RST-Style Discourse Parsing and Its Applications in Discourse Analysis

by

Vanessa Wei Feng

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Computer Science
University of Toronto

Abstract

RST-Style Discourse Parsing
and Its Applications in Discourse Analysis

Vanessa Wei Feng
Doctor of Philosophy
Graduate Department of Computer Science
University of Toronto
2015

Discourse parsing is the task of identifying the relatedness and the particular discourse relations among various discourse units in a text. In particular, among various theoretical frameworks of discourse parsing, I am interested in Rhetorical Structure Theory (RST). I hypothesize that, given its ultimate success, discourse parsing can provide a general solution for use in many downstream applications.

This thesis is composed of two major parts. First, I overview my work on discourse segmentation and discourse tree-building, which are the two primary components of RST-style discourse parsing. Evaluated on the RST Discourse Treebank (RST-DT), both of my discourse segmenter and tree-builder achieve the state-of-the-art performance.

Later, I discuss the application of discourse relations to some specific tasks in the analysis of discourse, including the evaluation of coherence, the identification of authorship, and the detection of deception. In particular, I propose to use a set of application-neutral features, which are derived from the discourse relations extracted by my discourse parser, and compare the performance of these application-neutral features against the classic application-specific approaches to each of these tasks. On the first two tasks, experimental results show that discourse relation features by themselves often perform as well as those classic application-specific features, and the combination of these two kinds of features usually yields further improvement. These results provide strong evidence for my hypothesis that discourse parsing is able to pro-

vide a general solution for the analysis of discourse. However, we failed to observe a similar effectiveness of discourse parsing on the third task, the detection of deception. I postulate that this might be due to several confounding factors of the task itself.

Acknowledgements

I am sincerely grateful to my supervisor Professor Graeme Hirst at the Department of Computer Science, University of Toronto. It has been my great pleasure to work with him since five years ago, when I began my life in Toronto as a master's student under his supervision and then proceeded as a Ph.D. student in 2011. Graeme is such a gentleman with great sense of humor. He never fails to provide me with insightful thoughts, recommendations, and inspiration throughout my research. He is a true mentor, who always shows respect and kindness to his students. Moreover, without his patient and careful editing of all my research papers, I would still be a novice in scientific writing, struggling for each presentation that I need to give.

I would like to thank my committee members, Professor Suzanne Stevenson and Professor Gerald Penn, for their helpful advice and criticism while I revolved the directions of my research. Although I normally work exclusively with my supervisor on my research projects, the regular checkpoint meetings with my committee members offered great opportunities to learn interesting and useful ideas from different perspectives. I am also grateful to Professor Michael Strube from HITS gGmbH, Germany, who was very kind to agree to serve as my external examiner; to Professor Frank Rudzicz from the CL group and to Professor Jack Chambers from the Linguistics Department, who agreed to be the new committee member for my final thesis defence. Without their valuable suggestions and insightful criticism, my final thesis work would have been of much less quality.

I am also indebted to my parents Huaying Sun and Hanjie Feng, who stayed in my hometown, Shanghai, China, when I was pursuing my Ph.D. studies in a foreign country. Without their support, I would have a much harder time in Toronto.

I would like to express my gratitude to all my colleagues in the CL group at University of Toronto, which is such an amazing group of talents, and all my friends in Toronto, who spent their time hanging out with me, preventing me from becoming a dull Ph.D. nerd.

Finally, I want to thank the Natural Sciences and Engineering Research Council of Canada and the University of Toronto for their financial support for my research.

Contents

1 Introduction **1**

1.1 Rhetorical Structure Theory 3

1.1.1 Elementary Discourse Units 4

1.1.2 Inventory of Discourse Relations 5

1.1.3 An Example of RST-Style Discourse Tree Representation 5

1.1.4 RST-Style Discourse Parsing Pipeline 8

1.1.5 Issues with RST and RST-DT 9

1.2 The Penn Discourse Treebank and PDTB-Style Discourse Parsing 11

1.3 Differences Between the Two Discourse Frameworks 13

I Discourse Parsing **16**

2 Discourse Segmentation **17**

2.1 Previous Work 17

2.2 Methodology 19

2.3 Features 21

2.4 Comparison with Other Models 22

2.5 Error Propagation to Discourse Parsing 25

2.6 Feature Analysis 27

2.6.1 Feature Ablation across Different Frameworks 27

2.6.2 Error Analysis 30

2.7 Conclusion and Future Work 32

3 Discourse Tree-Building and Its Evaluation 33

3.1 Evaluation of Discourse Parse Trees 33

3.1.1 Marcu's Constituent Precision and Recall 34

3.1.2 Example 35

3.2 Tree-Building Strategies 37

3.2.1 Greedy Tree-Building 37

3.2.2 Non-Greedy Tree-Building 39

3.2.2.1 Intra-Sentential Parsing Model 40

3.2.2.2 Multi-Sentential Parsing Model 41

4 Greedy Discourse Tree-Building by Rich Linguistic Features 43

4.1 Method 44

4.1.1 Raw Instance Extraction 44

4.1.2 Feature Extraction 45

4.1.3 Feature Selection 47

4.2 Experiments 47

4.2.1 Structure Classification 49

4.2.2 Relation Classification 52

4.3 Conclusion 53

5 A Linear-Time Bottom-up Discourse Parser with Constraints and Post-Editing 55

5.1 Introduction 55

5.2 Overall Work Flow 56

5.3 Bottom-up Tree-Building 57

5.3.1 Structure Models 59

5.3.2 Relation Models 59

5.4 Post-Editing 61

5.4.1 Linear Time Complexity . . . 63
5.4.2 Intra-Sentential Parsing . . . 63
5.4.3 Multi-Sentential Parsing . . . 64
5.5 Features . . . 64
5.6 Experiments . . . 66
5.7 Results and Discussion . . . 66
5.7.1 Parsing Accuracy . . . 66
5.7.2 Parsing Efficiency . . . 68
5.8 Conclusion . . . 69

II Applications of Discourse Parsing 72

6 The Evaluation of Coherence 73

6.1 Introduction . . . 73
6.1.1 The Entity-based Local Coherence Model . . . 74
6.1.2 Evaluation Tasks . . . 75
6.1.3 Extensions . . . 76
6.2 Extending the Entity-based Coherence Model with Multiple Ranks . . . 77
6.2.1 Experimental Design . . . 78
6.2.1.1 Sentence Ordering . . . 78
6.2.1.2 Summary Coherence Rating . . . 79
6.2.2 Ordering Metrics . . . 79
6.2.3 Experiment 1: Sentence Ordering . . . 81
6.2.3.1 Rank Assignment . . . 81
6.2.3.2 Entity Extraction . . . 82
6.2.3.3 Permutation Generation . . . 82
6.2.3.4 Results . . . 83
6.2.3.5 Conclusions for Sentence Ordering . . . 86

6.2.4 Experiment 2: Summary Coherence Rating 87
6.2.4.1 Results 88
6.2.5 Conclusion 90
6.3 Using Discourse Relations for the Evaluation of Coherence 90
6.3.1 Discourse Role Matrix and Discourse Role Transitions 91
6.3.1.1 Entity-based Feature Encoding 94
6.3.1.2 PDTB-Style Feature Encoding 94
6.3.1.3 Full RST-Style Feature Encoding 95
6.3.1.4 Shallow RST-Style Feature Encoding 97
6.3.2 Experiments 98
6.3.2.1 Sentence Ordering 98
6.3.2.2 Essay Scoring 100
6.3.3 Results 100
6.3.4 Conclusion 102
6.4 Summary of This Chapter 103

7 The Identification of Authorship **105**
7.1 Introduction 105
7.1.1 Authorship Attribution 105
7.1.1.1 Lexical Features 106
7.1.1.2 Character Features 106
7.1.1.3 Syntactic Features 106
7.1.2 Authorship Verification 107
7.1.2.1 Unmasking 107
7.1.2.2 Meta-Learning 108
7.2 Local Coherence Patterns for Authorship Attribution 109
7.2.1 Local Transitions as Features for Authorship Attribution 110
7.2.2 Data 111

7.2.3 Method . . . 113
7.2.4 Results . . . 114
7.2.4.1 Pairwise Classification . . . 114
7.2.4.2 One-versus-Others Classification . . . 117
7.2.5 Discussion . . . 117
7.2.6 Conclusion . . . 119
7.3 Using Discourse Relations for the Identification of Authorship . . . 121
7.3.1 General Feature Encoding by Discourse Role Transitions . . . 121
7.3.2 Discourse Relations for Authorship Attribution . . . 122
7.3.2.1 Chunk-based Evaluation . . . 124
7.3.2.2 Book-based Evaluation . . . 124
7.3.3 Discourse Relations for Authorship Verification . . . 126
7.3.3.1 Experiments . . . 127
7.3.4 Conclusion . . . 131
7.4 Summary of This Chapter . . . 132

8 The Detection of Deception 134

8.1 Introduction . . . 134
8.1.1 Unsupervised Approaches . . . 135
8.1.2 Supervised Approaches . . . 135
8.1.2.1 op_spam_v1.3 Dataset . . . 136
8.1.2.2 The op_spam_v1.4 Dataset . . . 138
8.1.2.3 Li et al.'s Cross-Domain Dataset . . . 139
8.2 Using Discourse Relations for the Detection of Deception . . . 140
8.2.1 Previous Work: Detecting Deception using Distributions of Discourse Relations . . . 141
8.2.2 A Refined Approach . . . 142
8.2.3 Data . . . 142

8.2.4 Features . . . 143
8.2.5 Results . . . 143
8.2.6 Discussion . . . 145
8.2.6.1 Nature of the Dataset . . . 145
8.2.6.2 Unreliability of Automatic Discourse Parser . . . 146
8.2.6.3 Wrong Intuition . . . 147
8.2.7 Conclusion . . . 149
8.3 Summary of This Chapter . . . 149

III Summary **154**

9 Conclusion and Future Work **155**
9.1 Future Work for Discourse Parsing . . . 156
9.1.1 On the Local Level: Tackling Implicit Discourse Relations . . . 157
9.1.2 On the Global Level: Better Parsing Algorithms . . . 158
9.1.3 Domain Adaption . . . 159
9.2 More Potential Applications . . . 160
9.2.1 Machine Translation . . . 160
9.2.2 Anti-Plagiarism . . . 160
9.2.3 Detecting Stylistic Deception . . . 161

List of Tables

1.1 Organization of the relation types in the RST Discourse Treebank. 6

1.2 The 41 distinct relation classes in the RST Discourse Treebank. 7

1.3 Definition of the CONDITION relation class. 7

2.1 Characteristics of the training and the test set in RST-DT. 22

2.2 Performance of our two-pass segmentation model on the *B* class. 24

2.3 Performance of our two-pass segmentation model on the *B* and *C* classes. . . . 24

2.4 The result of discourse parsing using different segmentation. 25

2.5 The effect of feature ablation across different segmentation frameworks. 29

2.6 Comparisons of error between our CRF-based segmentation models with different feature settings. 31

3.1 Computing constituents accuracies under various evaluation conditions for the example in Figure 3.1. 37

4.1 Number of training and testing instances used in *Structure classification*. 48

4.2 *Structure* classification performance of the instance-level evaluation. 51

4.3 *Relation* classification performance of the instance-level evaluation. 52

5.1 Performance of text-level discourse parsing by different models, using gold-standard EDU segmentation. 67

5.2 Characteristics of the 38 documents in the test set of RST-DT. 69

5.3 The parsing time for the 38 documents in the test set of RST-DT. 69

6.1 The entity grid for the example text with three sentences and eighteen entities. . 75
6.2 Accuracies of extending the standard entity-based coherence model with multiple ranks using **Coreference**+ option. 85
6.3 Accuracies of extending the standard entity-based coherence model with multiple ranks using **Coreference**− option. 86
6.4 Accuracies of extending the standard entity-based coherence model with multiple ranks in summary rating. 89
6.5 A fragment of PDTB-style discourse role matrix. 93
6.6 A fragment of the full RST-style discourse role matrix. 97
6.7 The characteristics of the source texts and the permutations in the WSJ dataset. 99
6.8 Accuracy of various models on the two evaluation tasks. 101

7.1 The list of authors and their works used in our experiments. 112
7.2 The list of stylometric features. 113
7.3 Accuracies of pairwise authorship attribution experiments. 115
7.4 Aggregated accuracies of pairwise authorship attribution experiments. 116
7.5 F_1 scores of one-class authorship attribution experiments. 118
7.6 Aggregated F_1 scores of one-class authorship attribution experiments. 119
7.7 The data used in our authorship experiments. 123
7.8 The chunk-based performance of pairwise authorship classification. 125
7.9 The book-based performance of pairwise authorship classification. 126
7.10 Performance of authorship verification using words as features in unmasking. . 128
7.11 Performance of authorship verification using discourse roles as features in unmasking. 129
7.12 Performance of authorship verification using words and discourse roles as features in unmasking. 130
7.13 The best performance of each feature set in building the base classifier. 131

8.1 Statistics of Li et al.'s (2014a) cross-domain dataset. 140
8.2 Statistics of the dataset used in our experiments. 143
8.3 Classification performance of various models on reviews of each domain. . . . 144
8.4 Comparison between the classification performance of our DIS features and Rubin and Vashchilko's DIS^{RV} features. 148

List of Figures

1.1 An example text fragment composed of four EDUs, and its RST discourse tree representation. 6

1.2 An example text fragment composed of three EDUs, where e_2 is an embedded EDU. 11

2.1 An example of a sentence with three EDUs and the label sequence for each token in the sentence. 18

2.2 Our segmentation model in the form of a linear-chain CRF. 20

2.3 Our segmentation model with no pairing features. 27

2.4 Our segmentation model in the framework of independent binary classification. 29

2.5 Example sentences where the full segmentation model is correct while the weaker model makes mistakes. 31

3.1 The gold-standard discourse parse tree T_g vs.the automatically generated discourse parse tree T_a. 36

3.2 Joty et al.'s intra- and multi-sentential Condition Random Fields. 41

5.1 The work flow of our proposed discourse parser. 57

5.2 Intra-sentential structure model M^{struct}_{intra}. 58

5.3 Multi-sentential structure model M^{struct}_{multi}. 59

5.4 Intra-sentential relation model M^{rel}_{intra}. 60

5.5 Multi-sentential relation model M^{rel}_{multi}. 60

6.1 An example text fragment composed of three sentences, and its PDTB-style discourse relations. 92
6.2 An example text fragment composed of seven EDUs, and its RST discourse tree representation. 96

Chapter 1

Introduction

No unit of a well-written text is completely isolated; interpretation requires understanding the relation between the unit and the context. Most rhetorical theories assume a hierarchical structure of discourse, where several small units of texts are related to each other to form a larger unit, which can then be related to other units. From this perspective, building the hierarchical discourse structure for a given text is similar to syntactic parsing, whose purpose is to build a hierarchical structure of a given sentence with respect to the grammatical relations among its text units. Therefore, discovering the hierarchical discourse relations in the text is termed "discourse parsing".

My ultimate hypothesis in this thesis is that discourse parsing can be successfully done automatically with sufficiently high accuracy, and, given its success, discourse parsing would be able to provide a **general** solution to a variety of problems in the analysis of discourse structures. In this thesis, in order to evaluate my hypothesis, I will first present our work on developing an automatic discourse parser and compare its performance against human judgment. Moreover, based on our parser, I will apply discourse parsing on three particular applications of discourse analysis, and observe how features derived from discourse parsing affect those applications.

The generality of discourse parsing is two-fold: Firstly, it can work on different levels

of granularity — from sentences to paragraphs, and finally the whole document. Secondly, discourse parsing aims to discover not only the relatedness of two given text units, e.g., whether they belong to the same subtopic or not, but also the exact coherence relation between them, e.g., CONTRAST, CAUSAL, and EXPLANATION, which can, but normally does not have to, depend on any specific target application. Therefore, discourse parsing is able to provide rich information about the content and the discourse structure of the text, which is clearly a powerful tool for many applications in the analysis of discourse.

In this chapter, I will first introduce Rhetorical Structure Theory, one of the most widely accepted frameworks for discourse analysis. In addition, I will also briefly introduce the Penn Discourse Treebank, a corpus developed in accordance with another popular discourse framework, and its related work, to shed some light on the discussion of discourse analysis from other theories and philosophies.

The thesis is organized as the following. In Part I, I will discuss the two major tasks in RST-style discourse parsing, namely, discourse segmentation and discourse tree-building, and the related work conducted on these two tasks. By the end of Chapter 3, all the necessary components of an RST-style discourse parser will have been presented. In Part II of this thesis, we will see several specific applications in discourse analysis, on which I will evaluate my ultimate hypothesis of the general usefulness of discourse parsing. Those applications include the evaluation of coherence (Chapter 6), the identification of authorship (Chapter 7), and the detection of deception (Chapter 8). I will first describe the application-specific approaches to each of these problems, which are well-established and classic solutions to each specific problem. Afterwards, I will proceed to discuss how information derived from our application-neutral discourse parser can be incorporated into each of these problems and enhance the overall performance, and therefore provide evidence to support the postulated generality of discourse parsing.

1.1 Rhetorical Structure Theory

Rhetorical Structure Theory (RST) (Mann and Thompson, 1988) is one of the most widely accepted frameworks for discourse analysis, and was adopted in the pioneering work of discourse parsing by Marcu (1997). In the framework of RST, a coherent text, or a fairly independent text fragment, can be represented as a discourse tree. In an RST-style discourse tree, the leaf nodes are non-overlapping text spans called *elementary discourse units* (EDUs) — these are the minimal text units of discourse trees (see Section 1.1.1) — and internal nodes are the concatenation of continuous EDUs. Adjacent nodes are related through particular discourse relations (see Section 1.1.2 for detail) to form a discourse subtree, which can then be related to other adjacent nodes in the tree structure. In this way, the hierarchical tree structure is established.

As discussed in length by Taboada and Mann (2006), in its original proposal, RST was designed as an open system, allowing flexibility for researchers working on different domains and applications. There are only a few fixed parts enforced in the original design of RST, including dividing a text into a set of non-overlapping discourse units and the tightness between discourse relations and text coherence. Therefore, in order to proceed with introducing the fine detail of the theory, below, I will make connection to a particular annotation scheme and its resulting corpus, the RST Discourse Treebank (RST-DT), and focus on the corresponding definitions as provided by the annotation guidance in this corpus.

The RST Discourse Treebank (RST-DT) (Carlson et al., 2001), is a corpus annotated in the framework of RST, published by the Linguistic Data Consortium (LDC) with catalog number LDC2002T07 and ISBN 1-58563-223-6[1]. It consists of 385 documents (347 for training and 38 for testing) from the *Wall Street Journal*. RST-DT has been widely used as a standard benchmark for research in RST-style discourse parsing, as it provides a systematic guideline in defining several intuition-based concepts in the original development of RST by Mann and Thompson, including the definitions of EDUs and several discourse relations. Throughout this thesis, the term *RST-style discourse parsing* will refer to the specific type of discourse parsing

[1] `https://catalog.ldc.upenn.edu/LDC2002T07`.

in accordance with the annotation framework in RST-DT.

1.1.1 Elementary Discourse Units

As stated by Mann and Thompson (1988, p. 244), RST provides a general way to describe the relations among clauses in a text, whether or not they are grammatically or lexically signalled. Therefore, elementary discourse units (EDUs), which are the minimal discourse units, are not necessarily syntactic clauses, nor are there explicit lexical cues to indicate boundaries.

In RST-DT, to provide a balance between the consistency and the granularity of annotation, the developers chose clauses as the general basis of EDUs, with the following set of exceptions.

1. Clauses that are subjects or objects of a main verb are not treated as EDUs.
2. Clauses that are complements of a main verb are not treated as EDUs.
3. Complements of attribution verbs (speech acts and other cognitive acts) are treated as EDUs.
4. Relative clauses, nominal postmodifiers, or clauses that break up other legitimate EDUs, are treated as embedded discourse units.
5. Phrases that begin with a strong discourse marker, such as *because*, *in spite of*, *as a result of*, *according to*, are treated as EDUs.

For example, according to Exception 1 above, the sentence

Deciding what constitutes "terrorism" can be a legalistic exercise.

consists of one single EDU, instead of two EDUs segmented before *can*. So simply relying on syntactic information is not sufficient for EDU segmentation, and more sophisticated approaches need to be taken. In Chapter 2, I will present my work on developing a discourse segmentation model for determining EDU boundaries.

1.1.2 Inventory of Discourse Relations

According to RST, there are two types of discourse relation, hypotactic ("mononuclear") and paratactic ("multi-nuclear"). In mononuclear relations, one of the text spans, the *nucleus*, is more salient than the other, the *satellite*, while in multi-nuclear relations, all text spans are equally important for interpretation.

In RST-DT, the original 24 discourse relations defined by Mann and Thompson (1988) are further divided into a set of 78 fine-grained rhetorical relations in total (53 mononuclear and 23 multi-nuclear), which provides a high level of expressivity.

The 78 relations can be clustered into 16 relation classes, as shown in Table 1.1. For example, the class CAUSE is a coarse-grained clustering of the relation types *cause*, *result*, and *consequence*. Moreover, three relations are used to impose structure on the tree: TEXTUAL-ORGANIZATION, SPAN, and SAME-UNIT (used to link parts of units separated by embedded units or spans). With nuclearity attached, there are 41 distinct types of discourse relation class, as shown in Table 1.2. For example, there can be three distinct types of CONTRAST relation: CONTRAST[N][N] (both spans are nucleus), CONTRAST[N][S] (the first span is nucleus and the other is satellite), and CONTRAST[S][N]. And these 41 distinct types of relation class are the level of granularity on which most current work on classifying RST-style discourse relations is focused.

The definition of each particular RST relation is based on four elements: (1) Constraints on the nucleus; (2) Constraints on the satellite; (3) Constraints on the combination of nucleus and satellite; and (4) Effect achieved on the text receiver. For example, Table 1.3 illustrates the definition of the CONDITION class, with respect to the four definition elements described above[2].

1.1.3 An Example of RST-Style Discourse Tree Representation

The example text fragment shown in Figure 1.1 consists of four EDUs (e_1-e_4), segmented by square brackets. Its discourse tree representation is shown below in the figure, following the

[2]Taken from the website of RST at `http://www.sfu.ca/rst/01intro/definitions.html`.

Relation class	Relation type list
ATTRIBUTION	*attribution*, *attribution-negative*
BACKGROUND	*background*, *circumstance*
CAUSE	*cause*, *result*, *consequence*
COMPARISON	*comparison*, *preference*, *analogy*, *proportion*
CONDITION	*condition*, *hypothetical*, *contingency*, *otherwise*
CONTRAST	*contrast*, *concession*, *antithesis*
ELABORATION	*elaboration-additional*, *elaboration-general-specific*, *elaboration-part-whole*, *elaboration-process-step*, *elaboration-object-attribute*, *elaboration-set-member*, *example*, *definition*
ENABLEMENT	*purpose*, *enablement*
EVALUATION	*evaluation*, *interpretation*, *conclusion*, *comment*
EXPLANATION	*evidence*, *explanation-argumentative*, *reason*
JOINT	*list*, *disjunction*
MANNER-MEANS	*manner*, *means*
TOPIC-COMMENT	*problem-solution*, *question-answer*, *statement-response*, *topic-comment*, *comment-topic*, *rhetorical-question*
SUMMARY	*summary*, *restatement*
TEMPORAL	*temporal-before*, *temporal-after*, *temporal-same-time*, *sequence*, *inverted-sequence*
TOPIC-CHANGE	*topic-shift*, *topic-drift*

Table 1.1: The 17 coarse-grained relation classes and the corresponding 78 fine-grained relation types (53 mononuclear and 23 multi-nuclear) in the RST Discourse Treebank. Note that relation types which differ by nuclearity only, e.g., *contrast* (mononuclear) and *contrast* (multi-nuclear), are clumped into one single type name here.

[Catching up with commercial competitors in retail banking and financial services,]e_1 [they argue,]e_2 [will be difficult,]e_3 [particularly if market conditions turn sour.]e_4

wsj_0616

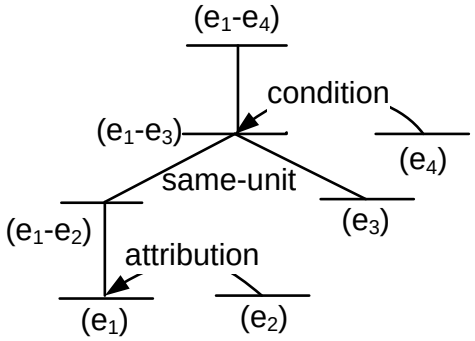


Figure 1.1: An example text fragment composed of four EDUs, and its RST discourse tree representation.

Relation class	Nuclearity associations		
	[N][S]	[S][N]	[N][N]
ATTRIBUTION	√	√	
BACKGROUND	√	√	
CAUSE	√	√	√
COMPARISON	√	√	√
CONDITION	√	√	√
CONTRAST	√	√	√
ELABORATION	√	√	
ENABLEMENT	√	√	
EVALUATION	√	√	√
EXPLANATION	√	√	√
JOINT			√
MANNER-MEANS	√	√	
TOPIC-COMMENT	√	√	√
SUMMARY	√	√	
TEMPORAL	√	√	√
TOPIC-CHANGE	√		√
TEXTUAL-ORGANIZATION			√
SAME-UNIT			√

Table 1.2: The 41 distinct relation classes in the RST Discourse Treebank with nuclearity attached.

Definition element	Description
Constraints on the nucleus, N	None
Constraints on the satellite, S	S represents a hypothetical, future, or otherwise unrealized situation (relative to the situational context of S)
Constraints on N + S	Realization of N depends on realization of S
Effect on text receiver, R	R recognizes how the realization of N depends on the realization of S

Table 1.3: Definition of the CONDITION relation class, with respect to the four definition elements.

notational convention of RST. The two EDUs e_1 and e_2 are related by a mononuclear relation ATTRIBUTION, where e_1 is the more salient span, as denoted by the arrow pointing to e_1. The span (e_1-e_2) and the EDU e_3 are related by a multi-nuclear relation SAME-UNIT, where they are equally salient, as denoted by the two straight lines connecting (e_1-e_2) and e_3. Finally, the span (e_1-e_3) is related to e_4 with a mononuclear relation CONDITION to form the complete discourse tree for the sentence. In this way, we have a tree-structured hierarchical representation corresponding to the entire sentence.

Note that no constraint is imposed to the scope for an RST-style discourse tree representation, in the sense that the tree-structured representation could be used to describe the discourse structures for texts on different levels: from sentences, to paragraphs, and finally to the entire text. Due to such a capacity to represent discourse relations on different levels of granularity, RST is of particular interest to many researchers in the field of discourse analysis. More importantly, it fits nicely with the goal outlined in the beginning of this chapter, i.e., to provide a general solution to a variety of problems in the analysis of discourse structures. As we shall see in later chapters, a number of problems of discourse analysis do benefit from identifying RST-style discourse relations in texts.

1.1.4 RST-Style Discourse Parsing Pipeline

Due to the nature of the tree-structured representation of discourse relations, RST-style discourse parsing typically adopts a pipeline framework which consists of two individual stages:

1. **Discourse segmentation**: Segment a raw text into non-overlapping EDUs, which are the bottom-level discourse units of the text-level discourse tree representation.

2. **Discourse tree-building**: Given the set of segmented EDUs from Stage 1, adopt appropriate strategies to build the discourse tree corresponding to the full text, e.g., the example discourse tree shown in Figure 1.1.

In Part I, Chapters 2 and 3 will discuss related work and my own work on these two stages in detail.

1.1.5 Issues with RST and RST-DT

Over its history of nearly three decades, RST has gained unparalleled popularity among various discourse theories, and has been applied to a variety of applications, not only for text generation — its original motivation and design purpose — but also for a large number of tasks in text understanding. Not coincidentally, there also has been much literature dedicated to questioning or criticizing several aspects of RST.

However, as mentioned previously, according to Taboada and Mann (2006), most of these criticisms stem from misunderstanding of, or digression from, the original design of RST. In contrast, RST should be considered as an open system with a high extent of flexibility, and encourages innovations and adaption for specific applications and domains. In fact, only the following general rules are enforced when applying RST-style discourse analysis:

> *Analysis of a text is performed by applying schemas that obey constraints of completedness (one schema application contains the entire text); connectedness (each span, except for the span that contains the entire text, is either a minimal unit or a constituent of another schema application); uniqueness (each schema application contains a different set of text spans); and adjacency (the spans of each schema application constitute one contiguous text span).*
>
> — Taboada and Mann (2006), p. 5.

Nevertheless, in terms of current computational approaches toward RST-style discourse analysis, especially due to the use of RST-DT as the benchmark dataset, there are indeed several commonly accepted formulations which are in fact questionable. Here, I briefly talk about some most prominent issues with regard to RST-DT and RST in general.

First of all, the clause-based EDU segmentation rule has been criticized as being too coarse-grained and being unable to capture a few linguistic phenomena. For example, as specified by RST-DT, clauses that are subjects or objects of a main verb are not treated as EDUs (see Section 1.1.1); therefore, the following sentence is regarded as one single EDU.

His studying hard makes him pass the exam.

However, this segmentation is not sufficiently fine-grained, as it precludes any representation of the underlying causal relation between the two actions *studying hard* and *passing the exam*.

Furthermore, there are concerns about whether it is feasible to represent a text by a tree-shaped discourse structure, and whether such a tree-shaped representation is the only valid representation for the given text. Admittedly, it might be a too strong assumption that a single tree is able to capture the discourse structure in the entire text: For a text written by an average writer, it is normal to see occasional digression from the main topic, or gradual development of thoughts, such that there is a certain degree of coherence within a small text fragment, while relations between different fragments are rather loose. Therefore, to deal with these complications in real texts, Wolf and Gibson (2005) propose to use an alternative graph-based data structure for analysis, which allows cross dependencies and nodes with more than one parent. However, despite their greater expressivity, graph-based representations also impose greater challenges to automatic discourse parsing.

Finally, the adjacency constraint in RST, i.e., the spans of each discourse relation constitute one contiguous text span, is not entirely justified either, and the subtlety lies in the presence of embedded discourse units. According to the definition in RST-DT, an embedded discourse unit has one or both of the following properties: (1) It breaks up a unit which is legitimately an EDU on its own; (2) It modifies a portion of an EDU only, not the entire EDU. For instance, Figure 1.2 shows a text fragment with three EDUs, where the second EDU is an embedded one. The embedded EDU e_2 breaks up e_1 and e_3, which, when concatenated, is a legitimate EDU on its own. Therefore, in order to characterize the coherence between e_1 and e_3, which is essentially a continuation, the developers of RST-DT had to invent a pseudo-relation, called

[But maintaining the key components of his strategy]e_1 [— a stable exchange rate and high levels of imports —]e_2 [will consume enormous amounts of foreign exchange.]e_3

wsj_0300

Figure 1.2: An example text fragment composed of three EDUs, where e_2 is an embedded EDU.

SAME-UNIT. However, in this way, the adjacency constraint is violated by the presence of the embedded EDU e_2.

1.2 The Penn Discourse Treebank and PDTB-Style Discourse Parsing

The Penn Discourse Treebank (PDTB) (Prasad et al., 2008) is another annotated discourse corpus. Its text is a superset of that of RST-DT (2159 Wall Street Journal articles). Unlike RST-DT, PDTB does not follow the framework of RST; rather, it follows Discourse Lexicalized Tree Adjoining Grammar (D-LTAG) (Webber, 2004), which is a lexically grounded, predicate-argument approach with a different set of predefined discourse relations. In this framework, a discourse connective (e.g., *because*) is considered to be a predicate that takes two text spans as its arguments. The argument that the discourse connective structurally attaches to is called **Arg2**, and the other argument is called **Arg1**; unlike in RST, the two arguments are not distinguished by their saliency for interpretation.

An example annotation from PDTB is shown in Example 1.1, in which the explicit connective (*when*) is underlined, and the two arguments, **Arg1** and **Arg2**, are shown in *italics* and **bold** respectively. The example is annotated with its three-level hierarchical relation type: it is of the CONTINGENCY class, the CAUSE type, and the REASON subtype.

Example 1.1. *Use of dispersants was approved* when **a test on the third day showed some positive results**. (CONTINGENCY:CAUSE:REASON)

(wsj_1347)

In PDTB, relation types are organized hierarchically: there are 4 *classes*: EXPANSION, COMPARISON, CAUSE, and TEMPORAL, which can be further divided into 16 *types* and 23 *subtypes*.

After the release of PDTB, several attempts have been made to recognize PDTB-style relations. The corpus study conducted by Pitler et al. (2008) showed that overall discourse connectives are mostly unambiguous and allow high accuracy classification of discourse relations: they achieved over 90% accuracy by simply mapping each connective to its most frequent sense. Therefore, the real challenge of discourse parsing lies in implicit relations (discourse relations which are not signaled by explicit connectives), and recent research emphasis is on recognizing these implicit discourse relations.

In particular, Lin et al. (2009) attempted to recognize such implicit discourse relations in PDTB by using four classes of features — contextual features, constituent parse features, dependency parse features, and lexical features — and explored their individual influence on performance. They showed that the production rules extracted from constituent parse trees are the most effective features, while contextual features are the weakest. Subsequently, they fully implemented an end-to-end PDTB-style discourse parser (Lin et al., 2014). Pitler et al. (2009) adopted a similar set of linguistically motivated features, and performed a series of one vs. others classification for recognizing implicit discourse relations of various types.

Later, based on the insight of Pitler et al. (2008) described above, Zhou et al. (2010) proposed to solve the problem of recognizing implicit relations by first predicting the appropriate discourse connective and then mapping the predicted connective to its most frequent discourse sense. Specifically, Zhou et al. trained a language model to evaluate the perplexity of a set of synthetic texts, which are formed by inserting every possible discourse connective into the implicit discourse relation of interest. The most probable connective is chosen from the synthetic text with the lowest perplexity. However, this approach did not achieve much success.

The main reason is that the synthetic texts formed in this way differ by the inserted connective only; therefore, the computation of perplexity would take into account a very limited number of contextual words near the connective (typically trigram sequences are used in the computation). In fact, usually a much larger proportion of the text is required for correctly interpreting the particular implicit relation.

A more recent research focus of recognizing implicit discourse relations is on feature refinement. Park and Cardie (2012) applied a simple greedy feature selection on the sets of features previously used by Pitler et al. (2009) to enhance the performance on implicit relation recognition. Recently, Rutherford and Xue (2014) argued that word pairs, which are shown to be the most effective features for recognizing implicit relations, suffer from sparsity issue when available training samples are limited. Therefore, they proposed to overcome this sparsity issue through representing relations by Brown word cluster pairs[3] and coreference patterns. Rutherford and Xue achieved the current state-of-the-art one-vs-others classification performance of recognizing Level-1 implicit relations in PDTB, ranging from an F_1 score of 28% (TEMPORAL vs. others) to 80% (EXPANSION vs. others).

1.3 Differences Between the Two Discourse Frameworks

As the two most popular frameworks in the study of discourse parsing, RST and PDTB have several inherent distinctions, which make the two frameworks potentially useful for different kinds of application. In Part II, we will see several specific applications of discourse analysis, and the different effects of the analysis generated by the two frameworks on the applications.

The most important difference between the two frameworks is that, in RST-style parsing, the text is ultimately represented as a discourse tree, and thus the discourse structure is fully annotated on different granularities of the text; in PDTB, however, there does not necessarily exist a tree structure covering the full text, i.e., PDTB-style discourse relations exist only in

[3]Brown word clustering is a form of hierarchical clustering of words based on the classes of previous words, proposed by Brown et al. (1992).

a very local contextual window. As will be demonstrated in Section 6.3, the full hierarchy of discourse structure can be quite useful for some particular applications.

Moreover, since, in RST-style parsing, a text is first segmented into non-overlapping EDUs, which are the smallest units in the final discourse tree representation, any given valid discourse unit in the text therefore participates in at least one discourse relation. In other words, the discourse relations in RST-style parsing cover the entire text. However, this is generally not true in PDTB-style discourse parsing. Therefore, the RST-style discourse relations have better coverage of the text than PDTB-style discourse relations. This property of better coverage can be useful for some particular applications as well.

Finally, in general, RST-style discourse relations are more constrained than PDTB-style relations: RST-style relations can exist only between adjacent text spans (a single EDU or the concatenation of multiple continuous EDUs), and two RST-style discourse relations in a text can only be one of the two cases: the texts corresponding to the two relations are completely disjoint with each other, or the text span of one relation is a proper sub-sequence of the text span of the other relation, i.e., the two text spans cannot partially overlap with each other. However, this constraint is not found in PDTB-style discourse relations, and thus there is more flexibility in the annotation for PDTB-style relations.

The differences discussed above do not necessarily lead to a definite statement that one discourse framework is superior to the other; rather, they illustrate the differences between the underlying philosophies of the two frameworks, and thus, we should choose the more suitable one depending on the particular applications in which we are interested. For instance, due to the existence of hierarchical structure and complete coverage in RST-style discourse representation, RST-style discourse parsing is probably more suitable for those applications where global understanding of the text is required, such as the applications to be discussed in later parts of this thesis. In contrast, because PDTB-style discourse parsing is lexically grounded and represents discourse relations in a fairly local context window, it is thus more effective for those applications where we wish to pinpoint the relevant information and may have little in-

terest in the remaining of the text. Examples of such applications include information retrieval and question answering.

Part I

Discourse Parsing

Chapter 2

Discourse Segmentation

As described in Section 1.1, for RST-style discourse parsing, identifying the boundaries of discourse units is the very first stage in the pipeline workflow; therefore, its performance is crucial to the overall accuracy. In this chapter, I will first present some previous work on RST-style discourse segmentation, and then discuss about my own CRF-based discourse segmenter.

2.1 Previous Work

Conventionally, the task of automatic EDU segmentation is formulated as: given a sentence, the segmentation model identifies the boundaries of the composite EDUs by predicting whether a boundary should be inserted before each particular token in the sentence. In particular, previous work on discourse segmentation typically falls into two major frameworks.

The first is to consider each token in the sentence sequentially and independently. In this framework, the segmentation model scans the sentence token by token, and uses a binary classifier, such as a support vector machine or logistic regression, to predict whether it is appropriate to insert a boundary before the token being examined. Examples following this framework include Soricut and Marcu (2003), Subba and Di Eugenio (2007), Fisher and Roark (2007), and Joty et al. (2012).

The second is to frame the task as a sequential labeling problem. In this framework, a

[Some analysts are concerned , however ,] [**that** Banco Exterior may have waited too long] [**to** diversify from its traditional export-related activities .] (**wsj_0616**)

Label sequence: C C C C C C **B** C C C C C C C C **B** C C C C C C C

Figure 2.1: An example of a sentence with three EDUs. The tokens are separated by whitespaces and the EDUs are segmented by square brackets. The corresponding label sequence for the tokens (excluding the first token) is shown below the sentence.

given sentence is considered as a whole, and the model assigns a label to each token, indicating whether this token is the beginning of an EDU. Conventionally, the class label *B* is assigned to those tokens which serve as the *beginning* of an EDU, and the label *C* is assigned to other tokens. Because the beginning of a sentence is trivially the beginning of an EDU, the first token in the sentence is excluded in this labeling process. For example, Figure 2.1 illustrates this sequential labeling process. The example sentence consists of 23 tokens, separated by whitespaces, and the last 22 tokens are considered in the sequential labeling process. Each token is assigned a label, *B* or *C*, by the labeling model. If the token is labeled as *B*, e.g., the token *that* and the token *to* in boldface, an EDU boundary is placed before it. Therefore, the sentence is segmented into three EDUs, indicated by the square bracket pairs. A representative work following this sequential labeling framework is Hernault et al. (2010a), in which the sequential labeling is implemented using Conditional Random Fields (CRFs).

An interesting exception to the above two major frameworks is Bach et al.'s (2012) reranking model, which obtains the best segmentation performance reported so far: for the *B* class, the F_1 score is 91.0% and the macro-average over the *B* and *C* classes is 95.1%. The idea is to train a ranking function whose input is the *N*-best output of a base segmenter and outputs a reranked ordering of these *N* candidates. In their work, Bach et al. used a similar CRF-based segmenter to Hernault et al.'s as a base segmenter.

Because the reranking procedure is almost orthogonal to the implementation of the base segmenter, it is worthwhile to explore the enhancement of base segmenters for further performance improvement. With respect to base segmenters, which typically adopt the two ma-

jor frameworks introduced previously, the best performance is reported by Fisher and Roark (2007), with an F_1 score of 90.5% for recognizing in-sentence EDU boundaries (the B class), using three individual feature sets: basic finite-state features, full finite-state features, and context-free features.

Existing base segmentation models, as introduced in the beginning of this section, have certain limitations. First, the adopted feature sets are all centered on individual tokens, such as the part-of-speech of the token, or the production rule of the highest node in the syntactic tree which the particular token is the lexical head of. Although contextual information can be partially captured via features such as n-grams or part-of-speech n-grams, the representation capacity of these contextual features might be limited. In contrast, we hypothesize that, instead of utilizing features centered on individual tokens, it is beneficial to equally take into account the information from pairs of adjacent tokens, in the sense that the elementary input unit of the segmentation model is a pair of tokens, in which each token is represented by its own set of features. Moreover, existing models never re-consider their previous segmentation decisions, in the sense that the discourse boundaries are obtained by running the segmentation algorithm only once. However, since individual decisions are inter-related with one another, by performing a second pass of segmentation incorporating features which encode global characteristics of the segmentation, we may be able to correct some incorrect segmentations of the initial run. Therefore, in this work, we propose to overcome these two limitations by our pairing features and a two-pass segmentation procedure, to be introduced in Section 2.2.

2.2 Methodology

Figure 2.2 shows our segmentation model in the form of a linear-chain Conditional Random Field. Each sentence is represented by a single linear chain. For each pair of adjacent tokens in a sentence, i.e., T_{i-1} and T_i, there is an associated binary node L_i to determine the label of the pair, i.e., the existence of a boundary in between: if $L_i = B$, an EDU boundary is inserted

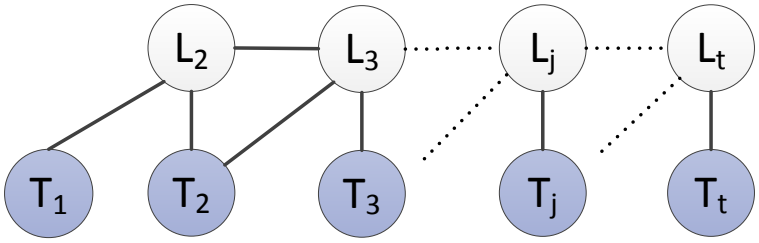


Figure 2.2: Our segmentation model in the form of a linear-chain CRF. The first layer consists of token nodes T_i's, $1 \leq i \leq t$, and the second layer represents the label L_i of each pair of tokens T_{i-1} and T_i.

before T_i; if $L_i = C$, the two adjacent tokens are considered a continuous portion in an EDU.

We choose a CRF-based model to label the whole sequence of tokens in a sentence, because a CRF is capable of taking into account the sequential information in the context, and solving the problem of determining boundaries in one single pass, which has been shown to be effective by Hernault et al. (2010a) and Bach et al. (2012). This sequential labeling framework is also beneficial to the training process, in the sense that no additional effort needs to be made to deal with the sparsity of EDU boundaries in the data, which is usually an issue for traditional binary classifiers.

As introduced previously, our segmentation model differs from previous work on RST-style discourse segmentation in two important ways.

First, rather than using a feature representation centered on a single token (possibly with some specifically designed features to partially incorporate contextual information), our boundary nodes take the input from a pair of adjacent tokens, to fully incorporate contextual information, allowing competition between neighboring tokens as well.

Secondly, rather than producing predictions of EDU boundaries by one pass of model application, we adopt a two-pass segmentation algorithm, which works as follows. We first apply our segmentation model once for each sentence. We then perform a second pass of segmentation, by considering some **global features** (to be described in Section 2.3) derived from the initial segmentation. The intuition behind these novel global features is that whether a given

token should be tagged as an EDU boundary sometimes depends on the neighboring EDU boundaries. For example, as suggested by Joty et al. (2012), since EDUs are often multi-word expressions, the distance between the current token and the neighboring boundaries can be a useful indication. In addition, it is also helpful to know whether the tokens between the current token and the neighboring boundary form a valid syntactic constituent. Since these global indicators are available only if we have an initial guess of EDU boundaries, a second pass of segmentation is necessary.

2.3 Features

As shown in Figure 2.2, each boundary node B_i in the linear-chain CRF takes the input of a pair of adjacent tokens, T_i and T_{i+1}, in the sentence. Each such pair is encoded using a list of surface lexical and syntactic features, as shown below. The features are partitioned into three subsets: basic features, global features, and contextual features, where the basic and contextual features are applicable for both the first and second pass, and the global features are applicable for the second pass only.

1. **Basic features (for both passes)**

   (a) The part-of-speech tag and the lemma of T_i / T_{i+1}.

   (b) Whether T_i / T_{i+1} is the beginning or the end of the sentence.

   (c) The top syntactic tag of the largest syntactic constituent starting from or ending with T_i / T_{i+1}.

   (d) The depth of the largest syntactic constituent starting from or ending with T_i / T_{i+1}.

   (e) The top production rule of the largest syntactic constituent starting from or ending with T_i / T_{i+1}.

2. **Global features (for the second pass)**

	Training	Test
# of documents	347	38
# of sentences	7,455	992
# of EDUs	18,765	2,346
# of in-sentence boundaries	11,310	1,354

Table 2.1: Characteristics of the training and the test set in RST-DT.

(a) The part-of-speech tag and the lemma of left / right neighboring EDU boundary.

(b) The distance to the left / right neighboring EDU boundary.

(c) Number of syntactic constituents formed by the tokens between T_i / T_{i+1} and the left / right neighboring EDU boundary.

(d) The top syntactic tag of the lowest syntactic subtree that spans from T_i / T_{i+1} to the left / right neighboring EDU boundary.

3. **Contextual features (for both passes)**: The previous and the next feature vectors.

2.4 Comparison with Other Models

We first study how our proposed two-pass discourse segmenter based on pairing features performs against existing segmentation models. In this experiment, we train our linear-chain CRF models on the RST Discourse Treebank (RST-DT) (Carlson et al., 2001), which is a large discourse corpus annotated in accordance with RST. By convention, the corpus is partitioned into a training set of 347 documents and a test set of 38 documents. The detailed characteristics of the corpus are shown in Table 2.1.

The data are preprocessed using Charniak and Johnson's reranking parser (Charniak and Johnson, 2005) to obtain syntactic structures. Our linear-chain CRFs are designed using CRFSuite (Okazaki, 2007), which is a fast implementation of linear-chain CRFs.

To apply our two-pass segmentation strategy (introduced in Section 2.2), we first train our model by representing each sentence with a single linear chain, using the basic features and

the contextual features as shown in Section 2.3. For both the training and the test set, we apply the trained one-pass model to obtain an initial EDU segmentation for each sentence. We then derive global features from this initial segmentation, and train our second-pass CRF model, together with the basic and the contextual features.

Two evaluation methods have been used in this task: the first is to evaluate the precision, recall, and F_1 scores of retrieving the in-sentence boundaries (the B class), which is the class that we care more about. The second is to evaluate the performance of both the two classes, B and C, based on the macro-averaged precision, recall, and F_1 scores of retrieving each class.

Table 2.2 demonstrates the performance evaluated on the B class. We compare against several existing models. In the first section, **CRFSeg** (Hernault et al., 2010a) is a model that adopts a similar CRF-based sequential labeling framework as ours, but with no pairing and global features involved. The second section lists four previous works following the framework of independent binary classification for each token, including **SPADE** (Soricut and Marcu, 2003), **S&E** (Subba and Di Eugenio, 2007), **Joty et al.** (Joty et al., 2012), and **F&R** (Fisher and Roark, 2007). The last model, **Reranking** (Bach et al., 2012), implements a discriminative reranking model by exploiting subtree features to rerank the N-best outputs of a base CRF segmenter, and obtained the best segmentation performance reported so far[1]. As can be seen, in comparison to all the previous models, our two-pass model obtains the best performance on all three metrics across two classes. In fact, we obtain the same recall as Joty et al., but their precision is noticeably lower than ours. With respect to the F_1 score, our model achieves an error-rate reduction of 17.8% over the best baseline, i.e., **Reranking**, and approaches to level of 95% of human performance on this task[2]. Moreover, since the reranking framework of Bach et al. is almost orthogonal to our two-pass methodology, in the sense that our two-pass segmentation model can serve as a stronger base segmenter, further improvement can be expected by plugging our two-pass model into the reranking framework.

[1]The results of the previous models in Tables 2.2 and 2.3 are the ones originally reported in the papers cited, not from our re-implementation.

[2]The human performance is measured by Soricut and Marcu (2003).

Model	**Precision**	**Recall**	F_1 **score**
CRFSeg	91.5	87.6	89.5
SPADE	85.4	84.1	84.7
S&E	85.6	86.6	86.1
Joty et al.	88.0	**92.3**	90.1
F&R	91.3	89.7	90.5
Reranking	91.5	90.4	91.0
Ours	**92.8**	**92.3**	**92.6**
Human	*98.5*	*98.2*	*98.3*

Table 2.2: Performance (%) of our two-pass segmentation model in comparison with other previous models and human performance, evaluated on the *B* class.

Model	**Class**	**Prec**	**Rec**	F_1
CRFSeg	*B*	91.5	87.6	89.5
	C	99.0	99.3	99.2
	Macro-Avg	95.2	93.5	94.3
Reranking	*B*	91.5	90.4	91.0
	C	99.3	99.4	99.2
	Macro-Avg	95.4	94.9	95.1
Ours	*B*	**92.8**	**92.3**	**92.6**
	C	**99.5**	**99.5**	**99.5**
	Macro-Avg	**96.1**	**95.9**	**96.0**

Table 2.3: Performance (%) of our two-pass segmentation model in comparison with other previous models and human performance, evaluated on the *B* and *C* classes and their macro-average.

Level	Segmentation	Span	Nuc	Rel
Intra-sentential	Joty et al.	78.7	70.8	60.8
	Ours	**85.1**	**77.5**	**66.8**
	Manual	96.3	87.4	75.1
Multi-sentential	Joty et al.	**71.1**	49.0	33.2
	Ours	**71.1**	**49.6**	**33.7**
	Manual	72.6	50.3	34.7
Text-level	Joty et al.	75.4	61.7	49.1
	Ours	**78.7**	**64.8**	**51.8**
	Manual	85.7	71.0	58.2

Table 2.4: The result of discourse parsing using different segmentation. The performance is evaluated on intra-sentential, multi-sentential, and text level separately, using the unlabeled and labeled F-score.

Table 2.3 demonstrates the performance evaluated on both classes and their macro-average. Only two previous models, CRFSeg and Reranking, reported their performance based on this evaluation, so other previous models are not included in this comparison. As can be seen, among the three models considered here, our two-pass segmentation model with pairing features performs the best not only on the *B* class but also on the *C* class, resulting in a macro-averaged F_1 score of 96.0%.

2.5 Error Propagation to Discourse Parsing

As introduced previously, discourse segmentation is the very first stage in an RST-style discourse parser. Therefore, it is helpful to evaluate how the overall performance of discourse parsing is influenced by the results of different segmentation models.

We use our own state-of-the-art RST-style discourse parser (to be introduced in Chapter 5) as the target discourse parser, and feed the parser with three sets of EDUs: (1) **manual**: the gold-standard EDUs as in the annotation of RST-DT; (2) **Joty et al.**, which is the EDUs segmented using the released version of Joty et al.'s (2012) segmenter[3]; and (3) **ours**: the

[3]The code is vailable at `http://alt.qcri.org/discourse/Discourse_Parser_Dist.tar.gz`. Note

EDUs segmented using our own model.

To evaluate the performance, we use the standard unlabeled and labeled F-score for Span, Nuclearity, and Relation, as defined by Marcu (2000). Moreover, to further illustrate the effect of automatic segmentation on different levels of the text, we conduct the evaluation on the intra-sentential, multi-sentential, and text levels separately. On the intra-sentential level, the evaluation units are discourse subtrees which do not cross sentence boundaries. On the multi-sentential level, all discourse subtrees which span at least two sentences are considered. On the text level, all discourse subtrees are evaluated.

The results are shown in Table 2.4. As can be seen, on the intra-sentential level, the influence of segmentation is significant. Evaluated on Span, Nuclearity, and Relation, using our own segmentation results in a 10% difference in F-score ($p < .01$ in all cases)[4], while the difference is even larger when using Joty et al.'s segmentation. Nonetheless, the overall parsing performance is significantly better ($p < .01$) when using our segmentation model than using Joty et al.'s.

However, the difference between using manual and automatic segmentation almost disappears when evaluated on multi-sentential level. In fact, the absolute difference on all metrics is less than 1% and insignificant as well. Actually, this is not a surprising finding: Most discourse constituents in an RST-style discourse parse tree conform to the sentence boundaries, in the sense that EDUs rarely span over multiple sentences. Moreover, the target discourse parser we adopt in this experiment takes a two-stage parsing strategy: in the first stage, sentences are processed to form sentence-level discourse subtrees, which in turn serve as the basic processing unit in the second parsing stage. Therefore, due to the nature of the RST-style discourse trees and the particular parsing algorithm in the target discourse parser, the influence of different segmentation is very much confined within each sentence, and thus has little effect on

that, in this released version, sentence splitting is incorporated as part of the preprocessing procedure of the software. For the sake of fair comparison, to rule out the complication of different sentence splitting between their software and our own models, we modified their code to ensure all EDU segmenters are fed with the same set of sentences as input.

[4] All significance tests are performed using the Wilcoxon signed-rank test.

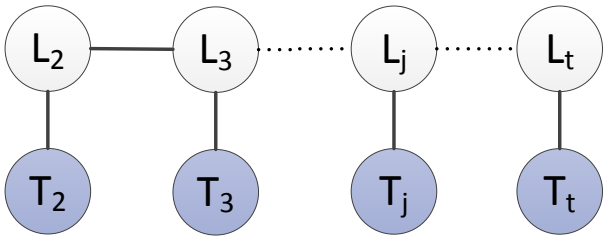


Figure 2.3: Our segmentation model with no pairing features. The first layer consists of token nodes T_i's, and the second layer represents the label L_i of the single token T_i, representing whether an EDU boundary exists before the token.

higher levels of the tree. Based on the analysis, the influence of segmentation on the text level is almost entirely attributed to its influence on the intra-sentential level.

2.6 Feature Analysis

In this section, we study the effect of our pairing and global features, the two distinct characteristics of our two-pass segmentation model, on the overall performance, and their generality across different segmentation frameworks.

2.6.1 Feature Ablation across Different Frameworks

First, starting from our full model, we perform a series of feature ablation experiments. In each of these experiments, we remove one of the component features or their combinations from the feature set in training, and evaluate the performance of the resulting model.

Removing Pairing Features ($-p$) By removing pairing features, our CRF-based segmentation model shown in Figure 2.2 reduces to the one shown in Figure 2.3, in which the input to each label node L_i is a single token T_i, rather than a pair of adjacent tokens T_{i-1} and T_i. Note that the first token T_1 is excluded in the sequence because there always exists an EDU boundary (the beginning of the sentence) before T_1. In accordance, the features listed in Section 2.3 now reduce to features describing each single token T_i.

Removing Global Features ($-g$) By removing global features, our two-pass segmentation model reduces to a simple one-pass model, in which only the basic and contextual features in Section 2.3 are used in training the model.

Removing Both Features ($-pg$) In this case, our model reduces to a simple one-pass model, in which only the basic and contextual features are used, and all features are based on each individual token T_i, rather than the token pair T_i and T_{i+1}.

Moreover, we wish to explore the generality of our pairing features and the two-pass strategy, by evaluating their effects across different segmentation frameworks. In particular, since our two-pass segmentation model itself is a CRF-based sequential labeling model, in this experiment, we also study the effect of removing pairing and global features in the framework of independent binary classification. Recall that in the framework of independent binary classification, each token (excluding T_1) in a sentence is examined independently in a sequence, and a binary classifier is used to predict the label for that token.

Figure 2.4 shows our models in the framework of independent binary classification. If pairing features are enabled, as shown in Figure 2.4a, in each classification, a pair of adjacent tokens, rather than a single token, are examined, and the classifier predicts whether an EDU boundary exists in between. If pairing features are disabled, the model reduces the one shown in Figure 2.4b.

In this experiment, we explore two underlying classifiers in independent binary classification: Logistic Regression (LR) and a linear-kernel Support Vector Machine (SVM). We implement these two classifiers using Scikit-learn (Pedregosa et al., 2011). For LR, all parameters are kept to their default values, while for SVM, we use auto class-weights, which are adjusted based on the distribution of different classes in the training data, to overcome the sparsity of class B in the dataset.

Table 2.5 demonstrates the results of our feature analysis. The first section lists the performance of our full model in different segmentation frameworks. As can be seen, our full

(a) With pairing features. (b) No pairing features.

Figure 2.4: Our segmentation model in the framework of independent binary classification.

Model	**Precision**	**Recall**	F_1 **score**
CRF	92.8	92.3	92.6
LR	**92.9**	92.2	92.5
SVM	92.6	**92.8**	**92.7**
CRF^{-p}	91.3	91.1	91.2
LR^{-p}	91.0	90.5	90.7
SVM^{-p}	90.4	92.5	91.4
CRF^{-g}	92.5	91.0	91.7
LR^{-g}	91.7	91.0	91.3
SVM^{-g}	84.7	94.7	89.4
CRF^{-pg}	87.0	82.5	84.7
LR^{-pg}	86.9	83.0	84.9
SVM^{-pg}	70.5	94.6	80.8

Table 2.5: The effect of removing pairing features ($-p$), removing global features ($-g$), and removing both ($-pg$), in comparison with the full model in the first section, across different segmentation frameworks. CRF stands for our standard two-pass segmentation models based on linear-chain CRFs, while LR and SVM stand for two different classifiers in the framework of independent binary classification.

models perform similarly across different frameworks, where the absolute difference in F_1 is less than 0.2% and insignificant. This is consistent with Hernault et al.'s (2010a) finding that, when a large number of contextual features are incorporated, binary classifiers such as SVM can achieve competitive performance with CRFs. The second section lists the performance of our models with no pairing features ($-p$). For all three resulting models, CRF^{-p}, LR^{-p}, and SVM^{-p}, the performance is significantly lower ($p < .01$) than their corresponding full model in the first section. A similar trend is observed when global features are removed ($-g$) in the third section. However, with respect to the underlying frameworks themselves, SVM is significantly

worse than CRF and LR ($p < .01$), while such a significant difference is not observable when pairing features are removed. Finally, when both sets of features are removed ($-pg$), as shown in the last section, the performance of our models drops drastically (from above 90% to below 85%). This suggests that the pairing and global features, which have an important effect on the performance by themselves, are even more important in their combination.

In this experiment, we demonstrate that the pairing features and the global features have individual effect in improving the overall segmentation performance, and such an improvement is significant. Moreover, we observe similar effects across different frameworks, which suggests the generality of these two novel aspects of our segmentation model.

2.6.2 Error Analysis

We now conduct a token-by-token error analysis to study the distributions of the errors made by our CRF-based models with different feature settings. In particular, we evaluate the labeling errors made on each token in the test set by our fully-fledged two-pass segmentation model or the models trained with pairing or global features removed. Here, we restrict our comparisons to models following the sequential labeling framework, i.e., the CRF-based models with $-p$ or $-g$ superscript in Table 2.5. Once again, all tokens which are the beginning of the sentence are not included in this analysis.

The results are shown in Table 2.6. One interesting observation is that, as demonstrated in the second section of the table, on top of CRF^{-g}, by adding global features, our full model is able to correct the 21 errors made by CRF^{-g} while introducing no additional errors in the process. In addition, as illustrated in the third section of the table, pairing and global features are almost complementary to each other, in the sense that the 39% of the errors made by CRF^{-p} occur on cases where CRF^{-g} is correct, and reciprocally, 32% of errors made by CRF^{-g} happen on cases where CRF^{-p} is correct.

Finally, in Figure 2.5, we show some examples sentences, which our fully-fledged two-pass segmentation model labels correctly, while the weaker models make some errors.

		CRF		
		¬Error	**Error**	**Total**
CRF^{-p}	**¬Error**	20357	52	20409
	Error	100	149	249
	Total	20457	201	20658

		CRF		
		¬Error	**Error**	**Total**
CRF^{-g}	**¬Error**	20436	0	20436
	Error	21	201	222
	Total	20457	201	20658

		CRF^{-p}		
		¬Error	**Error**	**Total**
CRF^{-g}	**¬Error**	20339	97	20436
	Error	70	152	222
	Total	20409	249	20658

Table 2.6: Comparisons of error between our CRF-based segmentation models with different feature settings.

[“ Oh , I bet] [it ’ll be up 50 points on Monday , ”] [said Lucy Crump , a 78-year-old retired housewife in Lexington , Ky.] (**CRF**)

[“ Oh ,] [I bet] [it ’ll be up 50 points on Monday , ”] [said Lucy Crump , a 78-year-old retired housewife in Lexington , Ky.] (**CRF**$^{-p}$)

[They argue] [that the rights of RICO defendants and third parties] [not named in RICO indictments] [have been unfairly damaged .] (**CRF**)

[They argue] [that the rights of RICO defendants and third parties] [not named in RICO indictments have been unfairly damaged .] (**CRF**$^{-g}$)

Figure 2.5: Example sentences where the full model (**CRF**) is correct while the weaker model, (**CRF**$^{-p}$) or (**CRF**$^{-g}$), makes mistakes.

2.7 Conclusion and Future Work

In this chapter, we presented our two-pass RST-style discourse segmentation model based on linear-chain CRFs. In contrast to the typical approach to EDU segmentation, which relies on token-centered features in modeling, the features in our segmenter are centered on pairs of tokens, to equally take into account the information from the previous and the following token surrounding a potential EDU boundary position. Moreover, we propose a novel two-pass segmentation strategy. After the initial pass of segmentation, we obtain a set of global features to characterize the segmentation result in a whole, which are considered in the second pass for better segmentation.

Comparing with several existing discourse segmentation models, we achieved the best performance on identifying both the boundaries and non-boundaries. Moreover, we studied the effect of our novel pairing and global features, and demonstrated that these two sets of features are both important to the overall performance, and such importance is observable across different segmentation frameworks and classifiers. Finally, we experimented with our segmentation model as a plug-in in our discourse parser and evaluated its influence to the overall parsing accuracy. We found that the automatic segmentation has a huge influence on the parsing accuracy, when evaluated on intra-sentential level; however, such an influence is very minor on multi-sentential level.

For future work, we wish to explore the incorporation of our two-pass segmentation model in the reranking framework of Bach et al. (2012). Since our model is shown to be a stronger base segmenter, with the reranking procedure, further improvement in segmentation accuracy may be expected.

Chapter 3

Discourse Tree-Building and Its Evaluation

The second stage in an RST-style discourse parsing pipeline is the tree-building process, which iteratively links adjacent discourse units to form a larger discourse unit, by identifying discourse-related units and assigning the specific type of discourse relation to those units. This process starts from EDUs on the bottom level and moves on to larger discourse units on higher levels in later times.

3.1 Evaluation of Discourse Parse Trees

Before introducing the related work in the tree-building process, let us first examine the issue of how to evaluate the correctness of a discourse parse tree generated after the tree-building process of an automatic discourse parser, given a gold-standard discourse-annotated dataset, such as RST-DT.

There is no systematic discussion of which evaluation methodology is generally mostly suitable for evaluating text-level RST-style discourse parse trees. However, Marcu's constituent precision and recall are relatively well-accepted measures for our domain of interest, providing us with the possibility of conducting direct comparison with other existing text-level discourse

parsers.

3.1.1 Marcu's Constituent Precision and Recall

Marcu (2000) applied constituent precision and recall, which are extensively used measurements for evaluating syntactic parsing, to measure the correctness of automatically generated discourse parse trees against gold-standard ones. Constituent recall reflects the number of constituents correctly identified by the discourse parser with respect to the number of constituents in the corresponding gold-standard tree, while constituent precision reflects the number of constituents correctly identified by the discourse parser with respect to the total number of constituents identified by the parser.

In practice, we can define different granularities of the constituents to evaluate the performance from various perspectives. For example, the following aspects are commonly considered in evaluating constituents in discourse parse trees:

1. **Spans**: the correctness of the span of each constituent.

2. **Nuclearity**: the correctness of the nuclearity of each constituent, which reflects the assignment of the saliency to the left and the right subtrees making up the constituent. For example, if two adjacent constituents form a discourse relation CONTRAST[N][S], then the nuclearity assigned to the left and right constituent is N (nucleus) and S (satellite) respectively.

3. **Relations**: the correctness of the relation assigned to each constituent, e.g., CONTRAST. By convention in RST-DT, for a pair of adjacent constituents, if they form a mononuclear relation (e.g. ELABORATION[N][S]), then the relation label (ELABORATION) is assigned to the satellite, while a special SPAN label is assigned to the nucleus; if two constituents from a multi-nuclear relation (e.g. LIST[N][N]), then the relation label (LIST) is assigned to both constituents.

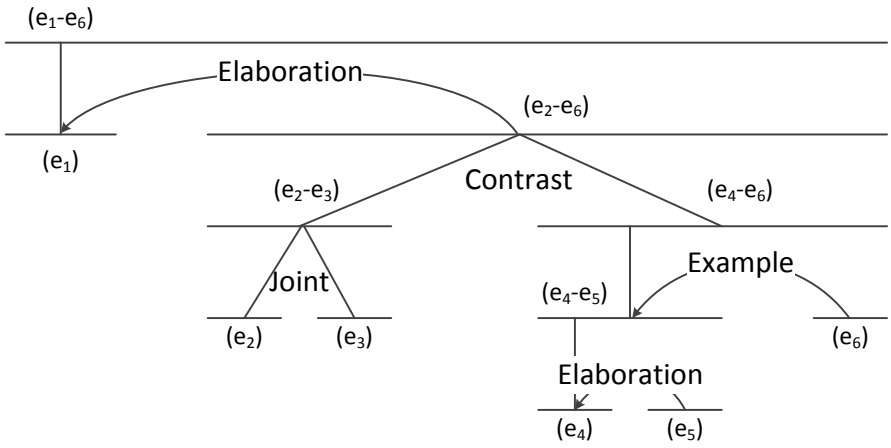
Based on the evaluation conventions outlined above, the root node of a discourse parse tree is excluded from the set of constituents to be evaluated, since the defined nuclearity and relation assignment is not applied to it.

Note that, in order to rule out the complication of different EDU segmentation in T_a and T_g in the evaluation of tree-building strategies, it is a common practice to build T_a from the same set of gold-standard EDUs as in T_g. Under this evaluation scenario, the total number of constituents to be evaluated in the two discourse parse trees will be the same, i.e., $2n - 2$ (after converting non-binary subtrees into a cascade of right-branching binary subtrees), where n is the number of EDUs in the text (recall that the root node is excluded from evaluation). Thus, both constituent precision and recall are equal to the accuracy of correctly identifying a constituent in the gold-standard parse tree by the discourse parser. I will refer to the resulting metric as *constituent accuracy*.

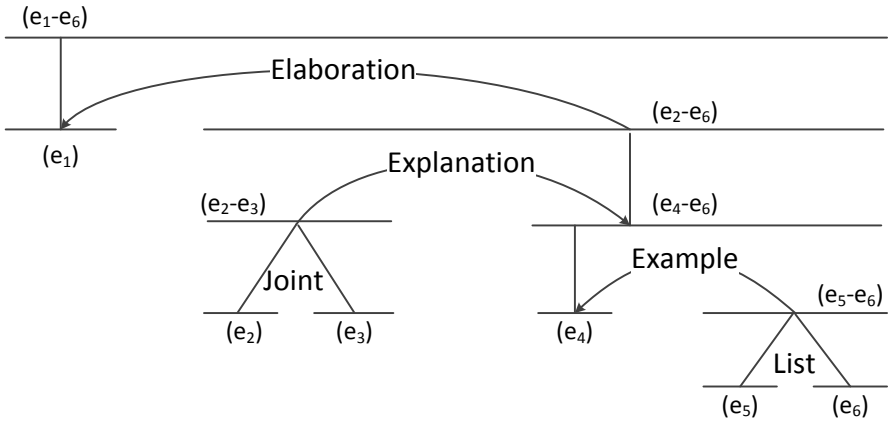
3.1.2 Example

Suppose the gold-standard discourse parse tree is T_g, and the discourse parse tree generated by an automatic discourse parser is T_a, as shown in Figure 3.1. Constituent accuracy is computed by listing the constituents in the gold-standard parse tree T_g and the ones in the automatically generated tree T_a. Constituent accuracy is computed under various evaluation conditions: Spans, Nuclearity, and Relations, following the conventions outlined in Section 3.1.1.

However, it is worth pointing out that, under the circumstance of ground-truth EDU segmentation, for the evaluation of tree structures (Spans), constituent accuracy usually produces an overestimation of the true performance. The reason is that these constituents on the bottom level, as shown in the first section in Table 3.1, will always be correctly identified, and therefore, even if the discourse parser makes a mistake on every other constituent in the discourse parse tree, it will have an accuracy of $n/(2n - 2) > 50\%$, where a random guess may approach even higher accuracy. In fact, by excluding the constituents in the first section from the computation of constituent accuracy, the result drops from 90% to 75% (3/4 is correct). Therefore,



(a) The gold-standard discourse parse tree T_g.



(b) The automatically generated discourse parse tree T_a.

Figure 3.1: The gold-standard discourse parse tree T_g vs. the automatically generated discourse parse tree T_a.

	Span		Nuclearity		Relation	
Constituents	T_g	T_a	T_g	T_a	T_g	T_a
$(e_1 - e_1)$	*	*	N	N	SPAN	SPAN
$(e_2 - e_2)$	*	*	N	N	JOINT	JOINT
$(e_3 - e_3)$	*	*	N	N	JOINT	JOINT
$(e_4 - e_4)$	*	*	N	N	SPAN	SPAN
$(e_5 - e_5)$	*	*	S	N	ELABORATION	LIST
$(e_6 - e_6)$	*	*	S	N	EXAMPLE	LIST
$(e_2 - e_3)$	*	*	N	N	CONTRAST	EXPLANATION
$(e_4 - e_5)$	*		N		SPAN	
$(e_4 - e_6)$	*	*	N	N	CONTRAST	SPAN
$(e_2 - e_6)$	*	*	S	S	ELABORATION	ELABORATION
Total accuracy	9/10 = 90%		7/10 = 70%		5/10 = 50%	

Table 3.1: Computing constituents accuracies under various evaluation conditions for the example in Figure 3.1.

to produce a more suitable estimation of the true performance, it might make more sense to exclude EDUs from the computation of constituent accuracy.

3.2 Tree-Building Strategies

There are two major kinds of such tree-building strategies: (1) greedy strategies which consider only one possible solution at each parsing step, and (2) non-greedy strategies which consider multiple or even all possible solutions at each parsing step, and choose the best one among them.

3.2.1 Greedy Tree-Building

For greedy tree-building strategies, at each step, the tree-builder always makes the best decision given the current situation, by identifying the most suitable pair of adjacent constituents to be merged into a larger discourse unit. In later times, the tree-builder does not re-consider the decisions previously made, even if they may seem inferior at this later time. This greedy

strategy is straightforward and quite efficient in practice, but it has the major disadvantage of being vulnerable to local errors.

Algorithm 1 Greedy tree-building algorithm.

Input: Segmented EDUs: $e_1, e_2, \ldots, e_n$

$L \leftarrow [e_1, e_2, \ldots, e_n]$ ▷ The list of nodes to be processed.
$S \leftarrow \emptyset$ ▷ The list of predicted scores for adjacent pairs of nodes in L.
for $i \leftarrow 1$ to $n - 1$ **do**
    $S[i] = Score(e_i, e_{i+1})$

while $|L| > 1$ **do**
    Pick $j \in [1, |S|]$ according to some heuristic $\hbar$. ▷ Select two nodes to merge.
    Make a new node T by merging the two nodes stored in $L[j]$ and $L[j + 1]$.
    Assign a relation label to T.

    Remove $S[j]$ from S. ▷ Modify the score list S.
    if j is not the first index in S **then**
        Compute $s_1 = Score(L[j - 1], T)$
        $S[j - 1] \leftarrow s_1$
    if j is not the last index in S **then**
        Compute $s_2 = Score(T, L[j + 1])$
        $S[j] \leftarrow s_2$

    Remove $L[j]$ and $L[j + 1]$ from L. ▷ Modify the node list L.
    Insert T to the j^{th} position in L.

Output: $L[1]$ ▷ The root node of the fully built discourse tree.

Algorithm 1 shows the workflow of a general greedy tree-building algorithm. It maintains two lists, L and S, where the former stores the intermediate nodes to be processed, and the latter stores the score information of pairs of adjacent nodes. To begin with, L contains the input EDUs of the text, and S stores the score of each pair of adjacent EDUs. The score can be computed in various ways, based on the predictions of local classifiers.

An example of using greedy tree-building strategies is the HILDA discourse parser (duVerle and Prendinger, 2009; Hernault et al., 2010c), which is the first fully-implemented supervised discourse parser that works at the full text level. Other work on greedy discourse parsing algorithms include SPADE (Soricut and Marcu, 2003) (a sentence-level discourse parser), Reitter (2003) (not implemented), Baldridge and Lascarides (2005) (using a more specific frame-

work of Segmented Discourse Representation Theory (Asher and Lascarides, 2003) rather than RST), Subba and Di Eugenio (2009), etc. The HILDA discourse parser adopts a bottom-up parsing approach: In each step of the tree-building process, two classifiers are applied in cascade — a binary structure classifier *Struct* to determine whether two adjacent text units should be merged to form a new subtree, and a multi-class classifier *Label* to determine which discourse relation label should be assigned to the new subtree. HILDA is a greedy discourse parser, in the sense that it always merges the two adjacent constituents with the highest score predicted by the *Struct* classifier to form a larger subtree.

One possible variant of this greedy tree-building algorithm is to use different heuristic $\hbar$ to select the pair of adjacent nodes in L to merge (Line 7 in Algorithm 1). However, as suggested by my previous exploration, usually the heuristic of choosing the pair of nodes with the highest score in S yields the best performance.

Another variant of the greedy tree-building algorithm is to adopt different feature representations in local classifiers, e.g., the *Struct* and *Relation* classifiers in HILDA. Chapter 4 will discuss our previous work on improving greedy tree-building by enhancing the local classifiers with rich linguistic features.

3.2.2 Non-Greedy Tree-Building

In contrast to greedy tree-building, non-greedy strategies consider multiple or even all possible solutions at each parsing step, and choose the **best** one among them. A representative example of non-greedy tree-building strategy is that of Joty et al. (2012; 2013), who utilize Conditional Random Fields (CRFs) as local classifiers, and build a discourse tree from bottom-up following a CKY-like parsing algorithm. They achieved the state-of-the-art performance at that time, i.e., a constituent accuracy of 55.3% for identifying relations, on RST-DT, which outperforms HILDA by a large margin.

Specifically, Joty et al.'s approach to text-level discourse parsing has two distinct features.

First, they differentiated text spans of different scopes — intra- or multi-sentential — by

employing an individual model to deal with text spans of the corresponding scope. The intuition of using separate models for text spans of different scopes is that the distribution of discourse relations is very different for text spans of the same sentence and for those belonging to different sentences. There was a similar conclusion by Feng and Hirst (2012b) as well (to be discussed in Section 4).

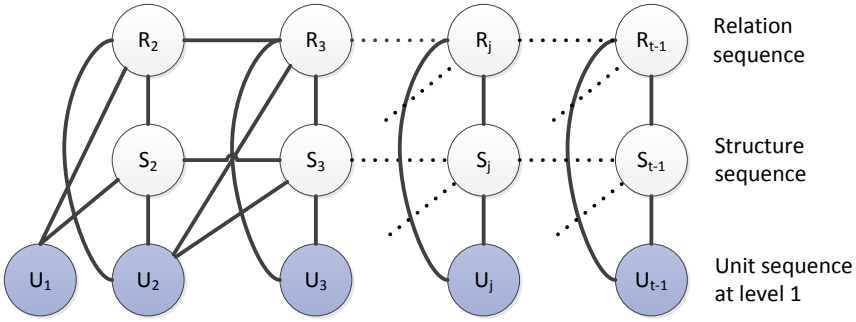
Secondly, Joty et al. jointly modeled the problem of determining the structure and the label for a given a pair of text spans, in contrast to other text-level discourse parsers (Feng and Hirst, 2012b; Hernault et al., 2010b), where two local classifiers are adopted in cascade to first determine whether to merge two spans, and then to determine which discourse relation should be assigned to the pair. In particular, for a given input text, their intra-sentential and multi-sentential parsing models assign a probability to each of the constituents of all possible discourse trees at the sentence level and at the text level, respectively.

Formally, given the model parameters Θ, for each possible constituent $R[i, m, j]$, i.e., the discourse relation relating the text spans containing EDUs i through m and the span containing EDUs $m + 1$ through j, at the sentence or document level, the parsing model estimates $P(R[i, m, j] \mid \Theta)$, which specifies a joint distribution over the label R and the structure $[i, m, j]$ of the constituent. Then, their model searches for the most likely discourse parse tree, i.e., the one with the highest probability $P(R[1, k, N] \mid \Theta), 1 \leq k \leq N$, where N is the total number of EDUs in the text. Therefore, their discourse parser in fact adopts an *optimal* tree-building strategy.

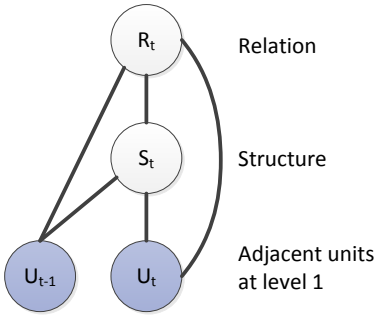
Now let us examine their intra- and multi-sentential parsing model in detail.

3.2.2.1 Intra-Sentential Parsing Model

Figure 3.2a shows a diagram of Joty et al.'s intra-sentential parsing model, expressed as a Dynamic Conditional Random Field (DCRF) (Sutton et al., 2007). On the bottom layer, each observed node U_j represents a discourse unit, either an EDU or a larger text span. The first layer of hidden nodes are the structure nodes, where $S_j \in \{0, 1\}$ denotes whether two adjacent discourse units U_{j-1} and U_j should be merged or not. The second layer of hidden nodes is



(a) Intra-sentential parsing model.



(b) Multi-sentential parsing model.

Figure 3.2: Joty et al.'s intra- and multi-sentential Condition Random Fields.

the relation nodes, in which each $R_j \in \{1, \ldots, M\}$ denotes the discourse relation between U_{j-1} and U_j, where M is the total number of predefined discourse relations[1]. Using the connections between adjacent structure nodes, we can enforce constraints such as the fact that a $S_j = 1$ must not follow a $S_{j-1} = 1$. Moreover, the connections between the two hidden layers can jointly model the structure and the relation of a sentence-level discourse subtree.

3.2.2.2 Multi-Sentential Parsing Model

For multi-sentential discourse parsing, a natural approach is to apply a new DCRF, similar to the one shown in Figure 3.2a, but at higher levels of the discourse tree. However, as argued by Joty et al., for a given input text with n sentences, the number of all possible text units

[1]In Joty et al.'s model $M = 41$, i.e., the 41 distinct relation types (with nuclearity attached) in RST-DT.

at multi-sentential level is $O(n^3)$, which leads to an overall time complexity of $O(M^2n^3)$ if using forwards-backwards exact inference on the DCRF, where M is the number of predefined discourse relations. Therefore, it is impractical to use the kind of "flat" chain structure, as in Figure 3.2a, for multi-sentential discourse parsing[2].

Instead, a separate model, as shown in Figure 3.2b, is adopted for multi-sentential discourse parsing, by breaking the chain structure. Joty et al. experimented with two different approaches to define the constituents, i.e., the adjacent units at level 1 in Figure 3.2b, to handle the complication that a sentence does not necessarily correspond to a single discourse subtree, but no significant difference was observed with the two approaches.

Although Joty et al.'s optimal discourse parser achieved state-of-the-art performance at that time, evaluated on RST-DT, it has a huge disadvantage in its inefficiency (to be shown in Section 5.1) due to its CKY-like bottom-up parsing algorithm. Therefore, in Chapter 5, I will introduce our recent work on a linear-time bottom-up discourse parser, which is both more efficient and more accurate than Joty et al.'s model.

[2]Note that there exist several approximate inference algorithms to train DCRFs, using the junction tree algorithm and DBN-style inference (Sutton et al., 2007), which result in lower time complexity. However, here, we simply restate Joty et al.'s original claim about the infeasibility of using flat chain structure on multi-sentential level.

Chapter 4

Greedy Discourse Tree-Building by Rich Linguistic Features

In this chapter, I will introduce our work on greedy discourse tree-building by incorporating rich linguistic features in local classifiers[1]. In this work, we use the HILDA discourse parser (Hernault et al., 2010c) (introduced in Section 3.2.1) as the basis of our work, and refine HILDA's original feature set by incorporating our own features as well as some adapted from Lin et al. (2009). We choose HILDA because it is a fully implemented text-level discourse parser with the best reported performance at the time of this work. On the other hand, we also follow the work of Lin et al. (2009), because their features can be good supplements to those used by HILDA, even though Lin et al.'s work was based on PDTB (introduced in Section 1.2). More importantly, Lin et al.'s strategy of performing feature selection prior to classification proves to be effective in reducing the total feature dimensions, which is favorable since we wish to incorporate rich linguistic features into our discourse parser.

[1]This work is first published as Feng and Hirst (2012b).

4.1 Method

First, as a brief overview, the HILDA discourse parser adopts a greedy bottom-up parsing approach: In each step of the tree-building process, two classifiers are applied in cascade — a binary structure classifier *Structure* to determine whether two adjacent text units should be merged to form a new subtree, and a multi-class classifier *Relation* to determine which discourse relation label should be assigned to the new subtree.

In this work, rather than focusing on the tree-building strategies, we aim to improve the overall parsing accuracy by enhancing the performance of local models in HILDA, i.e., the *Structure* and *Relation* local classifiers.

4.1.1 Raw Instance Extraction

Because HILDA adopts a bottom-up approach for discourse tree building, errors produced on lower levels will certainly propagate to upper levels, usually causing the final discourse tree to be very dissimilar to the gold standard. While appropriate post-processing may be employed to fix these errors and help global discourse tree recovery, we feel that it might be more effective to directly improve the raw instance performance of the *Structure* and *Relation* classifiers. Therefore, in our experiments, all classifications are conducted and evaluated on the basis of individual instances.

Each instance is of the form (S_L, S_R), which is a pair of adjacent text spans S_L (left span) and S_R (right span), extracted from the discourse tree representation in RST-DT. From each discourse tree, we extract positive instances as those pairs of text spans that are siblings of the same parent node, and negative examples as those pairs of adjacent text spans that are not siblings in the tree structure. In all instances, both S_L and S_R must correspond to a constituent in the discourse tree, which can be either an atomic EDU or a concatenation of multiple consecutive EDUs.

4.1.2 Feature Extraction

Given a pair of text spans (S_L, S_R), we extract the following seven types of feature.

HILDA's Features: We incorporate the original features used in the HILDA discourse parser with slight modification, which include the following four types of feature occurring in S_L, S_R, or both: (1) N-gram prefixes and suffixes; (2) lexical heads; (3) syntactic tag prefixes and suffixes; and (4) top syntactic tags.

Lin et al.'s Features: Following Lin et al. (2009), we extract the following three types of feature: (1) word pairs in (S_L, S_R); (2) dependency parse features in S_L, S_R, or both; and (3) syntactic production rules in S_L, S_R, or both.

Contextual Features: For a globally coherent text, we believe that there exist particular sequential patterns in the local usage of different discourse relations. Contextual features attempt to encode the discourse relations assigned to the preceding and the following text span pairs, given (S_L, S_R), the pair of text spans of interest. Lin et al. also incorporated contextual features in their feature set, but their work was based on PDTB, which has a very different annotation framework than RST-DT (see Sections 1.1 and 1.2). Since in RST-DT, a full text is represented as a discourse tree structure, the *previous* and the *next* discourse relations are not well-defined, whereas in PDTB, annotated discourse relations can form a chain-like structure such that contextual features can be more readily extracted.

Suppose $S_L = (e_i\text{-}e_j)$ and $S_R = (e_{j+1}\text{-}e_k)$, where $i \leq j < k$. To find the previous discourse relation REL_{prev} that immediately precedes (S_L, S_R), we look for the largest span $S_{prev} = (e_h\text{-}e_{i-1}), h < i$, such that it ends right before S_L and all its leaves belong to a single subtree which neither S_L nor S_R is a part of. If S_L and S_R belong to the same sentence, S_{prev} must also be a intra-sentential span, and vice versa if S_L and S_R is a multi-sentential span pair. REL_{prev} is therefore the discourse relation which dominates S_{prev}. Finding the next discourse relation REL_{next} that immediately follows (S_L, S_R) works in the analogous way.

Admittedly, when building a discourse tree using a greedy bottom-up approach, as adopted by the HILDA discourse parser, REL_{prev} and REL_{next} are not always available; therefore these

contextual features represent an idealized situation. In our experiments we wish to explore whether incorporating perfect contextual features can help better recognize discourse relations, and if so, set an upper bound of performance in more realistic situations.

Discourse Production Rules: Inspired by Lin et al.'s syntactic production rules as features, we develop another set of production rules, namely discourse production rules, derived directly from the tree structure representation in RST-DT.

For example, with respect to the RST discourse tree shown in Figure 1.1, we extract the following discourse production rules: ATTRIBUTION → NO-REL NO-REL, SAME-UNIT → ATTRIBUTION NO-REL, and CONDITION → SAME-UNIT NO-REL, where NO-REL denotes a leaf node in the discourse subtree.

The intuition behind using discourse production rules is that the discourse tree structure is able to reflect the relatedness between different discourse relations — discourse relations on a lower level of the tree can determine the relation of its direct parent on some degree. Hernault et al. (2010c) attempt to capture such relatedness by traversing a discourse subtree and encoding its traversal path as features, but since they used a depth-first traversal order, the information encoded in a node's direct children is separated too far away, while we believe most useful information can be gained from the relations dominating these direct children.

Semantic Similarities: Semantic similarities are useful for recognizing relations such as COMPARISON, when there are no explicit syntactic structures or lexical features signaling such relations.

We use two subsets of similarity features for verbs and nouns separately. For each verb in either S_L or S_R, we its most frequent verb class ID in VerbNet[2], and specify whether that verb class ID appears in S_L, S_R, or both. For nouns, we extract all pairs of nouns from (S_L, S_R), and compute the average similarity among these pairs. In particular, we use *path_similarity*, *lch_similarity*, *wup_similarity*, *res_similarity*, *jcn_similarity*, and *lin_similarity* provided in *nltk.wordnet.similarity* package (Loper and Bird, 2002) for computing WordNet-

[2] `http://verbs.colorado.edu/~mpalmer/projects/verbnet`

based similarity, and always choose the most frequent sense for each noun.

Cue Phrases: We compile a list cue phrases, the majority of which are connectives collected by Knott and Dale (1994). For each cue phrase in this list, we determine whether it appears in S_L or S_R. If a cue phrase appears in a span, we also determine whether its appearance is in the beginning, the end, or the middle of that span.

4.1.3 Feature Selection

If we consider all possible combinations of the features listed in Section 4.1.2, the resulting data space can be horribly high-dimensional and extremely sparse. Therefore, prior to training, feature selection is first conducted to effectively reduce the dimension of the data space.

We employ the same feature selection method as Lin et al. (2009). Feature selection is done for each feature type separately. Among all features belonging to the feature type to be selected, we first extract all possible features that have been seen in the training data, e.g., when applying feature selection for *word pairs*, we find all word pairs that appear in some text span pair that have a discourse relation between them. Then for each extracted feature f belonging to feature type T, we compute its mutual information (MI) with all 18 discourse relation classes defined in RST-DT, and use the highest MI to evaluate the effectiveness of that feature. All extracted features of type T are sorted to form a ranked list by effectiveness. After that, we use the threshold 0.0005 to select the top features from that ranked list. The total number of selected features used in our experiments is 21,410.

4.2 Experiments

As discussed previously, our research focus in this work is the tree-building step of the HILDA discourse parser, which consists of two classifications: *Structure* and *Relation* classification. The binary *Structure* classifier determines whether a discourse relation is likely to hold between consecutive text spans, and the multi-class *Relation* classifier determining which discourse re-

Level	Dataset	Positive #	Negative #	Total #
Intra-sentential level	Training	11,087	10,188	21,275
	Testing	1,340	1,181	2,521
Multi-sentential level	Training	6,646	49,467	56,113
	Testing	882	6,357	7,239
All	Training	17,733	59,655	77,388
	Testing	2,222	7,539	9,761

Table 4.1: Number of training and testing instances used in *Structure classification*.

lation label holds between these two text spans, if the *Structure* classifier predicts the existence of such a relation.

Although HILDA's bottom-up approach is aimed at building a discourse tree for the full text, it does not explicitly employ different strategies for *intra-sentential* text spans and *multi-sentential* text spans. However, we believe that discourse parsing is significantly more difficult for text spans on higher levels of the discourse tree structure. Therefore, we conduct the following three sub-experiments to explore whether the two classifiers behave differently on different discourse levels.

Intra-sentential level: Trained and tested on text span pairs belonging to the same sentence.

Multi-sentential level: Trained and tested on text span pairs belonging to different sentences.

All: Trained and tested on all text span pairs.

In particular, we conduct *Structure* and *Relation* classification by incorporating our rich linguistic features, as listed in Section 4.1.2. To rule out all confounding factors, all classifiers are trained and tested on the basis of individual text span pairs, by assuming that the discourse subtree structure (if any) dominating each individual text span has been already correctly identified (no error propagation).

4.2.1 Structure Classification

The number of raw training and testing instances used in this experiment for different discourse level is listed in Table 4.1. Raw instances are extracted in the way as described in Section 4.1.1. We notice that the distribution of positive and negative instances is extremely skewed for *multi-sentential* instances, while for all discourse levels, the distribution is similar in the training and the testing set.

In this experiment, classifiers are trained using the SVM^{perf} classifier (Joachims, 2005) with a linear kernel.

Structure classification performance for all three discourse levels is shown in Table 4.2. The columns **Full** and **NC** (No Context) denote the performance of using all features listed in Section 4.1.2 and all features but *contextual features* respectively. As discussed in Section 4.1.2, contextual features represent an ideal situation which is not always available in real applications; therefore, we wish to see how they affect the overall performance by comparing the performance obtained with and without them as features. The column **HILDA** lists the performance of using Hernault et al.'s (2010c) original features, and **Baseline** denotes the performance obtained by always picking the more frequent class. Performance is measured by four metrics: accuracy, precision, recall, and F_1 score on the test set, shown in the first section in each sub-table. In the second section of each sub-table, we also list the F_1 score on the training data, which provides us some interesting insight into the direct comparison of how well each individual model fits the training data.

On the *intra-sentential* level, we observe that, surprisingly, incorporating contextual features is able to boost the overall performance by a large margin, even though it requires only 38 additional features. Looking at the F_1 score obtained on the training data, we can see that using full features and without contextual features both performs better on the training data than HILDA. Note that such superior performance is not due to the possible over-fitting on the training data, because we are using significantly fewer features (21,410 for **Full** and 21,372 for **NC**) compared to Hernault et al.'s (136,987); rather, it suggests that using carefully selected

rich linguistic features is able to better model the problem itself.

On the *multi-sentential* level, our features result in lower accuracy and precision than HILDA's features. However, for this discourse level, the distribution of positive and negative instances in both training and test sets is extremely skewed, which makes it more sensible to compare recall and F_1 score for evaluation. In fact, our features achieve much higher recall and F_1 score in exchange for a much lower precision and a slightly lower accuracy.

In comparison with the results on the *intra-sentential* level, we can see that the binary classification problem of whether a discourse relation is likely to hold between two adjacent text spans is much more difficult on the *multi-sentential* level. One major reason is that a lot of features that are predictive for *intra-sentential* instances are no longer applicable (e.g., *Dependency parse features*). In addition, given the extremely imbalanced nature of the dataset on this discourse level, we might need to employ special approaches to deal with this needle-in-a-haystack problem. Such difficulty can also be perceived from the training performance. Compared to *intra-sentential* level, all features fit the training data much more poorly. This suggests that additional sophisticated features or models in addition to our rich linguistic features must be incorporated in order to fit the problem sufficiently well. Unfortunately, this under-fitting issue cannot be resolved by exploiting any abundant linguistic resources for feature vector extension (e.g., Hernault et al. (2010b)), because the poor training performance is no longer caused by the unknown features found in test vectors. Nevertheless, our features are able to model the problem significantly better than HILDA's.

On the *All* level, the performance of **Full** features is astonishingly good, probably because we have more available training data than the other two discourse levels. However, with contextual features removed, our features perform similarly with Hernault et al.'s, but still with a marginal improvement on recall and F_1 score.

	Full	**NC**	**HILDA**	**Baseline**
		Intra-sentential level		
Accuracy	**91.0***	85.2*	83.7	53.2
Precision	**92.7***	85.4*	84.8	53.2
Recall	**90.2***	87.0*	84.6	100.0
F_1	**91.5***	86.2*	84.7	69.4
Train F_1	**97.9***	96.2*	95.4	68.5
		Multi-sentential level		
Accuracy	87.7	86.7	**89.1**	87.8
Precision	49.6	44.7	**61.9**	–
Recall	**64.0***	39.5*	28.0	0.0
F_1	**55.9***	41.9*	38.6	–
Train F_1	**87.3***	71.9*	49.0	–
		All level		
Accuracy	**95.6***	87.0	87.0	77.2
Precision	**94.8***	74.2	79.4	–
Recall	**85.9***	66.0*	58.2	0.0
F_1	**89.5***	69.8*	67.1	–
Train F_1	**93.2***	80.8*	72.1	–

Table 4.2: *Structure* classification performance (in percentage) on text spans of *intra-sentential*, *multi-sentential*, and *all* level. Performance that is significantly superior to that of HILDA ($p < .01$) is denoted by *.

	Full	**NC**	**HILDA**	**Baseline**
		Intra-sentential level		
MAFS	**49.0**	48.5	44.6	–
WAFS	**76.3**	76.2	74.0	–
Acc	78.1	**78.1**	76.4	31.4
Train Acc	99.9	**99.9**	99.3	33.4
		Multi-sentential level		
MAFS	**19.4**	18.4	12.7	–
WAFS	**33.4**	32.9	31.6	–
Acc	**46.8**	46.7	45.7	42.5
Train Acc	**78.3**	67.3	57.7	47.8
		All level		
MAFS	**44.0**	42.8	37.9	–
WAFS	**60.7**	60.4	58.8	–
Acc	**65.3**	65.1	64.2	35.8
Train Acc	**100.0**	100.0	90.1	38.8

Table 4.3: *Relation* classification performance on text spans of *intra-sentential*, *multi-sentential*, and *all* level.

4.2.2 Relation Classification

The *Relation* classifier has 18 possible output labels, which are the coarse-grained relation classes defined in RST-DT (introduced in Section 1.1.2). We do not consider nuclearity when classifying different discourse relations, i.e., ATTRIBUTION[N][S] and ATTRIBUTION[S][N] are treated as the same label. The training and test instances in this experiment are from the positive subset used in *Structure* classification.

In this experiment, classifiers are trained using LibSVM classifier (Chang and Lin, 2011) with a linear kernel and probability estimation.

Relation classification performance for all three discourse levels are shown in Table 4.3. We list the performance achieved by **Full**, **NC**, and **HILDA** features, as well as the majority baseline, which is obtained by always picking the most frequent class label (ELABORATION in all cases).

Following Hernault et al. (2010b), we use Macro-averaged F-scores (MAFS) to evaluate the performance of each classifier. Macro-averaged F-score is not influenced by the number of instances that exist in each relation class by equally weighting the performance of each relation class. Therefore, the evaluation is not biased by the performance on those prevalent classes such as ATTRIBUTION and ELABORATION. For the purpose of brevity, we do not show the class-wise F-scores, but in our results, we find that using our features consistently provides superior performance for most class relations than HILDA's features, and therefore result in higher overall MAFS on all discourse levels. Other metrics such as Weight-averaged F-score (WAFS), which weights the performance of each relation class by the number of its existing instances, and the testing accuracy (Acc) are also listed, but they are relatively more biased evaluation metrics in this task. Similar to *Structure* classification, the accuracy on the training data (Train Acc)[3] is listed in the second section of each sub-table. It demonstrates that our carefully selected rich linguistic features are able to better fit the classification problem, especially on the *multi-sentential* level.

Similar to our observation in *Structure* classification, the performance of *Relation* classification on the *multi-sentential* level is also much poorer than that on *intra-sentential* level, which again reveals the difficulty of text-level discourse parsing.

4.3 Conclusion

In this chapter, I presented our work on improving greedy tree-building by using rich linguistic features in local classifiers, which is the first step of our objective of developing an RST-style text-level discourse parser. We chose the HILDA discourse parser (Hernault et al., 2010c) as the basis of our work, and significantly improved its tree building step by incorporating our own rich linguistics features, together with the features borrowed from Lin et al. (2009). We analyzed the difficulty with extending traditional sentence-level discourse parsing to text-level

[3]Here we use accuracy instead of MAFS as the evaluation metric on the training data, because it is the metric that the training procedure is optimized toward.

parsing, by showing that using exactly the same set of features, the performance of *Structure* and *Relation* classification on the *multi-sentential* level is consistently inferior to that on the *intra-sentential* level. We also explored the effect of contextual features on the overall performance. We showed that contextual features are highly effective for both *Structure* and *Relation* classification on all discourse levels. Although perfect contextual features are available only in idealized situations, when they are correct, together with other features, they can almost correctly predict the tree structure and better predict the relation labels. Therefore, if adopting an iterative updating approach, which progressively updates the tree structure and the labeling based on the current estimation, we may still push the final results toward this idealized end.

In our later work (to be introduced in Chapter 5), based on our observation of the different characteristics of intra- and multi-sentential instances, we decomposed the tree-building process into two stages to deal with intra- and multi-sentential instances separately. Moreover, in this later work, we moved one step forward from instance-level evaluation only to complete discourse parse tree-building.

Chapter 5

A Linear-Time Bottom-up Discourse Parser with Constraints and Post-Editing

In this chapter, I will introduce our follow-up work on text-level discourse tree-building[1]. In this work, we develop a much faster model whose time complexity is linear in the number of sentences. Our model adopts a greedy bottom-up approach, with two linear-chain CRFs applied in cascade as local classifiers. To enhance the accuracy of the pipeline, we add additional constraints in the Viterbi decoding of the first CRF. In addition to efficiency, our parser also significantly outperforms the state of the art. Moreover, our novel approach of post-editing, which modifies a fully-built tree by considering information from constituents on upper levels, can further improve the accuracy.

5.1 Introduction

As introduced in Section 3.2.2, Joty et al.'s CRF-based tree-building algorithm achieved much better performance than HILDA's greedy tree-building strategy. Their success can be attributed to two aspects: (1) the ability of incorporating contextual information by the use of CRFs as local classifiers, and (2) the optimal tree-building strategy by the use of CKY-like bottom-up

[1]This work is first published as Feng and Hirst (2014b).

parsing algorithm. However, despite their success, Joty et al.'s model has a major defect in its inefficiency, or even infeasibility, for application in practice. Interestingly, the inefficiency is also attributed to the two aspects which account for their success: their DCRF-based joint model, on which inference is usually slow, and their CKY-like parsing algorithm, whose issue is more prominent. Due to the $O(n^3)$ time complexity, where n is the number of input discourse units, for large documents, the parsing simply takes too long[2].

Based on the analysis above, in our work, we aim to develop a text-level discourse parser, which has similar or even better accuracy than Joty et al.'s optimal parser, and with a tree-building strategy much more efficient than their $O(n^3)$ CKY-like parsing algorithm. The core idea is to maintain the greediness in the bottom-up tree-building process, as in HILDA. But unlike HILDA, we use two **linear-chain** CRFs in cascade to serve as the local classifiers. In this way, we are able to take into account the sequential information from contextual discourse constituents, which cannot be naturally represented in HILDA with SVMs as local classifiers. Therefore, our model incorporates the strengths of both HILDA and Joty et al.'s model, i.e., the efficiency of a greedy parsing algorithm, and the ability to incorporate sequential information with CRFs. In addition to efficiency, our parser also significantly outperforms the previous state of the art obtained by Joty et al.

5.2 Overall Work Flow

Figure 5.1 demonstrates the overall work flow of our discourse parser. The general idea is that, similar to Joty et al. (2013), we perform a sentence-level parsing for each sentence first, followed by a text-level parsing to generate a full discourse tree for the whole document. However, in addition to efficiency (to be shown in Section 5.4.1), our discourse parser has a distinct

[2]The largest document in RST-DT contains over 180 sentences, i.e., $n > 180$ for their multi-sentential CKY parsing. Intuitively, suppose the average time to compute the probability of each constituent is 0.01 second, then in total, the CKY-like parsing takes over 16 hours. It is possible to optimize Joty et al.'s CKY-like parsing by replacing their CRF-based computation for upper-level constituents with some local computation based on the probabilities of lower-level constituents. However, such optimization is beyond the scope of this work.

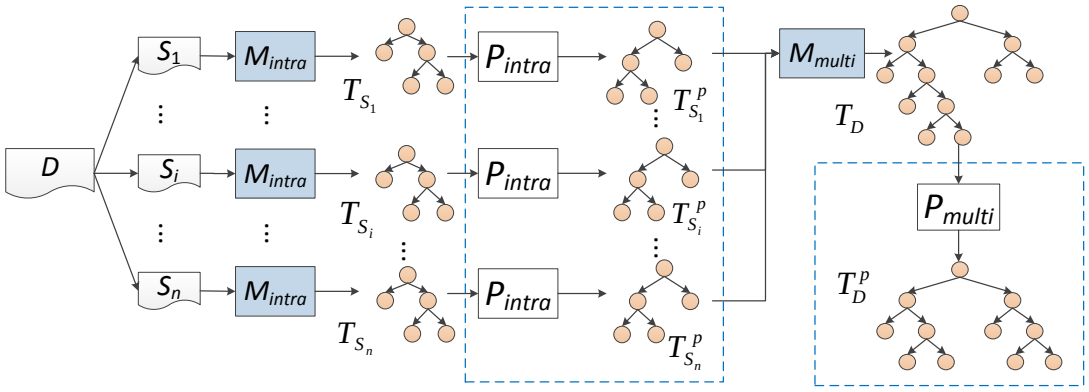


Figure 5.1: The work flow of our proposed discourse parser. In the figure, M_{intra} and M_{multi} stand for the intra- and multi-sentential bottom-up tree-building models, and P_{intra} and P_{multi} stand for the intra- and multi-sentential post-editing models.

feature, which is the post-editing component (to be introduced in Section 5.4), as outlined in dashes.

Our discourse parser works as follows. A document D is first segmented into a list of sentences. Each sentence S_i, after being segmented into EDUs (not shown in the figure), goes through an intra-sentential bottom-up tree-building model M_{intra}, to form a sentence-level discourse tree T_{S_i}, with the EDUs as leaf nodes. After that, we apply the intra-sentential post-editing model P_{intra} to modify the generated tree T_{S_i} to $T^p_{S_i}$, by considering upper-level information.

We then combine all sentence-level discourse tree $T^p_{S_i}$'s using our multi-sentential bottom-up tree-building model M_{multi} to generate the text-level discourse tree T_D. As in intra-sentential parsing, we also post-edit T_D using P_{multi} to produce the final discourse tree T^p_D.

5.3 Bottom-up Tree-Building

For both intra- and multi-sentential parsing, our bottom-up tree-building process adopts a similar greedy pipeline framework to that of HILDA discourse parser, to guarantee efficiency for large documents. In particular, starting from the constituents on the bottom level (EDUs for intra-sentential parsing and sentence-level discourse trees for multi-sentential parsing), at each

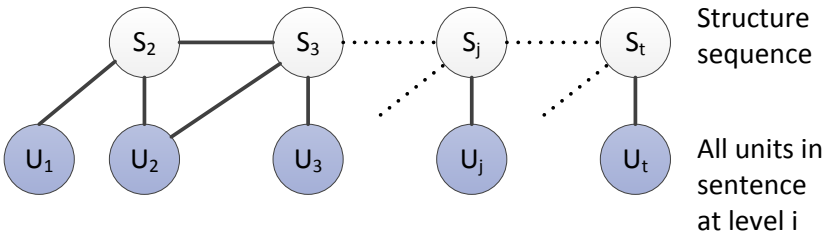


Figure 5.2: Intra-sentential structure model M_{intra}^{struct}.

step of the tree-building, we greedily merge a pair of adjacent discourse constituents such that the merged constituent has the highest probability as predicted by our *structure* model. The *relation* model is then applied to assign the relation to the new constituent.

Therefore, rather than jointly modeling the structure and relation, as Joty et al. did, we choose to use two individual models to model structure and relation separately. This choice is for the following reasons.

First, it is not entirely appropriate to model the structure and the relation at the same time. For example, with respect to Figure 3.2a, it is unclear how the relation node R_j is represented for a training instance whose structure node $S_j = 0$, i.e., the units U_{j-1} and U_j are disjoint. Assume a special relation NO-REL is assigned for R_j. Then, in the tree-building process, we will have to deal with the situations where the joint model yields conflicting predictions: it is possible that the model predicts $S_j = 1$ and $R_j =$ NO-REL, or vice versa, and we will have to decide which node to trust (and thus in some sense, the structure and the relation are no longer jointly modeled).

Secondly, as a joint model, it is mandatory to use a dynamic CRF, for which exact inference is usually intractable or slow. In contrast, for linear-chain CRFs, efficient algorithms and implementations for exact inference exist.

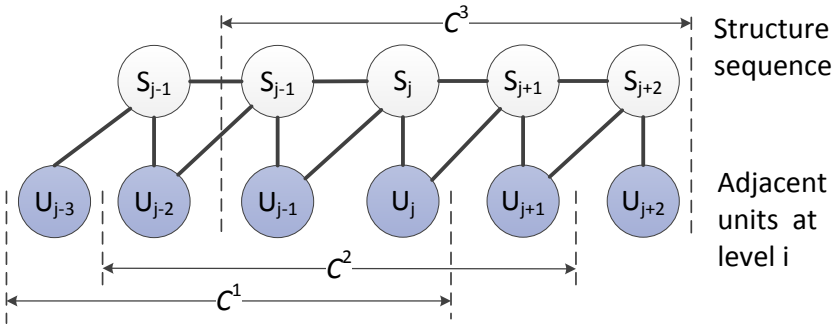


Figure 5.3: Multi-sentential structure model M^{struct}_{multi}. C^1, C^2, and C^3 denote the three chains for predicting U_j and U_{j+1}.

5.3.1 Structure Models

Figures 5.2 and 5.3 illustrate our structure models in the form of linear-chain CRFs. Similar to Joty et al.'s intra-sentential model, the first layer of the chain is composed of discourse constituents U_j's, and the second layer is composed of binary nodes S_j's to indicate the probability of merging adjacent discourse constituents. For intra-sentential parsing, at each step in the bottom-up tree-building process, as shown in Figure 5.2, we generate a single sequence, consisting of all the current discourse constituents in the sentence that need to be processed. For multi-sentential parsing, as shown in Figure 5.3, due to the computational difficulty of applying the exact inference on the entire sequence, we take a sliding-window approach to form CRF chains for a particular pair of constituents. For each pair U_{j-1} and U_j, we form three chains,C^1, C^2, and C^3, each of which contains two contextual constituents. We then find the chain $C^t, 1 \leq t \leq 3$, with the highest joint probability over the entire sequence, and assign its marginal probability $P(S^t_j = 1)$ to $P(S_j = 1)$.

5.3.2 Relation Models

Figures 5.4 and 5.5 illustrate our relation models, which are very similar to their counterparts in the structure models. However, unlike the structure models, in the relation models, adjacent

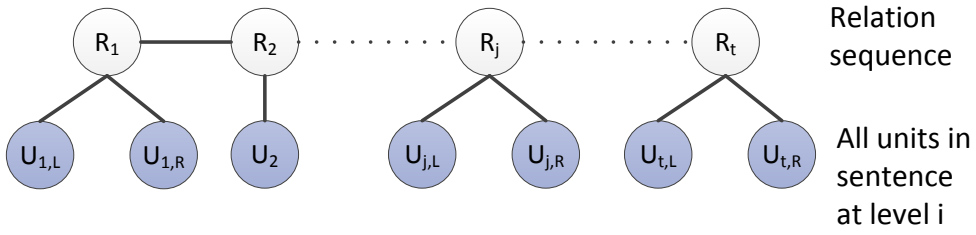


Figure 5.4: Intra-sentential relation model M^{rel}_{intra}.

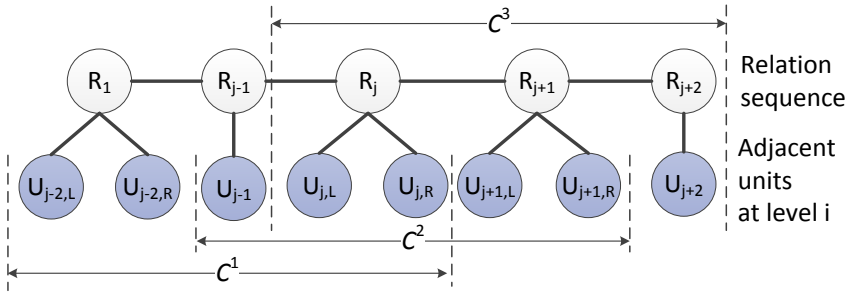


Figure 5.5: Multi-sentential relation model M^{rel}_{multi}. C^1, C^2, and C^3 denote the three sliding windows for predicting $U_{j,L}$ and $U_{j,R}$.

relation nodes do not share discourse constituents on the first layer. Rather, each relation node R_j attempts to model the relation of one single constituent U_j, by taking U_j's left and right subtrees $U_{j,L}$ and $U_{j,R}$ as its first-layer nodes; if U_j is a single EDU, then the first-layer node of R_j is simply U_j, and R_j is a special relation symbol LEAF[3]. Since we know, a priori, that the constituents in the chains are either leaf nodes or the ones that have been merged by our structure model, we never need to worry about the NO-REL issue outlined above.

In the bottom-up tree-building process, after merging a pair of adjacent constituents using M_{intra}^{struct} into a new constituent, say U_j, we form a chain consisting of all current constituents in the sentence to decide the relation label for U_j, i.e., the R_j node in the chain. In fact, by performing inference on this chain, we produce predictions not only for R_j, but also for all other R nodes in the chain, which correspond to all other constituents in the sentence. Since those non-leaf constituents are already labeled in previous steps in the tree-building, we can now **re-assign** their relations if the model predicts differently in this step. Therefore, this re-labeling procedure can compensate for the loss of accuracy caused by our greedy bottom-up strategy to some extent.

5.4 Post-Editing

After an intra- or multi-sentential discourse tree is fully built, we perform a post-editing to consider possible modifications to the current tree, by considering useful information from the discourse constituents on upper levels, which is unavailable in the bottom-up tree-building process.

The motivation for post-editing is that some particular discourse relations, such as TEXTUAL-ORGANIZATION, tend to occur on the top levels of the discourse tree; thus, information such as the depth of the discourse constituent can be quite indicative. However, the exact depth of a discourse constituent is usually unknown in the bottom-up tree-building process; therefore, it

[3]These leaf constituents are represented using a special feature vector `is_leaf = True`; thus the CRF never labels them with relations other than LEAF.

Algorithm 2 Post-editing algorithm.

Input: A fully built discourse tree T.

1: **if** $|T| = 1$ **then**
2: **return** T ▷ Do nothing if it is a single EDU.
3: $L \leftarrow [U_1, U_2, \ldots, U_t]$ ▷ The bottom-level constituents in T.
4: **while** $|L| > 2$ **do**
5: $i \leftarrow \text{PREDICTMERGING}(L, P^{struct})$
6: $p \leftarrow \text{PARENT}(L[i], L[i+1], T)$
7: **if** p = NULL **then**
8: **break**
9: Replace $L[i]$ and $L[i+1]$ with p
10: **if** $|L| = 2$ **then**
11: $L \leftarrow [U_1, U_2, \ldots, U_t]$
12: $T^p \leftarrow \text{BUILDTREE}(L, P^{struct}, P^{rel}, T)$

Output: T^p

might be beneficial to modify the tree by including top-down information after the tree is fully built.

The process of post-editing is shown in Algorithm 2. For each input discourse tree T, which is already fully built by bottom-up tree-building models, we do the following:

Lines 3 – 9: Identify the lowest level of T on which the constituents can be modified according to the post-editing structure component, P^{struct}. Here, we consider *structural* modifications only. To do so, at each time, for the given list of constituents L, we use the corresponding structural model, M^{struct}_{intra} or M^{struct}_{multi}, to propose a pair of adjacent tokens $L[i]$ and $L[i+1]$ to merge (Line 5). If the proposed pair is not merged (they do not share a common parent node) in the original tree T, we then modify the structure by merging $L[i]$ and $L[i+1]$ and terminate the loop (Lines 6 – 8); otherwise, we move up to a level higher in the tree, and proceed with this inspection (Line 9).

Lines 10 – 12: If modifications have been proposed in the previous step, we build a new tree T^p using P^{struct} as the structure model, and P^{rel} as the relation model, from the constituents on which modifications are proposed. Otherwise, T^p is built from the bottom-level constituents of T. The upper-level information, such as the depth of a discourse constituent, is derived from the initial tree T.

5.4.1 Linear Time Complexity

Here we show that the time complexity of each component in our discourse parser is linear in the number of sentences in the input text. The following analysis is focused on the bottom-up tree-building process, but a similar analysis can be carried out for the post-editing process. Since the number of operations in the post-editing process is roughly the same (1.5 times in the worst case) as in the bottom-up tree-building, post-editing shares the same complexity as the tree-building.

5.4.2 Intra-Sentential Parsing

Suppose the input document is segmented into n sentences, and each sentence S_k contains m_k EDUs. For each sentence S_k with m_k EDUs, the overall time complexity to perform intra-sentential parsing is $O(m_k^2)$. The reason is the following. On level i of the bottom-up tree-building, we generate a single chain to represent the structure or relation for all the $m_k - i$ constituents that are currently in the sentence. The time complexity for performing forward-backward inference on the single chain is $O((m_k - i) \times M^2) = O(m_k - i)$, where the constant M is the size of the output vocabulary[4]. Starting from the EDUs on the bottom level, we need to perform inference for one chain on each level during the bottom-up tree-building, and thus the total time complexity is $\Sigma_{i=1}^{m_k} O(m_k - i) = O(m_k^2)$.

The total time to generate sentence-level discourse trees for n sentences is $\Sigma_{k=1}^{n} O(m_k^2)$. It is fairly safe to assume that each m_k is a constant, in the sense that m_k is independent of the total number of sentences in the document. Therefore, the total time complexity $\Sigma_{k=1}^{n} O(m_k^2) \leq n \times O(\max_{1 \leq j \leq n}(m_j^2)) = n \times O(1) = O(n)$, i.e., linear in the total number of sentences.

[4] $M = 41$ in our case, which is the number of distinct relation types in RST-DT. For details, see Section 5.6.

5.4.3 Multi-Sentential Parsing

For multi-sentential models, M^{struct}_{multi} and M^{rel}_{multi}, as shown in Figures 5.3 and 5.5, for a pair of constituents of interest, we generate multiple chains to predict the structure or the relation.

By including a constant number k of discourse units in each chain, and considering a constant number l of such chains for computing each adjacent pair of discourse constituents ($k = 4$ for M^{struct}_{multi} and $k = 3$ for M^{rel}_{multi}; $l = 3$), we have an overall time complexity of $O(n)$. The reason is that it takes $l \times O(kM^2) = O(1)$ time, where l, k, M are all constants, to perform exact inference for a given pair of adjacent constituents, and we need to perform such computation for all $n-1$ pairs of adjacent sentences on the first level of the tree-building. Adopting a greedy approach, on an arbitrary level during the tree-building, once we decide to merge a certain pair of constituents, say U_j and U_{j+1}, we only need to recompute a small number of chains, i.e., the chains which originally include U_j or U_{j+1}, and inference on each chain takes $O(1)$. Therefore, the total time complexity is $(n-1) \times O(1) + (n-1) \times O(1) = O(n)$, where the first term in the summation is the complexity of computing all chains on the bottom level, and the second term is the complexity of computing the constant number of chains on higher levels.

We have thus showed that the time complexity is *linear* in n, which is the number of sentences in the document. In fact, under the assumption that the number of EDUs in each sentence is independent of n, it can be shown that the time complexity is also linear in the total number of EDUs[5].

5.5 Features

In our local models, to encode two adjacent units, U_j and U_{j+1}, within a CRF chain, we use the following 10 sets of features, some of which are modified from Joty et al.'s model.

[5]We implicitly made an assumption that the parsing time is dominated by the time to perform inference on CRF chains. However, for complex features, the time required for feature computation might be dominant. Nevertheless, a careful caching strategy can accelerate feature computation, since a large number of multi-sentential chains overlap with each other.

1. **Organization features:** Whether U_j (or U_{j+1}) is the first (or last) constituent in the sentence (for intra-sentential models) or in the document (for multi-sentential models); whether U_j (or U_{j+1}) is a bottom-level constituent.

2. **Textual structure features:** Whether U_j contains more sentences (or paragraphs) than U_{j+1}.

3. **N-gram features:** The beginning (or end) lexical n-grams in each unit; the beginning (or end) POS n-grams in each unit, where $n \in \{1, 2, 3\}$.

4. **Dominance features:** The PoS tags of the head node and the attachment node; the lexical heads of the head node and the attachment node; the dominance relationship between the two units.

5. **Contextual features:** The feature vector of the previous and the next constituent in the chain.

6. **Substructure features:** The root node of the left and right discourse subtrees of each unit.

7. **Syntactic features:** Whether each unit corresponds to a single syntactic subtree, and if so, the top PoS tag of the subtree; the distance of each unit to their lowest common ancestor in the syntax tree (intra-sentential only).

8. **Entity transition features:** The type and the number of entity transitions across the two units. We adopt Barzilay and Lapata's (2008) entity-based local coherence model to represent a document by an entity grid (to be introduced in Section 6.1.1), and extract local transitions among entities in continuous discourse constituents. We use bigram and trigram transitions with syntactic roles attached to each entity.

9. **Cue phrase features:** Whether a cue phrase occurs in the first or last EDU of each unit. The cue phrase list is based on the connectives collected by Knott and Dale (1994).

10. **Post-editing features:** The depth of each unit in the initial tree.

5.6 Experiments

For pre-processing, we use the Stanford CoreNLP (de Marneffe et al., 2006; Klein and Manning, 2003; Recasens et al., 2013) to syntactically parse the texts and extract coreference relations, and we use Penn2Malt[6] to lexicalize syntactic trees to extract dominance features.

For local models, our structure models are trained using MALLET (McCallum, 2002) to include constraints over transitions between adjacent labels, and our relation models are trained using CRFSuite (Okazaki, 2007), which is a fast implementation of linear-chain CRFs.

The data that we use to develop and evaluate our discourse parser is RST-DT. Following previous work on RST-DT (Feng and Hirst, 2012b; Hernault et al., 2010c; Joty et al., 2012, 2013), we use 18 coarse-grained relation classes, and with nuclearity attached, we have a total set of 41 distinct relations. Non-binary relations are converted into a cascade of right-branching binary relations.

5.7 Results and Discussion

5.7.1 Parsing Accuracy

We compare four different models using manual EDU segmentation. In Table 5.1, the *j*CRF model in the first row is the optimal CRF model proposed by Joty et al. (2013). gSVM^{FH} in the second row is our implementation of HILDA's greedy parsing algorithm using our rich linguistic features (Feng and Hirst, 2012b), which are described in Chapter 4. The third model, gCRF, represents our greedy CRF-based discourse parser, and the last row, gCRF^{PE}, represents our parser with the post-editing component included.

In order to conduct a direct comparison with Joty et al.'s model, we use the same set of

[6] `http://stp.lingfil.uu.se/~nivre/research/Penn2Malt.html`.

Model	Span	Nuclearity	Relation	
			Accuracy	MAFS
*j*CRF	82.5	68.4	55.7	N/A
gSVM^{FH}	82.8	67.1	52.0	27.4/23.3
gCRF	84.9*	69.9*	57.2*	35.3/31.3
gCRF^{PE}	**85.7***†	**71.0***†	**58.2***†	**36.2/32.3**
Baseline	74.9	44.3	29.6	1.8/1.1
Human	88.7	77.7	65.8	N/A

$*$: significantly better than gSVM^{FH} ($p < .01$)
†: significantly better than gCRF ($p < .01$)

Table 5.1: Performance of different models using gold-standard EDU segmentation, evaluated using the constituent accuracy (%) for span, nuclearity, and relation. The baseline is obtained from always merging current span with the next span to form a new discourse subtree, and labeling the resulting subtree with the relation ELABORATION[N][S]. For relation, we also report the macro-averaged F1-score (MAFS) for correctly retrieved constituents (before the slash) and for all constituents (after the slash). The significance test is performed with Wilcoxon signed-rank test.

evaluation metrics, i.e., the unlabeled and labeled constituent accuracy as defined in Section 3.1.1. For evaluating relations, since there is a skewed distribution of different relation types in the corpus, we also include the macro-averaged F_1-score (MAFS) as another metric, to emphasize the performance of infrequent relation types. We report the MAFS separately for the correctly retrieved constituents (i.e., the span boundary is correct) and all constituents in the reference tree.

As demonstrated by Table 5.1, our greedy CRF models perform significantly better than the other two models. Since we do not have the actual output of Joty et al.'s model, we are unable to conduct significance testing between our models and theirs. But in terms of overall accuracy, our gCRF model outperforms their model by 1.5%. Moreover, with post-editing enabled, gCRF^{PE} significantly ($p < .01$) outperforms our initial model gCRF by another 1% in relation assignment, and this overall accuracy of 58.2% is close to 90% of human performance[7]. With respect to the macro-averaged F_1-scores, adding the post-editing component

[7]The human performance is reported by Joty et al. (2013), measured on the doubly-annotated portion of RST-DT.

also obtains about 1% improvement. Moreover, in terms of the lower-bound of automatic discourse parsing models, we compute a majority baseline by always merging the current span with the next span to form a new discourse subtree, and labeling the resulting subtree with the relationELABORATION[N][S]. This lower-bound is an accuracy of 74.9% for Span, 44.3% for Nuclearity, and 29.6% for Relation. As a baseline obtained by a simple heuristic, the accuracy for Span, i.e., 74.9%, seems a bit too high. This echoes my previous discussion in Section 3.1.2 regarding the over-estimated accuracy for Span when using gold-standard discourse segmentation. Nevertheless, on Nuclearity and Relation, current discourse models perform substantially better than the simple baseline.

However, the overall MAFS is still at the lower end of 30% for all constituents. Our error analysis shows that, for two relation classes, TOPIC-CHANGE and TEXTUAL-ORGANIZATION, our model fails to retrieve any instance, and for TOPIC-COMMENT and EVALUATION, our model scores a class-wise F_1 score lower than 5%. These four relation classes, apart from their infrequency in the corpus, are more abstractly defined, and thus are particularly challenging.

5.7.2 Parsing Efficiency

We further illustrate the efficiency of our parser by demonstrating the time consumption of different models.

First, as shown in Table 5.2, the average number of sentences in a document is 26.11, which is already too large for optimal parsing models, e.g., the CKY-like parsing algorithm in *j*CRF, let alone the fact that the largest document contains several hundred of EDUs and sentences. Therefore, it should be seen that non-optimal models are required in most cases.

In Table 5.3, we report the parsing time[8] for the last three models, since we do not know the time of *j*CRF. Note that the parsing time excludes the time cost for any necessary preprocessing. As can be seen, our *g*CRF model is considerably faster than $g\text{SVM}^{FH}$, because, on one hand, feature computation is expensive in $g\text{SVM}^{FH}$, since $g\text{SVM}^{FH}$ utilizes a rich set

[8]Tested on a Linux system with four duo-core 3.0GHz processors and 16G memory.

	Avg	Min	Max
# of EDUs	61.74	4	304
# of Sentences	26.11	2	187
# of EDUs per sentence	2.36	1	10

Table 5.2: Characteristics of the 38 documents in the test data.

Model	**Parsing Time (seconds)**		
	Avg	Min	Max
gSVM^{FH}	11.19	0.42	124.86
gCRF	5.52	0.05	40.57
gCRF^{PE}	10.71	0.12	84.72

Table 5.3: The parsing time (in seconds) for the 38 documents in the test set of RST-DT. Time cost of any pre-processing is excluded from the analysis.

of features; on the other hand, in gCRF, we are able to accelerate decoding by multi-threading MALLET (we use four threads). Even for the largest document with 187 sentences, gCRF is able to produce the final tree after about 40 seconds, while jCRF would take over 16 hours assuming each DCRF decoding takes only 0.01 second.

We see that enabling post-editing doubles the time consumption (though the overall time is still acceptable in practice); the gain of parsing accuracy, however, as shown in Table 5.1 is relatively minor. Therefore, in practice, we might want to disable the post-editing component, for a better trade-off between efficiency and accuracy. In fact, with respect to the applications to be discussed in later chapters, I use the gCRF version (no post-editing) for parsing relatively large documents; e.g., chapters of novels and student essays.

5.8 Conclusion

In this chapter, I presented our work on developing an efficient text-level discourse parser with time complexity linear in the total number of sentences in the document. Our approach was to adopt a greedy bottom-up tree-building, with two linear-chain CRFs as local probabilistic models, and enforce reasonable constraints in the first CRF's Viterbi decoding. While signifi-

cantly outperforming the state-of-the-art model by Joty et al. (2013), our parser is much faster in practice. In addition, we proposed a novel idea of post-editing, which modifies a fully-built discourse tree by considering information from upper-level constituents. We showed that, although doubling the time consumption, post-editing can further boost the parsing performance to close to 90% of human performance.

Summary of Part I

At this point, we have presented the necessary components of our RST-style discourse parser.

For discourse segmentation, which is discussed in Chapter 2, our CRF-based EDU segmenter with novel global features significantly outperforms the previously reported best performance of Bach et al. (2012), and achieved an F_1 score of 92.4 on the RST-DT corpus.

For discourse tree-building, in Chapter 4, we proposed the use of rich linguistic features for enhancing the performance of local classifications in the framework of the HILDA discourse parser, which is the first fully-implemented text-level discourse parser. As discussed in Chapter 5, we developed an efficient linear-time tree-building algorithm, by adopting a greedy bottom-up tree-building strategy with linear-chain CRFs as local classifiers. We showed that our algorithm is not only considerably faster, but also more accurate than the optimal tree-building strategy of Joty et al. (2013), which achieved the state-of-the-art overall parsing accuracy reported previously.

Since we have a quality discourse parser, we are now at an appropriate stage to examine our previous intuition about the usefulness of discourse parsing, i.e., whether it is able to provide a general solution to many downstream applications in the analysis of discourse.

In the following parts, we will see several specific applications in discourse analysis, including the evaluation of coherence, the identification of authorship, and detection of deceptive opinions. I will describe the application-specific approaches to each of these problems, and discuss how information derived from our application-neutral discourse parser can be incorporated into each of these problems and affect the overall performance.

Part II

Applications of Discourse Parsing

Chapter 6

The Evaluation of Coherence

6.1 Introduction

In a well-written text, utterances are not simply presented in an arbitrary order; rather, they are presented in a logical and coherent form, so that the readers can easily interpret the meaning that the writer wishes to present. Therefore, coherence is one of the most essential aspects of text quality. Given its importance, the automatic evaluation of text coherence is one of the crucial components of many NLP applications. For example, in natural language generation, once the content of the text has been planned, a post-processing procedure is usually required to improve the coherence of the generated text. Similarly, in automatic multi-document summarization, excerpts from each individual document need to be rendered in a way such that the resulting summary is the most coherent. In addition, for essay scoring, since coherence is one of the major aspects based on which a score is assigned to an essay, a coherence evaluation module is a necessity in any automatic essay scoring system.

In this chapter, I will introduce a particular popular model for the evaluation of text coherence: the entity-based local coherence model of Barzilay and Lapata (B&L) (2005; 2008), which extracts mentions of entities in the text, and models local coherence by the transitions, from one sentence to the next, in the grammatical role of each mention. Numerous extensions

have been proposed to refine the entity-based local coherence model, most of which focus on enriching the feature sets used in B&L's original model. In this chapter, I will briefly overview some existing extensions on feature enrichment, and in Chapter 6.2, I will present our own extension in the direction of enhancing the learning procedure.

6.1.1 The Entity-based Local Coherence Model

For a document d, an entity grid is constructed, in which the columns represent the entities referred to in d, and rows represent the sentences. Each cell corresponds to the grammatical role of an entity in the corresponding sentence: subject (**S**), object (**O**), neither (**X**), or nothing (−), and an entity is defined as a class of coreferent noun phrases. If the entity serves in multiple roles in a single sentence, then we resolve its grammatical role following the priority order: $\mathbf{S} \succ \mathbf{O} \succ \mathbf{X} \succ -$.

For example, in Table 6.1, there is an example text fragment, consisting of three sentences, in which the entities are highlighted in boldface. The entity grid corresponding to the text is shown below[1]. A local transition is defined as a sequence $\{\mathbf{S}, \mathbf{O}, \mathbf{X}, -\}^n$, representing the occurrence and grammatical roles of an entity in n adjacent sentences. Such transition sequences can be extracted from the entity grid as continuous subsequences in each column. For example, the entity *dollar* in Table 6.1 has a bigram transition $\mathbf{S} \rightarrow \mathbf{S}$ from sentence 1 to 2. The entity grid is then encoded as a feature vector $\mathbf{\Phi}(d) = (p_1(d), p_2(d), \ldots, p_m(d))$, where $p_t(d)$ is the normalized frequency of the transition t in the entity grid, and m is the number of transitions with length no more than a predefined length k. $p_t(d)$ is computed as the number of occurrences of t in the entity grid of document d, divided by the total number of transitions of the same length. Moreover, entities are differentiated by their salience — an entity is deemed to be salient if it occurs at least l times in the text, and non-salient otherwise — and transitions are computed

[1]Text elements are considered as a single entity of multiple mentions if they refer to the same object or concept in the world, even if they have different textual realizations, e.g., *dollar* in S_1 and *U.S. unit* in S_3 refer to the same entity. However, as can be imagined, existing automatic corereference resolution tools are usually unable to capture such sophisticated cases.

S_1: [**The dollar**]$_{\mathbf{S}}$ finished lower [**yesterday**]$_{\mathbf{X}}$, after tracking [**another rollercoaster session**]$_{\mathbf{O}}$ on [**Wall Street**]$_{\mathbf{X}}$.
S_2: [**Concern**]$_{\mathbf{S}}$ about [**the volatile U.S. stock market**]$_{\mathbf{X}}$ had faded in [**recent sessions**]$_{\mathbf{X}}$, and [**traders**]$_{\mathbf{S}}$ appeared content to let [**the dollar**]$_{\mathbf{S}}$ languish in [**a narrow range**]$_{\mathbf{X}}$ until [**tomorrow**]$_{\mathbf{X}}$, when [**the preliminary report**]$_{\mathbf{S}}$ on [**third-quarter U.S. gross national product**]$_{\mathbf{X}}$ is released.
S_3: But [**seesaw gyrations**]$_{\mathbf{S}}$ in [**the Dow Jones Industrial Average**]$_{\mathbf{X}}$ [**yesterday**]$_{\mathbf{X}}$ put [**Wall Street**]$_{\mathbf{O}}$ back in [**the spotlight**]$_{\mathbf{X}}$ and inspired [**market participants**]$_{\mathbf{O}}$ to bid [**the U.S. unit**]$_{\mathbf{O}}$ lower.

	dollar	**yesterday**	**session**	**Wall Street**	**concern**	**market**	**sessions**	**traders**	**range**	**tomorrow**	**report**	**GNP**	**gyrations**	**DJIA**	**spotlight**	**participants**
S_1	S	X	O	X	-	-	-	-	-	-	-	-	-	-	-	-
S_2	S	-	-	-	S	X	S	X	X	X	S	X	-	-	-	-
S_3	O	X	-	O	-	-	-	-	-	-	-	-	S	X	X	O

Table 6.1: The entity grid for the example text with three sentences and eighteen entities. Grid cells correspond to grammatical roles: subjects (**S**), objects (**O**), or neither (**X**).

separately for salient and non-salient entities.

6.1.2 Evaluation Tasks

In their work, to evaluate their entity-based model, B&L used two tasks: *sentence ordering* and *summary coherence rating*.

In sentence ordering, a set of random permutations is created for each source document, and the learning procedure is conducted on this synthetic mixture of coherent and incoherent documents. The rationale behind this permutation-based approach is to simulate situations where, given a set of predefined information-bearing items, the problem is to determine the best order to present those items. B&L experimented on two datasets: news articles on the topic of earthquakes (*Earthquakes*) and narratives on the topic of aviation accidents (*Accidents*). A training data instance is constructed as a pair consisting of a source document and one of its

random permutations, and the permuted document is always considered to be less coherent than the source document. The entity transition features are then used to train a support vector machine ranker (Joachims, 2002) to rank the source documents higher than the permutations. The model is tested on a different set of source documents and their permutations, and the performance is evaluated as the fraction of correct pairwise rankings in the test set.

In summary coherence rating, a similar experimental framework is adopted. However, in this task, rather than training and evaluating on a set of synthetic data, system-generated summaries and human-composed reference summaries from the Document Understanding Conference (DUC 2003) were used. Human annotators were asked to give a coherence score on a seven-point scale for each item. The pairwise ranking preferences between summaries generated from the same input document cluster (excluding the pairs consisting of two human-written summaries) are used by a support vector machine ranker to learn a discriminant function to rank each pair according to their coherence scores.

6.1.3 Extensions

Since the initial publication of B&L's entity-based local coherence model, a number of extensions have been proposed, the majority of which are focused on enriching the original feature set. For example, Filippova and Strube (2007) applied B&L's entity-based coherence model in the task of *sentence ordering* on TüBa-D/Z, a German corpus of newspaper articles with manual syntactic, morphological, and NP coreference annotations provided. They further clustered entities by semantic relatedness as computed by the WikiRelated! API (Strube and Ponzetto, 2006); however, the improvement was not significant.

Cheung and Penn (2010) adapted the standard entity-based coherence model to the same German corpus as Filippova and Strube (2007), but replaced the original linguistic dimension used by B&L — grammatical role — with topological field information, and showed that for German text, such a modification improves accuracy.

Elsner and Charniak (2011) augmented the original features used in the standard entity-based coherence model with a large number of entity-specific features, and their extension significantly outperformed the standard model on two tasks: *sentence ordering* and *sentence insertion*. The task of *sentence insertion* is to test whether the system prefers to re-insert a sentence to its original place when removed from the text.

We observe that B&L's entity-based coherence model, from the observation that their original model was trained using only pairs of coherent and incoherent documents: a source document vs. its permutation, or a human-generated summary vs. a machine-generated summary. However, coherence is matter of degree rather than a binary distinction, so a model based only on such pairwise rankings is insufficiently fine-grained and cannot capture the subtle differences in coherence between the less-coherent documents. Therefore, we propose an extension by learning from more fine-grained coherence preferences in training data. In Section 6.2, I will present our extension to the entity-based local coherence model by multiple ranks.

6.2 Extending the Entity-based Coherence Model with Multiple Ranks

As introduced in Section 6.1.3, previous extensions of the entity-based coherence model are primarily focused on modifying or enriching the original feature set by incorporating other document information. By contrast, we wish to refine the learning procedure in a way such that the resulting model will be able to evaluate coherence on a more fine-grained level. Specifically, we propose a concise extension to the standard entity-based coherence model by learning not only from the original document and its corresponding permutations but also from ranking preferences among the permutations themselves[2].

We show that this can be done by assigning a suitable objective score for each permutation indicating its dissimilarity from the original one. We call this a *multiple-rank model* since

[2]This work is first published as Feng and Hirst (2012a).

we train our model on a multiple-rank basis, rather than taking the original pairwise ranking approach. This extension can also be easily combined with other extensions by incorporating their enriched feature sets. We show that our multiple-rank model outperforms B&L's basic model on two tasks, *sentence ordering* and *summary coherence rating*, evaluated on the same datasets as used by Barzilay and Lapata (2008).

6.2.1 Experimental Design

Following Barzilay and Lapata (2008), we wish to train a discriminative model to give the correct ranking preference between two documents in terms of their degree of coherence. We experiment with the same two tasks as in their work: *sentence ordering* and *summary coherence rating*.

6.2.1.1 Sentence Ordering

In the standard entity-based model, a set of random permutations is created for each source document, and the learning procedure is conducted on this synthetic mixture of coherent and incoherent documents. However, a model learned from the pairwise rankings between an original document and its permutation is not sufficiently fine-grained, since the subtle differences within the permutations are not learned. Our major contribution is to further differentiate among the permutations generated from the same source documents, rather than simply treating them all as being of the same degree of coherence.

Our fundamental assumption is that there exists a canonical ordering for the sentences of a document; therefore we can approximate the degree of coherence of a document by the similarity between its actual sentence ordering and that canonical sentence ordering. Practically, we automatically assign a proper objective score for each permutation to estimate its dissimilarity from the source document (see Section 6.2.2). By learning from all the pairs across a source document and its permutations, the effective size of the training data is increased while no further manual annotation is required, which is favorable in real applications when

available samples with manually annotated coherence scores are usually limited. For r source documents each with m random permutations, the number of training instances in the standard entity-based model is therefore $r \times m$, while in our multiple-rank model learning process, it is $r \times \binom{m+1}{2} \approx \frac{1}{2} r \times m^2 > r \times m$, when $m > 2$.

6.2.1.2 Summary Coherence Rating

Compared to the standard entity-based coherence model, our major contribution in this task is to show that by automatically assigning an objective score for each machine-generated summary, which estimates its dissimilarity from the human-generated summary from the same input document cluster, we are able to achieve competitive or even superior performance with B&L's model without knowing the true coherence score given by human judges.

It is crucial to evaluate our multiple-rank model in this task, since the task of summary coherence rating can more precisely approximate the coherence violations that the reader might encounter in real machine-generated texts, while the sentence-ordering task is only partially capable of doing so.

6.2.2 Ordering Metrics

As mentioned previously, the subtle differences among the permutations of the same source document can be used to refine the model learning process. Considering an original document $\boldsymbol{d}$ and one of its permutations, we call $\boldsymbol{\sigma} = (1, 2, \ldots, N)$ the *reference ordering*, which is the sentence ordering in $\boldsymbol{d}$, and $\boldsymbol{\pi} = (o_1, o_2, \ldots, o_N)$ the *test ordering*, which is the sentence ordering in that permutation, where N is the number of sentences being rendered in both documents.

In order to approximate different degrees of coherence among the set of permutations which bear the same content, we need a suitable metric to quantify the dissimilarity between the test ordering $\boldsymbol{\pi}$ and the reference ordering $\boldsymbol{\sigma}$. Such a metric needs to satisfy the following criteria: (1) It can be automatically computed while being highly correlated with human judgments of coherence, since additional manual annotation is certainly undesirable. (2) It depends on

the particular sentence ordering in a permutation while remaining independent of the entities within the sentences; otherwise our multiple-rank model might be trained to fit particular probability distributions of entity transitions rather than true coherence preferences.

In our work we use three different metrics: *τ distance*, *average continuity*, and *edit distance*.

Kendall's τ distance: This metric has been widely used in evaluation of sentence ordering (Bollegala et al., 2006; Lapata, 2003, 2006; Madnani et al., 2007)[3]. It measures the disagreement between two orderings $\boldsymbol{\sigma}$ and $\boldsymbol{\pi}$ in terms of the number of inversions of adjacent sentences necessary to convert one ordering into another. Kendall's τ distance is defined as

$$\tau = \frac{2m}{N(N-1)},$$

where m is the number of sentence inversions necessary to convert $\boldsymbol{\sigma}$ to $\boldsymbol{\pi}$, which can be efficiently computed using a divide-and-conquer method.

Average continuity (AC): Following Zhang (2011), we use average continuity as the second ordering metric. It was first proposed by Bollegala et al. (2006). This metric estimates the quality of a particular sentence ordering by the number of correctly arranged continuous sentences, compared to the reference ordering. For example, if $\boldsymbol{\pi} = (\ldots, 3, 4, 5, 7, \ldots, o_N)$, then $\{3, 4, 5\}$ is considered as continuous while $\{3, 4, 5, 7\}$ is not. Average continuity is calculated as

$$AC = \exp\left(\frac{1}{n-1}\sum_{i=2}^{n}\log\left(P_i + \alpha\right)\right),$$

where $n = \min(4, N)$ is the maximum number of continuous sentences to be considered, and $\alpha = 0.01$. P_i is the proportion of continuous sentences of length i in $\boldsymbol{\pi}$ that are also continuous in the reference ordering $\boldsymbol{\sigma}$. To represent the *dis*-similarity between the two orderings $\boldsymbol{\pi}$ and $\boldsymbol{\sigma}$, we use its complement $AC' = 1 - AC$, such that the larger AC' is, the more dissimilar two

[3]Filippova and Strube (2007) found that their performance dropped when using this metric for longer rankings; but they were using data in a different language and with manual annotations, so its effect on our datasets is worth trying nonetheless.

orderings are[4].

Edit distance (ED): Edit distance is a commonly used metric in information theory to measure the difference between two sequences. Given a test ordering π, its edit distance is defined as the minimum number of edits (i.e., insertions, deletions, and substitutions) needed to transform it into the reference ordering σ. For permutations, the edits are essentially movements, which can be considered as equal numbers of insertions and deletions.

6.2.3 Experiment 1: Sentence Ordering

Our first set of experiments is on sentence ordering (discussed in Section 6.2.1.1). Following Barzilay and Lapata (2008), we use all transitions of length ≤ 3 for feature extraction. In addition, we explore three specific aspects in our experiments: rank assignment, entity extraction, and permutation generation.

6.2.3.1 Rank Assignment

In our multiple-rank model, pairwise rankings between a source document and its permutations are extended into a longer ranking with multiple ranks. We assign a rank to a particular permutation, based on the result of applying a chosen ordering metric in Section 6.2.2 (τ, AC, or ED) to the sentence ordering in that permutation.

We experiment with two different approaches to assigning ranks to permutations, while each source document is always assigned a zero (the highest) rank.

In the **raw** option, we rank the permutations directly by their dissimilarity scores to form a full ranking for the set of permutations generated from the same source document. Since a full ranking might be too sensitive to noise in training, we also experiment with the **stratified** option.

In the **stratified** option: C ranks are assigned to the permutations generated from the same source document. The permutation with the smallest dissimilarity score is assigned the same

[4]We will refer to AC' as AC from now on.

(the highest) rank as the source document, and the one with the largest score is assigned the lowest ($C - 1$) rank; then ranks of other permutations are uniformly distributed in this range according to their raw dissimilarity scores. We experiment with 3 to 6 ranks (the case where $C = 2$ reduces to the standard entity-based model).

6.2.3.2 Entity Extraction

B&L's best results were achieved by employing an automatic coreference resolution tool (Ng and Cardie, 2002) for extracting entities from a source document, and the permutations were generated only afterwards — entity extraction from a permuted document depends on knowing the correct sentence order and the oracular entity information from the source document — since resolving coreference relations in permuted documents is too unreliable for an automatic tool.

We implement our multiple-rank model with full coreference resolution using Ng and Cardie's coreference resolution system, and entity extraction approach as described above — the **Coreference**+ condition. However, as argued by Elsner and Charniak (2011), to better simulate the real situations where human readers might encounter in machine-generated documents, such oracular information should not be taken into account. Therefore we also employ an alternative approach for entity extraction: use no coreference resolution, i.e., group head noun clusters by simple string matching — the **Coreference**− condition.

6.2.3.3 Permutation Generation

The quality of the model learned depends on the set of permutations used in training. We are not aware of how B&L's permutations were generated, but we assume that they are generated in a perfectly random fashion.

However, in reality, the probabilities of seeing documents with different degrees of coherence are not equal. For example, in an essay-scoring task, if the target group is (near) native speakers with sufficient education, we should expect their essays to be less incoherent — most

of the essays will be coherent in most parts, with only a few minor problems regarding discourse coherence. In such a setting, the performance of a model trained from permutations generated from a uniform distribution may suffer some accuracy loss.

Therefore, in addition to the set of permutations used by B&L (PS_{BL}), we create another set of permutations for each source document (PS_M) by assigning most of the probability mass to permutations which are mostly similar to the original source document. Besides its capability of better approximating real-life situations, training our model on permutations generated in this way has another benefit: in the standard entity-based model, all permuted documents are treated as incoherent; thus there are many more incoherent training instances than coherent ones (typically the proportion is 20:1). In contrast, in our multiple-rank model, permuted documents are assigned different ranks to further differentiate the different degrees of coherence within them. By doing so, our model will be able to learn the characteristics of a coherent document from those near-coherent documents as well, and therefore the problem of lacking coherent instances can be mitigated.

Our permutation generation algorithm is shown in Algorithm 3, where $\alpha = 0.05, \beta = 5.0$, $MAX\_NUM = 50$, and K and K' are two normalization factors to make $p(swap\_num)$ and $p(i, j)$ proper probability distributions. For each source document, we create the same number of permutations as PS_{BL}.

Algorithm 3 Permutation Generation.

Input: $S_1, S_2, \ldots, S_N$; $\boldsymbol{\sigma} = (1, 2, \ldots, N)$

1: Choose a number of sentence swaps *swap_num* with probability $e^{-\alpha \times swap\_num}/K$
2: **for** $i = 1 \rightarrow swap\_num$ **do**
3: Swap a pair of sentences (S_i, S_j) with probability $p(i, j) = e^{-\beta \times |i-j|}/K'$

Output: $\boldsymbol{\pi} = (o_1, o_2, \ldots, o_N)$

6.2.3.4 Results

In this task, we use the same two sets of source documents (*Earthquakes* and *Accidents*, see Section 6.1.2) as B&L. Each contains 200 source documents, equally divided between training

and test sets, with up to 20 permutations per document. We conduct experiments on these two domains separately. For each domain, we accompany each source document with two different sets of permutations: the one used by B&L (PS_{BL}), and the one generated from our model described in Section 6.2.3.3 (PS_M). We train our multiple-rank model and B&L's standard two-rank model on each set of permutations using the SVM^{rank} package (Joachims, 2006), and evaluate both systems on their test sets. Accuracy is measured as the fraction of correct pairwise rankings for the test set.

Full Coreference Resolution with Oracular Information (Coreference+) In this experiment, we implement B&L's fully-fledged standard entity-based coherence model, and extract entities from permuted documents using oracular information from the source documents (see Section 6.2.3.2).

Results are shown in Table 6.2. For each test situation, we list the best accuracy for each chosen scoring metric, with the corresponding rank assignment approach. C represents the number of ranks used in stratifying raw scores (N if using **raw** configuration, see Section 6.2.3.1 for details). Baselines are accuracies trained using the standard entity-based coherence model[5].

Our model outperforms the standard entity-based model on both permutation sets for both datasets. The improvement is not significant when trained on the permutation set PS_{BL}, and is only achieved using one of the three metrics; but when trained on PS_M (the set of permutations generated from our biased model), our model's performance significantly exceeds B&L's[6] for all three metrics, especially as their model's performance drops for dataset *Accidents*.

From these results, we see that in the ideal situation where we extract entities and resolve their coreference relations based on the oracular information from the source document, our

[5]There are discrepancies between our reported accuracies and those of Barzilay and Lapata (2008). The differences are due to the fact that we use a different parser: the Stanford dependency parser (de Marneffe et al., 2006), and might have extracted entities in a slightly different way than theirs, although we keep other experimental configurations as close as possible to theirs. But when comparing our model with theirs, we always use the exact same set of features, so the absolute accuracies do not matter.

[6]Following Elsner and Charniak (2011), we use the Wilcoxon signed-rank test for significance.

Condition: Coreference+					
Perms	Metric	*Earthquakes*		*Accidents*	
		C	Acc	*C*	Acc
PS_{BL}	τ	3	79.5	3	82.0
	AC	4	85.2	3	**83.3**
	ED	3	**86.8**	6	82.2
	Baseline		85.3		83.2
PS_M	τ	3	**86.8**	3	**85.2***
	AC	3	**85.6**	1	**85.4***
	ED	*N*	**87.9***	4	**86.3***
	Baseline		85.3		81.7

Table 6.2: Accuracies (%) of extending the standard entity-based coherence model with multiple-rank learning in sentence ordering using **Coreference+** option. Accuracies which are significantly better than the baseline ($p < .05$) are indicated by *.

model is effective in terms of improving ranking accuracies, especially when trained on our more realistic permutation sets PS_M.

No Coreference Resolution (Coreference−) In this experiment, we do not employ any coreference resolution tool, and simply cluster head nouns by string matching. Results are shown in Table 6.3.

Even with such a coarse approximation of coreference resolution, our model is able to achieve around 85% accuracy in most test cases, except for dataset *Earthquakes*, training on PS_{BL} gives poorer performance than the standard model by a small margin. But such inferior performance should be expected. As analyzed by Barzilay and Lapata (2008), in dataset *Earthquakes*, entities tend to repeat on a pronoun basis. Therefore, coreference resolution is crucial to this dataset, because simple string matching would introduce too much noise into training, especially when our model wants to train a more fine-grained discriminative system than B&L's. However, we can see from the result of training on PS_M, if the permutations used in training do not involve swapping sentences which are too far away, the resulting noise would be reduced, and our model would outperform theirs. And for dataset *Accidents*, our

Condition: Coreference−					
Perms	Metric	*Earthquakes*		*Accidents*	
		C	Acc	*C*	Acc
PS_{BL}	τ	4	82.8	*N*	**82.0**
	AC	3	78.0	3	**84.2****
	ED	*N*	78.2	3	**82.7***
	Baseline		**83.7**		80.1
PS_M	τ	3	**86.4****	*N*	**85.7****
	AC	4	**84.4***	*N*	**86.6****
	ED	5	**86.7****	*N*	**84.6****
	Baseline		82.6		77.5

Table 6.3: Accuracies (%) of extending the standard entity-based coherence model with multiple-rank learning in sentence ordering using **Coreference**− option. Accuracies which are significantly better than the baseline are indicated by * ($p < .05$) and ** ($p < .01$).

model consistently outperforms the baseline model by a large margin (with significance test at $p < .01$).

6.2.3.5 Conclusions for Sentence Ordering

Considering the particular scoring metric used in training, we find that *edit distance* usually stands out from the other two metrics. *Kendall's τ distance* proves to be a fairly weak metric, which is consistent with the findings of Filippova and Strube (2007). In addition, when our model significantly outperforms the baseline, usually larger C (4 or 5, or even using **raw** option) results in better accuracies.

Combining our multiple-rank model with simple string matching for entity extraction is a robust option for coherence evaluation, regardless of the particular distribution of permutations used in training, and it significantly outperforms the baseline in most conditions.

6.2.4 Experiment 2: Summary Coherence Rating

In the summary coherence rating task, we are dealing with a mixture of multi-document summaries generated by systems and written by humans. B&L did not assume a simple binary distinction among the summaries generated from the same input document cluster; rather, they had human judges give scores for each summary based on its degree of coherence (see Section 6.2.1.2). Therefore, it seems that the subtle differences among incoherent documents (system-generated summaries in this case) have already been learned by their model.

But we wish to see if we can replace human judgments by our computed dissimilarity scores so that the original supervised learning is converted into unsupervised learning and yet retains competitive performance. However, given a summary, computing its dissimilarity score is a bit involved, due to the fact that we do not know its correct sentence order. To tackle this problem, we employ a simple sentence alignment between a system-generated summary and a human-written summary originating from the same input document cluster. Given a system-generated summary $D_s = (S_{s1}, S_{s2}, \ldots, S_{sn})$ and its corresponding human-written summary $D_h = (S_{h1}, S_{h2}, \ldots, S_{hN})$ (here it is possible that $n \neq N$), we treat the sentence ordering $(1, 2, \ldots, N)$ in D_h as $\boldsymbol{\sigma}$ (the original sentence ordering), and compute $\boldsymbol{\pi} = (o_1, o_2, \ldots, o_n)$ based on D_s. To compute each o_i in $\boldsymbol{\pi}$, we find the most similar sentence S_{hj}, $j \in [1, N]$ in D_h by computing their cosine similarity; if all sentences in D_h have zero cosine similarity with S_{si}, we assign -1 to o_i.

Once $\boldsymbol{\pi}$ is known, we can compute its "dissimilarity" from $\boldsymbol{\sigma}$ using a chosen metric. But because now $\boldsymbol{\pi}$ is not guaranteed to be a permutation of $\boldsymbol{\sigma}$ (there may be repetition or missing values, i.e., -1, in $\boldsymbol{\pi}$), Kendall's τ cannot be used, and we use only *average continuity* and *edit distance* these two as scoring metrics in this experiment.

The remaining experimental configuration is the same as that of Barzilay and Lapata (2008), with the optimal transition length set to ≤ 2.

6.2.4.1 Results

As explained in Section 6.2.1.2, we employ a simple sentence alignment between a system-generated summary and its corresponding human-written summary to construct a test ordering π and calculate its dissimilarity between the reference ordering σ from the human-written summary. In this way, we convert B&L's supervised learning model into a fully unsupervised model, since human annotations for coherence scores are not required. We use the same dataset as Barzilay and Lapata (2008), which includes multi-document summaries from 16 input document clusters generated by five systems, along with reference summaries composed by humans.

In this experiment, we consider only *average continuity (AC)* and *edit distance (ED)* as scoring metrics, with **raw** configuration for rank assignment, and compare our multiple-rank model with the standard entity-based model using either full coreference resolution by running the coreference resolution tool on all documents or no resolution for entity extraction. We train both models on the ranking preferences (144 in all) among summaries originating from the same input document cluster using the SVM^{rank} package (Joachims, 2006), and test on two different test sets: *same-cluster test* and *full test*. *Same-cluster* test is the one used by Barzilay and Lapata (2008), in which only the pairwise rankings (80 in all) between summaries originating from the same input document cluster are tested; we also experiment with *full-test*, in which pairwise rankings (1520 in all) between all summary pairs excluding two human-written summaries are tested.

Results are shown in Table 6.4. **Coreference+** and **Coreference−** denote the configuration of using full coreference resolution or no resolution separately. First, clearly for both models, performance on *full-test* is inferior to that on *same-cluster* test, but our model is still able to achieve competitive performance with the standard model, even if our fundamental assumption about the existence of canonical sentence ordering in similar documents may break down on those test pairs not originating from the same input document cluster. Secondly, for the baseline model, using **Coreference−** configuration yields better accuracy in this task (80.0% vs. 78.8%

Entities	Metric	Same-cluster	Full
Coreference+	*AC*	**82.5**	**72.6***
	ED	**81.3**	**73.0****
	Baseline	78.8	70.9
Coreference−	*AC*	76.3	72.0
	ED	78.8	71.7
	Baseline	**80.0**	**72.3**

Table 6.4: Accuracies (%) of extending the standard entity-based coherence model with multiple-rank learning in summary rating. Baselines are the results of B&L's standard entity-based coherence model. Accuracies which are significantly better than the corresponding baseline are indicated by * ($p < .05$) and ** ($p < .01$).

on *same-cluster test*, and 72.3% vs. 70.9% on *full test*), which is consistent with the findings of Barzilay and Lapata (2008). But our multiple-rank model seems to favor the **Coreference+** configuration, and our best accuracy even exceeds B&L's best when tested on the same set: 82.5% vs. 80.0% on *same-cluster test*, and 73.0% vs. 72.3% on *full test*.

When our model performs poorer than the baseline (using **Coreference−** configuration), the difference is not significant, which suggests that our multiple-rank model with unsupervised score assignment can remain competitive with the standard model, which requires human annotations to obtain a more fine-grained coherence spectrum. This observation is consistent with previous discovery that human-generated summaries look quite extractive (Banko and Vanderwende, 2004). On the other hand, as discovered by Cheung and Penn (2013), human-generated summaries are more abstractive and aggregate information from multiple source text sentences, and are relatively less extractive than machine-generated summaries. It suggests that our unsupervised score assignment, although can approximate the degree of coherence to a fairly well degree, might not capture other important aspects of high-quality summaries, such as responsiveness and linguistic quality.

6.2.5 Conclusion

In this section, I presented our extension to the popular coherence model of Barzilay and Lapata (2008) by adopting a multiple-rank learning approach. This is inherently different from other extensions of this model, in which the focus is on enriching the set of features for entity-grid construction, whereas we simply keep their original feature set intact, and manipulate only their learning methodology. We show that this concise extension is effective and able to outperform B&L's standard model in various experimental setups, except for certain experimental configurations that are clearly unsuitable (see discussion in Section 6.2.3.5).

We experimented with two tasks: *sentence ordering* and *summary coherence rating*, following B&L's original framework. In sentence ordering, we also explored the influence of removing the oracular component in their original model and dealing with permutations generated from different distributions, showing that our model is robust for different experimental situations if using proper configurations. In summary coherence rating, we further extended their model in a way that their original supervised learning can be converted into unsupervised learning with competitive or ever superior performance.

Our multiple-rank learning model can be easily adapted into other extended entity-based coherence models with their enriched feature sets, and further improvement in ranking accuracies should be expected.

6.3 Using Discourse Relations for the Evaluation of Coherence

As introduced in Section 6.2, distinct from other previous extensions, the novelty of our extension to B&L's entity-based local coherence model lies in its efficient learning procedure by multiple ranks. In this section, we will explore another extension in an orthogonal dimension, i.e., the enrichment of the feature set. As discussed in Section 6.1.3, most previous work on

extending B&L's model fall into this dimension as well. However, those previously proposed enriched feature sets are usually application-specific, i.e., it requires certain expertise and intuition to conceive good features. In contrast, in our work[7], we seek insight on better feature encoding from discourse parsing, as it arguably provides a general solution to many problems in the analysis of discourse, including the evaluation of coherence. Therefore, one can expect that discourse parsing provides useful information for the evaluation of text coherence, because, essentially, the existence and the distribution of discourse relations are the basis of the coherence in a text.

6.3.1 Discourse Role Matrix and Discourse Role Transitions

In fact, there is already evidence showing that discourse relations can help better capture text coherence. Lin et al. (2011) use a PDTB-style discourse parser to identify discourse relations in the text, and they represent a text by entities and their associated discourse roles in each sentence.

For example, consider the same example text as in Table 6.1; its occurring PDTB-style discourse relations are shown in Figure 6.1. As introduced in Section 1.2, PDTB-style discourse parsing takes a lexically grounded, predicate-argument approach to extract discourse relations in the text, by identifying the (potentially implicit) discourse connective and its two arguments. For example, the explicit CONJUNCTION within S_2 is lexicalized by the connective *and*, and the clauses $C_{2.1}$ and $C_{2.2}$ its two arguments.

A fragment of Lin et al.'s PDTB-style discourse role matrix is shown in Table 6.5. In the matrix, columns correspond to the entities in the text and rows represent the contiguous sentences. Each cell $\langle E_i, S_j \rangle$ corresponds to the set of discourse roles that the entity E_i serves as in sentence S_j. For example, the entity *yesterday* from S_3 takes part in Arg2 of the last relation, so the cell $\langle yesterday, S_3 \rangle$ contains the role CONTRAST.Arg2. An entry may be empty (with a symbol *nil*, as in $\langle yesterday, S_2 \rangle$) or contain multiple discourse roles (as in $\langle dollar, S_2 \rangle$). A

[7]First published as Feng et al. (2014).

S_1: The dollar finished lower yesterday, after tracking another rollercoaster session on Wall Street.
S_2: [Concern about the volatile U.S. stock market had faded in recent sessions]$C_{2.1}$, [and traders appeared content to let the dollar languish in a narrow range until tomorrow, when the preliminary report on third-quarter U.S. gross national product is released.]$C_{2.2}$
S_3: But seesaw gyrations in the Dow Jones Industrial Average yesterday put Wall Street back in the spotlight and inspired market participants to bid the U.S. unit lower.

Three discourse relations are presented in the text above:

1. Implicit ENTREL between S_1 as Arg1, and S_2 as Arg2.
2. Explicit CONJUNCTION within S_2: $C_{2.1}$ as Arg1, $C_{2.2}$ as Arg2, with *and* as the connective.
3. Explicit CONTRAST between S_2 as Arg1 and S_3 as Arg2, with *but* as the connective.

Figure 6.1: An example text fragment composed of three sentences, and its PDTB-style discourse relations.

discourse role transition is defined as the sequence of discourse roles that a particular entity serves in n adjacent sentences. Lin et al. considered discourse role transitions of length 2 or 3, e.g., ENTREL.Arg1 → CONJUNCTION.Arg2 and ENTREL.Arg1 → *nil* → *Contrast*.Arg2, and calculated their frequencies with respect to the matrix. For example, the frequency of ENTREL.Arg1 → CONJUNCTION.Arg2 is $1/24 = 0.042$ in Table 6.5.

Contrasting Table 6.5 (discourse role matrix) with Table 6.1 (entity grid), in each entry of the discourse role matrix, the original syntactic role of each entity (**S**, **O**, **X**, or, −, as in the entity grid) is replaced by the discourse roles of each entity. Accordingly, a discourse role transition is a continuous subsequence extracted from each column in the discourse role matrix.

In their experiments, using discourse roles alone, Lin et al.'s model performs very similar to or even better than B&L's basic model. Combining their discourse role features with B&L's entity-based transition features further improves the performance.

However, PDTB-style discourse relations encode only very shallow discourse structures,

	dollar	**yesterday**	**session**	**Wall Street**	**concern**
S_1	ENTREL.Arg1	ENTREL.Arg1	ENTREL.Arg1	ENTREL.Arg1	*nil*
S_2	ENTREL.Arg2 CONJ.Arg2 CONTRAST.Arg1	*nil*	*nil*	*nil*	ENTREL.Arg2 CONJ.Arg1 CONTRAST.Arg1
S_3	CONTRAST.Arg2	CONTRAST.Arg2	*nil*	CONTRAST.Arg2	*nil*

Table 6.5: A fragment of Lin et al.'s PDTB-style discourse role matrix for the example text with the first six entities across three sentences.

i.e., the relations are mostly local, e.g., within a single sentence or between two adjacent sentences. Therefore, in general, features derived from PDTB-style discourse relations cannot capture long discourse dependency, and thus the resulting model is still limited to being a local model. Nonetheless, long-distance discourse dependency could be quite useful for capturing text coherence from a global point of view.

Therefore, in our work, we study the effect of deep hierarchical discourse structure in the evaluation of text coherence. In order to conduct a direct comparison between a model with features derived from deep hierarchical discourse relations and a model with features derived from shallow discourse relations only, we adopt two separate approaches:

1. We implement a model with features derived from RST-style discourse relations, and compare it against a model with features derived from PDTB-style relations.

2. In the framework of RST-style discourse parsing, we deprive the model of any information from higher-level discourse relations and compare its performance against the model utilizing the complete set of discourse relations. Moreover, as a baseline, we also re-implemented B&L's entity-based local coherence model, and we will study the effect of incorporating one of our discourse feature sets into this baseline model.

Therefore, we have four ways to encode discourse relation features, namely, entity-based, PDTB-style, full RST-style, and shallow RST-style.

6.3.1.1 Entity-based Feature Encoding

In entity-based feature encoding, our goal is to formulate a text into an entity grid, such as the one shown in Table 6.1, from which we extract entity-based local transitions. In our re-implementation of B&L, we use the same parameter settings as B&L's original model, i.e., the optimal transition length $k = 3$ and the salience threshold $l = 2$. However, when extracting entities in each sentence, e.g., *dollar*, *yesterday*, etc., we do not perform coreference resolution; rather, for better coverage, we follow the suggestion of Elsner and Charniak (2011) and extract all nouns (including non-head nouns) as entities. We use the Stanford dependency parser (de Marneffe et al., 2006) to extract nouns and their grammatical roles. This strategy of entity extraction also applies to the other three feature encoding methods to be described below.

6.3.1.2 PDTB-Style Feature Encoding

To encode PDTB-style discourse relations into the model, we parse the texts using an end-to-end PDTB-style discourse parser[8] developed by Lin et al. (2014). The F_1 score of this parser is around 85% for recognizing explicit relations and around 40% for recognizing implicit relations. A text is thus represented by a discourse role matrix in the same way as shown in Table 6.5. Most parameters in our PDTB-style feature encoding follow those of Lin et al. (2011): each entity is associated with the fully-fledged discourse roles, i.e., with type and argument information included; the maximum length of discourse role transitions is 3; and transitions are generated separately for salient and non-salient entities with a threshold set at 2. However, compared to Lin et al.'s model, there are two differences in our re-implementation, and evaluated on a held-out development set, these modifications are shown to be effective in improving the performance.

First, we differentiate between intra- and multi-sentential discourse relations, which is motivated by a finding in the field of RST-style discourse parsing — distributions of various discourse relation types are quite distinct between intra-sentential and multi-sentential instances

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

(see Chapter 4 and Joty et al. (2012)) — and we assume that a similar phenomenon exists for PDTB-style discourse relations. Therefore, we assign two sets of discourse roles to each entity: intra-sentential and multi-sentential roles, which are the roles that the entity plays in the corresponding intra- and multi-sentential relations.

Second, instead of Level-1 PDTB discourse relations (6 in total), we use Level-2 relations (18 in total) in feature encoding, so that richer information can be captured in the model, resulting in $18\times2 = 36$ different discourse roles with argument attached. We then generate four separate set of features for the combination of intra-/multi-sentential discourse relation roles, and salient/non-salient entities, among which transitions consisting of only *nil* symbols are excluded. Therefore, the total number of features in PDTB-style encoding is $4\times(36^2+36^3-2) \approx$ 192K.

6.3.1.3 Full RST-Style Feature Encoding

For RST-style feature encoding, we parse the texts using our end-to-end RST-style discourse parser (introduced in Chapter 5), which produces a discourse tree representation for each text, such as the one shown in Figure 6.2. For relation labeling, the overall accuracy of this discourse parser is 58%, evaluated on the RST-DT.

We encode the RST-style discourse relations in a similar fashion to PDTB-style encoding. However, since the definition of discourse roles depends on the particular discourse framework, here, we adapt Lin et al.'s PDTB-style encoding by replacing the PDTB-style discourse relations with RST-style discourse relations, and the argument information (Arg1 or Arg2) by the nuclearity information (nucleus or the satellite) in an RST-style discourse relation. More importantly, in order to reflect the hierarchical structure in an RST-style discourse parse tree, when extracting the set of discourse relations that an entity participates in, we find all those discourse relations that the entity appears in, excluding those relations that the entity does not occur in the main EDUs[9]. We then represent the role of the entity in each of these extracted

[9]The main EDUs of a discourse relation are the EDUs obtained by traversing the discourse subtree in which the relation of interest constitutes the root node, following the nucleus branches down to the leaves. For instance,

S_1: [The dollar finished lower yesterday,]e_1 [after tracking another rollercoaster session on Wall Street.]e_2
S_2: [Concern about the volatile U.S. stock market had faded in recent sessions,]e_3 [and traders appeared content to let the dollar languish in a narrow range until tomorrow,]e_4 [when the preliminary report on third-quarter U.S. gross national product is released.]e_5
S_3: [But seesaw gyrations in the Dow Jones Industrial Average yesterday put Wall Street back in the spotlight]e_6 [and inspired market participants to bid the U.S. unit lower.]e_7

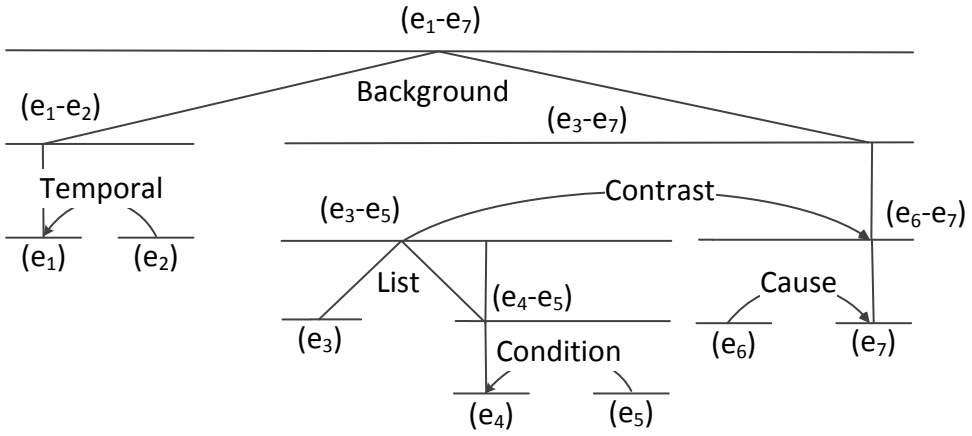


Figure 6.2: An example text fragment composed of seven EDUs, and its RST discourse tree representation.

	dollar	**yesterday**	**session**	**Wall Street**	**concern**
S_1	BACKGROUND.N TEMPORAL.N	BACKGROUND.N TEMPORAL.N	TEMPORAL.S	TEMPORAL.S	*nil*
S_2	LIST.N CONDITION.N CONTRAST.S	*nil*	*nil*	*nil*	LIST.N CONTRAST.S
S_3	CONTRAST.N BACKGROUND.N CAUSE.N	CAUSE.S	*nil*	CAUSE.S	*nil*

Table 6.6: A fragment of the full RST-style discourse role matrix for the example text with the first five entities across three sentences.

discourse relations. In this way, we can encode long-distance discourse relations for the most relevant entities. For example, considering the RST-style discourse tree representation in Figure 1.1, we encode the BACKGROUND relation for the entities *dollar* and *yesterday* in S_1, as well as the entity *dollar* in S_3, but not for the remaining entities in the text, even though the BACKGROUND relation covers the whole text. The corresponding full RST-style discourse role matrix for the example text is shown in Table 6.6.

Similarly, we differentiate between intra- and multi-sentential discourse relations; we use 18 coarse-grained classes of RST-style relations (introduced in Section 1.1.2) in feature encoding; the optimal transition length k is 3; and the salience threshold l is 2. The total number of features in RST-style encoding is therefore $4 \times (34^2 + 34^3 - 2) \approx 162\text{K}$[10], which is roughly the same as that in PDTB-style feature encoding.

6.3.1.4 Shallow RST-Style Feature Encoding

Shallow RST-style encoding is almost identical to RST-style encoding, except that, when we derive discourse roles, we consider shallow discourse relations only. To be consistent with

for the RST discourse tree in Figure 1.1, the main EDUs of the BACKGROUND relation on the top level are $\{e_1, e_7\}$, and the main EDUs of the LIST relation among (e_3-e_5) are $\{e_3, e_4\}$.

[10]Although we have 18 types of RST-style relation, there are only 34 distinct discourse roles, because the two relations JOINT and SAME-UNIT can only be multi-nuclear (with the left and right text spans being both nucleus). Therefore, while other relations have both REL.N and REL.S as possible discourse roles, these two relations do not have the satellite version.

the majority of PDTB-style discourse relations, we define shallow discourse relations as those relations which hold between text spans of the same sentence, or between two adjacent sentences. For example, in Figure 6.2, the BACKGROUND relation between (e_1-e_2) and (e_3-e_7) is not a shallow discourse relation (it holds between a single sentence and the concatenation of two sentences), and thus will be excluded from shallow RST-style feature encoding.

6.3.2 Experiments

To test our hypothesis about the usefulness of deep discourse structures, we conduct two series of experiments on two different datasets, each of which simulates a sub-task in the evaluation of text coherence, i.e., **sentence ordering** and **essay scoring**.

Since text coherence is a matter of degree rather than a binary classification, in both evaluation tasks we formulate the problem as a pairwise preference ranking problem. Specifically, given a set of texts with different degrees of coherence, we train a ranker which learns to prefer a more coherent text over a less coherent counterpart. Accuracy is therefore measured as the fraction of correct pairwise rankings as recognized by the ranker. In our experiments, we use the SVM^{light} package[11] (Joachims, 1999) with the ranking configuration, and all parameters are set to their default values.

6.3.2.1 Sentence Ordering

The task of sentence ordering, which has been extensively studied in previous work, attempts to simulate the situation where, given a predefined set of information-bearing items, we need to determine the best order in which the items should be presented. As argued by Barzilay and Lapata (2005), sentence ordering is an essential step in many content-generation components, such as multi-document summarization.

In this task, we use a dataset consisting of a subset of the Wall Street Journal (WSJ) corpus, in which the minimum length of a text is 20 sentences, and the average length is 41 sentences.

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

	Min	Max	Average	SdtVar
Sentences	20	163	41.04	18.73
Kendall's τ distance	0.14	0.80	0.50	0.06

Table 6.7: The characteristics of the 735 source texts and the corresponding 14,700 permutations in the WSJ dataset.

For each text, we create 20 random permutations by shuffling the original order of the sentences. In total, we have 735 source documents and $735 \times 20 = 14,700$ permutations. Because the RST-style discourse parser we use is trained on a fraction of the WSJ corpus, we remove the training texts from our dataset, to guarantee that the discourse parser will not perform exceptionally well on some particular texts. However, since the PDTB-style discourse parser we use is trained on almost the entire WSJ corpus, we cannot do the same for the PDTB-style parser.

To further characterize our dataset, for each permuted text, we compute the Kendall's τ distance between its sentence order and the correct sentence order in its source text. Kendall's τ is a metric between 0 and 1, which measures the pairwise disagreement between two ranking lists. If two rankings are the same, $\tau = 0$, and $\tau = 1$ if one ranking is completely the reverse of the other. The larger τ is, the more dissimilar two rankings are. The average τ of our dataset is 0.50, with a standard deviation of 0.06, suggesting that the random permutations in our dataset are of medium randomness, i.e., they are not too dissimilar from the source texts, nor too similar either.

In this experiment, our learning instances are pairwise ranking preferences between a source text and one of its permutations, where the source text is always considered more coherent than its permutations. Therefore, we have $736 \times 20 = 14,700$ total pairwise rankings, and we conduct 5-fold cross-validation on five disjoint subsets. In each fold, one-fifth of the rankings are used for testing, and the rest for training.

6.3.2.2 Essay Scoring

The second task is essay scoring, and we use a subset of International Corpus of Learner English (ICLE) (Granger et al., 2009). The dataset consists of 1,003 essays about 34 distinct topics, written by university undergraduates speaking 14 native languages who are learners of English as a Foreign Language. Each essay has been annotated with an organization score from 1 to 4 at half-point increments by Persing et al. (2010). We use these organization scores to *approximate* the degrees of coherence in the essays. The average length of the essays is 32 sentences, and the average organization score is 3.05, with a standard deviation of 0.59.

In this experiment, our learning instances are pairwise ranking preferences between a pair of essays on the same topic written by students speaking the same native language, excluding pairs with the same organization score. In total, we have 22,362 pairwise rankings. Similarly, we conduct 5-fold cross-validations on these rankings.

In fact, the two datasets used in the two evaluation tasks reflect different characteristics by themselves. The WSJ dataset, although somewhat artificial due to the permuting procedure, is representative of texts with well-formed syntax. By contrast, the ICLE dataset, although not artificial, contains occasional grammatical or spelling errors, because the texts are written by non-native English speakers. Therefore, using these two distinct datasets allows us to evaluate our models in tasks where different challenges may be expected.

6.3.3 Results

We now demonstrate the performance of our models with discourse roles encoded in one of the three ways: PDTB-style, full RST-style, or shallow RST-style, and compare against their combination with our re-implemented B&L's entity-based local transition features. The evaluation is conducted on the two tasks, sentence ordering and essay scoring, and the accuracy is reported as the fraction of correct pairwise rankings averaged over 5-fold cross-validation.

The performance of various models is shown in Table 6.8. The first section of the table shows the results of our re-implementation of B&L's entity-based local coherence model, rep-

Model		sentence ordering	essay scoring
No discourse structure	Entity	95.1	66.4
Shallow discourse structures	PDTB	97.2	82.2
	PDTB&Entity	97.3	83.3
	Shallow RST	98.5	87.2
	Shallow RST&Entity	98.8	87.2
Deep discourse structures	Full RST	99.1	**88.3**
	Full RST&Entity	**99.3**	87.7

Table 6.8: Accuracy (%) of various models on the two evaluation tasks: sentence ordering and essay scoring. For sentence ordering, accuracy difference is significant with $p < .01$ for all pairs of models except between PDTB and PDTB&Entity. For essay scoring, accuracy difference is significant with $p < .01$ for all pairs of models except between shallow RST and shallow RST&Entity. Significance is determined with the Wilcoxon signed-rank test.

resenting the effect with **no** discourse structure encoded. The second section shows the results of four models with **shallow** discourse structures encoded, including the two basic models, PDTB-style and shallow RST-style feature encoding, and their combination with the entity-based feature encoding. The last section shows the results of our models with **deep** discourse structures encoded, including the RST-style feature encoding and its combination with the entity-base feature encoding. With respect to the performance, we observe a number of consistent patterns across both evaluation tasks.

First, with **no** discourse structure encoded, the entity-based model (the first row) performs the poorest among all models, suggesting that discourse structures are truly important and can capture coherence in a more sophisticated way than pure grammatical roles. Moreover, the performance gap is particularly large for essay scoring, which is probably due to the fact that, as argued by Persing et al. (2010), the organization score, which we use to approximate the degrees of coherence, is not equivalent to text coherence. Organization cares more about the logical development in the texts, while coherence is about lexical and semantic continuity; but discourse relations can capture the logical relations at least to some extent.

Secondly, with **deep** discourse structures encoded, the full RST-style model in the third section significantly outperforms ($p < .01$) the models with shallow discourse structures, i.e.,

the PDTB-style and shallow RST-style models in the middle section, confirming our intuition that deep discourse structures are more powerful than shallow structures. This is also the case when entity-based features are included.

Finally, considering the models in the middle section of the table, we can gain more insight into the difference between PDTB-style and RST-style encoding. As can be seen, even without information from the more powerful deep hierarchical discourse structures, shallow RST-style encoding still significantly outperforms PDTB-style encoding on both tasks ($p < .01$). This is primarily due to the fact that the discourse relations discovered by RST-style parsing have wider coverage of the text[12], and thus induce richer information about the text. Therefore, because of its ability to annotate deep discourse structures and its better coverage of discourse relations, RST-style discourse parsing is generally more powerful than PDTB-style parsing, as far as coherence evaluation is concerned.

However, with respect to combining full RST-style features with entity features, we have contradictory results on the two tasks: for sentence ordering, the combination is significantly better than each single model, while for essay scoring, the combination is worse than using RST-style features alone. This is probably related to the previously discussed issue of using entity-based features for essay scoring, due to the subtle difference between coherence and organization.

6.3.4 Conclusion

In this section, we have studied the impact of deep discourse structures in the evaluation of text coherence by two approaches. In the first approach, we implemented a model with discourse role features derived from RST-style discourse parsing, which represents deep discourse structures, and compared it against our re-implemented Lin et al.'s (2011) model derived from PDTB-style parsing, with no deep discourse structures annotated. In the second approach, we

[12]The entire text is covered by the annotation produced by RST-style discourse parsing, while this is generally not true for PDTB-style discourse parsing.

compared our complete RST-style model against a model with shallow RST-style encoding. Evaluated on the two tasks, sentence ordering and essay scoring, deep discourse structures are shown to be effective for better differentiation of text coherence. Moreover, we showed that, even without deep discourse structures, shallow RST-style encoding is more powerful than PDTB-style encoding, because it has better coverage of discourse relations in texts. Finally, combining discourse relations with entity-based features is shown to have an inconsistent effect on the two evaluation tasks, which is probably due to the different nature of the two tasks.

In our future work, we wish to explore the effect of automatic discourse parsers in our methodology. As discussed previously, the PDTB- and RST-style discourse parsers used in our experiments are far from perfect. Therefore, it is possible that using automatically extracted discourse relations creates some bias to the training procedure; it is also possible that what our model actually learns is the distribution over those discourse relations which automatic discourse parsers are mostly confident with, and thus errors (if any) made on other relations do not matter. One potential way to verify these two possibilities is to study the effect of each particular type of discourse relation to the resulting model, and we leave it for future exploration.

6.4 Summary of This Chapter

In Sections 6.1 – 6.3, we studied an important problem in the analysis of discourse, i.e., the evaluation of discourse coherence. Specifically, in Chapter 6.2, we overviewed extension to a particularly popular model, Barzilay and Lapata's entity-based local coherence model. Unlike its numerous extensions, which are focused on extending the features in their original model, our extension is focused on extending the pairwise ranking learning procedure.

More importantly, aside from application-specific features, as those proposed in previous extensions to B&L's model, in Section 6.3, we explored using discourse relations, as a kind of application-neutral feature, to improve the performance in evaluating discourse coherence. In

particular, inspired by previous work of Lin et al.'s (2011) PDTB-style discourse role matrix, we proposed to incorporate deep hierarchical discourse structures into the role matrix, by deriving the discourse roles from RST-style discourse relations. In our experiments, our model derived from deep discourse structures is shown to be more powerful than a model derived from shallow discourse structures, and even ignoring deep discourse structures, RST-style feature encoding outperforms PDTB-style encoding due to its better coverage of discourse relations in the text. Moreover, encoding the application-neutral RST-style discourse relations into the model, we obtain significantly better performance than encoding the application-specific entity-based local transitions as features. This serves as the first support to our postulation that discourse parsing provides a general solution to problems in discourse analysis.

Furthermore, based on the results in Section 6.3, I argue that the full RST-style discourse role matrix can be used as a general representation of documents. And I will use the derived full RST-style feature encoding, i.e., probabilities of full RST-style discourse role transitions, as an application-neutral kind of feature for the other two applications to be discussed in Chapters 7 and 8.

Chapter 7

The Identification of Authorship

7.1 Introduction

Authorship identification is a well-studied problem in computational linguistics. There are two possible frameworks to approach this problem: **authorship attribution** and **authorship verification**. In this section, I will briefly overview the classic approaches to this problem, from these two perspectives separately.

7.1.1 Authorship Attribution

In the framework of **authorship attribution**, we have a disputed text known to be written by one of a relatively small set of authors, and our task is to find the closest match of the stylistic signature of the text to those of the candidate authors: Was this mystery play written by Shakespeare, Sheridan, or Shaw? We can answer the question by inducing a discriminative classifier from attested texts of the candidates and seeing which author it attributes the mystery text to. Typically, previous work on authorship attribution adopt a variety of lexico-syntactic stylometric features that operate on surface lexical or syntactic level, arose from linguistic intuition that are able to capture an author's writing style.

Juola (2006) and Stamatatos (2009) provide an extensive survey of modern methods in authorship attribution. Here, we briefly enumerate some of the most classic stylometric features, most of which will be used in our authorship experiments in the following few sections.

7.1.1.1 Lexical Features

Lexical stylometric features operate on the word level. Typical lexical features include the average length (in tokens) of sentences and paragraphs, and the vocabulary richness, as expressed by the token/type ratio, to represent the diversity an author's word usage. Another widely used example of lexical features is word frequencies, following a bag-of-words representation of documents. Moreover, it is also a common practice to distinguish content words from function words in computing the word frequencies. In the cases of texts which are not error free (such as student essays), spelling errors are also used to perform attribution.

7.1.1.2 Character Features

With respect to character features, a text is viewed as a mere sequence of characters, and character features are therefore computational simplistic to extract. Typical character features include frequencies of various characters, including alphabetical, digital, punctuation, upper-/lower-case characters, and so on. Another important and effective kind of character feature for authorship attribution is character *n*-grams.

7.1.1.3 Syntactic Features

Syntactic features are a more elaborate kind of stylometric feature, compared to lexical and character features described above. Typical syntactic features include frequencies of part-of-speech *n*-grams, frequencies of syntactic production rules, and part-of-speech entropy.

7.1.2 Authorship Verification

In the framework of **authorship verification**, the disputed text might or might not have been written by one particular author, but there are no specific known alternative candidates for authorship: Was this mystery play written by Shakespeare or not? In this situation, the notion of closest match does not apply, nor do we have any a priori notion of what would count as a close-enough match to the single candidate author; so the standard text classification methods of intrinsic authorship attribution cannot be used. Instead, a method known as unmasking may be used.

Unmasking is a technique developed by Koppel et al. (2007). It is based on the idea that if two texts were written by the same author, then any features a classifier finds that (spuriously) discriminate their authorship must be weak and few in number, and removal of these features from the classifier will seriously degrade its performance (thereby "unmasking" the spurious discrimination). On the other hand, if the texts were written by different authors, then many features will support their (correct) discrimination and removal of even a few strongly discriminating features will not seriously harm the classifier. Classifiers are induced not only for the disputed text and its candidate author but also for the disputed text and other authors similar to the candidate. The most discriminating features are progressively removed from each classifier and its performance is re-evaluated. If performance falls off far more rapidly for the candidate author, authorship is deemed to be verified.

More specifically, given a corpus consisting of works by a number of authors, the works are first segmented into small chunks, with approximately 500 words per chunk. And the verification method consists of two stages: *unmasking* and *meta-learning*.

7.1.2.1 Unmasking

In the unmasking stage, for each author A and each work X, a pair $\langle A_X, X\rangle$ is constructed, where A_X denotes the set of works by author A, excluding X if X is truly written by A. Then, the following procedure is performed m times on the pair:

1. Construct a base classifier from some features, which can potentially be any set of features, to discriminate A_X against X. In Koppel et al.'s work, they use a set of most frequent words in the pair $\langle A_X, X\rangle$ as the features in constructing the base classifier. We then measure the performance of the resulting base classifier by the cross-validation accuracy of the discrimination.

2. Eliminate the k most important features from the base classifier, e.g., the k features with the highest absolute weight in a linear-kernel SVM classifier, and re-evaluate the resulting base classifier.

3. Go to Step 2.

After the unmasking procedure above, we obtain a degradation curve corresponding to the accuracy of discriminating A_X against X, after each iteration of feature elimination. Based on the intuition outlined previously, if two texts were written by the same author, then the features that discriminate their authorship must be weak and few in number. Therefore, we expect to observe greater degradation if $\langle A_X, X\rangle$ is a *same-author* pair, i.e., A_X and X are written by the same author, compared to the case where $\langle A_X, X\rangle$ is a *different-author* pair.

7.1.2.2 Meta-Learning

In the second stage, meta-learning, the difference between *same-author* vs. *different-author* curves is quantified. In particular, Koppel et al. represent each curve as a numerical vector with the following features, where $i = 0, \ldots, m$.

- Accuracy after i elimination rounds.
- Accuracy difference between round i and $i + 1$.
- Accuracy difference between round i and $i + 2$.
- i^{th} highest accuracy drop in one iteration.

- i^{th} highest accuracy drop in two iterations.

A meta-learner is then trained to distinguish *same-author* vs. *different-author* curves, using the curve representation above. In their study, Koppel et al. observed that the following two conditions always hold for *same-author* curves while rarely hold for *different-author* curves, and a SVM-based meta-learner can thus differentiate the two types of curve with a very high accuracy.

- Accuracy after 6 elimination iterations is lower than 89%.
- The second highest accuracy drop in two iterations is greater than 16%.

The next two sections of this chapter are organized as the following: In Section 7.2, in the framework of authorship attribution, I will discuss our previous work of using local transition patterns, which are inspired by Barzilay and Lapata's entity-based local coherence model (introduced in Section 6.1.1), as a novel kind of authorship feature that operate on discourse text. And, in Section 7.3, I will further extend the idea of using features that operate above the surface lexical and syntactic level, for the task of authorship identification, by incorporating discourse relation features derived from our full RST-style discourse role matrix, introduced in Section 6.3.

7.2 Local Coherence Patterns for Authorship Attribution

Contemporary methods of authorship attribution and discrimination that are based on text classification algorithms invariably use low-level within-sentence aspects of the text as stylometric features[1]. In his survey, for example, Stamatatos (2009) lists twenty types of stylometric feature that mostly involve character and word unigrams and *n*-grams, part-of-speech tags, syntactic chunks, and parse structures. Koppel et al. (2009) adduce a similar set. Some examples of these low-level features are discussed in Section 7.1.

[1]Part of the text of this chapter is based on my co-authored paper with Hirst (Feng and Hirst, 2014a).

In this section, we experiment with a type of stylometric feature, patterns of local coherence, which, by contrast, is drawn from the discourse level across sentences. We look at the choice that a writer makes when referring to an entity as to which grammatical role in the sentence the reference will appear in, and how this choice changes over a sequence of sentences as the writer repeatedly refers to the same entity. We show that differences in the resulting patterns of reference and grammatical role are a powerful stylometric feature for identifying an author.

7.2.1 Local Transitions as Features for Authorship Attribution

Our features of local coherence patterns are inspired by Barzilay and Lapata's entity-based local coherence model, as introduced in Section 6.1.1. Recall that, in this model, a document is represented as a feature vector $\mathbf{\Phi}$, consisting of a set of transition probabilities between entities in continuous sentences.

Because the set of transition probabilities forms a feature vector $\mathbf{\Phi}$ for a text, we can investigate it as a possible stylometric feature for authorship attribution. That is, we hypothesize that authors differ in the patterns of local transitions that they use — their tendency to use some types of local transition rather than others — and this forms part of an author's unconscious stylistic signature. We can test this hypothesis by seeing whether we can use these feature vectors to build a classifier to identify authorship.

Of course, we are not suggesting that local transitions by themselves are sufficient for high-accuracy authorship attribution. Rather, we are hypothesizing only that they carry information about authorship, which, if correct, is an interesting new stylometric observation. And they could thus be a useful complement to lexical and syntactic features for attribution. In our experiments below, therefore, we put our emphasis on examining how much authorship information these features carry by themselves in comparison with a simple baseline as well as with traditional lexical and lexico-syntactic features and in combination with such features.

Any implementation of entity grids must deal with the question of how entities are iden-

tified and tracked through the text. Following B&L, we use the Reconcile 1.1 coreference resolution tool (Ng and Cardie, 2002) to resolve and extract entities in texts.

7.2.2 Data

We gather 19 works of nine 19th-century British and American novelists and essayists from Project Gutenberg; they are listed in Table 7.1. We split the texts at sentence boundaries into chunks of approximately 1000 words, regarding each chunk as a separate document by that author; leftovers of fewer than 1000 words were discarded. The imbalance between different authors in the number of documents for each is corrected by the sampling methods in our experiments (see Section 7.2.3 below).

We first apply coreference resolution to each document, using Reconcile 1.1 (Ng and Cardie, 2002). We then obtain a dependency parse of each sentence of each document to extract the grammatical role of each entity in the text, using the Stanford dependency parser (de Marneffe et al., 2006). We can then construct an entity grid and the corresponding coherence feature vector for each document. We take $n = 2$, that is only transition bigrams, so there are $4^2 = 16$ transition types. But we count transitions separately for salient entities — those entities with at least two occurrences in a document — and for non-salient entities. In the latter case, only 7 of the 16 transition types — those in which the entity appears in at most one sentence — can occur.

For each document, we also extract a set of 208 low-level lexico-syntactic stylistic features, as shown in Table 7.2. Of course, this list of stylometric features is not exhaustive, as it does not include several useful features such as features based on deep syntactic structures and semantic relatenedss. However, because our point here is to contrast coherence patterns, a kind of discourse-level features, with *low-level* lexico-syntactic features, we explicitly leave out those stylometric features which deal with deeper linguistic structures and information.

Text	Chunks
Anne Brontë	
Agnes Grey	78
The Tenant of Wildfell Hall	183
Jane Austen	
Emma	183
Mansfield Park	173
Sense and Sensibility	140
Charlotte Brontë	
Jane Eyre	167
The Professor	92
James Fenimore Cooper	
The Last of the Mohicans	156
The Spy	103
Water Witch	164
Charles Dickens	
Bleak House	383
Dombey and Son	377
Great Expectations	203
Ralph Waldo Emerson	
The Conduct of Life	67
English Traits	68
Emily Brontë	
Wuthering Heights	126
Nathaniel Hawthorne	
The House of the Seven Gables	106
Herman Melville	
Moby Dick	261
Redburn	27

Table 7.1: The list of authors and their works used in our experiments.

Lexical features (10)
Frequencies of function words: prepositions, pronouns, determiners, conjunctions, primary verbs (*be*, *have*, *do*), adverbs, and *to*
Frequencies of hapax legomena
Frequencies of hapax dislegomena
Character features (231)
Count of all alphabetic characters Count of all digit characters Upper-/lower-case character count (2) Letter frequencies (26) Count of punctuation marks Frequencies of the most frequent letter *n*-grams (100 bigrams and 100 trigrams)
Syntactic features (21)
Frequencies of the most frequent part-of-speech bigrams (20) The entropy of part-of-speech tags

Table 7.2: The list of stylometric features that we use. The numbers in parentheses denote the number of features in the category; categories not so marked are a single feature.

7.2.3 Method

We conduct two sets of authorship attribution experiments: pairwise and one-versus-others. In the former, we select two authors and build a classifier that attempts to discriminate them, using either the coherence feature set, the lexico-syntactic feature set, or a combination of both. In the latter, we select one author and build a classifier that attempts to discriminate that author from all others in the dataset, again using one or both of the feature sets. The classifier in all experiments is a one-hidden-layer feed-forward neural network. We choose this classifier because it is able to handle non-linearity relations among features, and it outperforms some other classifiers, such as decision trees and Support Vector Machines, in our development experiments.

Each experiment uses 5-fold cross-validation, in which one-fifth of the data is chosen as test data and the training data is derived from the other four-fifths. The process is repeated for each one-fifth of the data in turn. To prevent class imbalance, in each fold of cross-validation,

the training data partitions are further resampled (specifically, oversampled) in each iteration to obtain a balanced distribution in the training set between the two classes — that is, between the two selected authors in the pairwise experiments and between the selected author and all the others in the one-versus-others experiments. If the number of datapoints in one class is markedly fewer than that of the other, this procedure (implemented here by the Resample module of the Weka 3.6.8 toolkit (Hall et al., 2009)) replicates datapoints at random until the two classes are approximately equal in size. For pairwise classification, we also oversample the test set in the same way in order to set an appropriate baseline.

7.2.4 Results

7.2.4.1 Pairwise Classification

The results of pairwise classification are shown in Table 7.3. For each pair of authors, we show macro-averaged classification accuracy for four conditions: using only coherence features, using only traditional lexico-syntactic features, using all features, and, as a baseline, always guessing the author that has the greater representation in the training data. Because of our resampling procedure, the baseline is always close to, but not exactly, 50%, and it will be less than 50% when the more frequent author in the training data is not the more frequent one in the test data. Significant differences between the conditions for each pair of authors are indicated by superscripts ($p < .05$ in all cases).

As expected, the established lexico-syntactic stylometric features give accuracies that are significantly above the baseline (with one exception: Hawthorne versus Melville, where these features perform only 7 percentage points above baseline, a difference that is not significant because we have relatively little data for Hawthorne). And, as we hypothesized, our coherence features also significantly exceed the baseline in all cases, showing that these features contain a considerable amount of information about authorship. The combined feature set also significantly exceeds the baseline in all cases. However, there is no consistency or pattern as to the relative performance of the three feature sets. In some cases (denoted by ^{a} in the table), such as

		Austen	Charlotte	Cooper	Dickens	Emerson	Emily	Hawthorne	Melville
Anne	coherence	78.3^{d}	73.8^{d}	88.4^{d}	83.7^{ad}	85.5^{d}	81.9^{ad}	81.0^{d}	83.9^{d}
	lexico-syntactic	79.4^{d}	77.7^{d}	98.1^{bd}	78.4^{d}	90.6^{bd}	71.5^{d}	85.3^{d}	90.7^{bd}
	combined	86.7^{abd}	83.1^{abd}	99.1^{bd}	84.4^{ad}	90.9^{bd}	82.6^{ad}	90.1^{abd}	91.6^{bd}
	baseline	53.3	57.7	53.6	52.5	47.1	48.0	46.9	44.8
Austen	coherence		78.9^{d}	82.3^{d}	75.5^{d}	74.9^{d}	83.5^{d}	71.9^{d}	78.6^{d}
	lexico-syntactic		85.3^{bd}	97.5^{bd}	84.1^{bd}	97.4^{bd}	86.2^{d}	90.3^{bcd}	91.5^{bd}
	combined		91.7^{abd}	97.5^{bd}	86.4^{abd}	98.3^{bd}	93.9^{abd}	85.3^{bd}	90.5^{bd}
	baseline		47.2	49.3	53.6	45.4	45.2	46.1	46.4
Charlotte	coherence			93.2^{d}	80.6^{acd}	84.0^{d}	78.8^{acd}	84.1^{acd}	82.9^{ad}
	lexico-syntactic			92.8^{d}	58.1^{d}	93.3^{bd}	62.3^{d}	73.7^{d}	76.1^{d}
	combined			97.2^{abd}	77.3^{ad}	93.0^{bd}	73.4^{ad}	77.2^{d}	83.8^{ad}
	baseline			53.1	52.8	44.7	47.1	48.1	45.6
Cooper	coherence				81.3^{d}	79.6^{d}	92.9^{d}	73.5^{d}	74.2^{d}
	lexico-syntactic				92.8^{bd}	87.1^{bd}	93.3^{d}	81.0^{bd}	84.9^{bd}
	combined				95.3^{abd}	88.5^{bd}	96.6^{abd}	83.4^{bd}	87.7^{abd}
	baseline				53.7	44.6	46.0	46.1	46.0
Dickens	coherence					86.5^{d}	91.1^{acd}	75.2^{d}	79.9^{d}
	lexico-syntactic					87.3^{d}	77.8^{d}	74.3^{d}	83.2^{bd}
	combined					94.3^{abd}	87.3^{ad}	77.6^{ad}	85.7^{abd}
	baseline					48.8	49.1	49.6	49.0
Emerson	coherence						97.5^{d}	70.9^{d}	75.6^{d}
	lexico-syntactic						97.5^{d}	80.6^{bd}	89.9^{bd}
	combined						95.1^{d}	88.8^{abd}	94.2^{abd}
	baseline						51.5	53.6	51.6
Emily	coherence							78.9^{d}	94.6^{ad}
	lexico-syntactic							88.3^{bd}	86.0^{d}
	combined							91.1^{bd}	92.1^{ad}
	baseline							58.3	52.5
Hawthorne	coherence								67.9^{ad}
	lexico-syntactic								58.5
	combined								72.2^{ad}
	baseline								51.3

[a]Significantly better than lexico-syntactic features ($p < .05$). [b]Significantly better than coherence features ($p < .05$). [c]Significantly better than combined features ($p < .05$). [d]Significantly better than baseline ($p < .05$).

Table 7.3: Accuracy scores (%) of pairwise classification experiments.

Features	Accuracy
Coherence	81.3^{d}
Lexico-syntactic	83.7^{bd}
Combined	88.4^{abd}
Baseline	49.8

[a]Significantly better than lexico-syntactic features ($p < .05$). [b]Significantly better than coherence features ($p < .05$). [d]Significantly better than baseline ($p < .05$).

Table 7.4: Accuracy scores (%) of pairwise classification experiments, aggregated across all authors for each feature set.

Dickens versus Anne Brontë, the coherence features outperform the lexico-syntactic features, whereas in others (denoted by [b]), such as Dickens versus Austen, the reverse is true. To some extent, such a non-universal effect of coherence patterns and lexico-syntactic features across different authors suggests that these two types of features are independent and complementary. In a few cases (denoted by [c]), such as Charlotte Brontë versus Hawthorne, the coherence features also outperform the combined feature set, which is most likely due to some over-fitting issue on the training data, or the test data are inherently heterogeneous to the training data, in terms of lexical usage which depends on particular topics and contents. It is perhaps notable that, apart from Hawthorne versus Melville, all the pairs in which coherence features are superior to lexico-syntactic features involve a Brontë sister versus Dickens, Hawthorne, Melville, or another Brontë sister.

Generally speaking, however, Table 7.3 suggests that coherence features perform well but usually not quite as well as lexico-syntactic features, and that the combined feature set usually performs best. We confirm this generalization by aggregating the results for all pairs of authors by taking all predictions for all author pairs as a single set, and reporting accuracy for each set of features. The results, in Table 7.4, show this stated ordering for accuracy, with significant differences at each step.

7.2.4.2 One-versus-Others Classification

Unlike pairwise classification, where we are interested in the performance of both classes, in one-versus-others classification we are interested only in a single author class, and hence we can regard the problem as retrieval and report the results using the F_1 score of each class under each condition.

The results for each author are shown in Table 7.5, and aggregated results are shown in Table 7.6. A pattern similar to that of pairwise classification is observed. All feature sets perform significantly better than baseline, and the combined feature set is always significantly better than both others. The coherence features perform well, and significantly better than the lexico-syntactic features for Dickens and Charlotte Brontë; in addition, they perform notably better, albeit not significantly so, for Hawthorne and Emily Brontë. However, in aggregation, the lexico-syntactic features are significantly better than coherence features, and the combined set is significantly better again.

7.2.5 Discussion

Our experiments show that entity-based local coherence, by itself, is informative enough to be able to classify texts by authorship almost as well as conventional lexico-syntactic information, even though it uses markedly fewer features. And the two types of information together perform better than either alone. This shows that local coherence does not just represent a subset of the same information as lexico-syntactic features, which is not a surprise, given that they focus on different aspects of the text at different levels. On the other hand, given this point, we might expect that the performance of the two feature sets would be independent, and that there would be authorship discriminations that are difficult for one set of features but easy for the other. However, we do not find this; while each had cases in which it was significantly better than the other, the scores for the two feature sets were correlated significantly (pairwise task, $r = .3657, df = 34, p < .05$; one-versus-others task, $r = .7388, df = 7, p < .05$).

Although in aggregation the combination of lexico-syntactic and coherence feature sets

Author	Features	F_1
Anne	coherence	34.5^{d}
	lexico-syntactic	40.9^{bd}
	combined	48.0^{abd}
	baseline	15.1
Austen	coherence	39.1^{d}
	lexico-syntactic	60.3^{bd}
	combined	65.9^{abd}
	baseline	26.3
Charlotte	coherence	37.2^{ad}
	lexico-syntactic	25.2^{d}
	combined	38.0^{abd}
	baseline	16.2
Cooper	coherence	53.8^{d}
	lexico-syntactic	73.3^{bd}
	combined	78.0^{abd}
	baseline	25.8
Dickens	coherence	63.8^{ad}
	lexico-syntactic	61.3^{d}
	combined	70.6^{abd}
	baseline	50.7
Emerson	coherence	26.1^{d}
	lexico-syntactic	51.9^{bd}
	combined	61.6^{abd}
	baseline	9.1
Emily	coherence	29.5^{d}
	lexico-syntactic	25.6^{d}
	combined	32.7^{abd}
	baseline	7.9
Hawthorne	coherence	17.3^{d}
	lexico-syntactic	14.0^{d}
	combined	21.0^{abd}
	baseline	7.2
Melville	coherence	25.9^{d}
	lexico-syntactic	38.2^{bd}
	combined	39.6^{bd}
	baseline	12.1

[a]Significantly better than lexico-syntactic features ($p < .05$). [b]Significantly better than coherence features ($p < .05$). [d]Significantly better than baseline ($p < .05$).

Table 7.5: F_1 scores of one-class classification experiments.

Features	F_1
Coherence	40.5^{d}
Lexico-syntactic	47.9^{bd}
Combined	56.6^{abd}
Baseline	20.0

[a]Significantly better than lexico-syntactic features ($p < .05$). [b]Significantly better than coherence features ($p < .05$). [d]Significantly better than baseline ($p < .05$).

Table 7.6: F_1 scores of one-class classification experiments aggregated across all authors for each feature set.

outperform both feature sets individually, in a handful of cases, such as Hawthorne versus Austen in Table 7.3, the combined features obtain lower accuracy than using either lexico-syntactic or coherence features alone. We speculate that this is due to potential overfitting on the training data when using the combined feature set, which has a higher dimensionality than the other two.

7.2.6 Conclusion

We have shown that an author's presumably unconscious choices of transition types in entity-based local coherence is a stylometric feature. It forms part of an author's stylistic "signature", and is informative in authorship attribution and discrimination. As a stylometric feature, it differs markedly from the syntactic, lexical, and even character-level features that typify contemporary approaches to the problem: It operates at the discourse level, above that of sentences. Nonetheless, the correlation that we found between the performance of this feature set and the lower-level feature set suggests that there is some latent relationship between the two, and this requires further investigation.

We took Barzilay and Lapata's model very much as they originally specified it. However, there are many possible variations to the model that bear investigation. Apart from the obvious parameter settings — the value of n, the threshold for salience — the ranking of grammatical roles when an entity occurs more than once in a sentence may be varied. In particular, we

experimented with a variation that we called "multiple-transition mode" in which each occurrence of an entity in a sentence is paired individually in all combinations. For example, if a specific entity occurs three times, in the roles **S**, **O**, and **X**, in one sentence and then twice, in the roles **S** and **O** in the next, then we extract six transition bigrams (**S** → **S**, **O** → **S**, **X** → **S**, **S** → **O**, **O** → **O**, and **X** → **O**) rather than just the one with the highest priority, **S** → **S**. However, in some of our early experiments, this variation showed little difference in performance from Barzilay and Lapata's single-transition model, so we abandoned it as it adds significant complexity to the model. But we suspect that it might be more useful for characterizing authors, such as Dickens, who tend to write very long sentences involving complicated interactions among entities within a single sentence.

We see entity-based local coherence as possibly the first of many potential stylometric features at this level that may be investigated. These could include other aspects of repeated reference to entities. The present features consider only identical referents; but an author's use of other referential relationships such as various forms of meronymy (*the car . . . the wheels; the government . . . the minister*) could also be investigated. More generally, the set of various kinds of relationship used in lexical chains (Morris and Hirst, 1991) could be tried, but a too-promiscuous set would simply result in almost everything being related to almost everything else. Rather, instead of holding the relationships constant, as in the present approach, one would look for inter-author differences in the patterns of variation in the relationships. Such differences could also be sought in the surface forms of entity coreference across sentences (the author's choice of definite description, name, or pronoun); these forms are conflated in the present approach.

Understanding these kinds of discourse-level features better may bring new insight to this level of the stylistic analysis of text and the individuation of authors' style.

7.3 Using Discourse Relations for the Identification of Authorship

To further test the hypothesized ubiquitous benefit of using discourse relations as a kind of application-neutral feature for various applications in the analysis of discourse, in this section, we explore the effect of incorporating discourse relations for the problem of the identification of authorship. In particular, we apply the general feature encoding, derived from the discourse role matrix (see Section 7.3.1 for detail) on a set of identification problems, following the two frameworks of authorship identification separately: **authorship attribution** and **authorship verification**. As will be demonstrated, experimental results show that discourse relations are also useful for this application using both frameworks.

7.3.1 General Feature Encoding by Discourse Role Transitions

We define our general feature encoding of a document as the probabilities of full RST-style discourse role transitions. From now on, we will use the terms *general feature encoding* and *(full RST-style) discourse role transitions* interchangeably. Discourse role matrix and discourse role transitions are introduced in Section 6.3.1, and we revisit their definitions in this section as a useful reminder.

As introduced in Section 6.3.1, we can represent a text as an RST-style discourse role matrix, as shown in Table 6.6. In the matrix, the columns are the entities in the text, and the rows are the sentences. Each matrix entry captures the occurrence of each entity in each sentence, and the particular discourse roles (discourse relations plus the nuclearity information; e.g., BACKGROUND.N and TEMPORAL.N) associated with the entity in that sentence. We then derive our general feature encoding, i.e., the probabilities of each discourse role transition, from the discourse role matrix. Specifically, discourse role transitions are the subsequences extracted from each column in the discourse role matrix. For instance, BACKGROUND.N → CONDITION.N → CAUSE.N is a discourse role transition of length 3 extracted from the first column in Table

6.6.

As discussed previously, there are several free parameters in our general feature encoding. When extracting discourse roles, we differentiate between intra- and multi-sentential discourse relations; we use 18 coarse-grained classes of RST-style relations; the optimal transition length k is 3; and the salience threshold l is 2. The total number of features is approximately 162K.

7.3.2 Discourse Relations for Authorship Attribution

First, in the framework of authorship attribution (introduced in Section 7.1.1), we conduct a series of pairwise classifications on the same dataset as in our previous work of authorship attribution (introduced in Section 7.2), which consists of 19 works of nine 19th-century British and American novelists and essayists, and each work is chunked into fragments with at least 1000 words. However, in this study, we perform a clearer pre-processing[2] than we did in Section 7.2, resulting in different numbers of segmented chunks for each work, which are shown in Table 7.7.

In our experiments, we use three sets of features: (1) 208 basic lexico-syntactic stylometric features (as shown in Table 7.2); (2) local coherence patterns; and (3) full RST-style discourse roles. The first two sets are the same as used in the experiments in Section 7.2, and the third set is the same set of full RST-style feature encoding as described in our experiments on coherence evaluation (see Section 6.3.1.3).

In each pairwise experiment, for a particular pair of authors, we train and test on the subset of texts which are written by the two authors of interest. As in our previous work on authorship attribution, we conduct pairwise classification with five-fold cross-validation, where the models are trained using a linear-kernel SVMs. For each possible pair of authors, each time, one-fifth of the chunks are used for testing, and the rest are used for training.

To evaluate the performance of each feature set, we conduct two sets of evaluation: chunk-based and book-based evaluation.

[2]We resolved several incorrect chapter segmentation for each novel.

Text	Chunks
Anne Brontë	
Agnes Grey	62
The Tenant of Wildfell Hall	165
Jane Austen	
Emma	146
Mansfield Park	149
Sense and Sensibility	114
Charlotte Brontë	
Jane Eyre	158
The Professor	81
James Fenimore Cooper	
The Last of the Mohicans	138
The Spy	94
Water Witch	148
Charles Dickens	
Bleak House	343
Dombey and Son	340
Great Expectations	173
Ralph Waldo Emerson	
The Conduct of Life	62
English Traits	60
Emily Brontë	
Wuthering Height	109
Nathaniel Hawthorne	
The House of Seven Gables	93
Herman Melville	
Moby Dick	196
Redburn	112

Table 7.7: The data used in our authorship experiments. Each novel is segmented into chunks with at least 1000 words, without crossing paragraph breaks. Any leftover in each chapter with fewer than 1000 words is discarded. The numbers are different to those in Table 7.1 due to different pre-processing.

7.3.2.1 Chunk-based Evaluation

In chunk-based evaluation, the results are measured as the accuracy of correctly identifying the true author of each chunk, aggregated across all author pairs for each feature set over 5-fold cross-validation. The performance of different feature sets is shown in Table 7.8.

Consistent with our previous finding in Section 7.2 — in which the experiments and evaluation are also based on individual chunks — using coherence features alone does not perform as competitively as using lexico-syntactic features alone[3]. Moreover, using discourse roles as the single feature set also performs poorer than using lexico-syntactic features alone; however, the gap is much smaller than that between coherence features and lexico-syntactic features. In fact, discourse role features are shown to have significantly better performance than coherence features[4], and in our work on the evaluation of coherence (introduced in Section 6.3), we have a similar conclusion. More importantly, the combination of discourse role features with other features sets achieves significantly incremental performance than using these feature sets alone, which suggests the complementary effect of discourse roles to other feature sets. Finally, the best performance is achieved with the combination of all three feature sets, although the combination of discourse roles and lexico-syntactic features contributes to the most of the overall performance.

7.3.2.2 Book-based Evaluation

To further simulate the real situation of authorship attribution, we conduct a book-wise evaluation. The idea is that, in reality, we are given a whole document in question and a small set of suspect authors, and we are asked which suspect wrote this document. Therefore, rather than predicting based on the segmented chunks of the document, a more meaningful prediction should be a final decision for the whole document.

The book-based evaluation works in the following way: after predicting a probable author

[3]Note that the discrepancy between the accuracies in the first two rows, across Tables 7.4 and 7.8, is due to the different splitting of folds in two experiments and different pre-processing of the texts.

[4]The significance test is performed using the Wilcoxon's signed-rank test.

Features	**Acc (%)**
Coherence	73.9
Lexico-syntactic	85.5
Discourse roles	81.0
Coherence + Lexico-syntactic	86.4
Discourse roles + Coherence	83.0
Discourse roles + Lexico-syntactic	88.7
All	**89.4**

Table 7.8: The chunk-based performance of pairwise authorship classification. The results are measured with accuracy (in percentage) of correctly predicting the author of each chunk, aggregated across all author pairs for each feature set over 5-fold cross-validation. Accuracy difference is significant with $p < .01$ for all pairs of models.

for each chunk, for each book in the test set, we aggregate the chunk-based predictions to produce a final decision for the book. The final decision is produced by finding the author predicted for the majority of the chunks; ties are broken randomly.

The performance of different models for book-based evaluations shown in Table 7.9. In this evaluation, the results are measured as the average accuracy of correctly predicting the author of each book, aggregated across all author pairs for each feature set over five-fold cross-validation.

Contrary to chunk-based evaluation, as shown in Table 7.8, where lexico-syntactic features perform the best among all the three individual sets of features, they achieve the worst performance in the book-based evaluation. However, consistently across two classifications, discourse role features are superior to coherence features. This suggests that, discourse-level features, including coherence and discourse role features, tend to capture some more general characteristics of the texts and thus perform more stably across books of different topics and authors. In contrast, low-level features, i.e., lexico-syntactic features, which might be exceptionally good at discriminating texts of particular topics or authors, do not perform stably across different books. This result partially serves as another support for our hypothesis that discourse-level features, especially features derived from discourse relations, can characterize a text in a more general and application-neutral way than specially designed features, such as

Features	Correct	Incorrect	Acc (%)
Lexico-syntactic	720	40	93.9
Coherence	732	28	96.3[a]
Discourse roles	755	5	99.3[ab]
Coherence + Lexico-syntactic	748	12	98.2[ab]
Discourse roles + Lexico-syntactic	754	6	99.0[ab]
Discourse roles + Coherence	756	4	99.3[ab]
All	**758**	**2**	**99.6**[ab]

[a]Significantly better than lexico-syntactic features ($p < .01$). [b]Significantly better than coherence features ($p < .01$). [c]Significantly better than discourse role features ($p < .01$).

Table 7.9: The book-based performance of pairwise authorship classification. The results are measured with average accuracy (in percentage) aggregated across all author pairs for each feature set over 5-fold cross-validation.

the lexico-syntactic stylometric features here. Finally, the combination of all three features achieves the best accuracy, which almost perfectly discriminates the true author of each book for all author pairs in aggregation. However, the difference between the performance of using all features and using discourse roles by themselves is not significant ($p > .05$). In contrast, the addition of discourse role features significantly improves the performance on top of the combination of coherence and lexico-syntactic features.

7.3.3 Discourse Relations for Authorship Verification

In Section 7.1.2, we introduced Koppel et al.'s unmasking technique for authorship verification. In their original work, they used the 250 most frequent words in the two texts as the feature set for building the base classifier in the unmasking step. Koppel et al. argued that potentially any set of features can be used to build the base classifier, as long as the base classifier could reliably find a relatively larger number of discriminative features if the two texts are of different authors, compared to the cases where the texts are of the same author.

However, it has been rarely explored how the overall verification accuracy is affected by different sets of features in building the base classifier. One may expect that, other than the most

frequent words, more interesting features, which can better discriminate an author's essential writing style, may result in more distinct patterns between unmasking texts of the same author and unmasking those of different authors.

Therefore, it follows naturally to study whether we could incorporate our general feature encoding derived from discourse relations in building the base classifier for more effective unmasking.

7.3.3.1 Experiments

In our experiments, we use the same set of 19-century British and American novels, as the one we used in our authorship attribution experiments in Section 7.3.2, as shown in Table 7.7. However, since the chunks we use here have lengths of approximately 1000 words, whereas in Koppel et al.'s original work, 500-word chunks were used, we might need to tune a few parameters on our data. We experiment with the following parameter options, and we report the resulting performance for each set of parameters.

1. $k \in \{3, 5\}$: the number of most important features eliminated in each iteration.

2. $m \in \{5, 10, 15, 20\}$: the number of iterations for unmasking.

Moreover, to evaluate the effectiveness of our general feature encoding by discourse relations, we experiment with three different sets of features in building the base classifier in unmasking:

1. The 250 most frequent words. Although the set of text chunks in our experiments are double the lengths of those in Koppel et al.'s original work, we choose to stick to their reported optimal number of frequent words, because empirical results indicate that 250 is a reasonable rough boundary between common words and words tightly related to a particular text (Koppel and Schler, 2004).

2. The N most frequent RST-style discourse role transitions. Similar to the feature set of the most frequent words, we extract the most frequent RST-style discourse role transitions

Features eliminated (k)	**Iterations** (m)	**Correct *same*** (out of 17)	**Correct *diff*** (out of 152)	**Macro-F_1**
3	5	14	145	0.852
3	10	**15**	147	0.894
3	15	14	**149**	**0.902**
3	20	14	147	0.876
5	5	14	142	0.820
5	10	13	147	0.857
5	15	**15**	145	0.870
5	20	14	145	0.852

Table 7.10: Performance of leave-one-out cross-validation using the 250 most frequent words as features in base classifiers.

in the two texts, weighting frequency equally between the two texts, as the features to build the base classifier. We experiment with choices of $N \in \{250, 500, 1000\}$.

3. The combination of the two feature sets above.

Following Koppel et al., we conduct a series of verifications using leave-one-out cross-validation, and the performance is measured by the macro-averaged F_1 score across the two classes: *same-author* and *different-author* pairs (see Section 7.1.2).

Frequent Words as Base Features The result of using the 250 most frequent words as base features is shown in Table 7.10. The best performance, i.e., a macro-averaged F_1 of 0.902, is achieved using $k = 3$ and $m = 15$, which is close to the optimal setting reported by Koppel et al., i.e., $k = 3$ and $m = 10$. The reason why we need a slightly larger number of iterations here is probably due to the fact that we are using chunks with longer lengths in our experiments (1000 words vs. 500 words), and thus we need more iterations to deal with the larger variation of word usage in longer chunks.

Frequent Discourse Role Transitions as Base Features The result of using the N most frequent discourse role transitions as base features is shown in Table 7.11. The best performance, i.e., a macro-averaged F_1 of 0.916, is achieved using $k = 5$, $N = 1000$, and $m = 20$. Note

Features eliminated (k)	**Discourse transitions** (N)	**Iterations** (m)	**Correct *same*** (out of 17)	**Correct *diff*** (out of 152)	**Macro-F_1**
3	250	5	14	145	0.852
3	500	5	**15**	145	0.870
3	1000	5	**15**	144	0.858
3	250	10	12	144	0.803
3	500	10	13	149	0.882
3	1000	10	13	142	0.802
3	250	15	10	145	0.771
3	500	15	12	147	0.836
3	1000	15	13	145	0.833
3	250	20	12	146	0.825
3	500	20	13	145	0.833
3	1000	20	14	148	0.888
5	250	5	**15**	147	0.894
5	500	5	**15**	144	0.858
5	1000	5	**15**	145	0.870
5	250	10	14	149	0.902
5	500	10	12	143	0.792
5	1000	10	14	146	0.863
5	250	15	12	144	0.803
5	500	15	12	147	0.836
5	1000	15	14	148	0.888
5	250	20	13	148	0.869
5	500	20	12	147	0.836
5	1000	20	14	**150**	**0.916**

Table 7.11: Performance of leave-one-out cross-validation using the most frequent discourse role transitions as features in base classifiers.

that, in comparison with using the most frequent words as base features, to obtain the optimal performance, we need to use a considerably larger number of discourse role transitions (1000 vs. 250), which is probably attributed to the sparsity of discourse role transitions[5].

Frequent Words + Discourse Role Transitions as Base Features Table 7.12 demonstrates the performance of using the combination of the 250 most frequent words and the N most frequent discourse role transitions as base features, using various parameter settings. Here,

[5]Recall that when we encode a document by bigram and trigram full RST-style discourse role transitions, the resulting dimension of the feature space is approximately 162K.

Features eliminated (k)	**Discourse transitions** (N)	**Iterations** (m)	**Correct *same*** (out of 17)	**Correct *diff*** (out of 152)	**Macro-F_1**
3	250	5	14	146	0.863
3	500	5	**16**	146	0.899
3	1000	5	14	148	0.888
3	250	10	15	149	0.920
3	500	10	15	150	0.935
3	1000	10	15	148	0.907
3	250	15	15	150	0.935
3	500	15	15	148	0.907
3	1000	15	14	**152**	0.947
3	250	20	15	150	0.935
3	500	20	**16**	**152**	**0.983**
3	1000	20	15	**152**	0.965
5	250	5	15	146	0.881
5	500	5	14	141	0.810
5	1000	5	15	144	0.858
5	250	10	15	150	0.935
5	500	10	14	149	0.902
5	1000	10	15	147	0.894
5	250	15	15	151	0.950
5	500	15	15	152	0.965
5	1000	15	15	150	0.935
5	250	20	15	151	0.950
5	500	20	14	**152**	0.947
5	1000	20	14	151	0.931

Table 7.12: Performance of leave-one-out cross-validation using both the 250 most frequent words and the most frequent discourse role transitions as features in base classifiers.

the best performance, a macro-averaged F_1 of 0.983, is achieved using $k = 3$, $N = 500$, and $m = 20$.

Comparison between Different Base Features Finally, we evaluate the effect of different features used in the base classifier, by summarizing the best performance obtained of each specific set of features in Table 7.13. We see that using frequent words and frequent discourse role transitions achieve similar performance, which is slightly above 0.9 of macro-averaged F_1. However, the combination of the two sets achieves a much higher performance, a macro-

Base features	***Same*** F_1	***Diff*** F_1	**Macro-**F_1
Words	0.824	0.980	0.902
Discourse roles	0.848	0.984	0.916
Words + discourse roles	**0.970**	**0.993**	**0.983**

Table 7.13: Summary of the best performance of each feature set in building the base classifier.

averaged F_1 of 0.983, suggesting that discourse role transitions, as a general kind of discourse-level feature, do reveal some complementary aspects to surface-level lexical features, in terms of distinguishing the essential writing style of authors. And we had a similar observation in our attribution experiments in Section 7.3.2.

To summarize, our experiments in the authorship verification framework also demonstrate the effectiveness of discourse relations for the task of identification of authorship. Therefore, it serves as another support for our hypothesis of the general usefulness of discourse relations in the analysis of discourse.

7.3.4 Conclusion

In this section, first in the framework of **authorship attribution**, we studied the effectiveness of discourse role features, as encoded in the same way as the full RST-style feature encoding (described in our experiments of coherence evaluation in Section 6.3), in comparison with the low-level lexico-syntactic stylometric features and coherence pattern features. In particular, we conducted a series of pairwise experiments between texts of all pairs of authors, and used two methods for evaluation: chunk-based and book-based pairwise evaluation. In chunk-based evaluation, although yielding inferior performance than the standard lexico-syntactic stylometric features, discourse role features performed significantly better than coherence patterns as features. In book-based evaluation, where the evaluation is based on the aggregated prediction of each book, discourse role features achieved the best performance among three individual sets of features, while the lexico-syntactic features, which were shown to be the most effective single set of features for chunk-based classification, were the set of features with the poorest

performance. This suggests the limitation of these low-level features, i.e., lexico-syntactic features, as they are very application-specific, are susceptible to the variation of content topics and author style. In contrast, since discourse role features are generally neutral to any application, they tend to perform stably across texts of various topics and authorship.

Next, in the framework of framework of **authorship verification**, we considered Koppel et al.'s technique of unmasking and meta-learning (introduced in Section 7.1.2), and explored the use of different base features in the unmasking stage. Comparing with the original set of base features used in Koppel et al.'s work, i.e., the most frequent words, we studied the effect of using the most frequent discourse role transitions and their combination with the most frequent words as base features. Experiments suggested that discourse role transitions, as a general kind of discourse-level feature, can improve the accuracy of verification, when combined with low-level lexical features.

Therefore, following two distinct frameworks separately, we found our second support for our hypothesis of the general usefulness of discourse relations in the application of identification of authorship.

7.4 Summary of This Chapter

First, in Section 7.1, I introduced the general problem of identifying authorship, including its two major frameworks, authorship attribution and authorship verification, and the classic approaches previously proposed targeting each framework, which are considered as application-specific approaches.

Then, in Section 7.2, I studied the problem of whether coherence patterns, which are a type of discourse-level feature, can provide additional indicators to the classic lexico-syntactic features in authorship attribution. Our experiments showed that, although coherence patterns do not perform as well as surface-level features when used in isolation, their combination with the surface features often yields better performance. This suggests that discourse-level features

do capture some unconscious features of an author's writing style that are complementary to surface-level features.

Finally, in Section 7.3, I further extended the idea of using discourse-level features for authorship attribution, by encoding RST-style discourse relations in the same way as our full RST-style feature encoding in Section 6.3.1.3. In the framework of authorship attribution, experimental results showed that these discourse relations are also more powerful than discourse coherence patterns for authorship attribution, and when evaluated on the basis of books, discourse relations are also more powerful than those surface-level application-specific features. In the framework of authorship verification, when used in isolation, our discourse role features perform as well as the lexical features used by Koppel et al., and the combination of two sets of features achieves very high verification accuracy. Therefore, we obtain the second support to my stated hypothesis, i.e., the general usefulness of discourse relations in the applications of discourse analysis.

Chapter 8

The Detection of Deception

8.1 Introduction

With the rapid development of e-commerce and the increasing popularity of various product review websites, people are more used to making purchase decisions based on the reported experience of other customers. A product rated positively by its previous users is able to attract potential new customers, while a poorly rated product is certainly not a good option for most new customers. Given this influential power of product reviews, there comes a huge potential for *deceptive opinion spam* to distort the true evaluation of a product. The promoters of a product may post false complimentary reviews, and competitors may post false derogatory reviews.

In this chapter, I will first present some previous work on detecting deceptive opinions, from heuristics-based unsupervised approaches to learning-based supervised approaches, including our own work of using profile compatibility as indicators of deception. However, as will be seen, these approaches are all very specific to this particular task of detecting deceptive opinions; therefore, it is interesting to explore whether discourse relations, as a general kind of discourse-level feature, are also beneficial for this task, as they are for other problems in the analysis of discourse that we have seen in previous chapters.

8.1.1 Unsupervised Approaches

Although the task of detecting *deceptive opinion spam* can be formulated as a traditional classification problem with two classes, *deceptive* and *truthful*, where supervised learning can be applied, it is essentially very challenging. The difficulty arises from the lack of reliably labeled data, because it is extremely difficult for humans to identify through reading (Ott et al., 2011).

Therefore, much previous work on detecting deceptive opinions usually relies on some meta-information, such as the IP address of the reviewer or the average rating of the product, rather than the actual content of the review. Lim et al. (2010) proposed several behavior models to identify unusual reviewer patterns; e.g., spammers may contribute a fake review soon after product launch to maximally affect subsequent reviewers. Each of these individual models assigns a numeric score indicating the extent to which the reviewer has engaged in apparent spam behaviors, and these scores are then combined to product a final spam score. Jindal et al. (2010) employed data-mining techniques to discover unexpected class association rules; e.g., reviewers who give all high ratings to products of a brand while most other reviews are generally negative about the brand. Wu et al. (2010) proposed to use the distortion of popularity rankings as an indicator of opinion spamming, in the sense that deleting a set of random reviews should not overly disrupt the popularity ranking of a list of entities, while deleting fake reviews should significantly distort the overall rankings, under the assumption that deceptive reviews usually express a sentiment at odds with legitimate reviews. Feng et al. (2012b) postulated that for a given domain, there exists a set of representative distributions of review rating scores, and fake reviews written by hired spammers will distort such natural distributions. Experimenting with a range of pseudo-gold-standard datasets, they provided quantitative insights into the characteristics of natural distributions of opinions in various product review domains.

8.1.2 Supervised Approaches

With respect to supervised approaches, Jindal and Liu (2008) first conducted a tentative study of detecting deceptive opinions. In the absence of gold-standard datasets, they trained models

using features from the review texts, as well as from meta-information on the reviewer and the product, to distinguish between *duplicate* reviews (regarded as deceptive), i.e., reviews whose major content appears more than once in the corpus, and *non-duplicate* reviews (regarded as truthful). However, these (near-)duplicate reviews are often not sophisticated in their composition, and therefore are relatively easy to identify, even by off-the-shelf plagiarism detection software (Ott et al., 2011).

Yoo and Gretzel (2009) constructed a small gold-standard dataset with 40 truthful and 42 deceptive hotel reviews, and manually inspected the statistical differences of psychologically relevant linguistic features for truthful and deceptive reviews.

8.1.2.1 op_spam_v1.3 Dataset

Ott et al. (2011) released the op_spam_v1.3 dataset, a much larger dataset with 400 truthful and 400 deceptive hotel reviews with *positive* comments, which allows the application of machine learning techniques for training models to automatically detect deceptive opinions. Due to the difficulty for humans to identify deceptive opinions, traditional approaches to constructing annotated corpora by recruiting human judges to label a given set of texts does not apply to the task of deception detection. Consequently, crowdsourcing services such as Amazon Mechanical Turk[1] (AMT) have been adopted as a better solution (Ott et al., 2011; Rubin and Vashchilko, 2012). By asking paid subjects on AMT to compose deceptive and/or truthful texts, corpora with reliable labels can be constructed. The dataset contains **positive** reviews for the 20 most-rated hotels on TripAdvisor[2] (TA) in Chicago. For each hotel, Ott et al. selected 20 deceptive reviews from submissions on AMT, and 20 truthful reviews from 5-star reviews on TA, resulting in 800 reviews in total for 20 hotels. The average length of deceptive reviews is 115.75 words, while the truthful reviews are chosen to have roughly the same average length.

Since the release of the op_spam_v1.3 dataset, it has been extensively used in recent work on detecting deceptive opinions. First, Ott et al. focused on the textual content of the reviews,

[1] http://mturk.com.
[2] http://tripadvisor.com.

and approached the task of detecting deceptive opinions by finding any *stretch of imagination* in the text — they treated fake reviews as imaginative writing, and used standard computational approaches of *genre identification*, *psycholinguistic deception detection*, and *text categorization* to identify them. Among the set of features explored in their classification experiments, they discovered that the features traditionally employed in either psychological studies of deception or genre identification were both significantly outperformed by a much simpler N-GRAM model with *n*-grams only as features, which achieved nearly 90% accuracy, where human performance is barely over the random baseline of 50%. Later, Feng et al. (2012a) proposed a strengthened model, N-GRAM+SYN, by incorporating syntactic production rules derived from Probabilistic Context Free Grammar (PCFG) parse trees. They obtained 91.2% accuracy on the op_spam_v1.3 dataset, which is a 14% error reduction.

While both Ott et al. and Feng et al.'s cues of deception come purely from the surface realization in the reviews, in our work (Feng and Hirst, 2013), we proposed the use of additional signals of truthfulness by characterizing the degree of *compatibility* between the personal experience described in a test review and a product profile derived from a collection of reference reviews about the same product. Our intuition came from the hypothesis that since the writer of a deceptive review usually does not have any actual experience with that product, the resulting review might contain some contradictions with facts about the product, or the writer might fail to mention those aspects of the product that are commonly mentioned in truthful reviews. Conversely, a writer of a truthful review is more likely to make similar comments about some particular aspects of the product as other truthful reviewers. In other words, we postulate that by aligning the profile of a product and the description of the writer's personal experience, some useful clues can be revealed to help identify possible deception.

A product's profile P is composed of a set of its aspects $A = \{a_1, \ldots, a_N\}$, where each a_i is an individual aspect of the product; e.g., *bathroom* and *swimming pool* are typical aspects of hotels. Each aspect a_i is associated with a description $D_i = \{p_1 : w_1, \ldots, p_m : w_m\}$, in which each element is a unique word p_j to describe the aspect (e.g., *spacious* and *clean* can

be description words for the aspect of *bathroom*), along with the weight w_j assigned to that word. We extracted two different kinds of aspects from the reviews: *distinct* aspects, $a_{i,d}$, were extracted as the proper noun phrases in the reviews, while general aspects, $a_{i,g}$, were extracted as other noun phrases. We then performed a hierarchical agglomerative clustering to cluster similar aspects, separately for the set of distinct aspects and the set of general aspects. With respect to the descriptions, distinct and general aspects were treated differently as well. For a distinct aspect, its description can contain only one word, *existence*, while for each *general* aspect, its description words are automatically extracted from the dependencies in the set of reviews R.

To test an unseen review about a particular product, we first constructed a collective profile Q for that product, using a separate set of related reviews gathered from TripAdvisor, and constructed the test profile P, using the unseen review. We then performed a *bidirectional* alignment between the test profile P and the collective profile Q, and computed a list of aspect-wise compatibility features, capturing two types of compatibility: (1) Compatibility with the **existence** of some *distinct* aspect of the product, such as the mention of the famous art museum nearby the hotel; (2) Compatibility with the **description** of some *general* aspect of the product, such as commenting that the breakfast at the hotel is charged for.

We showed that, by incorporating our computed profile alignment features with Feng et al. (2012a)'s n-grams and deep syntax features, we significantly improve on the state-of-art performance.

8.1.2.2 The op_spam_v1.4 Dataset

More recently, Ott et al. released a new version, the op_spam_v1.4 dataset, with gold-standard *negative* reviews included as well (Ott et al., 2013), which offers an opportunity to more extensively study the problem of detecting deceptive reviews.

Experimenting with the op_spam_v1.4 dataset, Li et al. (2013) proposed a generative LDA-based topic modeling approach for fake review detection. Their model can detect the subtle

differences between deceptive reviews and truthful ones and achieves about 95% accuracy on review spam datasets, outperforming previous discriminative models by a large margin.

8.1.2.3 Li et al.'s Cross-Domain Dataset

Although the release of op_spam datasets does provide useful resources for conducting supervised approaches to the detection of deception, there have been concerns about the generality and representativeness of these datasets. For instance, it is unclear whether the deceptive opinions collected by crowdsourcing are able to reflect the true characteristics of real deceptive reviews written by professionals. It is possible that the distinction between the participants on AMT and the real professional writers of deceptive reviews is so conspicuous that the model trained using the constructed datasets is actually learning the difference between two groups of people, instead of learning the differences between truthful and deceptive writing. In addition, since the op_spam datasets contain reviews of hotels only, it is also unclear how those previous approaches perform on reviews of other domains, or even hotels in other cities.

Therefore, moving one step further to understand the general nature of deceptive reviews, in recent work of Li et al. (2014a), they studied a more comprehensive set of truthful and deceptive reviews. In particular, there are two important features of Li et al.'s dataset. First, unlike op_spam datasets, which include reviews of a single domain, Li et al.'s dataset includes reviews of multiple domains, i.e., Hotel, Restaurant, and Doctor, allowing us to evaluate how our trained models perform out of the domain the models are trained on. Second, in addition to crowdsourced deceptive reviews (the *Turker* set), their dataset also includes deceptive reviews composed by domain experts (the *Employee* set), i.e., the staff working in the target hotels, restaurants, or clinics. Therefore, this new dataset also allows the study of how the group factor affects the performance of trained models. Same with the op_spam datasets, the truthful reviews in the dataset are written by true customers (the *Customer* set).

The statistics of Li et al.'s dataset are shown in Table 8.1. Note that for the *Hotel* domain, both positive and negative reviews are available, while for others, only positive reviews are

	Turker		*Employee*		*Customer*	
	positive	negative	positive	negative	positive	negative
Hotel	400	400	140	140	400	400
Restaurant	200	0	120	0	200	0
Doctor	200	0	32	0	200	0

Table 8.1: Statistics of Li et al.'s (2014a) cross-domain dataset.

included.

In their work, Li et al. discovered several interesting aspects of deceptive reviews. For example, considering *Turk* and *Customer* reviews, their experiments showed that truthful reviews (the *Customer* set) tend to use more nouns, adjectives, prepositions, and determiners, while deceptive writings tend to encode more verbs, adverbs, and pronouns. However, this pattern is mostly violated if we compare *Employee* reviews, which are equally deceptive, with truthful reviews. These aspects all suggest that there could be a number of confounding factors in the task of detecting deceptive opinions, and we still have a long way to go in terms of finding general rules of deception, until we have gathered sufficiently large and representative data as well as obtained solid understanding of the real process of soliciting deceptive opinions in the real world.

8.2 Using Discourse Relations for the Detection of Deception

The detection of deception might also benefit from the extraction and understanding of the discourse relations in the text. Intuitively, for the task of deception detection, it might not be safe to solely rely on lexical or syntactic cues of deception, such as the ones introduced in Section 8.1.2, because those surface indicators may be easily avoided or imitated by experienced fake writers. However, discourse relations, as a kind of information from the deeper level of discourse, could potentially capture those unconscious cues of deception, which generally require more effort on the author's side if she intends to make her writing seem more truthful.

For example, in the context of product reviews, it might be the case that truthful reviews

tend to include comparisons to other similar products as evidence to support the argument in the review, by using discourse relations such as COMPARISON, CONTRAST, and EVIDENCE, while deceptive reviews rarely utilize these discourse relations.

8.2.1 Previous Work: Detecting Deception using Distributions of Discourse Relations

In line with the intuition outlined above, Rubin and Vashchilko (2012) conducted a preliminary experiment showing that truthful and deceptive stories tend to have different distributions of various types of discourse relations. In their work, they collected a small dataset of personal stories from submissions on Amazon Mechanical Turk (AMT). The dataset contains 18 completely truthful stories and 18 stories with some degree of deception. By manually annotating the dataset with RST discourse structures, Rubin and Vashchilko encoded each story with a vector consisting of frequencies of the discourse relations appearing in the text. Then, based on this data representation, they performed a series of comparisons between truthful and deceptive texts, including clustering and relation-wise comparisons. They found that their proposed methodology of using RST-style discourse analysis tends to identify systematic differences between deceptive and truthful stories in terms of their coherence and structure.

However, as admitted in their paper, Rubin and Vashchilko's work, although providing promising support to my intuition of the effectiveness of discourse relations for detection of deception, is still very preliminary. First, their analysis is constrained to a very small dataset, i.e., with only 36 stories in total, and thus some results cannot be statistically verified. Second, their analysis was based on manual annotation of RST discourse structures; however, annotating RST structures is not an easy task as it requires good understanding of the underlying framework. Finally, their data representation is fairly unsophisticated, as it encodes nothing but purely discourse information. However, as shown by the previous work of Ott et al. and Feng et al. (introduced in Section 8.1.2), surface information such as lexical and syntactic choices are very useful. It is thus understandable that their analysis based on purely discourse infor-

mation came to a few contradictory conclusions, because the model itself is not sufficiently sophisticated.

8.2.2 A Refined Approach

As the final series of experiments to test the generality of discourse relations to the analysis of discourse, we apply my discourse role feature encoding to the problem of detecting deceptive opinions, using Li et al.'s (2014a) cross-domain dataset (introduced in Section 8.1.2.3).

Compared to Rubin and Vashchilko's work, our approach is more sophisticated. In our general feature representation derived from enhanced discourse role matrix, since discourse relation roles are bound to the entities in the texts, this representation can go beyond purely discourse information. Moreover, our discourse role feature encoding is able to represent the transition sequences among discourse units; it thus has greater representation capacity than Rubin and Vashchilko's vector representation with relation probabilities. At the same time, while Rubin and Vashchilko used 36 texts only, we now have access to much larger gold-standard datasets, i.e., Li et al.'s dataset, allowing us to perform more reliable supervised learning techniques. Finally, by leveraging my RST-style text-level discourse parser, as described in Part I, the analysis of discourse structures can be automated, and thus expensive and time-consuming manual annotation is no longer required.

8.2.3 Data

We use Li et al.'s (2014a) cross-domain dataset (introduced in Section 8.1.2.3) in our experiments. As discussed previously, this dataset contains reviews of multiple domains, i.e, Hotel, Restaurant, and Doctor, as well as fake reviews written by different groups, i.e., domain experts and workers on AMT who pretend to be customers. However, in our experiments, we only consider the deceptive reviews collected on AMT, because as shown in Table 8.1, the size of the available samples of deceptive writings by domain experts is relatively small. Table 8.2 demonstrates the detailed statistics of the subset of reviews used in our experiments.

	Truthful		*Deceptive*	
	positive	negative	positive	negative
Hotel	400	400	400	400
Restaurant	200	0	200	0
Doctor	200	0	200	0

Table 8.2: Statistics of the dataset used in our experiments.

8.2.4 Features

In our experiments of detecting deceptive reviews, we explore three sets of features and their combinations, in which, the first two sets were application-specific features, which have been used in several previous work introduced in Section 8.1.2.

1. UNIGRAM features: frequencies of unigrams in the text.

2. SYN features: frequencies of production rules derived from syntactic parse trees. We use Stanford parser (Klein and Manning, 2003) to produce syntactic parse trees.

3. DIS features: frequencies of full RST-style discourse role transitions. These are our general feature encoding derived from the full RST-style discourse role matrix (introduced in Section 6.3), which has been extensively used in our experiments in the other two applications of discourse analysis in this thesis.

All features are encoded as their tf-idf weights, with features appearing only once in the dataset eliminated.

8.2.5 Results

For each domain in our dataset, we conduct a series of binary classification experiments to differentiate deceptive vs. truthful reviews. For each domain, we train a linear-kernel SVM classifier using the LibSVM software (Chang and Lin, 2011) and conduct 5-fold cross-validation over the reviews of the particular domain.

Domain	Model	Deceptive	Truthful	Average
Hotel-positive	UNIGRAM	88.2	88.0	88.1
	SYN	89.9	89.6	89.7
	DIS	53.7	54.0	53.9
	UNIGRAM + SYN	**90.1**	**89.9**	**90.0**
	UNIGRAM + DIS	87.3	86.7	87.0
	SYN + DIS	88.1	87.6	87.9
	UNIGRAM + SYN + DIS	88.8	88.2	88.5
Hotel-negative	UNIGRAM	88.6	88.4	88.5
	SYN	89.5	89.5	89.5
	DIS	55.8	55.5	55.6
	UNIGRAM + SYN	90.4	90.3	90.4
	UNIGRAM + DIS	87.4	87.3	87.4
	SYN + DIS	89.4	89.4	89.4
	UNIGRAM + SYN + DIS	**90.8**	**90.7**	**90.7**
Restaurant	UNIGRAM	85.6	85.4	85.5
	SYN	86.2	85.8	86.0
	DIS	34.5	57.9	46.2
	UNIGRAM + SYN	**86.9**	**86.6**	**86.7**
	UNIGRAM + DIS	85.8	85.7	85.7
	SYN + DIS	86.2	85.8	86.0
	UNIGRAM + SYN + DIS	**86.9**	**86.6**	**86.7**
Doctor	UNIGRAM	87.9	74.0	80.9
	SYN	87.7	73.8	80.8
	DIS	78.1	0.0	39.0
	UNIGRAM + SYN	**88.5**	**76.2**	**82.4**
	UNIGRAM + DIS	88.0	74.4	81.2
	SYN + DIS	87.8	74.2	81.0
	UNIGRAM + SYN + DIS	88.5	75.6	82.1

Table 8.3: Classification performance of various models on reviews of each domain. Performance is measured by the 5-fold cross-validation F_1 score (%) of each class (deceptive and truthful), as well as their macro-average.

Classification performance of various models is reported in Table 8.2. Performance is measured by the 5-fold cross-validation F_1 score of each class (deceptive and truthful), as well as their macro-average. The random baseline for all domains is 50%.

Unfortunately, in these experiments, we do not observe any positive influence of incorporating discourse role features (DIS) into the model. In particular, the discourse role features, on their own, perform barely over the random baseline of 50% for hotel reviews, and even worse than the random baseline on the other two domains. Moreover, the addition of DIS features into other feature sets usually degrades the performance as well.

Therefore, in contrast to the other two applications discussed previously in this thesis, i.e., the evaluation of coherence (Chapter 6) and the identification of authorship (Chapter 7), we cannot verify our hypothesis of the general usefulness of discourse relations on the detection of deception. However, these results do not suggest that discourse relations are completely useless for detecting deception either, as there exist several confounding factors in our experiments (see below), which might prevent us from drawing a clear conclusion at this time.

8.2.6 Discussion

As shown by our experimental results, in comparison with surface-level lexical and syntactic features, discourse relations, as encoded by our general feature encoding derived from full RST-style discourse role matrix, are not helpful, or even detrimental to the recognition of deceptive reviews. However, we argue that we cannot thus falsify our hypothesis of the general usefulness of discourse relations on this particular application, merely based on these results. The reason is that there exist a number of confounding factors in our experiments, which may affect the overall results in an uncertain way.

8.2.6.1 Nature of the Dataset

Our negative results may be related to the nature of the dataset used in the experiments, which could influence our results in the following two important ways.

In fact, the reviews in the op_spam datasets and Li et al.'s cross-domain dataset are either extremely positive or extremely negative. For example, in op_spam datasets, the truthful hotel reviews are selected from the 5-star (highest-rated) reviews on TripAdvisor, and in Li et al.'s dataset, the truthful restaurant and doctor reviews are chosen in a similar fashion. In case of deceptive reviews, the participants are also asked to generate reviews with completely positive/negative comments. Due to such extreme sentiment in our data, some particular discourse relations, e.g., CONTRAST and COMPARISON, which, in our intuition, might be indicative of truthful or deceptive reviews in practice, occur rarely in our data, and therefore, the usefulness of discourse role transitions are not visible.

Moreover, our discourse relation features may also suffer from the issue of sparsity as well. Recall from Section 6.3.1.3 that the dimensionality of our full RST-style feature encoding is $\approx$ 192K, which is considerably large, and our classifiers may be overfitting during the training with discourse role features incorporated.

8.2.6.2 Unreliability of Automatic Discourse Parser

It is also possible that our discourse parser does not perform sufficiently well on the set of reviews in our dataset.

In our previous studies of coherence (Chapter 6) and authorship (Chapter 7), the materials used in our experiments are either written by professional writers or produced by students learning English. Therefore, discourse transitions are expected to be more frequently associated with explicit discourse markers, because the authors are trained to write in this way so that their writing can be understood more easily.

However, product reviews are quite distinct from the materials used in our previous studies, in the sense that discourse connectives are rarely adopted in the writing. A typical review is shown in the example below.

Dr. Lurie is an exceptional physician with a successful practice.

He has always been very very accommodating with his schedule whenever I need to visit with him on short notice.

Dr. Lurie and his staff are all very friendly and knowledgeable with years of experience and offer good advice regarding my health.

His office is warm and inviting unlike some doctor offices which helps to put me at ease the second I walk in the door.

I know myself and my family are in good hands with Dr. Lurie.

In this example, the bulk of the review is formed by the description of several facts presented in a sequence of sentences, whose internal discourse relations seem to be relatively loose.

And as introduced in Chapter 1, the major challenge for current discourse parsers is the recognition of discourse relations without explicit discourse cues. Considering the example review above, from the perspective of automatic discourse parsers, possibly no interesting relation, other than multiple ELABORATION relations, can be extracted. Therefore, product reviews might be a particularly difficult type of text for current discourse parsers, and the unreliable performance of automatic discourse parsing may also have a negative influence on our results.

8.2.6.3 Wrong Intuition

Finally, it is possible that our methodology is based on a wrong intuition, in the sense that, in fact, we **cannot** distinguish truthful and deceptive reviews by their distributions of various discourse relations. However, as introduced in Section 8.2.1, preliminary work of Rubin and Vashchilko demonstrated a promising support to this intuition, although they used a different feature encoding, i.e., a vector composed of frequencies of discourse relations.

Therefore, to further investigate the possibility of our intuition being wrong, we conduct another set of experiments, to rule out the confound of different feature encodings in our evaluation. Particularly, we replace the set of DIS features in Section 8.2.4 by the set of features used by Rubin and Vashchilko, i.e., frequency of each discourse relation, and conduct the same set

Domain	Model	Deceptive	Truthful	Average
Hotel-positive	DIS	53.7	54.0	53.9
	DIS^{RV}	53.6	55.6	54.6
Hotel-negative	DIS	55.8	55.5	55.6
	DIS^{RV}	62.7	57.1	59.9
Restaurant	DIS	34.5	57.9	46.2
	DIS^{RV}	59.6	59.9	59.7
Doctor	DIS	78.1	0.0	39.0
	DIS^{RV}	79.0	13.8	46.4

Table 8.4: Comparison between the classification performance of our DIS features and Rubin and Vashchilko's DIS^{RV} features. Performance is measured by the 5-fold cross-validation F_1 score (%) of each class (deceptive and truthful), as well as their macro-average.

of classification experiments — we call the resulting feature set DIS^{RV}.

Table 8.4 demonstrates the comparison between the classification performance of our DIS features and Rubin and of Vashchilko's DIS^{RV} features on 5-fold cross-validation classification. First, we see that DIS^{RV} also performs not substantially better than the random baseline of 50% on the first three domains, and even poorer on the Doctor domain. It suggests that discourse relations, when encoded by Rubin and Vashchilko's feature representation, which was previously shown to be able to discriminate deceptive and truthful reviews, are not effective for our data either. Moreover, we notice that DIS^{RV} performs better than our DIS features on all domains, and on the Restaurant and Doctor domains, the difference is especially large. This is an unexpected result, because, as argued in the beginning of Section 8.2.2, DIS^{RV} features are considerably less sophisticated than our DIS features, because no lexical information is encoded in DIS^{RV} features, whereas discourse role transitions in our DIS features are attached to the entities in the text, and thus lexical information is partially incorporated.

Therefore, based on these two observations, a more probable explanation to why our general feature encoding derived from discourse relations is not effective in our experiments is: (1) our automatic discourse parser does not perform sufficiently reliably on this particular dataset of product reviews; (2) our high-dimensional DIS feature set is overfitting on this relatively

small dataset.

8.2.7 Conclusion

In this section, we explored the employment of discourse relations in the task of detecting deceptive reviews. First, we experimented with representing documents by our feature encoding derived from full RST-style discourse role matrix. However, as shown by our experiments, this set of discourse relation features is not effective for recognizing deceptive reviews in Li et al.'s cross-domain dataset.

However, we argue that we could not simply make a general conclusion that discourse relations are not useful for the detection of deception, by pointing out several confounding factors in our experiments, including the relatively small size of the dataset, the unreliability of our discourse parser on the domain of product reviews, and the possibility that the intuition which motivates our method is untrue. Based on a set of supplementary experiments in Section 8.2.6.3, we postulate that the first two factors might have a major effect on the experimental results, and thus, we are yet unable to determine whether our discourse relation features are useful or not for the particular application of detecting deception. Therefore, a more comprehensive dataset and a more reliable discourse parser might be required for future exploration.

8.3 Summary of This Chapter

In this chapter, I discussed the application of the detection of deceptive opinions. This remains a relatively less well-studied problem in NLP, due to the difficulty of conducting traditional supervised learning on this task. Previous application-specific approaches typically utilize surface-level indicators, such as lexical *n*-grams and syntactic production rules. Beyond these surface-level features, I also introduced our previous work on developing another set of application-specific features, i.e., product profile compatibility, to tackle the problem of recognizing deceptive reviews.

Moreover, I applied our RST-style discourse relations to the problem of detecting deceptive opinions, to explore the effectiveness of these application-neutral features on this particular task. Unfortunately, experiments showed that our discourse relation features were not effective for identifying deceptive reviews. However, we argue that these preliminary results do not necessarily indicate that our hypothesized general usefulness of discourse parsing does not hold on this particular application. In contrast, it might be worthwhile developing a more comprehensive dataset and a more reliable automatic discourse parser on the domain of product reviews.

Summary of Part II

In Part II, we studied three specific problems in the analysis of discourse, including the evaluation of discourse coherence (Chapter 6), the identification of authorship (Chapter 7), and the detection of deceptive opinions (Chapter 8). For each problem, I first demonstrated several application-specific approaches, which represent the state-of-the-art or classic methodology for each individual task; then, I explored the possibility of using discourse relations, extracted by our RST-style discourse parser (introduced in Part I), as a general kind of feature in isolation, or in combination with these application-specific features.

In Chapter 6, for the evaluation of coherence, the application-specific approaches include the basic entity-based local coherence model proposed by Barzilay and Lapata (B&L) (2005; 2008) and its numerous extensions. While most of these prior extensions focus on the enrichment of feature set, one impressive example is Lin et al.'s (2011) work on incorporating PDTB-style discourse relations into B&L's entity-based model, by encoding discourse relations as a set of discourse roles for each entity in the text. However, since PDTB-style discourse parsing does not annotate the full discourse hierarchy of the text, nor does it annotate discourse relations for the entire text, it is promising that its performance can be superseded by RST-style discourse parsing, which can overcome these two limitations. In my experiments, full RST-style feature encoding of discourse role transitions is shown to be more powerful than Lin et al.'s PDTB-style encoding, and outperforms the basic entity-based model by a wide margin. Therefore, this full RST-style feature encoding can serve as a general kind of discourse-level feature for other tasks in the analysis of discourse.

In Chapter 7, for the identification of authorship, I first studied the problem of whether coherence patterns, which are a type of discourse-level feature, can provide additional indicators to the classic lexico-syntactic features in authorship attribution. Our experiments showed that, although coherence patterns do not perform as well as surface-level features when used in isolation, their combination with the surface features often yields better performance. This suggests that discourse-level features do capture some unconscious features of an author's writing style that are complementary to surface-level features. Then, in Section 7.3, I further extended the idea of using discourse-level features for authorship attribution, by encoding RST-style discourse relations in the same way as our full RST-style feature encoding in Section 6.3.1.3. In the framework of authorship attribution, experimental results showed that these discourse relations are also more powerful than discourse coherence patterns for authorship attribution, and when evaluated on the basis of books, discourse relations are also more powerful than those surface-level application-specific features. In the framework of authorship verification, when used in isolation, our discourse role features perform as well as the lexical features used by Koppel et al., and the combination of two sets of features achieves very high verification accuracy.

Finally, in Chapter 8, I discussed the last application, the detection of deceptive opinions. This remains a relatively less well-studied problem in NLP, due to the difficulty of conducting traditional supervised learning on this task. Previous application-specific approaches typically utilize surface-level indicators, such as lexical *n*-grams and syntactic production rules. Beyond these surface-level features, I first introduced our previous work on developing another set of application-specific features, i.e., product profile compatibility, to tackle the problem of recognizing deceptive reviews. Moreover, to further evaluate my hypothesis of the general usefulness of discourse parsing on the analysis of discourse, I also applied our RST-style discourse relations to the problem of detecting deceptive opinions. Unfortunately, as demonstrated by the experimental results in Section 8.2, our discourse relation features were not effective for identifying deceptive reviews. However, we argue that these preliminary results do not necessarily

indicate that our hypothesized general usefulness of discourse parsing does not apply on this particular application; rather, it might be a necessity to obtain a more comprehensive dataset and a more reliable automatic discourse parser on the domain of product reviews.

Part III

Summary

Chapter 9

Conclusion and Future Work

This thesis is composed of two major topics: discourse parsing and its applications in the analysis of discourse. Discourse parsing is the task of identifying the relatedness and the particular discourse relations among various discourse units in a text, and I hypothesize that it can provide a general solution to many downstream applications, such as automatic summarization, the evaluation of coherence, and natural language generation.

In Chapter 1, I first introduced the two major theoretical frameworks for discourse parsing, namely, RST-style discourse parsing and PDTB-style discourse parsing. Since there are a number of distinctions between these two frameworks, which result from the different annotation motivation of the two frameworks, RST-style discourse parsing and PDTB-style discourse parsing are potentially useful for different kinds of application. Between the two frameworks, I am particularly interested in RST-style discourse parsing, due to its full hierarchical discourse representation and complete coverage of the discourse relations in the text, and I postulate that these features of RST-style discourse parsing are beneficial to applications in which global understanding of texts is required.

In Part I, I first overviewed my work on discourse segmentation and discourse tree-building, which are the two primary components of RST-style discourse parsing. Both my discourse segmenter and my tree-builder are based on Conditional Random Fields (CRFs), which is

very effective in modeling interactions between contexts. Evaluated on the RST Discourse Treebank (RST-DT), both my discourse segmenter and my tree-builder achieve the state-of-the-art performance.

I proceeded to discuss the application of discourse relations to some specific tasks in the analysis of discourse, including the evaluation of coherence (Chapter 6), the identification of authorship (Chapter 7), and the detection of deception (Chapter 8). In particular, I discussed some application-specific approaches to each of these tasks, which represent the state-of-the-art performance for each tasks. Then, I described the general methodology of utilizing discourse relations in each task, by first processing the texts using my own RST-style discourse parser, and derived a set of discourse role features. With respect to these three tasks, I directly applied these discourse role features to each task, without any task-specific adaption. For the first two applications, our empirical results showed that discourse role features, by themselves, often perform as well as those classic application-specific features, and the combination with application-specific features usually yields further improvement. These results provide strong evidence for my hypothesis that discourse relations, especially those produced by RST-style discourse parsing, can lead to a general solution to many downstream applications in the analysis of discourse. However, we failed to observe a similar positive support on the detection of deception, and argued that further endeavor should be focused on developing a more comprehensive dataset and a more reliable cross-domain discourse parser.

9.1 Future Work for Discourse Parsing

In line with the framework of this thesis, the first direction of future work is to refine our discourse parsing model. As discussed in Part I, discourse segmentation is considered a relatively trivial task, and therefore, we will focus on the discussion of the tree-building phase. Specifically, future exploration could be done on the local and global level separately.

On the local level, given two text spans, we strive for better recognition of the particular

discourse relation relating the two spans. This is similar to the annotation philosophy of PDTB (introduced in Section 1.2), in which we are particularly interested in identifying discourse relations in a relatively small context window, and how discourse relations are lexicalized by various discourse connectives.

On the global level, we are interested in developing more accurate and efficient parsing algorithms to construct the text-level discourse parse tree. As demonstrated in Chapter 5, the time efficiency of text-level discourse parsing is also a crucial requirement for practical considerations.

9.1.1 On the Local Level: Tackling Implicit Discourse Relations

On the local level, we wish to enhance the performance of recognizing the particular discourse relation that holds between two given text spans. As demonstrated in Table 4.3, in this setting, the accuracy of recognizing intra-sentential relations is 78% and the accuracy of recognizing multi-sentential relations is lower than 50%, which is far from perfect. While the causes of the relatively low performance are a bit mixed in the context of RST-style parsing, the picture is much clearer on the side of PDTB-style parsing. As discussed in Section 1.2, the main challenge lies in recognizing implicit discourse relations, because those relations not associated with explicit discourse connectives, and thus world knowledge and semantics are required for the interpretation. These all suggest that the crucial area for future research is to tackle the recognition of implicit discourse relations.

In order to effectively incorporate semantics in modeling discourse relations, an interesting and promising direction is to take advantage of the recent progress in word embeddings and language modeling based on deep neural networks. Baroni et al. (2014) conducted a comprehensive comparison between word embeddings, i.e., word vectors learned from context-predicting neural language models, with their count-based counterparts, i.e., word vectors learned with classic distributional semantics approaches, such as Latent Semantics Analysis (LSA). Baroni et al. found that word embeddings often outperform or perform very closely to the state of the

art obtained by the count-based vectors, while the former requires considerably less effort in feature engineering.

Moreover, Mikolov et al. (2013) discovered that word embeddings are able to capture not only semantic but also syntactic regularities in language. For example, they observed that if we denote the vector for a word i as x_i, and there exists a simple numeric relation to capture singular/plural regularity: $x_{apple} - x_{apples} \approx x_{car} - x_{cars}$. More surprisingly, there exists a similar simple vector offset operation for a variety of semantic and syntactic relations. Therefore, it is worth exploring whether we can represent each discourse connective by a word vector, such that there exists an analogous numeric relation in the vector space between words in the two text spans and the particular discourse connective. If there does exist such a simple numeric relation between context words and the particular discourse connective, we then can use this relation to insert an appropriate discourse connective for implicit discourse relations. As discussed in Section 1.2, once we have a reliable prediction of discourse connectives for implicit relations, it is trivial to identify the associated relation type, by simply mapping the predicted connective to its most frequent sense in PDTB.

9.1.2 On the Global Level: Better Parsing Algorithms

On the global level, we aim to develop more accurate and efficient parsing algorithms to combine local decisions, i.e., the attachment of adjacent text spans and the assignment of discourse relations to a given pair of text spans, to form the final discourse tree corresponding to the full text.

One interesting piece of work is recently presented by Li et al. (2014b), who proposed to simulate text-level discourse parsing by parsing dependencies among EDUs. The idea is to frame each discourse relation as a dependency relation between a pair of EDUs, with the nuclear EDU being the head of the dependency relation. A discourse relation between larger text spans composed of multiple EDUs is modeled as a dependency relation between the first

main EDU[1] of the first span and that of the second span.

By taking advantage of classic dependency parsing techniques (e.g., the Eisner algorithm (Eisner, 1996) and the maximum spanning tree algorithm) and operate on the basis of individual EDUs, the overall time complexity can be greatly reduced, because we do not have to take care of the complex interior structures of the text spans on higher levels of the discourse tree.

However, Li et al.'s approach has a few limitations. First of all, it is unclear whether it is sufficient to model a composite text span by its very first main EDU, as useful information might be lost during this process. Second, because being framed as a dependency relation, each discourse relation can have only one head, and this certainly violates the cases of multi-nuclear relations, such as LIST and SAME-UNIT. Finally, because each dependency structure is not uniquely mapped to a discourse tree, additional tricks will be needed if our interest is to construct the text-level discourse tree. Therefore, in future work, we wish to explore whether Li et al.'s approach can be further refined to produce more reasonable discourse structures.

9.1.3 Domain Adaption

Furthermore, we are interested to explore how difficult it is to transfer our discourse parsing techniques trained on news texts to other domains. For example, as we saw in Section 8.2.6.2, our current discourse parser is likely to fail on product reviews, as product reviews demonstrate a distribution of discourse relations quite distinct from that in our training corpus (fewer explicit discourse relations). Therefore, does it mean that we need to construct annotated corpora separately for each domain that we are interested in? However, as can be imagined, this approach is highly expensive and time-consuming, and thus is not desirable. On the other hand, it might be possible that, with the advance in recognizing implicit discourse relations (possibly with the approach proposed in Section 9.1.1), domain adaptation is no longer a huge issue.

[1] As introduced in Chapter 6.3, the main EDUs of a discourse relation are the EDUs obtained by traversing the discourse subtree in which the relation of interest constitutes the root node, following the nucleus branches down to the leaves.

9.2 More Potential Applications

The other direction for future work is to investigate more potential applications of discourse parsing, in addition to the ones explored in this thesis, namely, the evaluation of coherence, the identification of authorship, and the detection of deception. Some previous examples of other successful employment of discourse analysis include classifying text-wide temporal relations (Ng et al., 2013) and non-factoid answer reranking in the domain of question-answering (Jansen et al., 2014).

9.2.1 Machine Translation

Guzmán et al. (2014) showed that discourse structures can help better evaluate the quality of machine translation systems. Therefore, it follows naturally that the process of developing machine translation models can benefit from directly taking into account the information from discourse structures. For example, we can adopt a re-ranking framework, in which we generate a set of candidate translations, and use information derived from discourse structures to order these candidates. Alternatively, we can also train a Support Vector Machine, with a customized kernel, which is designed to encode conventional features used in machine translation, as well as features describing the resulting discourse tree structure.

9.2.2 Anti-Plagiarism

Discourse parsing may also benefit the study of anti-plagiarism, which can be considered a derivative of traditional authorship identification (introduced in Chapter 7).

Anti-plagiarism is to identify un-authorized use of expressions or thoughts of other persons. One naïve approach is to identify information belonging to other publications or online resources (using some off-the-shelf anti-plagiarism software or querying search engines), which are not associated appropriate references (by parsing the discourse structure of the text and recognizing the lack of ATTRIBUTION relations). A more sophisticated approach is to take

advantage of authorship verification techniques to identify segments in the text which demonstrate a distinct writing style to the claimed author, and the lack of appropriate ATTRIBUTION relations.

9.2.3 Detecting Stylistic Deception

In Chapter 8, we discussed the detection of factoid deception, which is the type of deception formed by lying about certain facts, such as in the domain of product reviews that we explored. Another interesting type of deception is stylistic deception, which is about imitating another person's writing style or intentionally masking his or her own writing style. Therefore, the detection of this type of deception can be regarded as an intersection between authorship identification and deception detection. While our general discourse role transition features were shown to be not as effective as we expected for detecting factoid deception (see Section 8.2), they might be useful for detecting stylistic deception.

Bibliography

Nicholas Asher and Alex Lascarides. *Logics of Conversation*. Cambridge University Press, 2003.

Ngo Xuan Bach, Nguyen Le Minh, and Akira Shimazu. A reranking model for discourse segmentation using subtree features. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDial 2012)*, pages 160–168, Seoul, South Korea, July 2012.

Jason Baldridge and Alex Lascarides. Probabilistic head-driven parsing for discourse structure. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL 2005)*, pages 96–103, Ann Arbor, Michigan, June 2005.

Michele Banko and Lucy Vanderwende. Using N-grams to understand the nature of summaries. In *HLT-NAACL 2004: Short Papers*, pages 1–4, Boston, Massachusetts, USA, May 2004.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, pages 238–247, Baltimore, Maryland, June 2014.

Regina Barzilay and Mirella Lapata. Modeling local coherence: An entity-based approach. In *Proceedings of the 42rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 141–148, Ann Arbor, Michigan, June 2005.

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

Danushka Bollegala, Naoaki Okazaki, and Mitsuru Ishizuka. A bottom-up approach to sentence ordering for multi-document summarization. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL 2006)*, pages 385–392, Sydney, Australia, July 2006.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. Class-based n-gram models of natural language. *Computational Linguistuistics*, 18(4): 467–479, December 1992.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. Building a discourse-tagged corpus in the framework of Rhetorical Structure Theory. In *Proceedings of Second SIGDial Workshop on Discourse and Dialogue (SIGDial 2001)*, pages 1–10, Aalborg, Denmark, September 2001.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:1–27, 2011.

Eugene Charniak and Mark Johnson. Coarse-to-fine n-best parsing and maxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 173–180, Ann Arbor, Michigan, June 2005.

Jackie C. K. Cheung and Gerald Penn. Towards robust abstractive multi-document summarization: A caseframe analysis of centrality and domain. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1233–1242, August 2013.

Jackie Chi Kit Cheung and Gerald Penn. Entity-based local coherence modelling using topological fields. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 186–195, Uppsala, Sweden, July 2010.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, Genoa, Italy, May 2006.

David A. duVerle and Helmut Prendinger. 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 Asian Federation of Natural Language Processing (ACL/IJCNLP 2009)*, pages 665–673, Suntec, Singapore, August 2009.

Jason M. Eisner. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING 1996)*, pages 340–345, 1996.

Micha Elsner and Eugene Charniak. Extending the entity grid with entity-specific features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011)*, pages 125–129, Portland, Oregon, June 2011.

Song Feng, Ritwik Banerjee, and Yejin Choi. Syntactic stylometry for deception detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 171–175, Jeju Island, Korea, July 2012a.

Song Feng, Longfei Xing, Anupam Gogar, and Yejin Choi. Distributional footprints of deceptive product reviews. In *Proceedings of International AAAI Conference on Weblogs and Social Media (ICWSM 2012)*, Dublin, Ireland, June 2012b.

Vanessa Wei Feng and Graeme Hirst. Extending the entity-based coherence model with multiple ranks. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012)*, pages 315–324, Avignon, France, April 2012a.

Vanessa Wei Feng and Graeme Hirst. Text-level discourse parsing with rich linguistic features. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL 2012)*, pages 60–68, Jeju, Korea, July 2012b.

Vanessa Wei Feng and Graeme Hirst. Detecting deceptive opinions with profile compatibility. In *Proceedings of the 6th International Joint Conference on Natural Language Processing (IJCNLP 2013)*, pages 338–346, Nagoya, Japan, October 2013.

Vanessa Wei Feng and Graeme Hirst. Patterns of local discourse coherence as a feature for authorship attribution. *Literary and Linguistic Computing*, pages 171–190, 2014a.

Vanessa Wei Feng and Graeme Hirst. A linear-time bottom-up discourse parser with constraints and post-editing. In *Proceedings of the 52th Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, pages 511–521, Baltimore, USA, June 2014b.

Vanessa Wei Feng, Ziheng Lin, and Graeme Hirst. The impact of deep hierarchical discourse structures in the evaluation of text coherence. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING 2014)*, Dublin, Ireland, August 2014.

Katja Filippova and Michael Strube. Extending the entity-grid coherence model to semantically related entities. In *Proceedings of the Eleventh European Workshop on Natural Language Generation (ENLG 2007)*, pages 139–142, Germany, June 2007.

Seeger Fisher and Brian Roark. The utility of parse-derived features for automatic discourse segmentation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL 2007)*, pages 488–495, Prague, Czech Republic, June 2007.

Sylviane Granger, Estelle Dagneaux, Fanny Meunier, and Magali Paquot. *International Corpus of Learner English (Version 2)*. Presses universitaires de Louvain, 2009.

Francisco Guzmán, Shafiq Joty, Lluís Màrquez, and Preslav Nakov. Using discourse structure improves machine translation evaluation. In *Proceedings of the 52nd Annual Meeting of*

the Association for Computational Linguistics (Volume 1: Long Papers) (ACL 2014), pages 687–698, Baltimore, Maryland, June 2014.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA data mining software: An update. *SIGKDD Explorations*, 11(1), 2009.

Hugo Hernault, Danushka Bollegala, and Mitsuru Ishizuka. A sequential model for discourse segmentation. In *Proceedings of the 11th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing 2010)*, pages 315–326, Iasi, Romania, March 2010a.

Hugo Hernault, Danushka Bollegala, and Mitsuru Ishizuka. A semi-supervised approach to improve classification of infrequent discourse relations using feature vector extension. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, pages 399–409, Cambridge, MA, October 2010b.

Hugo Hernault, Helmut Prendinger, David A. duVerle, and Mitsuru Ishizuka. HILDA: A discourse parser using support vector machine classification. *Dialogue and Discourse*, 1(3): 1–33, 2010c.

Peter Jansen, Mihai Surdeanu, and Peter Clark. Discourse complements lexical semantics for non-factoid answer reranking. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Baltimore, Maryland, June 2014.

Nitin Jindal and Bing Liu. Opinion spam and analysis. In *Proceedings of First ACM International Conference on Web Search and Data Mining (WSDM 2008)*, pages 219–230, Stanford, California, USA, February 2008.

Nitin Jindal, Bing Liu, and Ee-Peng Lim. Finding unusual review patterns using unexpected rules. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM 2010)*, pages 1549–1552, Toronto, ON, Canada, October 2010.

Thorsten Joachims. Making large-scale SVM learning practical. In *Advances in Kernel Methods – Support Vector Learning*, chapter 11, pages 169–184. Cambridge, MA, 1999.

Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2002)*, pages 133–142, Edmonton, Alberta, Canada, July 2002.

Thorsten Joachims. A support vector method for multivariate performance measures. In *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005)*, pages 377–384, Bonn, Germany, August 2005.

Thorsten Joachims. Training linear SVMs in linear time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006)*, pages 217–226, Philadelphia, PA, USA, August 2006.

Shafiq Joty, Giuseppe Carenini, and Raymond T. Ng. A novel discriminative framework for sentence-level discourse analysis. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pages 904–915, Jeju Island, Korea, July 2012.

Shafiq Joty, Giuseppe Carenini, Raymond Ng, and Yashar Mehdad. Combining intra- and multi-sentential rhetorical parsing for document-level discourse analysis. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, pages 486–496, Sofia, Bulgaria, August 2013.

Patrick Juola. Authorship attribution. *Foundations and Trends in Information Retrieval)*, 1(3): 233–334, December 2006.

Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics (ACL 2003)*, pages 423–430, Sapporo, Japan, July 2003.

Alistair Knott and Robert Dale. Using linguistic phenomena to motivate a set of coherence relations. *Discourse Processes*, 18(1):35–64, 1994.

Moshe Koppel and Jonathan Schler. Authorship verification as a one-class classification problem. In *Proceedings of the Twenty-first International Conference on Machine Learning (ICML 2004)*, pages 489–495, Banff, Alberta, Canada, July 2004.

Moshe Koppel, Jonathan Schler, and Elisheva Bonchek-Dokow. Measuring differentiability: Unmasking pseudonymous authors. *The Journal of Machine Learning Research*, 8:1261–1276, 2007.

Moshe Koppel, Jonathan Schler, and Shlomo Argamon. Computational methods in authorship attribution. *Journal of the American Society for Information Science and Technology*, 60(1), January 2009.

Mirella Lapata. Probabilistic text structuring: experiments with sentence ordering. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics (ACL 2003)*, pages 545–552, Sapporo, Japan, July 2003.

Mirella Lapata. Automatic evaluation of information ordering: Kendall's tau. *Computational Linguistics*, 32(4):471–484, 2006.

Jiwei Li, Claire Cardie, and Sujian Li. Topicspam: a topic-model based approach for spam detection. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 217–221, Sofia, Bulgaria, August 2013.

Jiwei Li, Myle Ott, Claire Cardie, and Eduard Hovy. Towards a general rule for identifying deceptive opinion spam. In *Proceedings of the 52th Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, Baltimore, USA, June 2014a.

Sujian Li, Liang Wang, Ziqiang Cao, and Wenjie Li. Text-level discourse dependency parsing.

In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, pages 25–35, Baltimore, Maryland, June 2014b.

Ee-Peng Lim, Viet-An Nguyen, Nitin Jindal, Bing Liu, and Hady Wirawan Lauw. Detecting product review spammers using rating behaviors. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM 2010)*, pages 939–948, Toronto, ON, Canada, October 2010.

Ziheng Lin, Min-Yen Kan, and Hwee Tou Ng. 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, August 2009.

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

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

Edward Loper and Steven Bird. NLTK: The natural language toolkit. In *Proceedings of ACL'02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, pages 63–70, July 2002.

Nitin Madnani, Rebecca Passonneau, Necip Fazil Ayan, John M. Conroy, Bonnie J. Dorr, Judith L. Klavans, Dianne P. O'Leary, and Judith D. Schlesinger. Measuring variability in sentence ordering for news summarization. In *Proceedings of the Eleventh European Workshop on Natural Language Generation (ENLG 2007)*, pages 81–88, Schloss Dagstuhl, Germany, June 2007.

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

Daniel Marcu. The rhetorical parsing of natural language texts. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL 1997)*, pages 96–103, Madrid, Spain, July 1997.

Daniel Marcu. *The Theory and Practice of Discourse Parsing and Summarization*. The MIT Press, November 2000.

Andrew Kachites McCallum. MALLET: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2013)*, Atlanta, Georgia, June 2013.

Jane Morris and Graeme Hirst. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17(1), 1991.

Jun-Ping Ng, Min-Yen Kan, Ziheng Lin, Wei Feng, Bin Chen, Jian Su, and Chew Lim Tan. Exploiting discourse analysis for article-wide temporal classification. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, pages 12–23, Seattle, Washington, USA, October 2013.

Vincent Ng and Claire Cardie. Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL 2002)*, pages 104–111, Philadelphia, Pennsylvania, July 2002.

Naoaki Okazaki. CRFsuite: a fast implementation of conditional random fields (CRFs), 2007. http://www.chokkan.org/software/crfsuite/.

Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T. Hancock. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Asso-*

ciation for Computational Linguistics: Human Language Technologies (ACL 2011), pages 309–319, Portland, Oregon, USA, June 2011.

Myle Ott, Claire Cardie, and Jeffrey T. Hancock. Negative deceptive opinion spam. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Short Papers*, Atlanta, Georgia, USA, June 2013.

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

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Isaac Persing, Alan Davis, and Vincent Ng. Modeling organization in student essays. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, pages 229–239, Cambridge, MA, October 2010.

Emily Pitler, Mridhula Raghupathy, Hena Mehta, Ani Nenkova, Alan Lee, and Aravind Joshi. Easily identifiable discourse relations. Technical Report MS-CIS-08-24, Department of Computer and Information Science, University of Pennsylvania, June 2008.

Emily Pitler, Annie Louis, and Ani Nenkova. 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/AFNLP 2009)*, pages 683–691, Suntec, Singapore, August 2009.

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

Marta Recasens, Marie-Catherine de Marneffe, and Christopher Potts. The life and death of discourse entities: Identifying singleton mentions. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT/NAACL 2013)*, pages 627–633, Atlanta, Georgia, June 2013.

David Reitter. Simple signals for complex rhetorics: On rhetorical analysis with rich-feature support vector models. *LDV Forum*, 18(1/2):38–52, 2003.

Victoria L. Rubin and Tatiana Vashchilko. Identification of truth and deception in text: Application of vector space model to rhetorical structure theory. In *Proceedings of the Workshop on Computational Approaches to Deception Detection*, pages 97–106, Avignon, France, April 2012.

Attapol Rutherford and Nianwen Xue. Discovering implicit discourse relations through brown cluster pair representation and coreference patterns. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014)*, pages 645–654, Gothenburg, Sweden, April 2014.

Radu Soricut and Daniel Marcu. Sentence level discourse parsing using syntactic and lexical information. In *Proceedings of Human Language Technologies: The 41st Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL 2003)*, pages 149–156, Atlanta, Georgia, USA, May 2003.

Efstathios Stamatatos. A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology*, 60(3):538–556, March 2009.

Michael Strube and Simone Paolo Ponzetto. WikiRelate! Computing semantic relatedness using Wikipedia. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI 2006)*, pages 1219–1224, Boston, USA, July 2006.

Rajen Subba and Barbara Di Eugenio. Automatic discourse segmentation using neural networks. In *Proceedings of the 11th Workshop on the Semantics and Pragmatics of Dialogue (SemDial 2007)*, pages 189–190, 2007.

Rajen Subba and Barbara Di Eugenio. An effective discourse parser that uses rich linguistic information. In *Proceedings of Human Language Technologies: The 47th Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL 2009)*, pages 566–574, Boulder, Colorado, June 2009.

Charles Sutton, Andrew McCallum, and Khashayar Rohanimanesh. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. *The Journal of Machine Learning Research*, 8:693–723, May 2007.

Maite Taboada and William Mann. Applications of rhetorical structure theory. *Discourse Studies*, 8(4):567–588, 2006.

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

Florian Wolf and Edward Gibson. Representing discourse coherence: A corpus-based study. *Computational Linguistics*, 31(2):249–87, June 2005.

Guangyu Wu, Derek Greene, Barry Smyth, and Pádraig Cunningham. Distortion as a validation criterion in the identification of suspicious reviews. In *Proceedings of the First Workshop on Social Media Analytics (SOMA 2010)*, pages 10–13, Washington D.C., District of Columbia, July 2010.

Kyung-Hyan Yoo and Ulrike Gretzel. Comparison of deceptive and truthful travel reviews. In *Information and Communication Technologies in Tourism 2009*, pages 37–47. Springer Vienna, 2009.

Renxian Zhang. Sentence ordering driven by local and global coherence for summary generation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011): Student Session*, pages 6–11, Portland, Oregon, June 2011.

Zhi Min Zhou, Man Lan, Zheng Yu Niu, Yu Xu, and Jian Su. The effects of discourse connectives prediction on implicit discourse relation recognition. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDial 2010)*, pages 139–146, Tokyo, Japan, September 2010.