourse features that we use for summarization are slightly different from those for argumentative zoning: we ignore the SS cases (as we will be extracting entire sentences), and instead differentiate between Explicit and Implicit relations. For each sentence, if one argument span of a relation appears in this sentence, we extract the feature:

PS + Explicit/Implicit + relation type + argument tag

For instance, the feature for the second sentence in Example 6.1 for summarization is "PS_Implicit_Comparison_Arg2". This is indicative of the importance of the sentence, as it shows that the sentence is providing contrastive and updated information to its previous context. Similar to argumentative zoning, we also include the number of relations as a feature.

6.3 Experiments

We first describe our data sets, experimental setup, and the two respective baseline systems. We then demonstrate that incorporating discourse features significantly improves both tasks. While the focus of this chapter is to study the application of discourse parsing in summarization and argumentative zoning, it will be interesting to also look at whether

	AIM	BAS	BKG	CTR	OTH	OWN	TXT	Total
#instances	310	246	789	600	2019	8624	227	12815
%	2.42	1.92	6.16	4.68	15.75	67.30	1.77	

Table 6.1: Number and percentages of the instances of the AZ labels.

	AIM	BAS	BKG	CTR	OTH	OWN	TXT
% in abstract	27.14	2.00	8.57	8.00	6.86	47.43	0.00
% in body	1.65	1.92	6.09	4.59	16.01	67.89	1.82

Table 6.2: Percentages of AZ labels in abstract and body.

these two tasks have effect on each other, as this has not been properly studied before. For example, as shown in Table 6.2, AZ labels have very different distribution in the abstract and body of scientific papers. As we want to generate a summary similar to the abstract, taking AZ labels as input will help the summarization system to produce a summary with correct distribution of AZ labels. To this end, we propose an iterative model, and show that argumentative zoning and summarization can improve each other in this model.

6.3.1 Data and Setup

We utilize the corpus of 80 scientific articles from (Teufel, 1999) as-is. These papers were collected from the conferences and workshops of ACL, COLING, ANLP, and EACL, published during 1987–1996. The papers' sentences are annotated with the AZ labels by three trained annotators. There are 12,815 sentences in total, exhibiting a skewed zoning distribution: 67.30% of all sentences are marked as OWN. This skewed distribution is reasonable, as the bulk of a paper should describe the authors' own work. We give the label distribution on the corpus in Table 6.1 and the label distribution in abstract and body of the papers in Table 6.2.

The same 80 articles are also used in the summarization task. In scientific article summarization, abstracts can be deemed gold standard summaries (model summaries), a practice which we adopt. The average number of words in the abstracts is 102. Thus, we round this down and set the maximum length of the extracted summaries to be 100 words, which means that the last sentence of the extracted summary will need to be trimmed if needed, to keep it exactly 100 words. When summarizing the scientific article, its abstract is omitted from the summarization process.

Since we have a relatively small set of documents, we employ 10-fold cross validation to assess performance. The articles are sorted by their article numeric IDs and divided into 10 sets of 8 articles.

We employ the Robust Argumentative Zoning (RAZ) system (Teufel and Kan, 2011) as the baseline system for the zoning tasks. RAZ is *robust* to articles with plain text input, *i.e.*, without markup for titles, authors, section titles, tables, figures, citations, etc. RAZ uses the maximum entropy classifier from Le Zhang[1] for sentence classification. Features used by RAZ include raw tokens, bigrams and trigrams, cue words and phrases, position, citation presence, sentence length, title overlap, agent, and verb tense. Teufel (1999) also described a classifier for Argumentative Zoning. The differences between Teufel's original system and RAZ are that Teufel's system

1. Utilizes a Naïve Bayes classifier,
2. Uses a much larger set of 1000 – 2000 hand crafted patterns and rules, and
3. Needs fully marked up articles as input (*i.e.*, it needs markup for titles, authors, section titles, equations, tables, figures, and citations).

Teufel (1999) obtained a macro F_1 of 50%, while RAZ gives a macro F_1 of 46.79%. However, as RAZ is freely available online[2], we use RAZ directly, instead of reimplementing Teufel's system. The reason is that RAZ and Teufel's system share many of their feature classes, and if we observe significant improvement on RAZ with discourse features, we

[1]`http://homepages.inf.ed.ac.uk/lzhang10/maxent.html`
[2]`http://wing.comp.nus.edu.sg/raz/`

	Precision	Recall	F_1
AIM	60%	44%	51%
BAS	29%	18%	22%
BKG	32%	20%	24%
CTR	28%	14%	19%
OTH	35%	28%	31%
OWN	76%	87%	81%
TXT	74%	52%	61%

Table 6.3: RAZ performance on each label reported in (Teufel and Kan, 2011).

can conclude that discourse features are complementary to the current AZ feature set. Table 6.3 shows the precision, recall, and F_1 for each label reported in (Teufel and Kan, 2011).

We use a machine learning approach to train and test the summarization system, and use ROUGE (Lin and Hovy, 2003) to automatically evaluate the system. The problem with a machine learning approach for extractive supervised summarization is that we need to train a regression model for scoring sentences, but we do not have annotations of importance scores of sentences for training. We follow (Varma et al., 2009) to estimate sentence importance score by calculating the ROUGE-2 score between a sentence S and the sentences in the abstract:

$$ROUGE2\_score(S, abstract) = \frac{\sum_{S' \in abstract} |Bigrams_S \cap Bigrams_{S'}|}{|S|} \tag{6.2}$$

where $|Bigrams_S \cap Bigrams_{S'}|$ is the number of bigrams shared by the sentence S and the abstract sentence S', and the sum is normalized by the length of S. ROUGE-2 is one of the evaluation scores calculated by ROUGE which has been shown to have high correlation with humans. Since ROUGE-2 has high correlation with human judgment, we can use it to estimate the gold standard importance scores for the sentences.

With the estimated importance scores, we use a regression model to estimate a real value for each testing sentence. We apply Support Vector Regression (SVR) to train our summarization model, as SVR has been shown to perform well in summarization task (Varma et al., 2009). We use the SVM^{light} package[3] by (Joachims, 1999) with the regression configuration. After SVR assigns a score to each sentence, the system ranks the sentences in descending order of the scores and constructs a summary by selecting the first several sentences with a cutoff of 100 words.

As our summarization system uses supervised approach, we also choose a supervised baseline system for comparison. The baseline summarization system uses three features: the length of the sentence, the position of the sentence, and the LexRank score of the sentence. The first two are standard features used by most supervised and unsupervised summarization systems. LexRank (Erkan and Radev, 2004a) is a graph model that computes the sentence centrality by running a random walk on the lexical graph, where nodes are sentences and edges are lexical similarities. LexRank has been shown to perform very well in text summarization (Erkan and Radev, 2004b).

6.3.2 Results for Argumentative Zoning

Table 6.4 shows the evaluation results (precision, recall, and F_1) and confusion matrix for the baseline RAZ system. The final row shows the macro precision, recall and F_1 over all labels. The macro F_1 for the baseline RAZ system is 46.79%.

We add the discourse features from the output of the discourse parser into the RAZ system. The evaluation results for RAZ+Discourse is shown in Table 6.5. All macro precision, recall and F_1 scores are improved over the RAZ system, with a new macro F_1 of 47.88%. A two tailed paired t-test shows that this new macro F_1 is significantly higher than that for RAZ with $p < 0.01$. Comparing precision and recall scores, it is clear that recall goes up in some classes and down in others, while precision is improved in all classes except OWN. For example, the precision for AIM increases by 6.93% while its recall drops by 3.23%. One reason is that adding discourse information improves the

[3]`http://svmlight.joachims.org/`

	Predicted as									
	AIM	BAS	BKG	CTR	OTH	OWN	TXT	P	R	F_1
AIM	**177**	6	17	5	13	84	8	59.60	57.10	58.32
BAS	13	**63**	7	5	54	102	2	34.05	25.61	29.23
BKG	8	4	**226**	22	190	336	3	36.69	28.64	32.17
CTR	13	12	52	**119**	88	315	1	35.95	19.83	25.56
OTH	7	57	118	46	**647**	1140	4	38.15	32.05	34.83
OWN	71	41	191	134	689	**7453**	45	78.56	86.42	82.30
TXT	8	2	5	0	15	57	**140**	68.97	61.67	65.12
Macro								50.28	44.47	**46.79**

Table 6.4: Results for the baseline RAZ system.

	Predicted as									
	AIM	BAS	BKG	CTR	OTH	OWN	TXT	P	R	F_1
AIM	**167**	7	17	5	12	96	6	66.53	53.87	59.54 (+1.22)
BAS	11	**61**	6	4	64	97	3	34.08	24.80	28.71 (−0.52)
BKG	4	5	**240**	25	178	335	2	41.10	30.42	34.96 (+2.79)
CTR	5	10	49	**132**	84	318	2	38.82	22.00	28.09 (+2.53)
OTH	7	53	113	48	**635**	1160	3	39.27	31.45	34.93 (+0.10)
OWN	52	41	155	125	633	**7573**	45	78.55	87.81	82.92 (+0.62)
TXT	5	2	4	1	11	62	**142**	69.95	62.56	66.05 (+0.93)
Macro								52.61	44.70	**47.88** (+1.09)

Table 6.5: Results for RAZ+Discourse. A two tailed paired t-test shows that macro F_1 for RAZ+Discourse is significantly better than that for RAZ with $p < 0.01$. On the last column, $+$ and $-$ represent increase and drop, respectively, as compared to the RAZ baseline.

quality of labeled results but decreases the coverage of the classifier.

We now illustrate how discourse information helps in recognizing the rhetorical function of a sentence through examples. As explained, we expect that a Comparison discourse relation may help to identify CTR label, as it tends to compare others' work with authors' own work. In Example 6.1, the gold standard label for the second sentence is CTR (contrast). Before adding in the discourse features, RAZ incorrectly classifies it into OWN. As the discourse parser recognizes an Implicit Comparison relation between these two sentences and assigns the discourse feature "PS_Comparison_Arg2" to the second sentence, the system now learns that this sentence provides a contrasting rhetorical function. Therefore, the new model is able to correctly assign CTR to this sentence.

The previous example is an instance of an Implicit discourse relation. Some Explicit discourse connectives are captured by the cue word/phrase and n-gram features of RAZ. However, many connectives (such as “when” and “and”) are ambiguous and need the assistance of an external classifier to identify their actual discourse functions (cf. Chapter 4). Moreover, the RAZ features do not capture all connectives, such as “on the other hand” in Example 6.3. RAZ assigns an incorrect label of OWN to Example 6.3. As the discourse parser recognizes this sentence as the Arg2 span of an Explicit Comparison relation, the new model (RAZ+Discourse) is able to correctly classify this as CTR.

(6.3) On the other hand, the fact that these approaches extract co-occurrences without reliability on being verb-complements violates accuracy requirements.
(CMP-LG 9409004)

The above examples illustrate the influence of the Comparison relation on classifying the CTR instances. This is further elaborated in Table 6.6, which shows the top 20 (AZ label, discourse feature) pairs ranked by their mutual information. The first pair on the list, (CTR, PS_Comparison_Arg2), shows that CTR and the discourse feature PS_Comparison_Arg2 have the highest correlation. In the list, we observe that PS_EntRel_Arg1 and PS_EntRel_Arg2 appear very often, and the first two for EntRel are OTH and OWN. In Table 6.2 we see that OWN and OTH are the most common labels in the body, as authors usually spend most of the body text in explaining how they derive their results from others’ and their own work. As entity transition is commonly seen in such explanation, it gives the reason of the high correlation of OWN/OTH and EntRel features.

Comparing Table 6.4 and Table 6.5, we see that BKG and CTR contribute the most to the improvement in terms of F_1 scores: F_1 for BKG and CTR increases 2.79% and 2.53%, respectively. BKG, CTR, and OWN contribute the most to the performance in

CTR	PS_Comparison_Arg2
OTH	PS_EntRel_Arg2
OWN	PS_Contingency_Arg2
OTH	PS_Contingency_Arg2
CTR	SS_Comparison
OWN	PS_EntRel_Arg1
OTH	PS_EntRel_Arg1
BAS	PS_EntRel_Arg1
OTH	PS_Comparison_Arg1
AIM	SS_Contingency
AIM	PS_Expansion_Arg2
TXT	PS_Expansion_Arg2
OWN	SS_Contingency
AIM	PS_EntRel_Arg1
OTH	PS_Contingency_Arg1
AIM	PS_EntRel_Arg2
OWN	PS_Comparison_Arg1
OWN	PS_EntRel_Arg2
TXT	SS_Contingency
CTR	PS_EntRel_Arg2

Table 6.6: A list of top 20 (AZ label, discourse feature) pairs ranked by their mutual information in descending order.

terms of more correctly labeled instances (cf. the diagonals in the two tables). F_1 increases are observed for all labels, except BAS. This is explained by the fact that, among the top 20 pairs ranked by mutual information scores, there is only one pair that incorporates the BAS label, which is "PS_EntRel_Arg1". As "PS_EntRel_Arg1" has higher correlation with OWN and OTH, it does not provide enough information to differentiate BAS from other labels. BAS sentences describe the basis for the work in the paper, and usually are recognized by the appearances of citations, as shown in the following example. As citation features are included in RAZ and discourse features do not further address the problem of citation, it is not surprising that adding discourse features does not help.

(6.4) We present a different method that takes as starting point the back-off scheme of Katz (1987).

(CMP-LG 9405001)

6.3.3 Results for Summarization

The results for the baseline summarization system are shown in the first row in Table 6.7. ROUGE-1 and ROUGE-2 are the unigram and bigram co-occurrence scores, while ROUGE-SU4 measures the skip-unigram and -bigram co-occurrence with a word window of 4. These three metrics have been shown in past publications to have high correlation with humans (Lin and Hovy, 2003). As ROUGE is a recall-based metric, it shows how much of the abstract is covered by the selected sentences.

When we incorporate discourse features into the baseline, the performance is significantly improved, with $p < 0.05$ for ROUGE-1 and ROUGE-SU4 and $p < 0.01$ for ROUGE-2 (cf. Row 2 in Table 6.7). When we look at the sentences in the generated summaries, 26.48% of them are associated with the Contingency relation. As Contingency indicates how one argument causally influences the other, in scientific papers it is used to explain how the authors derive their research work and results from the previous context. Thus it makes sense to use it as an indicator of the salience of the sentence in

	ROUGE-1	ROUGE-2	ROUGE-SU4
Baseline	40.80	10.79	15.56
Baseline+Discourse	42.47*	12.73**	16.98*
Baseline+AZ(gs)	48.19**	16.73**	20.71**
Baseline+RAZ	48.12**	16.76**	20.52**
Baseline+RAZ+Discourse	48.31**	17.08**	20.75**

Table 6.7: Results for different summarization models. The first row shows the baseline performance, while the following four rows show the performance of the combined models. Double (**) and single (*) asterisks indicate that the respective model significantly outperforms the baseline at $p < 0.01$ and $p < 0.05$, respectively. We use a two tailed paired t-test.

summarization.

We also look at whether adding AZ information into summarization improves performance. Teufel and Moens (2002) suggested that argumentative zoning can provide indication for a summarization system of scientific articles to concentrate on the contributions of the article and highlight the differences with previous work. However, to the best of our knowledge, there is no supervised summarization system that utilizes AZ. We include the gold standard AZ label as a feature for each sentence in the baseline system. The results in Row 3 show that AZ information is very useful in our scientific document summarization system: the performance is significantly improved ($p < 0.01$), with 7.39%, 5.94%, and 5.15% increases in ROUGE-1, ROUGE-2, and ROUGE-SU4, respectively.

We then replace the gold standard AZ labels with the predicted labels from the RAZ system. As illustrated in Row 4, the improvement over the baseline is also statistically significant with $p < 0.01$, but slightly lower than that with gold standard labels for ROUGE-1 and ROUGE-SU4. Table 6.8 shows the distribution of AZ labels in the abstracts and generated summaries. Before adding the AZ labels, the distribution in the summaries generated by the baseline is not close to that in the abstracts: the majority label is BKG which does not make sense as background is usually discussed in introduction

	AIM	BAS	BKG	CTR	OTH	OWN	TXT
% in abstract	27.14	2.00	8.57	8.00	6.86	47.43	0.00
% in summary (Baseline)	8.49	2.70	40.15	7.34	21.62	19.31	0.39
% in summary (Baseline+RAZ)	31.25	4.08	11.41	4.08	9.78	19.57	19.84

Table 6.8: Percentages of AZ labels in abstracts and generated summaries.

but not in abstract. As shown in the first row, the focus in abstract is usually OWN and AIM. After adding predicted AZ labels onto the baseline system, the majority class is now changed to AIM, OWN, and TXT, and the distribution is closer to that in the abstracts. The reason for a high TXT percentage is that TXT sentences give a high level overview and textual organization of the work in the paper, and thus is included in the summaries. The following example shows a TXT sentence.

(6.5) In the following section we explain how the probabilities for these various processing stages are combined to select the most likely target word sequence .

(CMP-LG 9408014)

When we add in both predicted AZ labels and discourse information, as shown in the last row of Baseline+RAZ+Discourse, it outperforms both Baseline+Discourse and Baseline+RAZ.

However, discourse information contributes little to Baseline+RAZ, as shown in the last two rows. The improvement of Baseline+RAZ+Discourse over Baseline+RAZ is not statistically significant. This shows that discourse is providing little information beyond that already provided by RAZ.

Recall that we made use of two discourse feature classes in the Baseline+Discourse model: one is the relation count (*i.e.*, how many discourse relations that this sentence is involved in), and the other is “PS + Explicit/Implicit + relation type + argument tag”. To study how different kinds of discourse information contribute to the performance, we conduct an ablation test, in which we eliminate particular information sources from the

	ROUGE-1	ROUGE-2	ROUGE-SU4
Baseline+Discourse	42.47	12.73	16.98
– Relation type	41.84	11.23	16.06
– Argument tag	42.09	12.38	16.79
– Explicit/Implicit	41.71	11.93	16.37
– Relation count	42.72	12.29	16.85

Table 6.9: Summarization performance when ablating away discourse information sources.

features: relation type, argument tag, Explicit/Implicit information, and relation count. The results are shown in Table 6.9, in which the first row shows the Baseline+Discourse model and the next four rows show the results when we eliminate one information source at a time from Baseline+Discourse. When we eliminate Explicit/Implicit information, we do not differentiate whether a relation is Explicit or Implicit. The results show that all four discourse information sources are needed to achieve best performance. The performance on all three ROUGE scores drops when we eliminate any information source. Relation type seems to have the most impact, as when we remove it, performance suffers the most.

6.3.4 An Iterative Model

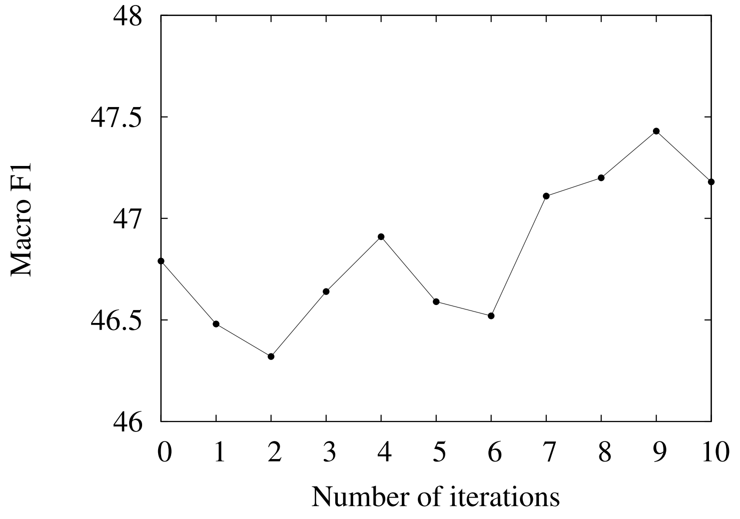
We have shown that argumentative zoning improves summarization of scientific articles. Can we utilize the output from the scientific document summarizer to assist in zoning? If this improves performance, we can view the two tasks as synergistic and perform a type of iterative co-training, by alternatingly utilizing the output from one system as input to the other task.

As RAZ uses binary features in the maximum entropy classifier and the summarization system assigns a score to each sentence, we need to convert this score into a binary feature. We include in RAZ a binary feature to check whether a sentence is selected as a summary sentence by the summarization system. As the model is designed to examine whether RAZ improves summarization and whether summarization improves RAZ, discourse features are not used in the model.

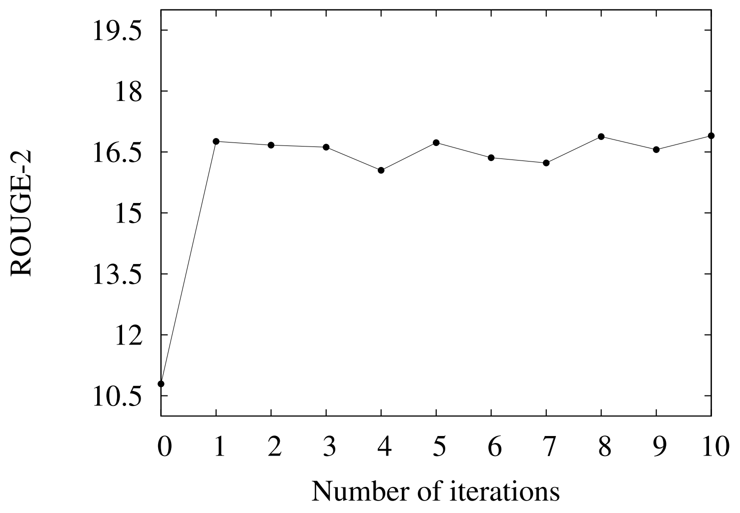
In the iterative model of the RAZ and the summarization systems, the output from a system is sent into the other system. Both systems are then re-trained and re-tested with the updated features. This process is repeated for several iterations. Figure 6.1 shows the results for 10 iterations for the learning process.

Figure 6.1(a) shows a sawtooth performance with an overall trend that is increasing over the 10 iterations. This suggests that when the summarization system is able to provide more accurate input, the performance of RAZ will be improved at the end of 10 iterations. However, the same is not seen in the summarization task (Figure 6.1(b)): here the improvement in the first iteration (which is the same as Row 4 of Table 6.7) is very large, but there is negligible improvement in the remaining nine iterations. This is due to the fact that the improvement of RAZ in Figure 6.1(a) is not significant enough to improve the summarization system.

Figure 6.2 summarizes the results from the experiments that we conducted in this section. It shows how the three NLP tasks – discourse parsing, argumentative zoning, and summarization – affect one another in terms of performance. An arrow pointing from task a to task b means that results from a can be used as features to improve b. It illustrates that features extracted from a discourse parser improve both downstream NLP tasks argumentative zoning and summarization, and output from argumentative zoning and summarization can also improve each other in an iterative manner.



(a) Argumentative zoning: macro F_1 for 10 iterations.



(b) Summarization: ROUGE-2 scores for 10 iterations.

Figure 6.1: Results for the iterative model of argumentative zoning and summarization.

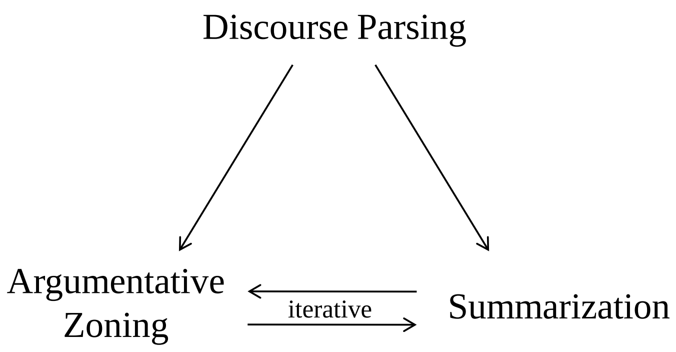


Figure 6.2: Application of discourse parsing in argumentative zoning and summarization. An iterative model for argumentative zoning and summarization.

6.4 Conclusion

In this chapter, we leveraged the information extracted from discourse structures and relations from our discourse parser, and demonstrated that incorporating discourse information significantly improves the performance on two NLP tasks – argumentative zoning and summarization of scholarly papers. This suggests that understanding the discourse structure of a text is needed to improve downstream tasks. We also proposed an iterative model for argumentative zoning and summarization, and demonstrated that both tasks can improve each other in this model. This interestingly shows how these three NLP tasks are interconnected with one another. Chapter 5 and this chapter together validate our hypothesis that **discourse parsing can provide information to model textual coherence and improve user tasks in natural language processing**. In our future work, we plan to apply discourse parsing in multi-document summarization in news domain, and to further show the applicability of discourse parsing in other NLP tasks, including question answering and textual entailment.

Chapter 7

Conclusion

In this thesis, we have designed and implemented a classification system to recognize Implicit discourse relations, and have conducted experiments in the Penn Discourse Treebank and showed significant improvement over the baseline. This classifier has been integrated into an end-to-end discourse parser, which we designed and developed to parse any free text into its discourse structure in the PDTB representation. The parser consists of different components to identify discourse relations and arguments for Explicit and Non-Explicit relation, and to recognize attribution spans. Then, to demonstrate its applicability, we applied this parser in two directions on coherence modeling and downstream NLP applications. We proposed a discourse role matrix to capture the discourse relation transition patterns, and showed that such model can be used to distinguish a coherent text from an incoherent one. Finally, we demonstrated that features extracted from the discourse parser can be leveraged in text summarization and argumentative zoning.

In this chapter, we recap the contributions of our research work. We briefly discuss the discourse parser that we made available to the research community. Finally, we will conclude this thesis with a number of future directions.

7.1 Main Contributions

This thesis makes four main contributions to the research in natural language processing. They include:

1. Classifying Implicit discourse relations. (Chapter 3)
2. An end-to-end discourse parser. (Chapter 4)
3. Modeling and evaluating text coherence. (Chapter 5)
4. Applying discourse parsing in summarization and argumentative zoning. (Chapter 6)

Classifying Implicit Discourse Relations. Implicit discourse relation classification is a harder task compared to its counterpart for Explicit relations, due to the absence of the discourse markers. In Chapter 3, we looked into this problem in the PDTB and proposed a supervised approach to tackle the problem. We implemented a classifier for Implicit discourse relations by leveraging four feature classes: (1) contextual features which check the states of the preceding and following relations, (2) production rules extracted from the constituent parses of the two arguments, (3) corresponding production rules extracted from the dependency trees of the two arguments, and (4) highly correlated word pairs extracted from the argument spans. We conducted feature selection based on mutual information, and adopted the maximum entropy classifier. The experiments conducted in the PDTB showed that the resultant classifier significantly outperforms the baseline system. We performed data analysis in the PDTB and identified four challenges to this task, which include: the ambiguity among the relation types, the need for using inference and a knowledge base, the analysis of the contextual information in understanding the arguments, and the access to world knowledge.

An End-to-End Discourse Parser. We designed a parsing algorithm and implemented it into a fully automatic, data-driven, end-to-end, discourse parser that is able to derive the discourse structure in the PDTB representation for any free text. This parser consists of three steps to identify and label Explicit relations, Non-Explicit relations, and attributions spans. In order to identify Explicit relations, we implemented a connective classifier to differentiate discourse connectives from non-discourse connectives. We also implemented an argument labeler to locate and label the Arg1 and Arg2 spans for Explicit relations. The Implicit classifier from Chapter 3 is adapted and re-trained to classify Non-Explicit relations. We performed a comprehensive evaluation of the parser from both component-wise and error-cascading perspectives. The parser gives an overall system F_1 score of 46.80% for partial matching with gold standard parses, and 38.18% with full automation. We also released the parser online for the benefit of the research community[1]. In Chapter 5 and 6 we showed that the parser improves downstream NLP tasks of coherence modeling, summarization, and argumentative zoning. However, as the overall performance is considered low, we believe that there are still much room for improvement before the parser can be applied in real products.

Modeling and Evaluating Text Coherence. Discourse researchers have shown that textual coherence and discourse structure are closely related to each other. However, there has not been research to empirically validate how automatic discourse parsing influences coherence modeling. Another contribution in our work is in applying discourse parsing in modeling and evaluating text coherence. We motivated the work by showing that coherent texts preferentially follow certain discourse relation patterns, which can be captured by analyzing the output of a discourse parser. We demonstrated that simply using the linear sequence of relation transition will lead to sparse features, and proposed a solution to this problem: a discourse role matrix. We showed that probabilities for relation transitions obtained from this matrix can be utilized in a preference ranking framework (which we used SVM) to differentiate coherent texts from incoherent ones. We compared the discourse-based coherence model with Barzilay and Lapata (2008)'s entity-based

[1]`http://wing.comp.nus.edu.sg/~linzihen/parser/`

model in the same task of differentiating a source (*i.e.*, coherent) text from its permuted (*i.e.*, incoherent) text, and showed significant improvement. To the best of our knowledge, this is the first study in which we showed that the output from an automatic discourse parser helps in coherence modeling.

Summarization and Argumentative Zoning. To further prove the applicability of a discourse parser in the downstream end-user NLP applications, we turned to text summarization and argumentative zoning in the scientific domain. We demonstrated that discourse features extracted from the discourse relations can significantly improve both tasks. Such features include, for example, the number of relations that the current sentence is involved in, and the current sentence is "the Arg2 span of an Implicit Comparison relation". We also proposed an iterative model for summarization and argumentative zoning, and showed that they can improve each other. This work demonstrated that our discourse parser is applicable in end-user tasks, and we believe that once the performance of the parser is improved, we will see its applications in other NLP tasks.

7.2 Future Work

Improvements for the Discourse Parser

As we have discussed in Chapters 3 and 4, there are several places in the discourse parser that we can improve upon. The first is in Implicit relation classification. We discussed four challenges in this task: ambiguity among relations, inference, contextual information, and world knowledge. In our current approach we do not look beyond the relation under consideration except the previous and next relations. Thus we plan to first explore contextual information in depth to look at how context can help better understand the Implicit relations.

The next area of improvement is in the Explicit classifier. Although the performance of the Explicit classifier is much better than that of the Implicit classifier, there is still much room for improvement. For example, we can extract informative features from the two arguments for the Explicit classifier, as the current classifier does not leverage the argument spans in the (C, Arg1, Arg2) tuple. The attribution span labeler only identifies and labels attribution spans, but does not indicate their direction, *i.e.*, whether an attribution is pointing to Arg1, Arg2, or the relation as a whole. The PDTB also provides annotations for the four dimensions for the attributions: their sources, types, scopal polarities, and determinacy. Thus, future steps also include identifying the attribution directions as well as the four dimensions. The current attribution span labeler is built upon two assumptions: (a) attribution spans always appear in clauses, and (b) we only examine clauses that are within the identify discourse relations from the previous steps. We would also like to identify attributions spans that are outside the discourse relations (*i.e.*, not within the Arg1 and Arg2 spans) and are more than or less than a clause. The above work would increase the end-to-end discourse parser's coverage of the entire spectrum of annotations done by the PDTB team.

The PDTB also examined work beyond formal discourse relations. In this work, AltLex relations are combined into Non-Explicit relations and handled together with Implicit, EntRel, and NoRel relations. Prasad et al. (2010b) showed that the set of AltLex expressions is open-ended and it is difficult to handle them by using syntactic patterns. We believe that although the set of AltLex relations is much smaller than that of Implicit, they should be handled separately from Implicit relations. Thus we would also like to look into this issue of open class of AltLex expressions and design a system to identify AltLex relations. EntRel relation models how entity transitions from a sentence to its following sentence. Both Expansion and EntRel are similar in the way that Expansion captures how the discourse of the same entity is expanded. In fact, Pitler et al. (2009) combined both into one class in their experiments. We are also interested in exploring approaches that we can utilize EntRel in other NLP tasks.

Coherence Model and its Application

Our discourse coherence model operates at the sentential level. However, as the PDTB relations connect clauses, sentences, and even multiple sentences, our discourse coherence model should also be extended to operate on clausal and multi-sentential levels as well. That is, instead of having only sentences as rows as shown in Table 5.4, we can have, for example, clauses as rows. The current model also does not capture the discourse hierarchy structure, *i.e.*, whether one relation is embedded within another relation. Another future plan is to look at how hierarchical information can be integrated into the discourse role matrix.

In the discourse role matrix that we proposed, each column of the matrix corresponds to a lexical chain (Morris and Hirst, 1991) for a particular term across the whole text. The key differences between the chains in the matrix and the traditional lexical chains are that our chain nodes' entities are simplified (they share the same stemmed form, instead being connected by WordNet relations), but are further enriched by being typed with discourse relations. One possible future work is to study how such a discourse model can be combined with a lexical chain to improve topic segmentation.

The Automatically Evaluating Summaries of Peers (AESOP) task that is part of the Text Analysis Conference (TAC) is a task to develop better summarization system evaluation[2]. Its purpose is to encourage and promote research of automatic evaluation metrics to evaluate the quality of machine generated summaries, in terms of their (1) content and (2) readability (*i.e.*, linguistic quality). The first is to measure how informative the machine generated summary is, while the second is to evaluate the linguistic qualities of the summary, which also include measuring how coherent this summary is.

Therefore, another future direction is to apply the coherence model that we proposed in Chapter 5 in a summarization evaluation system to measure readability. The

[2]`http://www.nist.gov/tac/2011/Summarization/AESOP.2011.guidelines.html`

current coherence model is applied in a pair-wise preference ranking approach (which we used SVM), because the initial idea was to differentiate between a coherent and incoherent texts. To adopt this model in a summarization evaluation metric to rank a set (instead of a pair) of summaries, we thus need to figure out how to generalize the pair-wise ranking model into list-wise. We can also frame the coherence model in a linear regression model to give a real value of coherence for each text.

References

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: an entity-based approach. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 141–148, Morristown, NJ, USA.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34:1–34, March.

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *Proceedings of the Human Language Technology Conference / North American Chapter of the Association for Computational Linguistics Annual Meeting 2004 (HLT-NAACL 2004)*, pages 113–120, Boston, Massachusetts, USA, May.

Gillian Brown and George Yule. 1983. *Discourse Analysis*. Cambridge University Press.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311.

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*, Morristown, NJ, USA.

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.

Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Rashmi Prasad, Aravind Joshi, and Bonnie Webber. 2005. Attribution and the (non)-alignment of syntactic and discourse arguments of connectives. In *Proceedings of the ACL Workshop on Frontiers in Corpus Annotation II: Pie in the Sky*, Ann Arbor, MI, USA.

David 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 (ACL-IJCNLP 2009)*, Singapore.

H. P. Edmundson. 1969. New methods in automatic extracting. *JACM: Journal of the ACM*, 16.

Micha Elsner and Eugene Charniak. 2008. Coreference-inspired coherence modeling. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers (HLT 2008)*, pages 41–44, Stroudsburg, PA, USA.

Micha Elsner and Eugene Charniak. 2011. Extending the entity grid with entity-specific features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers (ACL-HLT 2011)*, pages 125–129, Stroudsburg, PA, USA.

Micha Elsner, Joseph Austerweil, and Eugene Charniak. 2007. A unified local and global model for discourse coherence. In *Proceedings of the Conference on Human Language Technology and North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2007)*, Rochester, New York, USA, April.

Robert Elwell and Jason Baldridge. 2008. Discourse connective argument identification with connective specific rankers. In *Proceedings of the IEEE International Conference on Semantic Computing (ICSC 2010)*, Washington, DC, USA.

Günes Erkan and Dragomir R. Radev. 2004a. LexRank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479.

Günes Erkan and Dragomir R. Radev. 2004b. The University of Michigan at DUC 2004.

In *Proceedings of the Document Understanding Conference 2004 (DUC 2004)*, Boston, Massachusetts, USA, May.

Katherine Forbes, Eleni Miltsakaki, Rashmi Prasad, Anoop Sarkar, Aravind Joshi, and Bonnie Webber. 2003. D-LTAG system: Discourse parsing with a lexicalized tree-adjoining grammar. *Journal of Logic, Language and Information*, 12(3):261–279.

Barbara J. Grosz and Candace L. Sidner. 1986. Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3):175–204, July.

Barbara J. Grosz, Scott Weinstein, and Aravind K. Joshi. 1995. Centering: a framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225, June.

Michael A.K Halliday and Ruqaiya Hasan. 1976. *Cohesion in English*. Longman, London.

Marti A. Hearst. 1997. TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64.

Ryuichiro Higashinaka and Hideki Isozaki. 2008. Automatically acquiring causal expression patterns from relation-annotated corpora to improve question answering for why-questions. *ACM Transactions on Asian Language Information Processing (TALIP)*, 7(2):6.

Cong Duy Vu Hoang and Min-Yen Kan. 2010. Towards automated related work summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 427–435, Stroudsburg, PA, USA.

Jerry R. Hobbs. 1985. On the coherence and structure of discourse.

Le Thanh Huong, Geetha Abeysinghe, and Christian Huyck. 2004. Generating discourse structures for written texts. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, Morristown, NJ, USA.

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)*. Sydney, Australia.

Thorsten Joachims. 1999. Making large-scale support vector machine learning practical. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods – Support Vector Learning*, pages 169–184. MIT Press, Cambridge, MA, USA.

Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing: An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics*. Prentice-Hall.

Nikiforos Karamanis. 2007. Supplementing entity coherence with local rhetorical relations for information ordering. *Journal of Logic, Language and Information*, 16:445–464, October.

Alistair Knott. 1996. *A Data-Driven Methodology for Motivating a Set of Coherence Relations*. Ph.D. thesis, Department of Artificial Intelligence, University of Edinburgh.

Esther König. 1994. Syntactic-head-driven generation. In *Proceedings of the 15th Conference on Computational Linguistics (COLING 1994)*, pages 475–481, Stroudsburg, PA, USA.

Mirella Lapata and Regina Barzilay. 2005. Automatic evaluation of text coherence: Models and representations. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI 2005)*, Edinburgh, Scotland, UK.

Mirella Lapata. 2003. Probabilistic text structuring: experiments with sentence ordering. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, Morristown, NJ, USA.

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.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL 2003)*, pages 71–78, Morristown, NJ, USA.

Ziheng Lin, Tat-Seng Chua, Min-Yen Kan, Wee Sun Lee, Long Qiu, and Shiren Ye. 2007. NUS at DUC 2007: Using evolutionary models of text. In *Proceedings of the Document Understanding Conference 2007 (DUC 2007)*, Rochester, NY, USA, April.

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 (EMNLP 2009)*, Singapore.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2010. A PDTB-styled end-to-end discourse parser. Technical Report TRB8/10, School of Computing, National University of Singapore, August.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2011. Automatically evaluating text coherence using discourse relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*, pages 997–1006, Portland, Oregon, USA, June.

Ziheng Lin, Chang Liu, Hwee Tou Ng, and Min-Yen Kan. 2012. Combining coherence models and machine translation evaluation metrics for summarization evaluation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, Jeju, Korea, July.

Annie Louis and Ani Nenkova. 2012. A coherence model based on syntactic patterns. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural*

Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012), Jeju, Korea, July.

Annie Louis, Aravind Joshi, and Ani Nenkova. 2010. Discourse indicators for content selection in summarization. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL 2010)*, pages 147–156, Stroudsburg, PA, USA.

Hans Peter Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165.

Inderjeet Mani and Mark T. Maybury. 1999. *Advances in Automatic Text Summarization*. The MIT Press, Cambridge, Masschussets.

Inderjeet Mani. 2001. *Automatic Summarization*. John Benjamins Publishing Company, Amsterdam.

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)*, Morristown, NJ, USA.

Daniel Marcu. 1996. Distinguishing between coherent and incoherent texts. In *The Proceedings of the Student Conference on Computational Linguistics in Montreal*, pages 136–143.

Daniel Marcu. 1997. *The Rhetorical Parsing, Summarization, and Generation of Natural Language Texts*. Ph.D. thesis, University of Toronto.

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*, Montreal, Canada.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Stephen Merity, Tara Murphy, and James R. Curran. 2009. Accurate argumentative zoning with maximum entropy models. In *Proceedings of the ACL-IJCNLP 2009 Workshop on Text and Citation Analysis for Scholarly Digital Libraries (NLPIR4DL 2009)*, pages 19–26, Stroudsburg, PA, USA.

Eleni Miltsakaki, Rashmi Prasad, Aravind Joshi, and Bonnie Webber. 2004. The Penn Discourse Treebank. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004)*, Lisbon, Portugal.

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.

Saif Mohammad, Bonnie Dorr, Melissa Egan, Ahmed Hassan, Pradeep Muthukrishan, Vahed Qazvinian, Dragomir Radev, and David Zajic. 2009. Using citations to generate surveys of scientific paradigms. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2009)*, NAACL '09, pages 584–592, Stroudsburg, PA, USA.

Jane Morris and Graeme Hirst. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17:21–48, March.

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.

Kenji Ono, Kazuo Sumita, and Seiji Miike. 1994. Abstract generation based on rhetorical structure extraction. In *Proceedings of the 15th Conference on Computational Linguistics (COLING 1994)*, pages 344–348, Stroudsburg, PA, USA.

Martha. Palmer, Daniel. Gildea, and Paul. Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

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

Emily Pitler and Ani Nenkova. 2009. Using syntax to disambiguate explicit discourse connectives in text. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, Singapore.

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) Short Papers*, Manchester, UK.

Emily Pitler, Annie Louis, and Ani Nenkova. 2009. Automatic sense prediction for implicit discourse relations in text. 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 (ACL-IJCNLP 2009)*, Singapore.

Livia Polanyi and Remko Scha. 1984. A syntactic approach to discourse semantics. In *Proceedings of the 10th International Conference on Computational Linguistics (COLING 1984)*, pages 413–419. Association for Computational Linguistics.

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)*.

Rashmi Prasad, Aravind Joshi, and Bonnie Webber. 2010a. Exploiting scope for shallow discourse parsing. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC-2010)*, pages 2076–2083, Valletta, Malta, May.

Rashmi Prasad, Aravind Joshi, and Bonnie Webber. 2010b. Realization of discourse relations by other means: alternative lexicalizations. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters (COLING 2010)*, pages 1023–1031, Stroudsburg, PA, USA.

Vahed Qazvinian and Dragomir R. Radev. 2008. Scientific paper summarization using citation summary networks. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, pages 689–696, Stroudsburg, PA, USA.

Dragomir R. Radev, Hongyan Jing, Malgorzata Stys, and Daniel Tam. 2004. Centroid-based summarization of multiple documents. *Information Processing and Management*, 40:919–938, November.

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)*, New York, USA.

Remko Scha and Livia Polanyi. 1988. An augmented context free grammar for discourse. In *Proceedings of the 12th Conference on Computational Linguistics*, pages 573–577. Association for Computational Linguistics.

Peter Rossen Skadhauge and Daniel Hardt. 2005. Syntactic identification of attribution in the RST Treebank. In *Proceedings of the Recent Advances in Natural Language Processing (RANLP 2005)*, Borovets, Bulgaria.

Radu Soricut and Daniel Marcu. 2003. Sentence level discourse parsing using syntactic and lexical information. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2003)*, Edmonton, Canada.

Radu Soricut and Daniel Marcu. 2006. Discourse generation using utility-trained coherence models. In *Proceedings of the COLING/ACL Main Conference Poster Sessions*, pages 803–810, Morristown, NJ, USA.

Karen Sparck-Jones. 1998. Automatic summarising: factors and direction. In Inderjeet Mani and Mark T. Maybury, editors, *Advances in Automatic Text Summarization*. MIT Press.

Caroline Sporleder and Alex Lascarides. 2008. Using automatically labelled examples to classify rhetorical relations: An assessment. *Natural Language Engineering*, 14(3):369–416, July.

Simone Teufel and Min-Yen Kan. 2011. Robust argumentative zoning for sensemaking in scholarly documents. Springer Hot Topics.

Simone Teufel and Marc Moens. 2002. Summarizing scientific articles: experiments with relevance and rhetorical status. *Computational Linguistics*, 28(4):409–445.

Simone Teufel. 1999. *Argumentative Zoning: Information Extraction from Scientific Text*. Ph.D. thesis, School of Cognitive Science, University of Edinburgh, UK.

Vasudeva Varma, Praveen Bysani, Kranthi Reddy, Vijay Bharat, Santosh GSK, Karuna Kumar, Sudheer Kovelamudi, Kiran Kumar N, and Nitin Maganti. 2009. IIIT hyderabad at TAC 2009. In *Proceedings of Test Analysis Conference 2009 (TAC 2009)*.

WenTing Wang, Jian Su, and Chew Lim Tan. 2010. Kernel based discourse relation recognition with temporal ordering information. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, Uppsala, Sweden, July.

Bonnie Lynn Webber and Aravind K. Joshi. 1998. Anchoring a lexicalized tree-adjoining grammar for discourse. In *Coling/ACL Workshop on Discourse Relations and Discourse Markers*, pages 86–92.

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

Ben Wellner and James Pustejovsky. 2007. Automatically identifying the arguments of discourse connectives. In *Proceedings of the 2007 Joint Conference on Empirical*

Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007), Prague, Czech Republic.

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.

Ben Wellner. 2009. *Sequence Models and Ranking Methods for Discourse Parsing*. Ph.D. thesis, Brandeis University.

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)*, Morristown, NJ, USA.

Shiren Ye, Long Qiu, Tat-Seng Chua, and Min-Yen Kan. 2005. NUS at DUC 2005: Understanding documents via concepts links. In *Proceedings of DUC 2005*.

Appendix A

An Example for Discourse Parser

A.1 Features for the Classifiers in Step 1

Here are features extracted from the Explicit relation in Example A.1 for the classifiers in Step 1 of the parser. The constituent parse of Example A.1 is shown in Figure A.1.

(A.1) *Orders for durable goods were up 0.2% to $127.03 billion* <u>after</u> **rising 3.9% the month before.**

(Temporal.Asynchronous - wsj_0036)

A.1.1 Features for the Connective Classifier

C POS = IN

$prev$ + C = billion after

$prev$ POS = CD

$prev$ POS + C POS = CD IN

C + $next$ = after rising

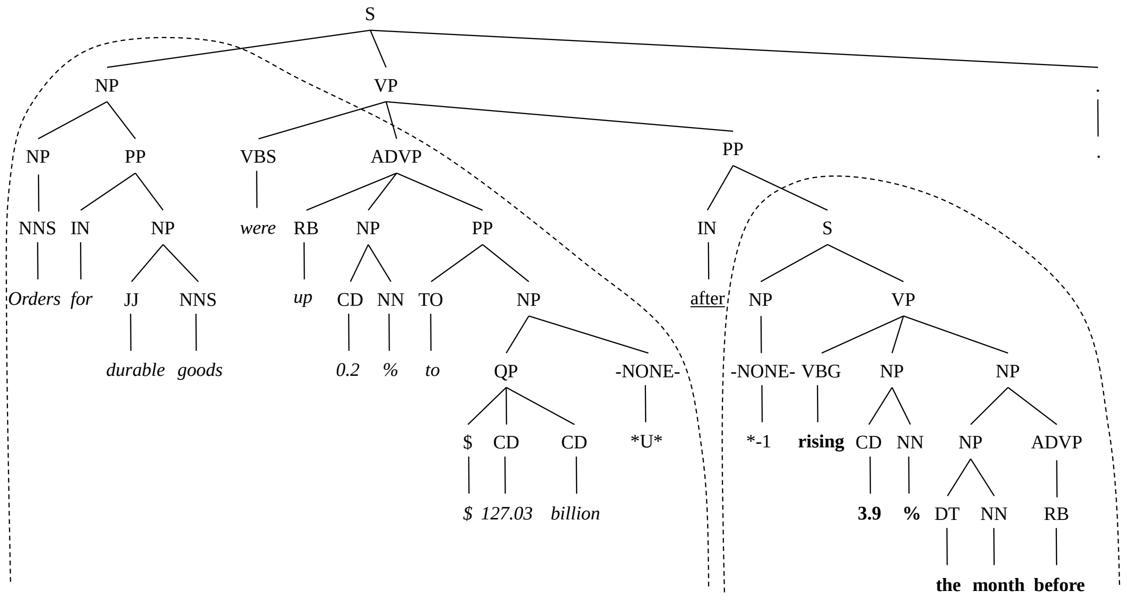


Figure A.1: The constituent parse tree for Example A.1.

$next$ POS = VBG

C POS + $next$ POS = IN VBG

path of C's parent $\rightarrow$ root = IN $\uparrow$ PP $\uparrow$ VP $\uparrow$ S

compressed path of C's parent $\rightarrow$ root = IN $\uparrow$ PP $\uparrow$ VP $\uparrow$ S

A.1.2 Features for the Argument Position Classifier

C string = after

C POS = IN

$prev_1$ = billion

$prev_1$ POS = CD

$prev_1$ + C = billion after

$prev_1$ POS + C POS = CD IN

$prev_2$ = 127.03

$prev_2$ POS = CD

$prev_2$ + C = 127.03 after

$prev_2$ POS + C POS = CD IN

A.1.3 Features for the Argument Node Identifier

In the parser tree (Figure A.1) for Example A.1, we need to identify the Arg1 and Arg2 nodes from the 18 internal nodes (except POS nodes). Here we list out the features used to label the S node that covers the Arg2 span.

C string = after

C's syntactic category = subordinating

numbers of left siblings of C = 0

numbers of right siblings of C = 1

the path P of C's parent $\rightarrow$ N = IN $\uparrow$ PP $\downarrow$ S

the relative position of N to C = right

A.1.4 Features for the Explicit Classifier

C string = after

C's POS = IN

C + $prev$ = billion after

A.2 Features for the Attribution Span Labeler in Step 3

The following shows features extracted from Example A.2 for the attribution span labeler. The $curr$ clause under consideration and its previous and next clauses are:

$curr$ clause = declared San . . . game two.

$prev$ clause = . . . averages,”

$next$ clause = “I’d . . .

(A.2) ... averages,” $\boxed{\text{declared San Francisco batting coach Dusty Baker after game two.}}$ “I’d ...

lowercased verb in $curr$ = declared

lemmatized verb in $curr$ = declare

the first term of $curr$ = declared

the last term of $curr$ = .

the last term of $prev$ = ”

the first term of $next$ = “

the last term of $prev$ + the first term of $curr$ = ” declared

the last term of $curr$ + the first term of $next$ = . “

the position of $curr$ in the sentence = middle

VP $\rightarrow$ VBD S

VBD $\rightarrow$ declared

NP $\rightarrow$ NNP NNP NN NN NNP NNP

NNP $\rightarrow$ San

NNP $\rightarrow$ Francisco

NN $\rightarrow$ batting

NN $\rightarrow$ coach

NNP $\rightarrow$ Dusty

NNP $\rightarrow$ Baker

PP $\rightarrow$ IN NP

IN $\rightarrow$ after

NP $\rightarrow$ NN CD

NN $\rightarrow$ game

CD $\rightarrow$ two