# Sequence Models and Ranking Methods for Discourse Parsing

A Dissertation

Presented to

The Faculty of the Graduate School of Arts and Sciences

Brandeis University

Computer Science

James Pustejovsky, Brandeis University, Advisor

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

by

Ben Wellner

February, 2009

This dissertation, directed and approved by Ben Wellner's committee, has been accepted and approved by the Graduate Faculty of Brandeis University in partial fulfillment of the requirements for the degree of:

**DOCTOR OF PHILOSOPHY**

Adam B. Jaffe, Dean of Arts and Sciences

Dissertation Committee:

James Pustejovsky, Brandeis University, Chair

Pengyu Hong, Brandeis University

Inderjeet Mani, The MITRE Corporation

Bonnie Webber, University of Edinburgh

Nianwen Xue, Brandeis University

# Acknowledgments

This dissertation would not have been possible without the help of many people. I'd like to first thank my advisor James Pustejovsky. He continually provided encouragement, helpful insight and a friendly and positive demeanor without which the completion of this disseration would have been much more difficult, and certainly much less pleseant. Among many things, James taught me to view problems from a different, deeper perspective and to appreciate and look for the linguistic phenomena that may lie behind certain empirical observations.

I am also grateful to the other members of my dissertation committee, Pengyu Hong, Inderjeet Mani, Bonnie Webber and Nianwen (Bert) Xue for their time spent reviewing and their valuable feedback in both the early and later stages of my dissertation.

At MITRE, my supervisor Marc Vilain has been continually supportive of my studies and was particularly helpful in shaping my responsibilities at MITRE to correspond well with my dissertation work. My office-mate, Scott Mardis, provided perspective and humor about things dissertation-related and not. I thank Lynette Hirschman, David Day, Warren Greiff, Mark Maybury and others for encouraging me to begin, helping to secure funding for, and supporting me during my graduate studies. Many colleagues at MITRE on various projects that I have worked on have all been very understanding and accommodating during this period. I have also benefited from helpful conversations with Inderjeet Mani and John Burger at various points that have led to ideas that have found their way into this dissertation.

Aravind Joshi, Alan Lee, Rashmi Prasad and others at UPenn provided valuable feedback on drafts of this work and I strongly benefited from discussions with them on matters related to the Penn Discourse Treebank. Andrew McCallum, during my time at UMass, helped instill in me a passion for and a solid background in machine learning.

Finally, but foremost, I would like to thank my family. My wonderful wife, Katharine, provided constant support and encouragement. My two sons Leif and Espen, have been a source of pure joy. Spending time with them was, and no doubt will remain, the perfect way to unwind after a long day in the office. And, I am forever thankful to my mother, Mary.

# Abstract

## Sequence Models and Ranking Methods for Discourse Parsing

A dissertation presented to the Faculty of
the Graduate School of Arts and Sciences of
Brandeis University, Waltham, Massachusetts

by Ben Wellner

Many important aspects of natural language reside beyond the level of a single sentence or clause, at the level of the *discourse*, including: reference relations such anaphora, notions of topic/focus and foreground/background information as well as rhetorical relations such as Causation or Motivation. This dissertation is concerned with data-driven, machine learning-based methods for the latter – the identification of rhetorical discourse relations between abstract objects, including events, states and propositions. Our focus is specifically on those relations based on the semantic *content* of their arguments as opposed to the *intent* of the writer.

We formulate a dependency view of discourse in which the arguments of a rhetorical relation are lexical heads, rather than arbitrary segments of text. This avoids the difficult problem of identifying the most elementary segments of the discourse. The resulting discourse parsing problem involves the following steps: 1) identification of discourse cue phrases that signal a rhetorical relation 2) identification of the two arguments of a rhetorical relation signaled by a discourse cue phrase and 3) determination of the *type* of the rhetorical relation.

To address the above problems, we apply a set of discriminative, statistical machine learning algorithms and explore the tradeoffs with various sets of features. We demonstrate how performance can be improved through learning architectures that allow for multiple co-dependent processing stages to be handled within a *single* model, rather than as a cascade of separate models. We capture additional dependencies with the novel application sequence-structured Conditional Random Fields to the problem of identifying discourse relations and their rhetorical types. The proposed Conditional Random Field model is more general than typically utilized in the literature, making use of non-factored feature functions to arrive at a conditional, sequential ranking model.

Finally, we demonstrate the general applicability of our proposed discourse parsing model by applying it to the problem of syntactic dependency parsing, itself an important determinant for discourse parsing. This points towards a layered sequential (re-)ranking architecture for complex language processing tasks applicable beyond discourse parsing.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

Utterances rarely exist in isolation. Rather, for any body of text there exist various discourse coherence relations spanning across sentence, clausal and phrasal boundaries connecting together various entities and eventualities appearing in the text. It is the nature and configuration of these relations that establish textual coherence and allow for the discourse to acquire meaning beyond the sum of its constituent utterances.

In this dissertation, we are concerned with a particular class of coherence relations called *rhetorical relations*. [1] Rhetorical relations are text structuring relations between *abstract objects* [Asher, 1993] (i.e., events, states, facts and propositions). They can be divided into *intentional* and *informational* relations. Briefly, intentional (or presentational) relations can be viewed as those relations attempting to modify or augment the hearer's (or reader's) belief state. Informational relations, however, aim to make the hearer aware

[1]Throughout this dissertation, the terms *rhetorical* relation and *discourse* relation will be used interchangeably.

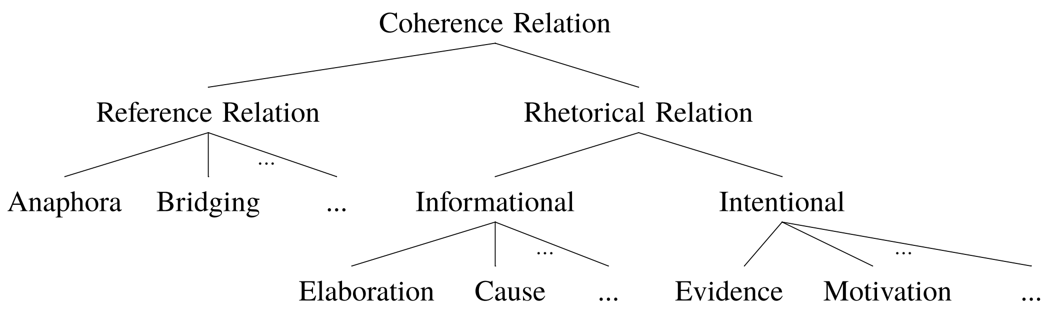


Figure 1.1: Coherence relation ontology

of semantic relations such as causality or elaboration that hold between abstract objects. To illustrate these differences, consider the simple discourse below [2]

(1) a George Bush supports big business.

b He's sure to veto House bill 1711.

At the intentional level, an EVIDENCE relation holds – (a) is providing evidence for (b) with the aim of increasing the reader's belief in the claim made in (b). In looking at the *content* of the two statements at the informational level, however, a CAUSE relation holds – the state of affairs in (a) is a cause for the action described in (b).

Rhetorical relations lie in contrast to identity-based *reference* relations such as anaphora (e.g., *He* referring to *George Bush* in 1), and meronymy (i.e., part-whole) relations like bridging and accommodation. Figure 1 provides an overview of coherence relations.

While all types of coherence relations are important for a complete picture of the discourse, and both informational and intentional rhetorical relations appear necessary [Moore & Pollack, 1992], the scope of this dissertation focuses on informational discourse pars-

[2]This example is from Moore & Pollack [1992].

ing. In particular, this dissertation addresses the problem of automatically *identifying* and *classifying the type* of informational discourse relations between abstract objects. For example, in (1) our goal is to establish the CAUSE relation between (a) and (b).

## 1.1 Motivation for Informational Discourse Parsing

There are two general motivations for pursuing automatic identification of informational discourse relations: 1) their use in natural language applications and 2) their use in facilitating better solutions to other semantic and pragmatic problems in natural language. We take these two areas in turn.

Discourse-level relations have the potential to benefit a variety of NLP applications. For example, in question answering, systems could use CAUSE relations to better answer questions about the causes or results of particular events. Information extraction systems, such as those developed for MUC-4, [3] were required to identify complex scenario-level events [4] which often involved relating various "atomic" events along with their participants. These atomic events, making up a scenario-level event were frequently scattered across multiple clauses or sentences. The ability to identify informational discourse relations was noted as key component technology for performing this level of analysis [Sundheim, 1992]. While many of the MUC-4 systems tackled the problem of identifying informational relations, either directly or indirectly, such systems were very much engi-

[3] A series of Message Understanding Conferences (MUCs) were sponsored by DARPA in the 1990s. These challenges involved identifying entities (e.g. *Person*, *Organization*), relations between them (e.g., *Employed-At*) as well as more complex events in newswire texts.

[4] The MUC-4 scenario-events were terrorism related involving arson attacks, kidnapings, bombings, etc.

neered for a particular task and domain. A *task-independent* layer of discourse analysis would likely enable a more rapid transition to new domains by avoiding the need to rebuild a discourse analysis component from scratch. The utility of discourse relations has also been established in areas such as automatic summarization [Marcu, 1998] and generation [Hovy, 1993].

Besides their use in applications, discourse relations are strongly intertwined with other problems in computational linguistics. In particular, when semantically interpreting an utterance it can help to understand the discourse relations the utterance is involved in. The framework of Segmented Discourse Representation Theory (SDRT) [Asher & Lascarides, 2003], in fact, focuses directly on this problem and formalizes how discourse, clause-level and lexical semantics interact. Asher & Lascarides [2003] demonstrate the benefit discourse relations provide for the analysis of: anaphora, temporal structure, bridging inferences, presuppositions, lexical ambiguity and other important problems in semantics and pragmatics. With regard to anaphora in particular, considerable work exists demonstrating how discourse structure can constrain the set of potential antecedents for anaphoric pronouns [Polanyi & van der Berg, 1999; Hirst, 1981; Hobbs, 1979].

## 1.2 Modeling Discourse

There are many different *levels* or granularities at which discourse coherence can be examined. The highest levels of coherence involve large bodies of text, possibly large sections or chapters. Texts must cohere at these high-levels (e.g., introductory material

One day after Delmed Inc. made top management changes and disclosed the end of an important business tie, its stock didn't trade and the company forecast a "significant" drop next year in sales of its core product. The disclosure came, a Delmed spokeswoman said, after the American Stock Exchange alerted the company that trading wouldn't resume in its stock until additional information about developments was provided. In addition to the forecast, the company also said it is examining potential cost cuts and reductions in overhead.

Figure 1.2: Example excerpt from the Penn TreeBank (file wsj_0970).

before expository material) as well as at middle layers of discourse involving coherence among scenes or complex events containing many sub-events in the case of narrative-like texts. Our focus in this work, however, is concerned mostly with low-level discourse involving relations between "atomic" eventualities, or *abstract objects*. This level of analysis arises out of focusing on relations that are lexicalized by *discourse connectives* such as *but*, *therefore*, and *moreover*.

The following steps are involved in low-level discourse parsing:

1. Segment the discourse into its most basic, usually non-overlapping, *discourse segments*.

2. Determine the structure (usually in a recursive manner) of the discourse by determining which segments and/or segment groups are related to each other at the discourse level

3. Identify the *type* of each rhetorical relation

Figure 1.2 provides an example discourse involving three sentences; Figure 1.3 illustrates one way in which such a discourse would be segmented — i.e, broken up into its

A One day after Delmed Inc. made top management changes
B and disclosed the end of an important business tie,
C its stock didn't trade
D and the company forecast a "significant" drop next year in sales of its core product.
E That disclosure came,
F a Delmed spokeswoman said,
G after the American Stock Exchange alerted the company
H that trading wouldn't resume in its stock
I until additional information about developments was provided.
J In addition to the forecast, the company also said it is examining potential cost cuts and reductions in overhead.

Figure 1.3: A segmentation of the discourse in Figure 1.2.

atomic discourse units. The discourse *connectives*, underlined, are the important discourse cues used to explicitly signal rhetorical relations.

The structure of the discourse is a hierarchical, ordered tree structure where each rhetorical relation is represented as a node in the tree with the node's children representing the relation's arguments. Figure 1.4 illustrates a potential discourse structure for the segments in Figure 1.3[5]. Note, however, that it is not strictly an ordered tree as the two ELABORATION relations exhibit *crossing dependencies*[6]; further, segment D has two parents.

## 1.3 Dissertation Overview

This dissertation is concerned with automatically identifying and assigning a type to discourse relations holding between abstract objects in English text. In contrast to most

[5]The rhetorical types shown here for the discourse relations are explained in detail in Chapter 8

[6]Crossing dependencies occur when the arguments for two different relations are sequentially *interleaved*. This phenomenon is discussed further in Section 5.4 and is a motivating case for identifying discourse relations jointly rather than independently.

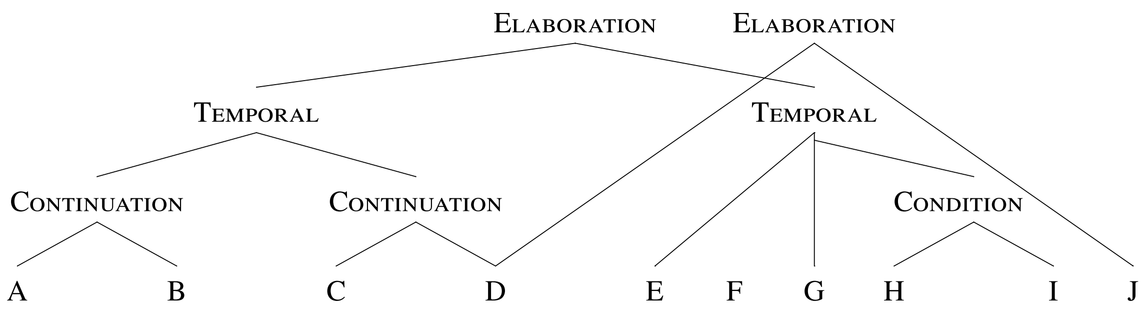


Figure 1.4: One possible discourse structure implied by the annotations provided in the Penn Discourse Treebank (WSJ article 0970), together with associated rhetorical relation types given the spans in 1.3.

approaches to discourse parsing, the approach taken is data-driven using the recently released Penn Discourse TreeBank (PDTB) [Miltsakaki et al., 2004b] as a source of gold-standard annotations from which to train and formally evaluate statistical machine learning methods. Before addressing the proposed work in detail, we first provide some background on the PDTB, the corpus used throughout this dissertation.

### 1.3.1 Overview of the Penn Discourse TreeBank

The Penn Discourse TreeBank provides a layer of discourse annotation over the entire Wall Street Journal portion of the Penn TreeBank [Marcus et al., 1993]. From a birds-eye view, the PDTB annotation effort aims at describing discourse structure as presented in Figure 1.4. In fact, the PDTB annotations do not explicitly construct a hierarchical representation. Instead, arguments are arbitrary text spans that *may* allow for a hierarchical interpretation when an argument span subsumes another connective and its two arguments. For example, the connective <u>and</u> in segment B in Figure 1.3 and its two arguments (in segments A and B) are subsumed within a single argument of the connective phrase

One day after.

Partly motivated by practical reasons, relations in the PDTB are annotated in just two cases: 1) when they are lexicalized by discourse connectives and 2) when they hold between adjacent sentence units not spanning paragraph boundaries. In the first case, the discourse connective is viewed as a predicate with two arguments, Arg1 and Arg2 (i.e., the two arguments participating in the relation). Connectives may be coordinating (e.g. *and*), subordinating (e.g., *when*) or adverbial (e.g., *nevertheless*). Subordinating and coordinating connectives both take their arguments *structurally*, based on clausal syntax and/or adjacency [Webber et al., 2003]. Adverbial connectives may, however, take their Arg1 *anaphorically*; it may reside anywhere prior in the discourse. The textual realizations of arguments of discourse relations may consist of arbitrary text spans (phrases, clauses, sentences and sequences of any of these) and may be overlapping. Thus, in contrast to most discourse frameworks, there isn't an explicit, a priori constrained notion of a discourse *segment* except the looser notion that the segment denotes an abstract object, which is typically a clause or clause-like construction.

These properties of the PDTB, while well-justified on methodological grounds, make the problem less amenable to traditional parsing approaches. Without well-defined, non-overlapping segments and with the inclusion of *anaphoric* connectives (e.g., the connective *also* in Figure 1.4) a purely tree-like, constituency-based structure is not possible.

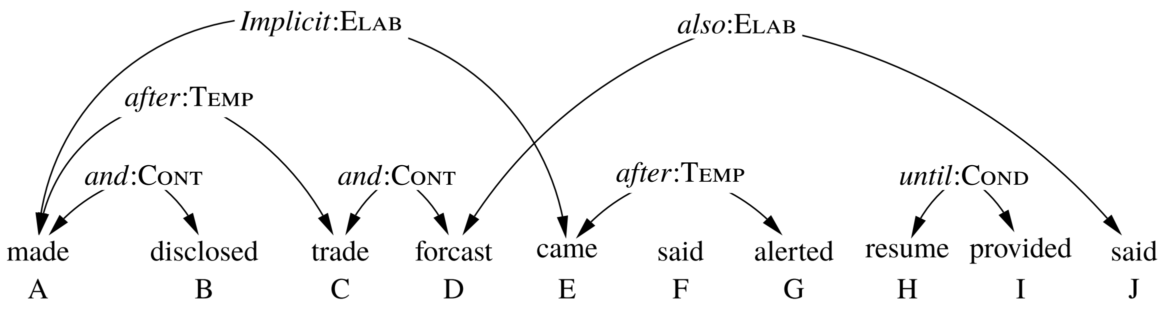


Figure 1.5: Non-projective dependency representation for the segmented discourse in Figure 1.3 with nodes in the graph representing the lexical heads of each discourse segment.

### 1.3.2 Dissertation Contributions

In this section, the major contributions of this work are summarized.

#### Dependency-based Representation for Discourse

Rather than confronting the issues just described and trying to augment a constituent parsing framework to accommodate them, in this dissertation we advocate recasting the target representation of the discourse to that of a multi-headed, non-projective *dependency structure*. Discourse relations are defined to hold between lexical heads, rather than argument extents. The dependency representation for the discourse in Figure 1.2 is shown in Figure 1.5. This representation accommodates shared arguments, avoids the problem of discourse segmentation altogether and also accommodates crossing dependencies in the discourse (e.g., between the two Elaboration relations shown in Figure 1.5).

Given the target discourse representation described, the discourse parsing problem in the context of this dissertation can be broken down into three components:

1. Identify discourse connectives among potential connective phrases.
2. Identify the arguments of discourse connectives
3. Ascertain the type of the relation holding over the two arguments for each connective.[7]

**Statistical Methods for Discourse Parsing and NLP**

A common shortcoming of many NLP systems is that they involve a series of components, often carried out in a sequential fashion with the output of previous stages available to down stream components. This makes the engineering task simpler, but has the potential for degraded accuracy since errors in earlier stages may propagate to later stages.

In this dissertation we explore a variety of discriminative, statistical machine learning methods to learn decision functions for the above problems. Specifically, we explore different types log-linear models that are especially well-suited to problem of selecting elements out of a set of candidate elements by viewing the task as a *ranking*, rather than classification problem.

The series of models put forth address the overall discourse parsing problem in an increasingly *global* manner along two dimensions. The first dimension is the degree to

[7]The PDTB also annotates relations between adjacent sentence units when there is not an explicit discourse connective signaling the relation.

which multiple processing stages are incorporated within a single step — e.g., identification of a discourse connective *together with* its arguments. The second dimension is the degree to which individual *decisions* are influenced by other (nearby) decisions in the discourse. Specifically, we examine the sequence of discourse connectives within a document or paragraph and capture dependencies between the arguments of adjacent discourse connectives within this sequence. This is achieved by generalizing sequence Conditional Random Fields to include non-factored feature functions.

We also demonstrate how the sequential statistical model developed here for discourse can be applied to syntactic dependency parsing and discuss interesting possibilities for future work using this type of model.

**Discourse Parser and Analysis**

A concrete artifact produced as part of this dissertation is a discourse parser which automatically annotates arbitrary text with discourse relationships following the three steps outlined above. As part of developing this parser, we carry out an extensive analysis of the contributions of various features for the tasks of 1) identifying connectives, 2) identifying their arguments and 3) ascertaining the rhetorical types of the relations. Additionally, we provide a detailed analysis of how various features affect performance in the context of the richer statistical models that capture multiple processing stages and/or capture sequential dependencies.

## 1.4 Organization

This dissertation is organized as follows:

- Chapter 2 provides a broad overview computational approaches to handling discourse. Many aspects of discourse are less-agreed upon than in syntax, for example, and this chapter aims to set this dissertation work within this larger context.

- Chapter 3 looks at the problem of identifying the arguments of discourse connectives within the PDTB 1.0

- Chapter 4 re-examines argument identification with a revised set of features and underlying syntactic representation, utilizes data from the final release of the PDTB (Version 2.0), and provides experiments on identifying arguments of implicit discourse connectives.

- Chapter 5 considers the full discourse parsing task which does not assume that the discourse connectives are given, but rather aims to identify the connectives along with their arguments, arriving at a discourse-level predicate-argument structure. Experiments in this chapter 1) compare independent predicate and argument identification vs. a joint model; and 2) present a the sequential ranking model and experimental results for identifying discourse predicates and arguments sequentially.

- Chapter 6 steps away from discourse and applies the sequential ranking model for identifying discourse structure introduced in Chapter 5 to syntactic dependency parsing, an important level of analysis for automatic discourse parsing.

- Chapter 7 returns to the full discourse parsing task, but makes use of automatic parses produced by the systems introduced in Chapter 6, rather than gold-standard parses.

- Chapter 8 examines the problem of identifying the rhetorical *types* of discourse predicates. Here, too, sequential dependencies are handled discriminatively within a Conditional Random Field model.

- Chapter 9 summarizes the major contributions of this work and outlines future directions.

# Chapter 2

# Modeling Discourse

This chapter provides some background on discourse processing. The first section gives a broad overview and some historical perspective, while the second section reviews more recent, data-driven methods for automatically identifying discourse structure. Finally, we introduce the Penn Discourse Treebank in detail and contrast it with other discourse frameworks.

## 2.1 Views of Discourse

Discourse theories and frameworks all aim to do roughly the following: explain how the meaning of text is more than a simple sum (or concatenation) of the meaning of its individual parts (i.e., clauses or sentences). Our focus here is on approaches that define *coherence relations* between units of text (usually clauses) in order to establish the discourse structure[1]. Approaches vary along a number of criteria, including: 1) the inventory and granularity of coherence relations, 2) whether or not and the degree to which a formal semantics is defined for the coherence relations 3) constraints imposed on the structure of the established coherence relations (e.g., tree vs. graph) 4) the degree to which there is a link between the coherence relations and the linguistic signals used to mark them and 5) the definition of a discourse unit (or *argument*) – e.g., whether they are (non-)overlapping text spans, propositions or other semantic abstractions associated with text spans, whether they can denote entities (or just abstract objects), etc.

We summarize here some important theories of discourse coherence relations, keeping in mind the criteria above. It is also worth noting that our discussion here emphasizes work focused on monologic, written text rather than dialog and/or spoken language.

**Surface-based Cohesive Relations**

One of the earliest influential accounts of discourse coherence relations is that of Halliday & Hasan [1976]. Their approach aims to describe how texts *cohere* by focusing on establishing the language-specific (in this case, English) linguistic devices used to explicitly establish conjunctive relations.[2] They only consider entire sentences as the units of discourse and provide little in the way of a precise semantics for the relations. The inventory of relations is of moderate detail, including various sub-types of the following relation categories: *additive* (i.e. parallel, elaboration), *contrastive*, *causal* and *temporal*.

[1]Note that other notions besides relations between text units can be useful for analyzing discourse such as focus, foreground and background, etc.

[2]Briefly, conjunctive relations are relations linking two units of "equal stature" - i.e., neither unit is subordinate to the other with its interpretation thusly constrained.

Related to Halliday and Hasan's approach is that of Martin [1992] which also focuses on conjunctive discourse relations. He allows for relations between clauses (not just sentences) and also makes precise the notion of an *implicit relation* which exists if a conjunctive discourse connective can be inserted into the text. This notion is revived in the Penn Discourse Treebank, where implicit discourse connectives are annotated.

Grimes [1975] provides an account of unlexicalized coherence relations where the arguments of the relations consist of propositions (not texts spans). Besides a more detailed inventory of coherence relations, Grimes introduces the fundamental notions of *paratactic* and *hypotactic* relations later leveraged in Rhetorical Structure Theory (see below), SDRT and elsewhere. Briefly, paratactic relations are coordinating-like relations between propositions whereas hypotactic relations are subordinating-like and hold between a primary proposition which dominates a secondary one.

## Coherence Relations of Hobbs

Hobbs [1985] places coherence relations within a larger computational approach to semantics and pragmatics based on *abductive reasoning*.[3] Hobbs defines a set of coherence relations, each with a relatively precise semantics. For example, Hobbs defines the *parallel* coherence relation roughly as follows:

(2) For two clauses, $C_1$ and $C_2$, infer the proposition $p(a_1, a_2, ...,)$ from $C_1$ and $p(b_1, b_2, ...)$ from $C_2$ where $\forall i.similar(a_i, b_i)$ for some definition of $similar$.

[3]Abductive reasoning is a form of unsound reasoning that aims to find the best *explanation* given a set of observations.

Within a framework based on abductive reasoning [Hobbs et al., 1988], Hobbs allows for discourse-level inferences, anaphora and lexical disambiguation to be handled in a unified way incorporating syntactic information as well as lexical semantics and world-knowledge. As these different stages of processing clearly influence each other, handling them *simultaneously* within a single framework, in this case using abduction, would seem advantageous.

Hobbs' inventory of relations and their semantics have strongly influenced other, more recent efforts at modeling coherence relations, including the Discourse GraphBank [Wolf & Gibson, 2005], SDRT [Asher, 1993; Asher & Lascarides, 2003] and The Penn Discourse Treebank [Miltsakaki et al., 2004b], the corpus of interest in this thesis.

### Rhetorical Structure Theory

Another discourse framework emerged in the 1980's with the work of Mann & Thompson [1988] in Rhetorical Structure Theory (RST). Rather than focusing on a rigorous model-theoretic approach to discourse semantics, RST focuses on providing a means to simply *describe* the way in which texts are structured. A set of 24 primary rhetorical relations was defined in the original RST proposal, motivated by careful analysis of a variety of texts (both dialogue and monologue). A simple piece of text and its corresponding RST analysis is shown in Figure 2.1.

RST constrains the structure of a discourse to that of a tree. One potential problem with this has to do with segments that appear in multiple relations. RST accommodates this to a limited degree by introducing a special ternary schema that operates over three

[Mary is in a bad mood]$^{1A}$ [because her son is ill.]$^{1B}$

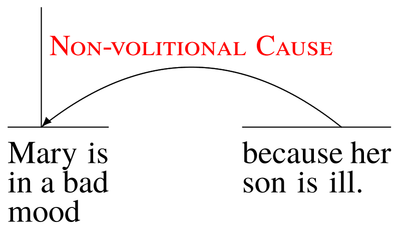


Figure 2.1: Causal relation with nucleus 1A and satellite 1B.

adjacent segments or instantiated schemas/relations and allows one segment to serve as a nucleus and the other two as satellites. The constituent structure is preserved by viewing all the segments within such a structure as siblings with a single parent. Figure 2.2 is an example of this.

[John went to the store]$^{2A}$ [in order to buy some tomatoes.]$^{2B}$ [He arrived just after it closed.]$^{2C}$

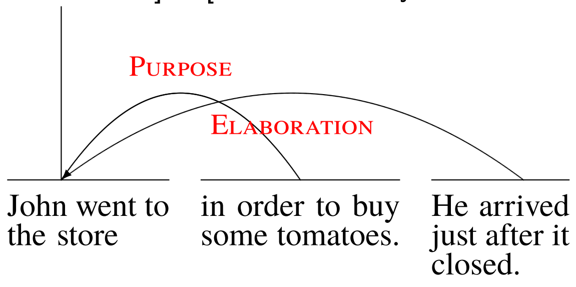

Purpose
Elaboration
John went to the store
in order to buy some tomatoes.
He arrived just after it closed.

Figure 2.2: A nucleus 2A with two satellites, 2B and 2C.

RST appears to run into difficulties, however, with discourses involving three segments where there is a shared argument between the two relations as in Figure 2.3. One remedy would be to model such cases differently as shown in Figure 2.4. Such an analysis seems

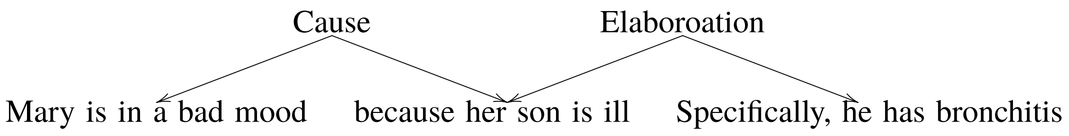


Figure 2.3: Discourse dependency structure indicating a shared argument.

less precise, however, as the content of the embedded ELABORATION relation is clearly not the most minimal unit of information required to establish the causal relation.

A final point which follows from the discussion here is that RST is unable to accommodate non-constituent structure (i.e., anaphoric or dependency structure [Webber, 2006]) in the discourse. For example, the ELABORATION relation established between the segments  and  in Figure 1.2 cannot be handled.

[Mary is in a bad mood]$^{3A}$ [because her son is ill.]$^{3B}$ [Specifically, he has bronchitis.]$^{3C}$

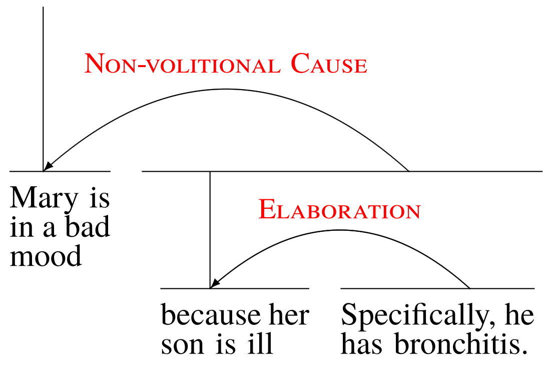


Figure 2.4: Hierarchical RST discourse structure.

Despite the above issues, RST has shown some potential for applications including text generation and summarization [Marcu, 1999b].

### Linguistic Discourse Model

The Linguistic Discourse Model (LDM) [Scha & Polanyi, 1988; Polanyi & Scha, 1984] provides a full account of discourse parsing by treating the task as an extension of sentence-level syntactic parsing. From a computational point of view LDM is appealing. The discourse segments, roughly clauses, are well-defined in terms of standard syntactic constructions. Further, discourses are structured purely as constituent trees making the target representation amenable to parsing methods designed for modeling sentence-level syntax.

LDM bears some similarities to RST as both are purely constituency-based. In contrast to the various schema within RST, however, LDM has three somewhat simpler re-write rules: *coordination* (for e.g., lists, narration), *subordination* and *n-ary* constructions over arbitrary rhetorical relations. The resulting structure for LDM is notably simpler than RST. All of the information regarding relations between segments exists within the constituency nodes of the resulting structure. In RST, however, a constituent node may serve simply to aggregate segments in which the actual rhetorical relations are denoted by relations between siblings.

### Grammar-Based Approaches

Along with LDM, other researchers have continued to explore the syntax-semantic interface at the discourse level with a specific aim at preserving *compositionality*, analogous to compositionality at the sentence-level present with most Montague-based approaches to semantics. Such approaches aim to extend grammars beyond the sentence level to the discourse level.

Gardent [1997] introduces a Feature-based Tree Adjoining Grammar framework for discourse parsing. Another, somewhat similar approach is the D-LTAG system, based on Lexicalized Tree Adjoining Grammar (LTAG) [Forbes et al., 2003]. More recent work in this vein has explored integrating synchronous Tree Adjoining Grammar and

SDRT [Danlos, 2008].

### Wolf and Gibson

Wolf & Gibson [2005] put forward a framework for which discourse is represented as a *chain graph*, a graph with both directed and undirected arcs. Interestingly, they arrived at their representation based on an empirical methodology rather than intuitions or preconceived constraints. Their approach started with a fixed set of coherence relations with relatively precise semantics. The relations roughly followed those of Hobbs [1985]. They had two annotators annotate 135 news articles for such relations without any preconceived notions about how the relations are structured. Given a set of guidelines on how to segment the texts (roughly at the clause level), the annotators carried out the following steps: 1) segmented the texts into basic discourse units 2) grouped the units into larger units when such units acted as an argument to a relation 3) identified when two segments (or segment groups) were connected via a discourse relation and 4) identified the type of relation.

In their analysis of the annotated texts, they found a surprising degree of non-tree discourse structure. In fact, when considering all the rhetorical relation types they annotated, they noticed that nearly 12.5% of the relations would need to be deleted to remove all crossing dependencies. They also report that 41% of all segments have two or more incoming arcs, indicating that shared arguments is a frequent phenomenon.

Consider the constructed example below from Wolf & Gibson [2005] which exhibits crossing dependencies:

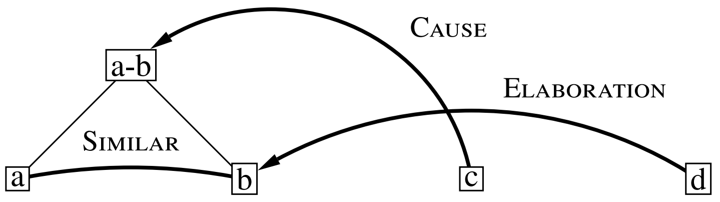


Figure 2.5: Graphical representation of the discourse in Example 3.

(3) a. Susan wanted to buy some tomatoes

b. and she also tried to find some basil

c. because her recipe asked for these ingredients.

d. The basil would probably be quite expensive at this time of year.

The discourse in 3 contains a crossing dependency but the Elaboration relation between (b) and (d) is essentially between an entity in the discourse (i.e., *basil*) rather than between propositions or facts. Of course, (d) could have stated the following in which the relation between (b) and (d) holds between abstract objects — i.e., the eventualities of *trying to find basil* in (b) and *not succeeding in finding basil* in (d).[4]

(4) She didn't succeed in finding any basil, however.

It does appear, however, that a significant portion of the crossing links in the Graph-Bank have to do with establishing *entity-level* coherence. Many other crossing dependencies are due to the *attribution* relation, which while arguably a discourse relation, certainly has a different nature than Hobbs' original relations. Nevertheless, the Graph-Bank results have provoked additional discussion and research into the structural con-

[4]Note that this particular example constructed by Wolf and Gibson might best be analyzed at an intentional level. An interesting open question is whether such crossing dependencies occur more frequently when analyzing discourse structures more naturally analyzed from an intentional view.

straints on discourse and corroborate other claims regarding the inadequacy of trees for discourse [Danlos, 2004].

### 2.1.1 Semantically-Driven Approaches

A variety of theories on discourse interpretation have emerged based on work on model-theoretic semantics for language [Montague, 1974; Dowty et al., 1981]. Most of these theories make use of some form of *dynamic semantics*, including Kamp's Discourse Representation Theory (DRT) [Kamp, 1981; Kamp & Reyle, 1993]. Briefly, DRT can be viewed as dynamically interpreting a discourse, one sentence at a time, along the way updating a representation of the discourse, known as a Discourse Representation Structure (DRS). While subsequent approaches (cf. [Groenendijk & Stokhof, 1991]) have removed the need for an explicit representation (and restored the potential for compositionality not present with DRT), these approaches are all related by their dynamic nature: the meaning of an utterance is determined by the grammar *and* the prior discourse.

Segmented Discourse Representation Theory (SDRT) [Asher, 1993; Asher & Lascarides, 2003] is a theory that extends DRT to handle *discourse relations*. Discourse relations are provided a precise semantics largely stemming from the work of Hobbs [1985]. In addition, a mechanism is provided for deriving discourse relations using defeasible reasoning. The simplest example of such default reasoning involves a basic assumption that a *narrative* (or *temporal*) relation holds between two adjacent sentences:

(5) John felt dizzy. He fell on the ground.

In the above, a causal relation (or result relation, to be precise) also appears to hold (though it can be argued to be ambiguous). In general, of course, this need not be the case. However, such interpretations can be overridden based on discourse cues as in:

(6) John felt dizzy *because* he fell on the ground.

While aiming to identify such relations is an important aspect of SDRT, its goals are much broader, aiming to integrate semantic and pragmatic interpretation within a single logical framework. SDRT adopts a rich view of lexical semantics based on Generative Lexicon (GL) [Pustejovsky, 1995]. An underlying aim of GL is to properly distribute the meaning of a sentence across its various components, rather than allow for some word types, verbs in particular, to dominate the meaning derivation. GL, in effect, moves beyond standard *compositionality* to allow for richer operations (i.e., beyond function application) to reflect the full degree to which context determines meaning. SDRT provides additional context with the formalisms that can aid in lexical disambiguation [Asher & Lascarides, 1995] in frameworks such as GL. Lexical semantics as provided in GL, in turn, also feeds back to the discourse layer and is required for cases such as:

(7) John fell. Max pushed him.

The standard discourse interpretation (i.e. *before*('fall','push')) is overridden by the lexical (or perhaps common sense) properties of the events in the two sentences. More specifically, it is the case that the event structure for *push* includes "lexical knowledge" to the effect that "pushing results in some kind of movement". This, coupled with similar lexical knowledge about *fall* can allow for the proper inference to be made.

While not necessarily related to SDRT, lexical semantics also plays an important role in *discourse verbs* [Danlos, 2006], such as *cause*, *lead* or *precede*. These verbs take abstract objects as arguments and can function in a similar capacity to discourse relations. Even prepositions can convey discourse relations [Danlos, 2007]:

(8) John died of cancer.

## 2.2 Data-driven Computational Models of Discourse

In this section we discuss recent approaches towards parsing discourse that have taken a data-driven, frequently machine-learning based, approach. We first discuss a methods for identifying discourse structure and subsequently look at approaches for categorizing the types of rhetorical relations.

### 2.2.1 Discourse Structure Parsing

#### Discourse Chunking

Within the framework of RST, rather than tackle the full discourse parsing task, Sporleder & Lapata [2005] present a framework for identifying the elementary discourse units (i.e. segments) and labeling each as a nucleus or satellite. While their system removes the important and difficult step of establishing a *relation* between two elementary discourse units (*edus*), they are able to achieve good performance at this task — 78.4% F-measure at identifying and labeling edus. They demonstrate the system's utility on a sentence compression task.

### Intra-sentential Discourse Parsing

Soricut & Marcu [2003] develop an intra-sentential discourse parser in the style of RST. They leverage the RST Discourse Bank that contains 385 Wall Street Journal articles from the Penn Treebank annotated with RST relations. For the sentences in the corpus that have associated with them a single RST discourse tree, they train and evaluate a statistical model able to 1) segment the discourse into elementary discourse units, 2) generate the (binary) discourse parse tree associated with the sentence (with the segments as leaves in the tree) and 3) associate with each internal node in the tree a rhetorical relation.

The segmentation model is a simple generative probabilistic model. The model estimates the conditional probability $P(b|w_i, t)$ where $b$ is a Bernoulli random variable indicating whether or not a segment boundary is present at word $w_i$ given the parse tree $t$ for the sentence. The distribution is estimated using counts over a single feature that captures the lexical and syntactic context around the word $w_i$. More specifically, the feature considers the uppermost node in the parse tree $t$ with the lexical head $w_i$ (using standard lexical head projection rules [Magerman, 1994; Collins, 1999]) along with that node's children and their lexical heads. Given that single feature, the probability $P(b|w_i, t)$ is estimated by the smoothed ratio of the number of times that feature occurs *with* a segment boundary over the total number of times it occurs. The segment model is able to identify segments with a balanced precision and recall F-measure of 83.1 using automatic (Charniak) parses and 84.7 with gold-standard parses.

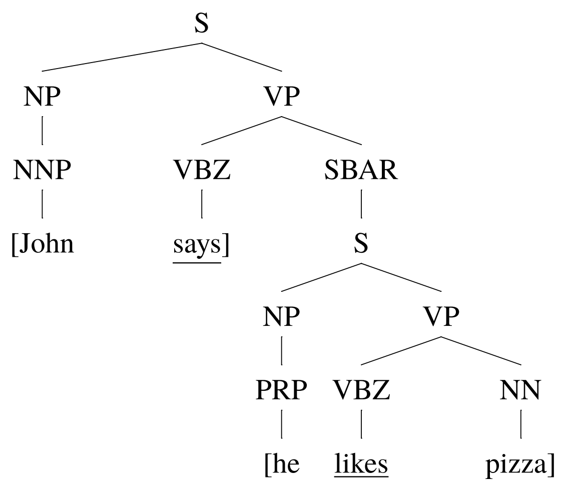


Figure 2.6: Syntactic tree for a sentence with a discourse boundary after “says”.

The discourse parsing model is also a simple generative model which computes the probability of a discourse parse tree based on a feature called the *dominance set*. Let $DST$ represent a sentence with a syntactic parse tree together with its discourse segments derived with the above segmentation model. The dominance set, $D$, of $DST$ is a set of pairs $(i, N) \prec (j, M)$ where $i$ and $j$ denote $i$th and $j$th segments within the sentence, respectively. $N$ denotes the head node of the $i$th segment (i.e., the highest node within the tree that has a the head word for the $i$ segment as its lexical head) while $M$ denotes the parent of $N$ and represents the node within segment $j$ to which segment $i$ *attaches*.

To make this concrete, given the simple sentence shown in Figure 2.6 the discourse segment boundary would lie between “says” and “he”. The dominance set would consist of a single pair (one pair for each boundary) $D = \{(2, SBAR(likes)) \prec (1, VP(says))$.

The probability for a particular constituent, $c$, of a discourse tree $DT$ is simply the product $P_s(ds(c)|filter_s(D, c)) \cdot P_r(rel(c)|filter_r(D, c))$ where $ds(c)$ denotes the discourse segment of the constituent $c$ and $rel(c)$ denotes its relation (or rhetorical) type and where the $filter_s$ and $filter_r$ functions simply extract the portions of the dominance set $D$ relevant to the candidate discourse constituent $c$. These probabilities are estimated using maximum likelihood estimation on the training corpus with various smoothing techniques. The parser will identify the discourse parse, $DT$, that maximizes $P(DT|D) = \prod_{c \in DT} P_s(ds(c)|filter_s(D, c)) \cdot P_r(rel(c)|filter_r(D, c))$. That parse can be identified using a bottom-up dynamic programming algorithm.

The resulting parser can identify unlabeled discourse relations with 70.5% accuracy using fully automatic methods, 73.0% accuracy with gold-standard syntactic parses and 96.2% accuracy with both gold-standard parses and discourse segments. Poor automatic discourse segmentation quality is thus the primary bottleneck for improved discourse parsing in this framework.

### RST-Style Rhetorical Parsing

Marcu [1999a] presents a machine learning-based approach for building up RST rhetorical structures. Marcu uses an approach inspired by early work on syntactic and semantic parsing involving two separate steps. The first stage identifies the elementary discourse segments. The second stage involves a parser that takes a sequence of discourse segments from a document and builds a tree using a series shift and reduce operations. SHIFT operations take an incoming segment and add it to the top of the stack. REDUCE operations take the top two elements from the stack (two trees or segments) and combine them into a new tree that is placed on the stack. Separate reduce operations are employed to handle different local structures (e.g. based on nuclearity) and rhetorical types. A decision procedure for selecting which operation to perform at each step was learned using C4.5 decision-tree induction that considered various structural features (e.g. of the top two trees in the stack), features regarding the presence of cue-phrases, as well semantic similarity measures based on lexeme-based cosine-similarity and Wordnet.

Despite being a data-driven approach, the shift-reduce parser approach would seem vulnerable to cascading errors. While the parser builds up a discourse structure in the form of a tree, it does so in a greedy fashion and mistakes early on in the process have the potential to adversely effect subsequent decisions.

### Probabilistic Parsing for Discourse

Baldridge & Lascarides [2005] used head-driven probabilistic parsing approaches developed for sentence-level syntax [Collins, 1999] and applied them to the task of inducing parsers for discourse. They annotated a set of dialogs from the Redwoods Corpus with a set of rhetorical relations espoused by SDRT.

There has also been work that has applied maximum spanning tree-based dependency parsing algorithms [McDonald, 2006] and discriminative training to discourse parsing with some promising results. Baldridge et al. [2007] present an dependency-based approach to discourse and appear to improve upon the results in Baldridge & Lascarides [2005] when evaluating against the task of identifying the lexical head of each discourse argument. This work bears some similarity to the work in this dissertation in that it uses a dependency representation as well as discriminative learning methods that can handle arbitrary features without the need to consider their interdependencies.

### 2.2.2 Classifying Rhetorical Relations

We have already discussed data-driven methods for identifying RST-based rhetorical relations. Outside of RST, we have previously leveraged the GraphBank corpus to classify coherence relations using a maximum entropy classifier [Wellner et al., 2006]. Sporleder & Lascarides [2005, 2008] and Marcu & Echihabi [2002] both present methods for identifying unmarked rhetorical relations[5] by leveraging discourse connectives to automatically label new examples in order to classify relations into subsets of the rhetorical relation types within SDRT and RST, respectively.

### 2.2.3 Temporal Discourse

Very much related to understanding informational discourse coherence relations is understanding *temporal* relations between abstract objects. Temporal relations may be viewed as a sub-class of discourse coherence relations. In many respects, however, it may be more natural to view temporal relations somewhat distinct from coherence relations generally [Mani & Pustejovsky, 2004]. In particular, isolating the temporal aspects of discourse provides a useful level of abstraction — e.g., it may be easier to infer that event $e_i$ occurs before $e_j$, but difficult to determine whether $e_i$ *caused* $e_j$. Conversely, focusing on temporal aspects alone facilitates a finer level of analysis. For example, the TimeBank

[5]Unmarked rhetorical relations are those that are not signaled by a discourse connective such as *but*, *because*, etc.

corpus [Pustejovsky et al., 2003] distinguishes between the case where $e_i$ occurs at *some interval* before $e_j$ and the case where it occurs *immediately* before $e_j$, a distinction not likely to be made amongst the relation type inventories of most discourse frameworks.

Data-driven methods for identifying and classifying temporal relations in the Time-Bank Corpus are presented in Mani et al. [2006], Bethard et al. [2007] and Lapata & Lascarides [2006].

## 2.3 The Penn Discourse Treebank

The PDTB, the corpus we use for our experiments, differs from most other discourse-level annotation efforts in its bottom-up, lexically-driven approach. Rather than identifying all possible discourse relations, the PDTB focuses first on annotating relations lexicalized by discourse connectives that explicitly occur in the text along with their two arguments. These discourse connectives include coordinating conjunctions (e.g., *and*, *or*), subordinating conjunctions (e.g., *because*, *when*, *since*) and discourse adverbials (e.g., *however*, *previously*, *nevertheless*). This aspect of the PDTB, its explicit connectives and their arguments, is somewhat theory neutral — i.e., by identifying such connectives, interpreting them as predicates and identifying their arguments no statement is being made a priori about 1) the structure of the discourse (e.g. whether it is tree-like) or 2) the types and semantics of the rhetorical relations realized as discourse predicates.

Discourse arguments in the PDTB represent abstract objects [Asher, 1993] which include facts, propositions and events. Each argument must include at least one predicate and can be realized as: a clause, a VP within VP coordination, a nominalization (in certain, restricted cases), an anaphoric expression or a response to a question. Each connective has two arguments: Arg2 is the argument syntactically connected to the connective in the same sentence and Arg1 is the other argument which may lie in the same sentence as the connective or, generally, anywhere prior in the discourse.

### 2.3.1 Examples

Below are a few examples from the PDTB. Each Arg1 is denoted in *italics* and each Arg2 is denoted in **bold**. The head-words for each argument are underlined. We discuss and motivate the identification of head-words in Section 3.1.

(9) *Choose 203 business executives, including, perhaps, someone from your own staff,* $\boxed{\text{and}}$ **put them out on the streets**, to be deprived for one month of their homes, families and income.

(9) shows an example of a coordinating connective *and* and its two arguments. In this case, the Arg1 lies in the same sentence as the connective. It is also possible for the Arg1 to lie outside the sentence (usually in the immediately preceding sentence) when the coordinating connective begins a sentence.

An example of the subordinating connective, *because* is shown below in (10). This example brings up some interesting ambiguities that arise quite regularly in the data. An alternative reading for this example might only include the extent *to duck liability* for the

ARG1. That is, the predicate *be able* could be read to include the discourse relation and its two arguments as an argument.

(10) *Drug makers shouldn't be able to duck liability* $\boxed{\text{because}}$ **people couldn't identify precisely which identical drug was used.**

Both coordinating and subordinating connectives are *structural* [Webber et al., 2003]. Discourse adverbials however, take one argument, ARG2, structurally but the other can be anaphoric: its ARG1 may be present anywhere in the current running discourse with little or no restriction. Example (11) shows the case in which the ARG1 lies in the previous sentence. In many cases, however, it resides in the same sentence as the connective or many sentences prior in the discourse.

(11) *France's second-largest government-owned insurance company, Assurances Generales de France, has been building its own Nagivation Mixte stake*, currently thought to be between 8% and 10%. Analysts said **they don't think it is contemplating a takeover**, $\boxed{\text{however}}$, and its officials couldn't be reached.

The PDTB, Version 1.0, contains a total of 18505 explicit connectives annotated with discourse arguments. The annotations are layered on top of the Penn Treebank-II (PTB) parse trees and cover all 25 Wall Street Journal (WSJ) sections amounting to over 1 million words. The final version of the PDTB, Version 2.0, was released in February 2008 and includes 18459 explicit connectives after refining the annotations in the earlier version.

| PDTB Relation Category | Number of Instances |
|---|---|
| Explicit | 18459 |
| Implicit | 16224 |
| AltLex | 624 |
| EntRel | 5210 |
| NoRel | 254 |

Table 2.1: Frequencies of the various discourse relation categories in the PDTB 2.0

### 2.3.2 Beyond Explicit Connectives

A core aspect of the PDTB is its annotation of explicit connectives and their arguments. As apparent when examining any body of text, *unlexicalized* discourse relations are present in many cases. That is, informational coherence relations exist without any particular discourse connective phrase signaling them, such as in Example (7). Rather than trying to identify all such relations as in Wolf & Gibson [2005], however, the PDTB 2.0 takes a more constrained approach and annotates relations between adjacent sentence units within the same paragraph exactly when there is no explicit connective signaling a relation with one argument in each sentences. These relations include: 1) *implicit* relations which are those relations that could be inferred by inserting a discourse connective phrase[6] at the beginning of the second sentence, 2) AltLex relations which are coherence relations between the two sentences that are signaled by some phrase other than a discourse connective and where the insertion of a connective phrase (to establish an *implicit* relation) would be redundant, 3) EntRel relations in which there is coherence between two adjacent sentence units but relating entirely to the entities in those sentences

[6]In fact, the PDTB also allows for *multiple* implicit connective phrases to be associated with a discourse relation.

and not over abstract objects and 4) NoRel where there is no coherence between the two sentence units. Below are simple examples of an implicit relation (constructed), an AltLex relation (from the PDTB) an EntRel relation (constructed), and a NoRel example, respectively:

(12) *John pushed Max.* Implicit= As a Result **Max fell.**

(13) *In September, she pleaded guilty and paid a $500 fine.* AltLex = Her alternative **was 90 days in jail**.

(14) *Howard is 89 years old* EntRel **He turns 90 in a couple of weeks**

(15) *The new Explorer sport-utility vehicle, set for introduction next spring, will also have rear-seat belts.* NoRel **Mr. Leinonen said he expects Ford to meet the deadline easily**.

Table 2.1, taken from [Prasad et al., 2008], provides the frequencies in the PDTB for the various relation categories.

### 2.3.3 Annotation of Relation Types

The PDTB 2.0 also annotates the type, or *sense*, of all explicit, implicit and AltLex discourse relations. This layer of annotation makes some theoretical commitments: the types of relations are *informational* relations as opposed to *intentional* relations and a particular inventory of relations was selected.[7] The relation types are inspired by those in

[7]Note, however, that it could easily be argued that the most salient coherence relations in the WSJ texts that constitute the PDTB are, in fact, better described with informational relations.

SDRT [Asher & Lascarides, 2003] and Hobbs [1985], but make a number of refinements. Interestingly, the PDTB relation type categories are hierarchical, including four top-level categories (or *classes*), 16 mid-level categories (or *types*) and 23 low-level categories (or *sub-types*).

Explicit and implicit (including AltLex) relations may be assigned one *or two* senses. Below is an example of an implicit relation annotated with the implicit connective phrase "because":

(16) *The government's construction spending figures contrast with a report issued earlier in the week by the F.W. Dodge Group.* Implicit = BECAUSE **Dodge reported an 8% increase in construction contracts awarded in September.**

This was annotated with both a REASON sense and a SPECIFICATION sense as the second sentence both explains why the "spending figures contrast with a report" and elaborates by providing more specific details on what was reported.

We defer a detailed discussion of the various sense distinctions until Chapter 8.

### 2.3.4 Inter-annotator Agreement

The PDTB, like many annotated corpora, was designed to try to achieve high inter-annotator agreement. Clearly, if humans frequently disagree on the presence, structure or rhetorical type of discourse relations we can not expect automated systems to perform well. Overall agreement for identifying the arguments (both ARG1 and ARG2) was 90.2% for explicit connectives and 85.1% for implicit connectives [Prasad et al., 2008], with generally lower scores for Arg1 and higher scores for Arg2 [Miltsakaki et al., 2004a]. When considering a *partial match* metric, which gives credit to arguments that overlap to a sufficient degree, those agreements move to 94.5% and 92.6% for the arguments of explicit and implicit connectives, respectively. Agreements for assigning relation types or senses to connectives were: 94% agreement at the *class* level, 84% agreement at the *type* level and 80% agreement at the *sub-type* level. These results seem comparable to other discourse annotation efforts. For example, in the RST Discourse Treebank [Carlson et al., 2003], agreements for identifying the two spans participating in a relation were in the range of 78% to 93%. Agreement percentages for assigning rhetorical types to given relations ranged from 68% to 79% for an inventory of 16 rhetorical types, which is slightly below the agreement rates in the PDTB.

# Chapter 3

# Identifying Arguments of Discourse Connectives

The study of discourse is concerned with analyzing how phrase, clause or sentence-level units of text are *related* to each other within a larger unit of text (e.g., a document). Long recognized as important in dialog and text generation, this level of analysis is important generally for applications needing to place events and propositions in their proper context such as scenario-level information extraction, question answering, summarization, sentiment analysis and others.

In line with much of the NLP research agenda, recently a number of annotated corpora have emerged which encode discourse-level phenomena, making it possible to apply supervised, empirically-driven techniques to identifying discourse relations. Some of these corpora were reviewed in the previous chapter. While these corpora differ in many ways, they all more or less encode the problems that involve: 1) identifying/segmenting the basic units of discourse (e.g., clauses, phrases), 2) determining for which pairs of segments (or segment groups) a discourse relation exists, and 3) characterizing the *type* of relation (cause, elaboration, etc.) between segment pairs.

In this chapter we focus on problems (1) and (2) above. However, rather than explicitly identifying the discourse segments and then deciding for which pairs a relation exists, we focus on identifying relations between the pairs of *head words* that *represent* the discourse segments. In this sense, the problem resembles that of predicate-argument identification where the predicates are discourse connectives and the arguments are single words which serve as anchors for the discourse segments.

To address the problem of identifying the arguments of discourse connectives we incorporate a variety of lexical and syntactic features in a discriminative log-linear ranking model. To capture dependencies between the two arguments of a connective we use a log-linear *re*-ranking model to select the best argument pair from a set of N-best argument pairs that are provided by simpler independent argument models. Further, we provide an analysis of the contribution of the various features demonstrating that features based on a dependency parse representation outperform features derived from a constituent tree parse.

Some of the material from this chapter was reported in Wellner & Pustejovsky [2007] and was carried out on Version 1.0 of the PDTB.

## 3.1 Head-Based Representation of the PDTB

In contrast to other annotations layered on the PTB such as PropBank [Palmer et al., 2005] and NomBank [Meyers et al., 2004], the arguments of a discourse connective frequently do not correspond to a single parse tree constituent. Arguments consist instead of a *set* of non-overlapping constituents from the parse tree. This target representation makes the process of identifying the arguments to discourse connectives difficult since the space of candidate arguments extents is considerably larger than for PropBank parsing, for example. Even without this added difficulty, discourse segmentation is one of the most difficult stages in discourse parsing [Soricut & Marcu, 2003]. While the segments themselves may be useful in certain contexts, for many applications, if not most, it will still be necessary to *interpret* these segments (e.g. at the predicate-argument level). As such, we argue that, in general, identifying the lexical *heads* of these discourse segments is sufficient and perhaps even preferable for this stage of processing. A problem arises, however, with arguments that consist of sequences of abstract objects represented as coordinated or subordinated sequences of VPs, clauses or sentences. What should the head be in such cases? By convention we designate the extent head as the head of the first element in the sequence for multi-clause arguments. In (4), the head of the Arg2 would be *went*, but its implicit scope includes the second VP coordinate headed by *caught*.

(17) Mr. Dozen even related *the indignity* <u>*suffered*</u> $\boxed{\text{when}}$ **he and two colleagues <u>went</u> on an overnight fishing expedition of the New Jersey shore and caught nothing.**

The problem then becomes how to determine the end of the sequence of abstract objects. In many cases, there is a “natural end” to such sequences based on the syntax. In (4), the natural end is simply the end of the VP coordination. Difficult cases remain, however, particularly with multi-sentential ARG1S of anaphoric connectives. Determining the end of the these arguments seems non-trivial. Nevertheless, identifying the beginning of the argument (via its head) is an important step in modeling these difficult cases.

### 3.1.1 Head Identification

Identifying the head of a discourse argument given its extent (as described by a set of constituent sub-trees in the PTB) consists of two steps. First, we construct a single syntactic tree formed by taking all of the sub-trees in the extent, finding their least common ancestor (LCA) node and including all intermediate nodes from the subtrees to the LCA node. Note that none of the *other* children of the intermediate nodes other than the subtrees that span the argument extent are included. So, for example, in Figure 3.1 if an argument extent included the entire sentence except the phrase “the Commerce Department said”, the NP “the Commerce Department” as well as the verb “said” would be pruned from the syntactic tree.

After the diminished tree is identified that spans just the argument extent, a slight variation of the head finding algorithm in [Collins, 1999] is applied to it in order to identify the head. Briefly, the head finding algorithm consists of a set of rules that indicate how to traverse a parse tree from a constituent phrase internal node to a leaf

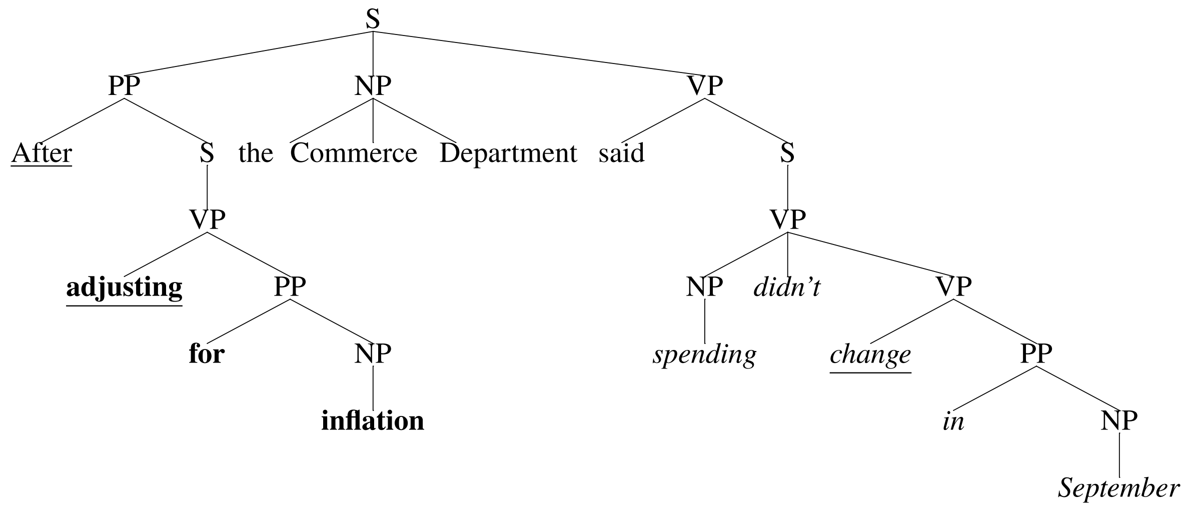


Figure 3.1: Syntactic structure and discourse arguments for the connective "After". The lexical heads of each argument are underlined.

node, where the leaf node represents the "head" of the original constituent specified. So, for example, a rule such as PP → Left → IN,TO,VBG,VBN,RP,FW indicates that for a PP internal node, select the child node to traverse by first identifying the first word with part-of-speech IN (i.e., a proposition according to Penn Treebank conventions) from the left. If no such word is found, look for the first word with part-of speech TO from the left, etc. The elements on the right hand side of a rule may be terminal nodes specified with parts-of-speech or other internal constituent nodes (e.g. VP, NP). Rules are then applied recursively until reaching a terminal node.

Figure 3.1 provides an example indicating the arguments to the connective "After" and the derived argument heads, which are underlined. Note that the rules used here identify the "semantic heads", which correspond to the verb most carrying the meaning for VPs and for predicate nominals and adjectives in copula constructions.

## 3.2 Discourse Argument Identification

Identifying the arguments of discourse connectives can be naturally formulated as a binary classification task where separate classifiers are trained for each type of argument — i.e., Arg1 and Arg2. First, a set of candidate arguments, $\alpha_i$ is gathered for each connective, $\pi$. Training instances, $\langle \alpha_i, \pi \rangle$, are then created for each candidate with respect to the connective. A training instance is positive if $\alpha_i$ is the true argument for $\pi$ and negative otherwise. At decoding time, the candidate classified positively with the highest probability (or score) compared to the other candidates is selected as the argument.

An alternative to using a standard classification approach is to use a *ranking* model. The advantage of the ranking model is that candidate instances are compared against each other *during training* as well as during decoding. In contrast, with a standard classifier, separate instances (i.e. candidates) are trained and classified as if they were completely independent. We use a log-linear ranking model. Such models have been used for a variety of other tasks including co-reference [Denis & Baldridge, 2007], question answering [Ravichandran et al., 2003] and parse re-ranking [Charniak & Johnson, 2005]. Appendix B provides further details on how the parameters are estimated for such models.

For a given Arg1 candidate, $\alpha_i$, the probability of that candidate being the argument given the connective, $\pi$, and the document, $x$, is defined according to the model as:

$$P_1(\alpha_i|\pi, x) = \frac{\exp\left(\sum_k \lambda_k f_k(\alpha_i, \pi, x)\right)}{\sum\limits_{\alpha_j \in C_1(\pi,x)} \exp\left(\sum_k \lambda_k f_k(\alpha_j, \pi, x)\right)} \tag{3.1}$$

where the $f_k$ are feature functions, the $\lambda_k$ are their weights and $C_1(\pi, x)$ is the set of candidate Arg1 arguments for the connective $\pi$ in the document $x$. The model for Arg2 is defined analogously, but may in fact use a different set of features or a different candidate generation function. Note that the function $C_2$ will denote the candidate generation function for Arg2 arguments and $\beta_j$ will refer to a specific Arg2 candidate.

At training time, all potential candidates of a particular type for a given connective are provided to the ranking model as a distribution: the correct gold-standard candidate receiving a probability mass of 1.0 and the other candidates receiving masses of 0.0. During decoding, we select candidates in the same way as for training and produce a distribution over these candidates according to Equation 3.1, selecting the candidate assigned the highest probability by the model as the argument.

We compared both the above ranking model and a standard binary Maximum Entropy model (i.e., logistic regression) and found the ranking model to have a small but consistent edge over the classifier. Accordingly, we only report results here using the ranking model.

### 3.2.1 Candidate Selection

Identifying the candidate arguments, $\alpha_i \in C_1(\pi, x), \beta_j \in C_2(\pi, x)$, is an important aspect of the problem. For Arg1 candidates in particular, there are conceivably very many possible candidates for a given connective stretching back from the sentence containing the connective to the beginning of the document. We employ two simple criteria to reduce the space of candidate argument head words. First, we only consider argument candidates that have an appropriate part-of-speech (all verbs, common nouns, adjectives).

Second, we only consider candidates that are within 10 “steps” of the connective where a single step includes a sentence boundary or a syntactic dependency link within a sentence (see Figure 3.2). Only candidates lying within the same sentence as the connective are considered for Arg2.

### 3.2.2 Features

We used a variety of features for identifying the discourse arguments of a connective.

**Baseline Features.** Our baseline features included simply the connective and argument words, where the connective appears in the sentence, whether the argument precedes or follows the connective and whether the argument is in the same sentence as the connective or not.

**Constituent Path Features.** As noted in work on semantic role labeling, features derived from the constituent parse of the sentence can be very helpful for deriving the argument structure of predicating verbs [Toutanova et al., 2005] and nouns [Jiang & Ng, 2006]. Syntax plays a strong role in identifying discourse arguments, too, though even for structural connectives it by no means “aligns” with the discourse structure [Dinesh et al., 2005]. We introduced a feature capturing the constituent tree path from the connective to the candidate argument as well as variants in which repeated nodes and part-of-speech nodes are removed from the path. If the argument lies in a different sentence, the path from the connective to the argument consists of the path from the connective to the top node of its sentence, followed by a series of virtual *SENT* nodes for the intervening sentences and then ending with the path from the top node of the sentence containing the argument to the argument head itself.

**Dependency Path Features.** We experimented with a number of syntactic features based on a *dependency* parse representation. The primary motivation here is that it provides for a more compact and natural representation of the syntax, providing for better syntactic features with less data sparseness than constituent path features. The dependency representation we use is that put forth in de Marneffe et al. [2006] and we apply their approach to deriving the dependency structure from the constituent parse. The features used here include the (shortest) dependency path from the connective to the prospective argument and two collapsed versions removing coordination links as well as repeated links of the same type. For argument candidates in prior sentences, we introduce *SENT* links for each intervening sentence.

**Connective Features.** Different discourse connectives behave differently depending on their type. A potentially important feature then involves capturing the connective type: (coordinating, subordinating or adverbial). We use the categorized lists of discourse connectives found in [Knott, 1996]; further, any connectives not appearing in these lists are considered discourse adverbials. As we would expect different syntax associated with different connectives we introduce conjunctive features such as the connective type and syntactic path.

**Lexico-Syntactic Features.** One of the prime difficulties in identifying the correct non-anaphoric argument has to do with *attribution*. In this situation the argument is the

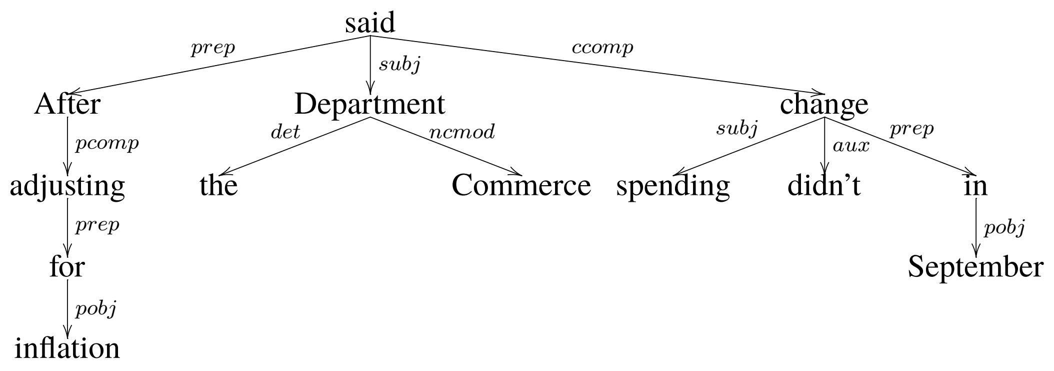


Figure 3.2: Dependency structure.

complement of a verb indicating attribution of the proposition denoted by the complement to an individual other than the writer. Figure 3.1 provides an example of this where the Arg1 of "After" is the complement of the verb "said" being attributed to "the Commerce Department". To model this situation we introduce features capturing whether the argument is a potentially attribution-denoting verb, whether it has a clausal complement, whether it is the clausal complement of another verb and whether the complementing verb is attributing.

A full listing of the features used for identifying arguments is shown in Table 3.1.

## 3.3 Experiments with Independent Argument Identification

For all of our experiments, we use sections 02-22 for training, sections 00-01 for development and sections 23-24 for testing. The development data was used to customize our features and to tune the Gaussian prior used to prevent over-fitting in the log-linear mod-

| | |
|---|---|
| Baseline Features | |
| A | Where in the sentence (beginning, middle, end) the connective resides |
| B | Whether the argument is in the same sentence as the connective (yes,no) |
| C | Connective phrase |
| D | Down-case connective phrase |
| E | Argument head word |
| F | Argument head prior or after connective |
| G | A & B |
| Constituent Features | |
| H | Path from argument to connective through the constituent tree |
| I | Length of path |
| J | Collapsed path without part-of-speech |
| K | Collapsed path removing repetitions of the same node type (e.g. VP-VP-VP $\rightarrow$ VP) |
| L | C & H |
| Dependency Features | |
| M | Dependency path from argument to connective |
| N | Path + head word of first link from connective |
| O | Collapsed path removing coordinating links |
| P | Collapsed path removing repetitions of links |
| Q | C & M |
| Connective Features | |
| R | coordinating, subordinating or adverbial connective |
| S | A & R |
| T | M & R |
| Lexico-Syntactic Features | |
| U | Argument is an attributing verb |
| V | Argument has a clausal complement |
| W | U & V |
| X | Argument is a clausal complement of a verb |
| Y | X & governing verb is an attributing verb |

Table 3.1: Feature types for discourse connective argument identification

els ( at $\sigma = 0.25$ for both the local and the re-ranking models). All results are reported on the testing data, sections 23-24.

We report results using both gold-standard parses and automatic parses using the Charniak-Johnson parser [Charniak & Johnson, 2005]. At training time, for simplicity, we used the gold-standard parses in the Penn Treebank. Ideally, the argument identification

| | Accuracy | | |
|---|---|---|---|
| Feature Set | ARG1 | ARG2 | Conn. |
| A-G | 32.7 | 60.7 | 21.6 |
| A-L | 60.6 | 85.5 | 53.6 |
| A-G;M-Q | 73.7 | 94.2 | 70.2 |
| A-Y | 75.0 | 94.2 | 71.7 |
| A-Y(auto) | 67.9 | 90.6 | 62.7 |

Table 3.2: Results for argument identification on the testing data (WSJ sections 23-24) gold standard parses (with various feature sets) and Charniak-Johnson parses (auto) for the full feature set A-Y.

system should be trained on automatic parses representative of the parsing performance on the evaluation data, by, for example, performing $n$-fold jackknifing (i.e., cross-fold application) over the training data. Thus, these results using the automatic parses should be viewed here as lower-bounds.

For evaluating ARG1 and ARG2 argument identification performance we report *accuracy* — i.e., the percentage of arguments correctly identified. An argument is correct if and only if it is the same head-word as derived from the argument extent as annotated in the PDTB (as described in Section 3.1.1). We also report *Connective Accuracy* which is the percentage of connectives for which *both* arguments were correctly identified.

Our results for the task of identifying arguments are shown in Table 3.2 for various feature combinations. It is interesting to compare the performance of the constituent parse features (A-L) vs. the dependency parse features (A-G;M-Q). The dependency parse features perform markedly better: 70.2 vs. 53.6 Connective Accuracy with gold-standard parses. Likely, this discrepancy can be explained in part by the better 'alignment' of the dependency parses to the target discourse argument representation. In particular, the dependency representation used here directly links semantic head words (which are the target discourse argument heads), making syntactic heads subordinate, obviating the need to follow the VP chain from the syntactic to the semantic head. Figure 3.2 provides an example of this where the *ccomp* dependency link directly links "said" with "change" which is the content-bearing, semantic head of the complement clause.

## 3.4 Experiments With Re-ranking

A drawback to the above approach is that the two arguments are identified independently. Ideally, one would like to consider both arguments and the connective simultaneously, taking into account global properties such as the pattern of the argument structure (e.g. Connective $\prec$ ARG2 $\prec$ ARG1 vs. ARG1 $\prec$ Connective $\prec$ ARG2) or properties of compatibility between the two arguments (e.g. agreement in tense). Considering all pairs of arguments outright, however, presents scalability issues as the number of such pairs can be very large (especially with anaphoric ARG1s). Indeed, a huge advantage of the lexicalized approach taken with the PDTB is that we *can* identify arguments independently using the connectives as anchors. Nevertheless, there is obvious potential gain from modeling pairs of arguments jointly.

One way to model these dependencies in a tractable fashion is to use *re-ranking* [Collins, 2000] which has proven successful in a variety of NLP tasks. The basic idea is to use a model with strong independence assumptions, $GEN_N(\pi)$, in this case based on the independent ARG1 and ARG2 models described above, to generate $N$ candidate argument

| | Accuracy | | |
|---|---|---|---|
| N | ARG1 | ARG2 | Conn. |
| 1 | 74.5 | 94.5 | 71.4 |
| 5 | 83.1 | 97.4 | 81.8 |
| 10 | 90.5 | 97.9 | 89.2 |
| 20 | 93.8 | 97.9 | 92.1 |
| 30 | 94.6 | 97.9 | 92.9 |

Table 3.3: $N$-best upper-bounds for different values of $N$ according to a product of independent argument ranker probabilities with the full feature set (A-Y)

pairs for a given connective, $\pi$. The re-ranking model is then used to re-rank these candidate pairs; the top-ranked pair is selected as the argument pair for the connective.

In our setting for a given connective, $\pi$, we define the *local probability* for a candidate argument pair, $\langle \alpha_i, \beta_j \rangle$ as:

$$P_{loc}(\alpha_i, \beta_j | \pi, x) = P_{\text{ARG1}}(\alpha_i | \pi, x) \cdot P_{\text{ARG2}}(\beta_j | \pi, x) \tag{3.2}$$

Thus, $GEN_N(\pi)$ generates the top $N$ argument pairs according to the $P_{loc}$. In practice, we also assert that $P_{loc}(\alpha_i, \beta_j | \pi, x) = 0$ when $\alpha_i = \beta_j$ since it is never the case that the true ARG1 is the same as the true ARG2 for the same connective.

Table 3.3 shows the oracle upper bounds on performance - the performance achieved by selecting the correct argument pair from $GEN_N(\pi)$ if it is in the list of argument pairs and otherwise selecting the first pair with one correct argument if such a pair exists. Note that performance on ARG2 plateaus at 97.9. This is due to 2.1 percent of the ARG2S not being reachable because they are not considered candidates (they are more than 10 “parse steps” away or an invalid part-of-speech).

### 3.4.1 Modeling Inter-Argument Dependencies

The model for re-ranking pairs of arguments is given by

$$P_r(\alpha_i, \beta_j|\pi, x) = \frac{\exp\left(\sum_k \lambda_k f_k(\alpha_i, \beta_j, \pi, x)\right)}{\sum_{\alpha_i, \beta_j \in GEN_N(\pi)} \exp\left(\sum_k \lambda_k f_k(\alpha_i, \beta_j, \pi, x)\right)} \tag{3.3}$$

Following previous work [Collins, 2000; Toutanova et al., 2005], we mix the local model into the final score along with the re-ranking model as:

$$P(\alpha_i, \beta_j|\pi, x) = P_{loc}(\alpha_i, \beta_j|\pi, x)^\gamma \cdot P_r(\alpha_i, \beta_j|\pi, x) \tag{3.4}$$

where $\gamma$ indicates the degree to which the local model influences the final score. Tuning $\gamma$ on the development data, we set $\gamma = 0.4$ for all our re-ranking experiments.

The re-ranking model is able to accommodate features over *both* candidate arguments. For example, we can test whether the two arguments are the same predicate or whether they are both reporting verbs. Another set of features consists of triples denoting the relative order of the arguments and the connective. For example, the feature *CONN*_ARG2_ARG1 indicates the connective and both arguments lie in the same sentence with the connective first, followed by ARG2 and then ARG1. The feature *Prev_CONN*_ARG2 indicates ARG1 is in the previous sentence and the connective precedes ARG2 within the sentence containing the connective. Other slight variations capture configurations where the ARG1 candidate lies further back in the discourse. Finally, we found some utility in comparing the syntactic arguments (e.g., subject, direct object) of the candidate argument pairs. For example, the arguments of the discourse adverbial *also* not only frequently involve the same predicate but also involve the same entities that appear as arguments to the predicate. Currently, we simply introduce features testing whether the argument strings are identical as a proxy for full co-reference.

Table 4 shows the results incorporating the re-ranking model for the different feature sets described earlier. The re-ranking models in each case are constructed from the features that would naturally be available to the re-ranker. For example, the re-ranking model for feature set A-Y uses a feature testing whether both candidate arguments are reporting verbs, whereas the re-ranking model for A-L doesn't.

| | Accuracy | | | | |
|---|---|---|---|---|---|
| Features | Arg1 | Arg2 | Conn | Indep. Conn | Err. |
| A-G | 44.1 | 59.6 | 30.6 | 21.6 | 11.5% |
| A-L | 64.7 | 85.6 | 58.1 | 53.6 | 9.6% |
| A-G;M-Q | 74.2 | 94.4 | 71.8 | 70.2 | 5.4% |
| A-Y | 76.4 | 95.4 | 74.2 | 71.7 | 8.8% |
| A-Y(auto) | 69.8 | 90.8 | 64.6 | 62.7 | 5.4% |

Table 3.4: Re-ranking results for argument identification on the testing data using gold-standard and Charniak-Johnson parses for the full feature set, A-Y (auto). The error reduction (Err.) is relative to the results in Table 2.

### 3.4.2 Discussion and Error Analysis

Not surprisingly, performance at identifying Arg2s is much higher than for Arg1s as the former are syntactically bound to the connective. Indeed, performance for identifying Arg2s may be at or very close to human levels of performance using gold-standard parses. Miltsakaki et al. [2004a] indicate 94.1% inter-annotator agreement for Arg2, 86.3% on Arg1 and 82.8% agreement per discourse connective with respect to the full

| Connective Type | Frequency | Indep. Acc | Joint Acc | Err. Reduction |
|---|---|---|---|---|
| Coord. | 662 | 75.5 | 78.3 | 11.4% |
| Subord. | 547 | 87.2 | 86.8 | -3.0% |
| Adv. | 386 | 42.2 | 49.0 | 11.8% |
| Total | 1595 | 71.7 | 74.2 | 8.8% |

Table 3.5: Frequency of each connective type and connective accuracy for the independent (Indep.) and re-ranking (Joint) approaches using gold-standard parses and features (A-Y).

argument extents for a set of 10 connectives. The disagreement rates, however, would likely be reduced considerably using our head-based representation since almost half of the disagreements reported were due to argument extent disagreements.

Many of the ARG2 errors we found had to do with attribution, such as:

(18) ..“We pretty much *have* a policy of not commenting on rumors, $\boxed{\text{and}}$ I **think(?)** that **falls** in that category.

where the system proposed “think” as the ARG2 and the annotated argument was “falls”.

The ARG1 errors were much more diverse with many involving arguments in previous sentences, such as the following case in which the system proposed *owned* as the argument yet the correct argument was *completed* found three sentences prior in the discourse.

(19) ..Quantum *completed* in August an acquisition of Petrolane... Petrolane is the second-largest... The largest, Suburban Propane, was already *owned(?)* by Quantum. $\boxed{\text{Still}}$, Quantum **has** a crisis to get past right now.

An examination of the errors by connective type is shown in Table 5. The re-ranking model provides considerable improvement for coordinating and adverbial connectives, but slightly *lowers* performance for subordinating connectives. One hypothesis for this is that the arguments subordinating connectives are more strongly determined by the syntactic relationship between the connective and each argument and that re-ranking argument pairs simply introduces unnecessary additional features that are superfluous for identifying arguments of subordinating connectives and appear as noise and/or cause model overfitting. For discourse adverbials, overall performance remains below 50% despite the improvement from re-ranking.

## 3.5 Related Work

Given the formulation of discourse relations as predicate-argument structures anchored on discourse connectives, our work here bears some resemblance to work in semantic role labeling that has focused on identifying semantic frames for verbs [Toutanova et al., 2005]. The task of identifying discourse relations is simpler in that there are only and exactly two arguments for each predicate; yet it is more difficult due to the fact that candidate arguments for certain connectives must be identified outside of a single sentence.

Within discourse parsing, our work bears some similarity to that of Soricut & Marcu [2003], which we described briefly in the previous chapter. However, they focus only on identifying (and labeling the type of) all intra-sentential discourse relations whereas we attempt to identify discourse relations spanning multiple sentences, provided they are lexicalized by a connective. While not directly comparable to our results, they report 73.0 F-measure at identifying intra-sentential discourse relations and segments using gold-standard parses. With gold-standard discourse segments provided, their system achieves human-levels of performance (96.2 F-measure), broadly comparable to our near-human levels of performance on identifying ARG2S with gold-standard parses. Sporleder & Lapata [2005] address intra-sentential discourse modeling with a chunking approach. They achieve 88.7 F-measure on identifying discourse segment boundaries and 76.3 F-measure when also labeling each segment as a nucleus or satellite.

## 3.6 Summary

We have presented a fully automated system capable of identifying the arguments of discourse connectives. Rather than identifying the full argument extents in the PDTB, we have proposed here an alternative problem formulation: that of identifying the heads of discourse arguments.

With such a representation our system achieves 74.2% accuracy using gold-standard parses and 64.6% accuracy using automatic parses on the task of correctly identifying both arguments of discourse connectives. We found that syntactic features based on a dependency parse representation provide more discriminative features over those based on a constituent tree representation. Additionally, we found a notable improvement by exploiting joint features over argument pairs in a re-ranking model in comparison to modeling the arguments independently.

# Chapter 4

# Identifying Arguments in PDTB 2.0

The results described in the previous chapter were carried out on Version 1.0 of the PDTB. Subsequent to the completion of those experiments, the final version of PDTB, Version 2.0, was released. The PDTB 2.0 [Prasad et al., 2008] includes the annotation of implicit discourse connectives for the entire WSJ portion of the Penn Treebank, rather than just for sections 08, 09 and 10 as in Version 1.0. It also includes rhetorical types (or *senses*) for all implicit relations as well as for all the explicit discourse connectives. Additionally, some of the annotations were modified and refined to improve consistency.

This chapter provides some revised results on the argument identification task using this new version of the data; additionally, important modifications have been made to the target representation and the underlying system and features used to identify the arguments. We also present results in this section on identifying the arguments of *implicit* discourse connectives, which are fully annotated across all WSJ sections in PDTB 2.0.

## 4.1 Revised Argument Identification of Explicit Connectives

This section describes revised results at identifying the arguments of explicit connectives using the PDTB 2.0. Besides using this newer, and final, version of the PDTB, the primary differences between the results discussed here and in Chapter 3 can be summarized as follows:

- Rather than using the semantic head of each discourse argument as the target representation, the syntactic head is used. [1] So, for example, in Figure 3.1 the syntactic head of the ARG1 *spending didn't change in September* would be *did* rather than the content-bearing verb *change* as in the representation used in Chapter 3.
- A new dependency parse representation is used based on the representations developed for the CoNLL 2007 Shared Task on Dependency Parsing [Johansson & Nugues, 2007; Nivre et al., 2007a]. This contrasts with the earlier dependency representation that followed the approach in [de Marneffe et al., 2006].
- Some additional feature types were used that aim to take into account additional discourse-level properties. These are discussed below in detail.
- We have taken a different, more standard, approach to re-ranking in which $n$-best lists of argument pairs are generated by 2-fold jackknifing over the training data.

[1] Briefly, the *syntactic* head is the lexical item directly attached to the highest VP node within a complex VP phrase, while the *semantic* head is typically the highest lexical item that is not a modal or auxiliary in the complex VP.

It is worth noting that some of the these changes were motivated in part by practical considerations in developing a more *robust* and simpler piece of software that could operate over real data. For example, the new dependency parsing representation employed makes it possible to leverage any of the parsers that have been built for the recent CoNLL dependency parsing tasks.

We discuss each of these changes in detail below.

### 4.1.1 Target Representation

In our prior work with discourse argument identification described in the previous chapter, we identified the argument with its *semantic* head. This representation was driven, in part, by the particular syntactic dependency representation we employed which was modeled after that in de Marneffe et al. [2006]. Additionally, the semantic head of an argument is generally a source for more useful features than the syntactic head (which may be an auxiliary verb).

The primary reasons for changing the target representation to target the syntactic head are: 1) the syntactic head is more straightforward to identify consistently, 2) it more readily facilitates extraction of the full argument extent in a dependency representation by simply following the transitive closure of the syntactic head, and 3) the syntactic head fits more naturally with features based on constituent-based syntax (since the modal auxiliary verb chain need not be traversed) and/or dependency representations that use the syntactic head for VPs.

| | | |
|---|---|---|
| ADJP | ← | NNS QP NN $ ADVP JJ VBN VBG ADJP JJR NP JJS DT FW RBR RBS SBAR RB |
| ADVP | → | RB RBR RBS FW ADVP TO CD JJR JJS IN NP JJS NN |
| CONJP | → | CC RB IN |
| FRAG | → | (NN \| NP) W* SBAR (PP \| IN) (ADJP \| JJ) ADVP PP |
| INTJ | ← | ** |
| LST | → | LS : |
| NAC | ← | NN NP NAC EX $ CD QP PRP VBG JJ JJS JJR ADJP FW |
| PP | → | IN TO VBG VBN RP FW |
| WHPP | → | IN TO VBG VBN RP FW |
| PRN | → | S* N* W* (PP \| IN) (ADJP \| JJ) (ADVP \| RB) |
| PRT | → | RP |
| QP | ← | ($ \| IN \| NNS \| NN \| JJ \| RB \| DT \| CD \| NCD \| QP \| JJR \| JJS) |
| RRC | → | VP NP ADVP ADJP PP |
| S | → | VP *-PRD S SBAR ADJP UCP NP FRAG SINV PP |
| SBAR | → | S SQ SINV SBAR FRAG IN DT |
| SBARQ | → | SW S SINV SBARW FRAG VP |
| SINV | → | VBZ VBD VBP VB MD *-PRD VP SQ FRAG S SINV SBAR SBARQ |
| SQ | → | VBZ VBD VBP VB MD *-PRD VP SQ FRAG S SINV SBAR SBARQ |
| UCP | → | ** |
| VP | → | VBD VBN MD VBZ VB VBG VBP VP *-PRD ADJP NN NNS NP S SINV SBAR SBARQ SQ |
| WHADJP | ← | CC WRB JJ ADJP |
| WHADVP | → | CC WRB |
| WHNP | ← | NN WDT WP* WHADJP WHPP WHNP |
| X | ← | ** |

Table 4.1: Rules for assigning heads to constituent phrases.

### 4.1.2 Revised Dependency Representation

The largest change to the argument identification system involved revising the syntactic dependency representation. For the most part, we followed the constituent to dependency conversion method described in Johansson & Nugues [2007] to arrive at our target dependency structure. Briefly, that method involves the following steps:

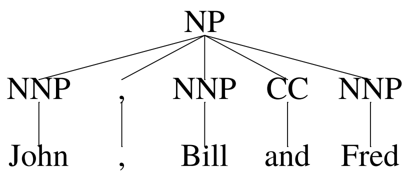
**Identify the Head Words** For each constituent the head-word is identified using the set of head-finding rules in Table 4.1. For a given non-terminal, $n$, the rule whose left-hand side matches $n$ is applied. The ordered children of $n$ are traversed in the direction specified by the arrow until a child matches the first node label expression on the right-hand-side. If such no child matches the node label expression, then the next node expression is tried and so forth until a match is found. The rules

**Algorithm 1** LabelLink($w, l$)

```
Require: A word, $w$, with incoming link, $l$, to label
Ensure: A label, $lab$, for link $l$
  $C \leftarrow$ highest phrase with $w$ as its head
  $P \leftarrow$ parent of $C$

  if $C$ is the root node return ROOT
  else if $P$ matches N-L where $member(L$,[ADV,DIR,EXT,LGS,LOC,MNR,PRD,SBJ,TMP]$)$ return $L$
  else if $C$ is an object return OBJ
  else if $C$ is PRN return PRN
  else if $C$ is RRC or SBAR and $P$ is NP return RCMOD
  else if $w$ is punctuation return P
  else if $C$ is PP, ADVP or SBAR and $P$ is VP or S* return ADV
  else if $C$ is PRT return PRT
  else if $P$ is a phrase containing coordination
        { if $C$ is CC return CONJ else return COORD }
  else if $C$ is VP and $P$ is VP, SQ, SINV, SBAR, S or SBARQ return VC
  else if $C$ is S-TPC-* return CCOMP
  else if $C$ is S* and $P$ is VP return CCOMP
  else if $P$ is S* or VP
        { if $C$ is CC return CONJ else if $C$ is DT return NMOD else if $C$ is WHNP-* return SBJ else
  return VMOD }
  else if $P$ is UCP
        { if $C$ is CC return CONJ else return COORD }
  else if $P$ NP or QP return NMOD
  else if $P$ is ADJP, ADVP, WHADJP or WHADVP return AMOD
  else if $P$ is PP or WHPP return PMOD
  else return DEP
```

are then recursively applied to the selected child. For example, the rule `CONJ` $\rightarrow$ `CC RB IN` in Table 4.1 indicates that the first CC from the left should be selected as the child; if no such CC is found, then the first RB from the left is selected. If none of the elements on the right-hand side of a rule match, the first non-punctuation element is selected (in the direction indicated by the arrow). A small set of special cases surround handling of coordination within NP, ADJP and ADVP phrases to identify the left-most conjunct as the head. For example, in the NP:



The head would be identified as the NNP John.

**Assign Governor Links** This is done for each word, $w_i$, in a sentence by first identifying the highest non-terminal node, $n$ in the tree that has $w_i$ as its head. A link is then established between $w_i$ and the head of the constituent that immediately dominates $n$, $p_i$ where the link is from $p_i$ to $w_i$.

**Label Dependencies** Each link in the dependency graph is then assigned a label according to Algorithm 1.

This process differs from the approach in Johansson & Nugues [2007] in a few ways. First, we added a link type RCMOD, which denotes relative clauses - these link types are annotated in the Penn Treebank (via RRC nodes) and appeared to us a useful syntactic distinction for reasons not confined to discourse parsing. We also added a clasual complement link CCOMP. Secondly, we did not exploit the secondary edges in the Penn Treebank or the explicative, cleft or gapping constructions. These constructions are what result in non-projective dependencies. Thirdly, we did not add additional structure to complex, non-coordinated noun phrases as was done in Johansson & Nugues [2007], but left the dependency structure of the NP 'flat', if it was so annotated in the PTB. Finally, the head percolation rules are slightly modified here compared to their approach. For example, we changed the rule direction for clausal non-terminals (S, SBAR, etc.) so that

| | |
|---|---|
| ADV | General adverbial |
| AMOD | Adjectival modifier |
| CCOMP | Clausal complement |
| CONJ | Conjunction (dependent of first conjunct) |
| COORD | Coordination |
| DEP | Unspecified dependency |
| DIR | Adverbial denoting direction |
| EXT | Adverbial denoting extent |
| LGS | Logical subject of passive voiced verbs |
| LOC | Locative modifier (adverbial or nominal) |
| MNR | Adverbial of manner |
| NMOD | Modifier of nominal |
| OBJ | Object |
| P | Punctuation |
| PMOD | Modifier of preposition |
| PRD | Predicative complement |
| PRN | Parenthetical |
| PRT | Particle |
| RCMOD | Relative clause modifier |
| ROOT | Root |
| SBJ | Subject |
| TMP | Temporal modifier |
| VC | Verb chain |
| VMOD | Verb modifier |

Table 4.2: List of dependency labels

clausal constructions coordinate in the same manner as verb phrases — i.e., with the left conjunct always being the head.

The full set of dependency labels is shown in Table 4.2. An example parse is provided in 4.1. It is worth emphasizing that the representation and set of dependency link types used here is somewhat arbitrary and that many different representations are used in the literature and within the recent dependency parsing shared tasks, in particular. This particular representation was chosen because it is projective[2] and the inventory of link

[2]The primary benefit of a projective dependency parse in the context of discourse parsing is that it is possible to trivially pull out the extent of a discourse argument given just its lexical head.

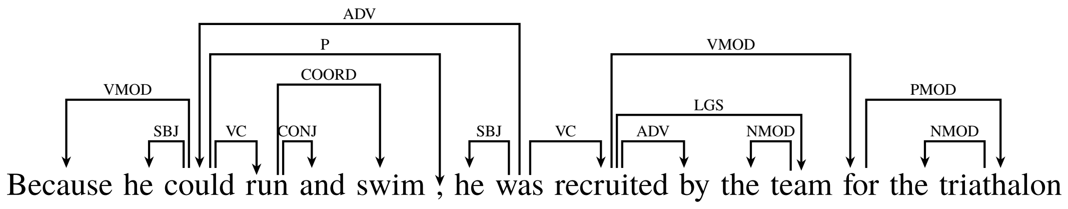

ADV
P
COORD
VMOD
SBJ
VC
CONJ
SBJ
VC
VMOD
LGS
ADV
NMOD
PMOD
NMOD
Because he could run and swim , he was recruited by the team for the triathalon

Figure 4.1: Dependency parse example.

types appears to be appropriate for the task of discourse parsing. It is left as future work to determine which syntactic representations and inventories of grammatical relation types are "optimal" with regard to discourse parsing.

## 4.1.3 Feature Additions and Modifications

Most of the features for the revised system are the same as described in Chapter 3 (cf. Table 3.1). We describe here a few additional features we found useful. Note that these features are described here in the context of the independent argument identification rankers; similar features were used within the re-ranking framework that considers ARG1-ARG2 pairs.

Before discussing the new features, recall that our features are broken down into various feature classes. We slightly reorganize the feature classes from Table 3.1 here as follows: The Baseline features (A-G) and the Connective features (R-T) constitute the BASE class of features, while the Constituent features, Dependency features and Lexico-Syntactic features map to classes CONSTPARSE, DEPPARSE, and LEXSYN, respectively.

**Paragraph Boundary Features** Paragraph boundaries provide important clues for the identification of distant ARG1S, in particular. We used paragraph information in two ways.

First, we included features such as a) whether the argument was in the same paragraph as the connective; b) within the same paragraph, but different sentence; c) if the argument is in the first sentence of a paragraph; d) distance of the argument from the connective in terms of paragraphs and e) various conjunctions of (a)-(d) based on the connective phrase and connective *type*. Secondly, we used paragraphs to define a different set of dependency relations at the discourse level. The head-word for each sentence is linked to the head-word of the previous sentence, just as before via a Sent link, except for the sentences which are the first sentence within a paragraph. For those sentences, the head-word is linked by a Para dependency link to the head-word of the *first* sentence in the preceding paragraph, if one exists. This representation is based on observation that distant or multi-sentence Arg1 arguments have a tendency to have their heads in the first sentence of a preceding paragraph. These features form a new feature class: Para.

**Intervening Features** These features include the distance in tokens between the connective and candidate argument (or argument pairs, for re-ranking) as well as all intervening parts-of-speech. Conjunctions of distance and intervening parts-of-speech are also used. These features are modeled, in part, on features used for syntactic dependency parsing. These features form a new feature class: Intervening.

**Additional Context Features** Following the recent work in Elwell & Baldridge [2008], we added features such as the previous and subsequent word to the candidate argument head and connective as well as features regarding whether a) the candidate argument is within quotes, b) the connective is within quotes and c) whether they appear within the same quotation. These features are added to the Base class of features.

### 4.1.4 Jackknifing To Create Representative $N$-best lists

Key to achieving good performance with re-ranking methods is providing them with $N$-best lists at *training* time that are representative of the $N$-best lists that will appear at *decoding* time. In the the re-ranking approach taken in Chapter 3, the $N$-best lists for the entire set of training were generated by the *local* Arg1 and Arg2 models *which were trained on the same data*. The result of this is that the $N$-best lists generated are of much better quality than would be generated on unseen data. To compensate for this, the re-ranking model is trained on these artificially high-quality $N$-best lists, but the final probability for an argument pair is defined as a mixture over the local and re-ranking probabilities (cf. Equation 3.4).

A more standard approach is to generate $N$-best lists using local models that haven't been trained on the same data to which they are applied. This is done by jackknifing. The data are split into a set of $M$ even-sized folds. $N$-best lists are generated for each $m_i \in M$ by training local Arg1 and Arg2 models on all $m_j$, where $j \neq i$, and applying those models to $m_i$. For simplicity, we used just two folds across the training data. Based on evaluation on the development data, we found this setup to provide somewhat better performance and did not require specialized tuning of the $\gamma$ parameter from Equation 3.4 to optimally weight the local and re-ranking probabilities. Thus, the probability for an argument pair is described as:

$$P(\alpha_i, \beta_j|\pi, x) = \frac{\exp\left(\sum_k \lambda_k f_k(\alpha_i, \beta_j, \pi, x)\right)}{\sum_{\alpha_i, \beta_j \in GEN'_N(\pi)} \exp\left(\sum_k \lambda_k f_k(\alpha_i, \beta_j, \pi, x)\right)} \tag{4.1}$$

This differs from Equation 3.3 in that the function $GEN'_N$ generates $N$ best lists according to Equation 3.2 but where we ensure that the conditional probabilities for the ARG1 and ARG2 candidates of a connective are produced by models estimated from *different* connectives, using the 2-fold jackknifing method.

## 4.2 Revised Argument Identification for Explicit Connectives

We describe here essentially the same set of experiments as in Sections 3.3 and 3.4, but with the PDTB 2.0 data, the syntactic heads as representatives for argument spans, the modified dependency representation and the augmented feature set. One other minor experimental difference is that we have chosen to move section 22 from the training data into the development data set, which now consists of sections 00, 01 and 22.

### 4.2.1 Heuristic Baseline

We found it insightful to look at the performance of a simple heuristic-based approach to identifying the arguments of discourse connectives. For subordinating connectives, adverbials and all sentence initial connectives, the ARG2 is identified as the governor of the head-word of the connective phrase in the dependency representation; the ARG1 is then simply the syntactic governor of the ARG2 or the root word of the previous sentence if the ARG2 is the root word in the sentence containing the connective. For sentence-medial coordinating connectives, the syntactic governor of the connective is identified as ARG1 and the ARG2 is identified as the dependent of a COORD syntactic link from the ARG1.

### 4.2.2 Results

The results are summarized in Table 4.3 for independent argument identification as well as joint identification using re-ranking. The heuristic baseline performs respectably, and provides insight into the degree to which the discourse structures in the PDTB align with syntax [Dinesh et al., 2005]. The overall results represent a slight improvement over the previous argument identification results [Wellner & Pustejovsky, 2007] and are due to the newly introduced properties of the system. This last point is clear since the new system performs at essentially the same level of accuracy on the PDTB 1.0 data — also at 75.6% for the joint model and slightly lower at 72.8% on independent argument identification.

The re-ranking model achieves a 10% error reduction over separate identification of the ARG1 and ARG2 arguments.

Results per connective type for the Re-rank ArgID system are shown in Table 4.4. Some connectives, sentence-initial coordinating connectives, specifically, behave in some aspects as a discourse adverbial. A full list of results broken down by individual connectives can be found in Appendix A.

| | Accuracy | | | |
|---|---|---|---|---|
| System | Arg1 | Arg2 | Conn | Old Conn |
| Heuristics | 70.2 | 89.2 | 65.8 | 65.9 |
| Indep. ArgID | 76.6 | 94.4 | 72.9 | 71.7 |
| Re-rank ArgID | 78.2 | 94.7 | 75.6 | 74.2 |

Table 4.3: Revised argument identification results for explicit connectives. For comparison, earlier results reported in [Wellner & Pustejovsky, 2007] and in Section 3.4 (Old Conn) are provided.

| | Accuracy | | |
|---|---|---|---|
| Connective Type | Arg1 | Arg2 | Conn |
| Subordinating | 89.2 | 98.4 | 88.5 |
| Coordinating | 82.7 | 93.0 | 78.1 |
| Adverbial | 54.8 | 92.2 | 53.0 |

Table 4.4: Connective type accuracies with the Re-ranking model.

| | Accuracy | | |
|---|---|---|---|
| Feature Set | Arg1 | Arg2 | Conn |
| Full | 78.2 | 94.7 | 75.6 |
| NoConstParse | 77.0 | 93.5 | 74.2 |
| NoDepParse | 76.0 | 92.8 | 72.5 |
| NoLexSyn | 78.3 | 94.4 | 75.8 |
| NoIntevening | 77.8 | 93.9 | 75.0 |
| NoPara | 77.7 | 95.0 | 75.3 |
| NoBase | 77.7 | 94.1 | 75.0 |
| NoSyntacticFeatures | 69.6 | 89.1 | 64.3 |

Table 4.5: Argument identification results with various feature subsets. The set of features for NoSyntacticFeatures consists of the intersection of NoConstParse, NoDepParse and NoLexSyn.

Finally, we provide some additional insight into the utility of the various feature sets through a set of ablation experiments in which each feature set is separately removed from the union of all the features. The results of these experiments are summarized in Table 4.5. As made clear by the table, the removal of dependency parse-based features (NoDepParse) hurts performance the most of any single feature type. At the other end of the spectrum, removal of lexical compatibility (NoLexSyn) features actually slightly improves overall performance. The bottom row in the table is the result of a system that makes no use of features that require a full parse to compute — i.e., dependency and constituent path features as well as lexico-syntactic features that look at the verbal predicate-argument structure.

## 4.3 Argument Identification for Implicit Connectives

As discussed briefly in Section 2.3, Version 2.0 of the PDTB contains annotations for *implicit* connectives. Frequently, discourse relations are not realized with explicit discourse connectives, but rather must be inferred by the reader. In general, the arguments of such relations may reside anywhere in the discourse [Wolf & Gibson, 2005], possibly at opposite ends of a document. A class of such relations, however, parallels the relations exposed by explicit connectives - namely relations between arguments in adjacent sentences that could be realized by inserting a discourse connective phrase at the beginning of the second sentence, as in:

(20) *In July, the Environmental Protection Agency imposed a gradual ban on virtually all uses of asbestos.* Implicit= As a Result **By 1997, almost all remaining uses of cancer-causing asbestos will be outlawed.**

In the PDTB, the Arg1 must reside in a prior sentence to the Arg2, though in general it may consist of multiple sentences and/or not the immediately preceding sentence. For practical reasons, however, both arguments must reside within the same paragraph. Note also that the arguments need not be entire sentences, but may be clausal complements, subordinating clauses or adjuncts just as with explicit discourse connectives. Below are a couple of examples illustrating these types of cases.

(21) According to the Audit Bureau of Circulations, *Time, the largest newsweekly, had average circulation of 4,393,237, a decrease of 7.3%. Newsweek's circulation for the first six months of 1989 was 3,288,453, flat from the same period last year.* Implicit = And **U.S. News' circulation in the same time was 2,303,328, down 2.6%.**

(22) Judge Curry ordered the refunds to begin Feb 1. and said *that he wouldn't entertain any appeals or other attempts to block his order by Commonwealth Edison.* Implicit= In other words **"The refund pool... may not be held hostage through another round of appeals,** " Judge Curry said.

Besides implicit connectives, in some cases it was observed by the PDTB annotators that there was no clear discourse relation between adjacent sentence units. Such instances are annotated with the label NoRel (with the two arguments in these cases always consisting of full sentences adjacent to each other). In some cases, annotators found alternative lexicalizations of discourse connectives, such as:

(23) *In many other instances, there is almost no difference between the real test and Learning Materials.* AltLex = What's more, **the test and Learning Materials are both produced by the same company ...**

Finally, in a large number of sentence pairs there was a coherence relation between pairs of adjacent sentences where the arguments of such a relation were not abstract objects, but entities. These were assigned the label EntRel as in the following example:

(24) *John R. Stevens, 49 years old, was named senior executive vice president and chief operating officer, both new positions.* EntRel **He will continue to report to Donald Pardus, president and chief executive officer.**

Identifying the arguments of implicit connectives can be carried out in the same manner as was done for explicit connectives. The only difference is that there isn't a particular phrase associated with the implicit connective. One approach would be to simply add the implicit connective to the discourse, just prior to the first word of each sentence and connect it to the sentence's root word using an existing or, perhaps, special dependency link type.

An alternative is to designate a particular word within a sentence as a proxy for the implicit connective. We have taken this approach and have designated the final punctuation within a sentence as the proxy for the implicit connective. This is done purely in order to coerce the task of identifying the arguments of implicit connectives into the same problem structure as for identifying arguments of explicit connectives. The final punctuation marker also has the advantage of always linking to the root word of the sentence. Syntactically it is similar to sentence initial connectives such as *But* or *And* which are linked via a Conj dependency link to the root word of the sentence, whereas final punctuation is linked to the root via a P link.

Results are shown in Table 4.6 using the same set of final features as was used for argument identification of explicit connectives. Interestingly, the overall connective accuracy is nearly identical to that for identifying explicit connectives. However, the Arg2 results are considerably worse. Another observation from the table is that the Arg1 and Arg2 independent argument accuracies are quite comparable to the re-ranker, but the re-ranking model has a a considerably higher Conn accuracy. This demonstrates clearly the advantage the re-ranking approach has for ensuring consistency between argument pairs, while not necessarily improving Arg1 or Arg2 individually across the test set.

While the results here for implicit connectives are roughly equal to the results for explicit connectives, the results in the case of implicits may in fact be better since there is more inter-annotator disagreement with regard to argument spans for implicit connectives. Prasad et al. [2008] indicate that exact match agreement was 90.2% for the arguments (both Arg1 and Arg2) of explicit connectives but only 85.1% for implicit connectives. While identification of the lexical head of the arguments would certainly have higher

| | Accuracy | | |
|---|---|---|---|
| System | ARG1 | ARG2 | Conn |
| Indep. ArgID | 82.1 | 87.5 | 72.9 |
| Joint ArgID | 82.7 | 87.1 | 75.1 |

Table 4.6: Results for identifying the arguments of *Implicit*, *AltLex*, *NoRel*, and *EntRel* connectives.

agreement, it seems likely that identifying the argument heads of implicit connectives is at least as difficult as identifying the argument heads for explicit connectives.

It's worth noting that these results are not too different from the results obtained at identifying the arguments of explicit *coordinating* connectives shown in Table 4.4. This correspondence seems intuitive since the implicit relations are, essentially, coordinating-type relations. Implicit relations introduce some added difficulty in that their arguments lie in difference sentences, which may explain the results for implicit connectives being slightly lower than for explicit coordinating connectives, which have a fair number of sentence-medial coordinating connectives where both arguments reside in the same sentence.

### 4.3.1 Identifying Both Explicit and Implicit Connectives with a Single Model

Explicit and implicit connectives are qualitatively different; though, as mentioned, implicit connectives might be viewed as "hidden" sentence initial coordinating connectives. A natural question that arises is whether a *single* model that identifies arguments for both connective types performs better than separate models for implicit and explicit con-

| | Accuracy | | |
|---|---|---|---|
| System | Arg1 | Arg2 | Conn |
| Indep. ArgID | 79.0 | 90.9 | 72.7 |
| Joint ArgID | 80.0 | 90.2 | 74.8 |

Table 4.7: Results for identifying the arguments both explicit and implicit connectives (including *AltLex*, *NoRel* and *EntRel*) relations.

nectives. A single model may be able to capture some generalizations across the two connective types. For example, some of the features look at attribution or consider the compatibility of the predicate-argument structure for the two discourse arguments; we might expect such features to generalize well across the arguments of both implicit and explicit connectives. On the other hand, separate models have the potential to provide more discriminative power for cases where the statistics are different across the two connective types.

Using the same set of features as for explicit and implicit connectives separately, the results using a single model for both connective types are shown in Table 4.7. The reranking model identifies both arguments correctly for 74.8% of the connectives, which is slightly below the combined accuracies for the two separate models that identify the arguments correctly for 75.3% of the connectives.

# Chapter 5

# Discourse Parsing

In this chapter we examine the full discourse parsing task which in the context of this dissertation involves identifying discourse connectives as well as their two arguments according to the Penn Discourse Treebank. First, we examine the task of identifying discourse connectives and their arguments separately and then propose a joint model for identifying connectives and their arguments. The joint model is a natural extension of the re-ranking model used for identifying argument pairs described in the previous chapter.

## 5.1 Discourse Connectives

In this section we examine the problem of determining whether a potential discourse connective word or phrase is, in fact, acting as a discourse connective. Many connective words/phrases can act as discourse connectives or serve in a different capacity depending on the context. The simplest example of this involves the connective *and* which can serve to coordinate two clauses (and thus abstract objects):

(25) John called Bill with news **and** he did not take it well.

or two simple noun phrases:

(26) The man **and** the tiger faced each other in the ring.

The task of identifying connectives is further complicated due to some particulars of the annotation scheme used for the PDTB. The first issue has to due with the requirement that arguments for connectives be *abstract objects*. This is a semantic distinction, not a syntactic one. An abstract object, according to the PDTB guidelines, is generally any clause or VP within VP coordination, though it may also be a nominalization or pronominal reference to an event. In only two cases are nominalizations allowed to serve as arguments to discourse connectives. The first involves a nominalization with an "extensional reading". Below is an example from the PDTB annotation manual [Prasad et al., 2006].

(27) .. and many are hoping *for major new liberalizations* $\boxed{\text{if}}$ **he is returned firmly to power**.

The above ARG1 can be read as "that there will be liberalizations". The second case is exemplified with:

(28) .. the court permitted *resurrection of such laws*, $\boxed{\text{if}}$ **they meet certain procedural requirements.**

In this case, the nominalization is clearly derivable from an equivalent verb phrase - e.g. “such laws to be resurrected”.

The second issue complicating identification of discourse connectives is that coordinating conjunctions within VP coordinations are not annotated. This makes the task somewhat more difficult than it might be otherwise since VP coordination surface constructions are similar to clausal coordination.

The important point here is that in order to determine whether a phrase is acting as a discourse connective, it may require examining the potential arguments of that phrase. This indicates that the problem of identifying discourse connectives and their arguments are very much co-dependent. An analogous situation can be found in co-reference where the problems of identifying the referent of an anaphor and determining the anaphoricity of a potential anaphor are co-dependent. [Denis & Baldridge, 2007].

## 5.2 Identifying Discourse Connectives

The first stage in identifying discourse connectives is to select a set of candidate discourse expressions. Using the training data sections we set aside (WSJ Sections 02-21), we compiled a lexicon of potential discourse connective phrases. As shown in Knott [1996], the set of valid discourse connective expressions is unbounded when considering adverbial modifications of the connectives. Accordingly, we remove such adverbial modifiers from the connective lexicon. For example, the connective phrase *only because* is removed as it is compositional. Multi-word connectives such as *in addition* or *on the contrary*, however, are non-compositional and remain as distinct entries in the lexicon. In a few cases, one phrase may be a substring of another. For example, *as* is a valid connective on its own, but also forms part of other connective phrases such as *as a result* and *as though*.

Given the lexicon of connectives, candidate connectives are identified by simply scanning an input text for all phrases up to length four (the longest connective phrase in the lexicon) and creating candidate phrases from the longest match against the lexicon.

The final step performed when identifying candidate discourse phrases is to designate the *head word* of the phrase. The head-word is defined as the left-most *preposition* if one is present. Otherwise, the head word is the right-most word in the phrase.

We describe two approaches to identifying connectives based on their syntactic context, one based on syntactic constituency structure and another on dependency structure.

### 5.2.1 Constituency-based Features for Connective Identification

We found a relatively simple set of features to be quite effective at identifying whether a phrase was, in fact, acting in a discourse capacity. These features make use of the constituent parse tree as well as properties of the candidate connective itself.

The features include:

**Connective Features** The entire connective phrase as well as the connective type: coordinating, subordinating or adverbial.

**Path Features** For a connective head word, $n_0$, let the sequence of non-terminals $path = n_1 \prec ... \prec n_m$ denote the non-terminals that dominate $n_0$, where $n_0 \prec n_1$ . The path features introduced consist of $n_1$, $n_2$, $n_3$ as well as the bi-grams $n_1n_2$ and $n_2n_3$ and the trigram $n_1n_2n_3$. We also included the entire *collapsed path* from $n_0$ to $n_m$ - i.e. the path where sequences of the same non-terminal label are collapsed to a single non-terminal.

**Syntactic Context Features** These include:

- **SimpleNP** TRUE if the first NP appears in $path$ before the first VP and an SBAR doesn't appear in $path$.
- **ComplexNP** TRUE if the first NP appears in $path$ before the first VP, and an SBAR appears before the first NP.
- **VPCoord** TRUE if two or more of the sister nodes to $n_1$ are VPs.
- **SOnly** TRUE if $path$ contains no VPs or NPs.
- **VPBeforeNP** TRUE if the first VP appears before the first NP in $path$
- **SBAROnly** TRUE if $path$ contains no VPs or NPs, but contains an $SBAR$

**Conjunctive Features** A final set of features involved conjunctive predicates of the connective phrase and connective type together with various path features. Specifically, we included the features of the cross product of: (*connective type*, *connective phrase*) $\otimes$ ($n_2$, $n_1n_2$, $n_2n_3$).

### 5.2.2 Dependency-based Features

A corresponding set of dependency-based features was used to identify discourse connectives. These include:

**Connective Features** As above for the constituent-based features.

**Contextual Features** We used the previous word; the previous part-of-speech; the connective phrase and previous word; and the connective phrase and previous part-of-speech. Similar bi-grams were used for the subsequent word and part-of-speech.

**Syntactic Features** These features involve properties of the parts-of-speech for various *siblings* of the candidate connective within the dependency graph. These include: 1) part-of-speech triple including the connective part-of-speech, its parent's part-of-speech and the sibling's part-of-speech, 2) the part-of-speech triple and the dependency link type from the parent to the sibling, 3) the connective part-of-speech, parent part-of-speech and the link type to the sibling (without part-of-speech) 4) same as (3) but coarse parts-of-speech used (e.g. $NN* \Rightarrow N$, $V* \Rightarrow V$).

**Clause Detection Features** Whether the parent of the connective has a syntactic subject and/or whether a sibling of the connective has a syntactic subject.

The task is more naturally modeled by a constituent-based syntactic analysis since clauses are identified explicitly. This is important since coordinating conjunctions within VP-coordination are not (supposed to be) annotated. Identifying whether a potential

| **Evaluation Data** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Constituent | | | Dependency | | | Both | | |
| | Rec | Prec | F1 | Rec | Prec | F1 | Rec | Prec | F1 |
| Coord | 75.20 | 96.83 | 84.65 | 73.16 | 97.80 | 83.70 | 74.18 | 97.57 | 84.28 |
| Subord | 90.84 | 97.44 | 94.02 | 97.07 | 98.14 | 98.06 | 96.88 | 97.96 | 97.42 |
| Adv | 96.25 | 98.71 | 97.46 | 97.32 | 99.09 | 98.20 | 96.96 | 99.09 | 98.01 |
| Overall | 87.94 | 97.77 | 92.59 | 89.83 | 98.41 | 93.92 | 89.96 | 98.29 | 93.94 |

| **Development Data** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Constituent | | | Dependency | | | Both | | |
| | Rec | Prec | F1 | Rec | Prec | F1 | Rec | Prec | F1 |
| Coord | 94.21 | 95.48 | 94.84 | 91.81 | 96.63 | 94.16 | 92.17 | 96.64 | 94.35 |
| Subord | 92.39 | 97.77 | 92.00 | 96.14 | 98.09 | 97.11 | 95.90 | 97.85 | 96.86 |
| Adv | 95.73 | 98.44 | 97.07 | 96.69 | 98.87 | 97.77 | 96.55 | 99.01 | 97.76 |
| Overall | 93.97 | 97.39 | 95.65 | 95.19 | 97.98 | 96.56 | 95.14 | 97.93 | 96.51 |

Table 5.1: Overall and per connective type accuracies for the task of identifying discourse connectives using constituent-based and dependency-based features.

argument for a coordinating conjunction is an actual clause is less straightforward with a dependency analysis.

Using the above features, we trained a binary Maximum Entropy (logistic regression) classifier to determine for each candidate discourse connective phrase whether or not the phrase is acting as a discourse predicate as determined by the PDTB annotation scheme. Maximum Entropy classifiers typically employ a Gaussian prior penalty over the parameters to avoid overfitting. We tuned the value for the Gaussian prior to 1.0 based on performance observations on the development data set.

### 5.2.3 Results and Analysis

Table 5.1 provides results at identifying discourse connectives using the constituent-based features, the dependency-based features and their union. The constituent-based features have a natural tendency to do a better job of identifying coordinating connectives. This is due to the fact that the distinction between clausal coordination and VP-coordination is articulated clearly in the phrase-structure. Determining whether the conjuncts of a coordinating connective are clauses is not as straightforward using a dependency representation, hence the heuristic features above that attempt to identify whether the potential VP arguments of the connective have an external argument (via the SBJ link).

Another observation from these results is that recall for coordinating connectives is much lower on the evaluation data than on the development data. This appears to be due to coordinating connectives within VP-coordination appearing frequently as discourse predicates in Section 24, whereas according to the PDTB 2.0 guidelines such connectives should not be annotated.

It may be surprising that the accuracy numbers are not higher. There appear to be a few sources of difficulty: 1) inconsistent annotation of coordinating connectives within VP coordination - sometimes these are annotated and other times not; 2) some difficulties in identifying multiple adjacent discourse connectives such as "And so" or "But then"; and 3) difficulties with connectives whose arguments are nominalizations. It remains unclear how well humans perform/agree on this task. The results here for identifying discourse connectives are higher than those found in Litman [1996] in which the error rate remained above 10%. However, the results are not directly comparable due to different guidelines about what constitutes a discourse connective as well as different data sets.

## 5.3 Joint Connective and Argument Identification

Having presented statistical models for 1) identifying discourse connectives and 2) identifying the arguments of discourse connectives, in this section we visit the full discourse parsing problem using a joint statistical model to identify discourse connectives and their arguments in a single step.

There are two primary motivations for taking this approach:

1. Determining whether a potential connective phrase is, in fact, acting in a discourse capacity in some cases requires identifying what its arguments *would be*. More specifically, only abstract objects may serve as arguments; further, not all nominalizations are allowed to serve as arguments even if they are, in fact, abstract objects.

2. In general, one would prefer a system to fail to recognize a relation (i.e. a connective and its arguments) altogether rather than to posit a spurious relation. The former amounts to a recall error whereas the latter introduces both a recall and precision error. Accordingly, if there is great uncertainty about what the arguments of a connective are, it may be preferable to "punt" on the connective altogether even when there is positive evidence for the connective acting in a discourse capacity.

### 5.3.1 Model Overview

The joint model is a natural extension of the argument identification re-ranking model discussed in Chapters 3 and 4 — cf. Equation 4.1 It makes two primary additions. First, rather than a ranking instance for each true discourse connective (according to the gold-standard), there is now a ranking instance for each *candidate* discourse phrase. The second change is that for each ranking instance, in addition to the top $N$ argument pairs as determined by $GEN'(\pi)$ via the independent local models, there is one additional outcome $Null$ indicating that this candidate connective phrase is not acting as a discourse connective.

The re-ranking model is formalized below in Equation 5.1.

$$P(\langle \alpha_i, \beta_j \rangle | \pi, x) = \frac{\exp\left(\sum_k \lambda_k f_k(\langle \alpha_i, \beta_j \rangle, \pi, x)\right)}{\sum_{\alpha_i, \beta_j \in GEN'(\pi) \cup Null} \exp\left(\sum_k \lambda_k f_k(\langle \alpha_i, \beta_j \rangle, \pi, x)\right)} \tag{5.1}$$

where $Null$ indicates the outcome associated with the connective *not* being a discourse connective. This additional outcome event includes as features the same exact features as derived for this instance using the connective identification classifier (cf. Section 5.1).

The ability for the model to accommodate different features for different outcomes is a particularly strong advantage of the log-linear *ranking* model. The features are *non-factored* in that the choice of features is not only dependent on the observed data, but also on the predicted or dependent variable.

As an example of this, consider the task of part-of-speech tagging — i.e, assigning a part-of-speech to each word in a sentence. This task can be formulated as a classification problem where there are $N$ classes, one for each part-of-speech and each word is a classification instance. Typically, with a multinomial (i.e., $N$-way) classifier, the same features are used over *all* the different outcomes. For example, if one of the features is to consider the last 2 characters of the word, this feature is used for all $N$ outcomes. This treatment of features is very much common-place within supervised learning, including decision trees, neural networks, Bayesian modeling, Maximum Entropy modeling, etc. Within a ranking model (or any model incorporating non-factored feature functions), however, it would be possible to only consider the aforementioned feature for some subset of the class outcomes. For example, for all verb parts-of-speech classes the last two characters could be used as features, while for noun classes, the last *four* characters could be used. As different features may have greater discriminative power for different sub-sets of the class labels, such a model provides more flexibility and potential for better classification accuracy.

An additional property of ranking, non-factored model formulations is that they appear to be more robust to data skew. For example identifying the arguments of a single connective could be carried out using a binary classifier that considers each candidate argument and labels each candidate as Yes or No. The candidate that receives the highest score for Yes could be selected as the argument using probability for probabilistic models such as Maximum Entropy or distance from the margin for margin-based classifiers like Support Vector Machines to assign a score. When training such a model, however, for each connective there may be many candidate arguments, but only one correct argument resulting in many instances labeled No and considerably fewer labeled Yes. A large skew in the class distribution is known to cause problems for machine learning methods.

### 5.3.2 Results

We report results here for the joint connective and argument identification system described above. It is interesting to compare this system with a two-stage system that 1) identifies discourse connectives and 2) identifies the two arguments for the identified connectives.

The primary metric used to evaluate these systems is *Predicate-ID* which is the recall, precision and balanced F-measure for the task of identifying connectives *and* their two arguments - i.e., the entire predicate. A recall error is introduced if a discourse connective is not identified; a precision error is introduced for each spurious discourse connective; finally, if a connective is properly identified, but one or both of its arguments are incorrectly identified, this results in both a precision and a recall error.

Two other metrics are *Connective-ID* which is the precision, recall and F-measure for identifying only the connectives (as in Table 5.1) and the *Arg ID Precision* which shows the accuracies for identifying arguments for just the correctly identified connectives.

Predicate-ID and Connective-ID results on the task of jointly identifying connectives and their arguments are shown in Table 5.2. Additionally, the table shows the Argument-ID Precision which is the argument identification accuracies for those connectives correctly identified. As the table indicates, the Full Joint model improves the overall

**Evaluation Data**

| | Connective-ID | | | Predicate-ID | | | ArgID Precision | | |
|---|---|---|---|---|---|---|---|---|---|
| | Rec | Prec | F1 | Rec | Prec | F1 | ARG1 | ARG2 | Conn |
| Cascade+Indep | 89.83 | 98.41 | 93.92 | 62.96 | 69.99 | 66.29 | 75.2 | 94.1 | 71.6 |
| Cascade+Joint | 89.83 | 98.41 | 93.92 | 65.22 | 72.51 | 68.67 | 76.3 | 95.2 | 74.1 |
| Full Joint | 89.52 | 98.21 | 93.66 | 67.23 | 73.76 | 70.34 | 77.6 | 95.0 | 75.1 |

**Development Data**

| | Connective-ID | | | Predicate-ID | | | ArgID Precision | | |
|---|---|---|---|---|---|---|---|---|---|
| | Rec | Prec | F1 | Rec | Prec | F1 | ARG1 | ARG2 | Conn |
| Cascade+Indep | 95.19 | 97.98 | 96.56 | 70.15 | 72.70 | 71.40 | 77.2 | 95.5 | 74.7 |
| Cascade+Joint | 95.19 | 97.98 | 96.56 | 72.26 | 74.87 | 73.54 | 78.9 | 95.1 | 76.9 |
| Full Joint | 94.82 | 97.32 | 96.05 | 73.56 | 75.50 | 74.52 | 79.5 | 94.9 | 77.6 |

Table 5.2: Results comparing independent (explicit) connective and independent argument identification (Cascade+Indep), independent connective and joint argument identification (Cascade+Joint) and simultaneous identification of both arguments and the connective (Full Joint) over the DevTest data (WSJ sections 00,01 and 22) and the Eval data (WSJ sections 23 and 24).

Predicate-ID score over a Cascade+Joint which first identifies the connectives and then identifies the two arguments for each identified connective, jointly. On the Eval Data, the Predicate-ID F-measure error reductions of the Full Joint over the Cascade+Indep model, which identifies arguments separately as well as the connective, are on the order of 12% with a comparable error reduction on the Development Data. Given that the Full Joint model doesn't improve Connective-ID performance, it seems fair to conclude that Full Joint model does a better job of selecting connectives for which it is more likely to be able to identify the arguments correctly. This is reflected in the higher Argument-ID precision scores for the Full Joint model.

## 5.4 Sequence Re-Ranking

In this section, we develop a discourse parsing model that takes yet more global information into account. Specifically, we consider the *sequence* of discourse connectives and identify the arguments for the sequence of connectives jointly. As is frequent in sequence modeling, we make the Markov assumption and allow the probability distribution over argument pairs for a connective $\pi_i$ at position $i$ to be conditionally independent of the argument pairs $\pi_j$ where $j < i - 1$ and $j > i + 1$ given $\pi_{i-1}$ and $\pi_{i+1}$. We utilize a general form of Conditional Random Fields to capture these Markov dependencies while allowing for rich non-independent features.

### 5.4.1 Background on Random Fields

Markov Random Fields (MRFs) are undirected graphical models in which each random variable in a complex probability distribution is represented as a vertex in a graph. Two vertices are connected in the graph if they are directly dependent. [1] Conditional Random Fields (CRFs) are a common special case of MRFs in which some of the variables are given (i.e. fixed or observed) and will always be conditioned upon.

Let $Y$ be a set of output variables and $X$ be a set of observed variables with $\mathbf{y}$ and $\mathbf{x}$ denoting particular assignments to those variable sets. In their most general form, CRFs model the conditional distribution of a set of output variables $\mathbf{y}$ given a set of observed

[1]More precisely, two variables $x_i$ and $x_j$ are *not* connected if and only if there exists a set of random variables $Y$, where $x_k \notin Y, x_j \notin Y$ s.t. $x_i$ and $x_j$ are conditionally independent of each other given $Y$. Note that $Y$ may be the empty set.

variables $\mathbf{x}$ as:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_X} \prod_{A \in G} \psi_A(\mathbf{y}_A, \mathbf{x}_A) \tag{5.2}$$

where $G$ is a set of subsets $A \subset X \cup Y$ with $\mathbf{x}_A$ denoting an assignment to the variables $A \cap X$ and $\mathbf{y}_A$ denoting an assignment to the variables in $A \cap Y$. Each $\psi_A$ is a non-negative function (called a *factor* or *compatibility function*) over the subset of variables $A$. The term $Z_X$ above is known as the *partition function* and is computed by marginalizing over all possible assignments to $Y$ and $X$:

$$Z_X = \sum_{\mathbf{y},\mathbf{x}} \prod_{\psi_A \in G} \psi_A(\mathbf{y}_A, \mathbf{x}_A) \tag{5.3}$$

Frequently, the compatibility functions take on an exponential form:

$$\psi_A(\mathbf{x}_A, \mathbf{y}_A) = \exp\left(\sum_k \lambda_k f_k(\mathbf{x}_A, \mathbf{y}_A)\right)$$

for a parameter vector $\Lambda = \{\lambda_i\}$ and corresponding *feature functions*, $f_k$.

The graphical structure of CRFs may be arbitrary, in general. Many instantiations of CRFs, however, restrict the graphical structure to linear chain-structures. Such a restricted structure offers the possibility for more efficient statistical inference in the model while remaining useful for a wide variety of applications.

Sequence structured Conditional Random Fields(CRFs) [Lafferty et al., 2001] have become one of the primary tools for modeling a wide-range of phenomena in natural language, including part-of-speech tagging [Wellner & Vilain, 2006], chunking, named entity tagging, string-similarity [Wellner et al., 2005; McCallum et al., 2005], citation extraction [Peng & McCallum, 2004], word-alignment [Blunsom & Cohn, 2006] and a variety of other applications.

Sequence structured CRFs model the conditional distribution of a sequence of *labels*, $\mathbf{y}$ given a sequence of *observations*, $\mathbf{x}$ as:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_{\mathbf{x}}} \exp\left(\sum_i \sum_k \lambda_k f_k(y_{i-1}, y_i, \mathbf{x}, i)\right) \tag{5.4}$$

where $i$ ranges over the positions in the sequence, $k$ ranges over the model's features, the $\lambda_k$ are the model parameters and the $f_k$ are real-valued *feature functions* over an output variable (i.e., a label) $y_i$, its predecessor, $y_{i-1}$ the observation sequence, $\mathbf{x}$ and the index $i$.

The normalization term, $Z_{\mathbf{x}}$ sums over over all possible labelings of the sequence:

$$Z_{\mathbf{x}} = \sum_{y'} \exp\left(\sum_i \sum_k \lambda_k f_k(y'_{i-1}, y'_i, \mathbf{x}, i)\right)$$

### 5.4.2 CRFs for Sequence Ranking

CRFs are standardly employed to label sequences with a small, fixed class of labels, such as part-of-speech tags (e.g., DT, NN, etc.) or states that encode a set of entity types (e.g., PERSON, ORGANIZATION, etc.). For such tasks with fixed categories, the features of a CRF are often factored as: $f_k(y_{i-1}, y_i, \mathbf{x}, i) = p(\mathbf{x}, i)q(y_{i-1}, y_i)$ where $p$ is a predicate over the observation sequence and current position only and $q$ is a predicate over the output label pair, $y_{i-1}, y_i$. With such a factoring, it is natural to view a feature as correlating a predicate of the observation sequence (at position $i$) with a particular label or label pair as defined by $q$. In particular, the correlation will be "triggered" only when $q(y_{i-1}, y_i) = 1$. Thus, for any given property of the observations, there may be multiple features in the model associating this predicate with a different labels or label pairs, each of which will receive a different learned weight. For example, let $p(\mathbf{x}, i) = 1$ if and only if "the word is `man` at position $i$ and `the` at position $i-1$". For part-of-speech tagging[2] , then, with the above predicate $p$, we might expect the weight for

$$f_k(y_{i-1} = \textsc{dt}, y_i = \textsc{nn}, \mathbf{x}, i) = p(\mathbf{x}, i)q(y_{i-1} = \textsc{dt}, i_t = \textsc{nn})$$

to be higher than the weight for the feature:

$$f_k(y_{i-1} = \textsc{vbd}, y_i = \textsc{in}, \mathbf{x}, i) = p(\mathbf{x}, i)q(y_{i-1} = \textsc{vb}, y_i = \textsc{in})$$

### 5.4.3 Non-Factored Feature Functions: An Example from Word Alignment

While the above feature factorization is typical for most applications of CRFs, there is no requirement that the features predicate over the *value* of an output variable assignment. More general features could view the output variable assignment as an index (or pointer) to some additional part of the problem structure. Feature functions of this type can then

[2] DT denotes a determiner (e.g. "the"), NN denotes a common noun, VBD denotes a past-tense verb and IN denotes a preposition.

consist of predicates over the referenced structure (along with the observation sequence, $\mathbf{x}$ and position $i$). A simple, but powerful, example of this view of features is the use of CRFs for discriminative word-alignment [Blunsom & Cohn, 2006].

The word-alignment problem (or sequence-alignment problem, more generally) is an important component problem in statistical machine translation (SMT) and other areas. For SMT, word alignment models are used to align the words of sentence-aligned bi-lingual parallel corpora. Statistics (such as word/phrase translation tables) are then gathered over the predicted word-aligned data and used for decoding (translating) source language sentences into the target language.

Given a foreign language sentence $\mathbf{s}$ (known as the *source*) and an English (or *target*) language sentence $\mathbf{t}$, the word-alignment problem can be formulated in CRFs to model many-to-one alignments, where each source word is aligned with zero or one target words. Given $\mathbf{s}$ and $\mathbf{t}$, an alignment, $\mathbf{a}$ is a sequence of *indices* where $\mathbf{a_i} = j$ indicates that the source word at position $i$, $\mathbf{s_i}$ is aligned with the target word at position $j$, $\mathbf{t_j}$. When $\mathbf{a_i} = null$ the source word $\mathbf{s_i}$ is not aligned to any target word.

The following CRF thus defines the conditional probability of an alignment given source and target sentences:

$$p(\mathbf{a}|\mathbf{s},\mathbf{t}) = \frac{1}{Z_{\mathbf{s},\mathbf{t}}} \exp\left(\sum_i \sum_k \lambda_k f_k(\mathbf{a_{i-1}}, \mathbf{a_i}, \mathbf{s}, \mathbf{t}, i)\right) \tag{5.5}$$

As the values of the output variables (i.e. the “labels”) for the alignment CRF are simply word-indices, they do not denote a category and cannot be meaningfully compared.

Further, each sentence has a different number of possible output variable values as the number of possible index values is equal to the length of the source sentence. Features that predicate directly on the output variable values are therefore entirely inappropriate. Instead, features are defined as predicates over the source-target word match implied by the output variable value. An example feature where the source language is Norwegian and the target language English might be:

$$f_k(\mathbf{a_{i-1}}, \mathbf{a_i}, \mathbf{s}, \mathbf{t}, i) = \begin{cases} 1 & \text{if } s_{\mathbf{a_i}} = \text{`on'} \wedge t_i = \text{`på'} \\ 0 & \text{otherwise} \end{cases}$$

## 5.4.4 CRFs for Sequential Re-Ranking

In this section, we use the non-factored feature representation described above to arrive at a CRF model for sequential re-ranking. Before doing so, it is worth noting that the non-factored, non output-predicating features (described above) offer a very flexible modeling framework that extends beyond re-ranking.

### Non-stationary Variable CRFs and Non-Factored Feature Functions

Non-factored features allow for modeling tasks involving the selection of a single outcome out of a set of possible outcomes (at each time-step in the sequence), but where the set of outcomes is different (and possibly of varying size) at each time-step. This flexibility is perhaps best illustrated with a concrete problem.

Consider the task of disambiguating mentions of entities in a body of text by mapping them to some sort of canonical unique identifier. This task could be *word sense disam-*

*biguation* (in which the unique identifiers would correspond to word senses) or a task such as gazetteering (where the identifiers would correspond to entries in a gazetteer). Each such mention will have a different set of potential unique identifiers so there is not fixed alphabet of output labels. Moreover, different mentions may very well have different numbers of possible identifiers.

Such models contain what might best be termed *index multinomial* variables. The variables are simply serving as indices to possible outcomes. The nature of the outcomes is determined entirely by the choice of features. In sequential models, when different numbers of outcomes are possible at each time step, the model is *non-stationary*[3] with respect to the *type* of random variable.

### Sequential Ranking CRFs

We first describe Sequential Re-Ranking CRFs with a general notation, and then formalize the models for sequentially re-ranking argument pairs of discourse connectives.

Let $\mathbf{x} = x_1, ..., x_{|T|}$ be a sequence of observations. Assume a function which generates an ordered set of $K$ outputs,$GEN_K(x_t)$, for a particular observation $x_t$. A sequential re-ranking of $\mathbf{x}$ is a sequence of indices, $\mathbf{y} = y_1, ..., y_n$ where $y_t = i$ indicates that output corresponding to $x_t$ is the $i$th element of $GEN_K(x_t)$. When the model is stationary with respect to the number of outcomes (i.e. when $K$ is fixed for all $t$), sequential re-ranking CRFs are defined as:

[3]A sequential model is stationary when features and parameters are *tied* across time-steps — i.e., their values are independent of the position in the sequence.

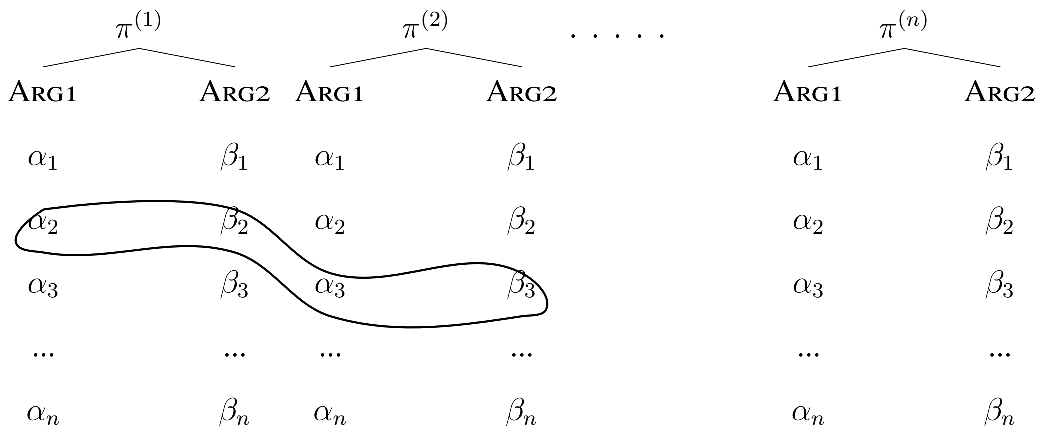

$\pi^{(1)}$
$\pi^{(2)}$
$\pi^{(n)}$
ARG1
ARG2
$\alpha_1$ $\alpha_2$ $\alpha_3$ ... $\alpha_n$
$\beta_1$ $\beta_2$ $\beta_3$ ... $\beta_n$

Figure 5.1: Argument re-ranking as a sequence modeling problem.

$$p(\mathbf{y}|\mathbf{x}) = \frac{\exp\left(\sum_{t=1}^{|T|}\sum_k f_k(y_{t-1}, y_t, \mathbf{x}, t)\right)}{\sum_{\mathbf{y}'\in\mathcal{N}_K^{|T|}} \exp\left(\sum_{t=1}^{|T|}\sum_k f_k(y'_{t-1}, y'_t, \mathbf{x}, t)\right)} \tag{5.6}$$

where $\mathcal{N}_K^{|T|}$ denotes the set of all possible index sequences of length $|T|$ with index values ranging from 1 to $K$.

## 5.5 Capturing Bi-Connective Argument Dependencies

### 5.5.1 Motivation

We propose using CRFs to identify the arguments of connectives, and to, in particular, model dependencies between the arguments of adjacent connectives. The motivation for this is based, in part, on established analyses that have examined the degree to which adjacent relations overlap and interact with each other [Lee et al., 2008] [Lee et al., 2006]. It is demonstrated, for example, that shared arguments occur 7.5% of the time but that crossing dependencies (e.g., where the ARG1 of a connective $\pi_t$ occurs between

| Count | | | | | | |
|---|---|---|---|---|---|---|
| 6486 | $\alpha^{(1)}$ | $\pi^{(1)}$ | $\beta^{(1)}$ | $\alpha^{(2)}$ | $\pi^{(2)}$ | $\beta^{(2)}$ |
| 1362 | $\alpha^{(1)}$ | $\pi^{(1)}$ | $\beta^{(1)}/\alpha^{(2)}$ | $\pi^{(2)}$ | $\beta^{(2)}$ | |
| 1264 | $\alpha^{(1)}$ | $\pi^{(1)}$ | $\beta^{(1)}$ | $\pi^{(2)}$ | $\beta^{(2)}$ | $\alpha^{(2)}$ |
| 1023 | $\pi^{(1)}$ | $\alpha^{(1)}$ | $\beta^{(1)}$ | $\alpha^{(2)}$ | $\pi^{(2)}$ | $\beta^{(2)}$ |
| 880 | $\alpha^{(2)}$ | $\alpha^{(1)}$ | $\pi^{(1)}$ | $\beta^{(1)}$ | $\pi^{(2)}$ | $\beta^{(2)}$ |
| 714 | $\alpha^{(1)}/\alpha^{(2)}$ | $\pi^{(1)}$ | $\beta^{(1)}$ | $\pi^{(2)}$ | $\beta^{(2)}$ | |
| 341 | $\pi^{(1)}$ | $\beta^{(1)}$ | $\alpha^{(1)}/\alpha^{(2)}$ | $\pi^{(2)}$ | $\beta^{(2)}$ | |
| 316 | $\alpha^{(1)}$ | $\pi^{(1)}$ | $\pi^{(2)}$ | $\beta^{(2)}$ | $\alpha^{(2)}/\beta^{(1)}$ | |

Table 5.3: Histogram indicating the relative order of adjacent explicit discourse connective pairs and their arguments across the entire PDTB corpus. The two adjacent connectives are denoted $\pi^{(1)}$ and $\pi^{(2)}$ with their arguments indexed accordingly. Shared arguments are denoted with a slash (i.e., '/'). Only the 8 most frequent patterns are included here.

a connective $\pi_{t-1}$ and the ARG2 of $\pi_{t-1}$) occur very infrequently (less than 1%) among adjacent connectives[4].

Table 5.3 provides a histogram of how adjacent discourse connectives and their arguments are ordered in the text throughout the PDTB. The histogram makes clear that there is considerable interaction among the arguments of adjacent connectives, the statistics of which could be utilized to constrain the potential arguments of discourse connectives. In particular, shared arguments occur quite frequently, such as in the following sentence :

(29) $\boxed{\text{When}}^{1}$ demand ***is***$^{1,2}$ stronger than suppliers can handle $\boxed{\text{and}}^{2}$ delivery times lengthen, prices *tend*$^{2}$ to rise.

The ARG2 for *When* is shared with the ARG2 for *and*[5].

[4]Counting crossing dependencies among non-adjacent connectives may result in higher frequencies of interleaved arguments.

[5]It is worth noting here that interaction among lexical *heads* is different than interaction among argument *spans*. For example, examining the patterns of heads conflates the case where there is a shared argument that is a matrix clause and the case where an argument for one connective subsumes both arguments of a different connective.

Crossing dependencies rarely occur among adjacent connectives. One such case is the following example taken from Lee et al. [2006]:

(30) Mr. Meek *said*$^1$ the evidence *pointed*$^2$ to wrongdoing by Mr. Keating "and others," $\boxed{\text{although}}^1$ he **did**$^1$n't allege any specific violation. Richard Newsom, a California state official who last year examined Lincoln's parent, American Continental Corp., said he $\boxed{\text{also}}^2$ **saw**$^2$ evidence that crimes had been committed.

The Arg1 for the connective *also*, headed by 'pointed' appears between the connective *although* and its Arg1 headed by 'said'.

Since crossing dependencies are rare, we might expect a model that captures dependencies among the arguments of adjacent connectives to assign lower probabilities to such configurations.

### 5.5.2 A Conditional Sequence Model for Identifying Arguments

The approach taken for modeling these dependencies based on adjacency can be viewed as a generalization of the joint argument identification system described in Section 5.3. Rather than re-ranking the arguments for each connective independently, the CRF allows for arguments to be selected in a co-dependent manner among adjacent connectives. Instead of having a fixed set of output labels as with a standard CRF formulation, however, here the output value at each position in the sequence corresponds to an index that refers to a particular argument pair in the set of candidate argument pairs for the connective at that position. More specifically, the feature functions in the model are *non-factored* — they do *not* predicate separately over the output variable assignments and the observations.

Let $\vec{\pi} = \pi_1, ..., \pi_{|T|}$ be a sequence of connectives within a document. Assume a function which generates a candidate set of $K$ argument pairs, $G_K(\pi_t)$, for a particular connective $\pi_t$. Let $\overrightarrow{\langle \alpha, \beta \rangle} = \langle \alpha, \beta \rangle^{(1)}, ..., \langle \alpha, \beta \rangle^{(|T|)}$ denote a sequence of argument pairs, one pair for each connective. The probability of such a sequence is defined as:

$$p(\overrightarrow{\langle \alpha, \beta \rangle} | \vec{\pi}) = \frac{1}{Z_{\vec{\pi}}} \exp \left( \sum_{t=1}^{|T|} \sum_{k} \lambda_k f_k(\langle \alpha, \beta \rangle^{(t-1)}, \langle \alpha, \beta \rangle^{(t)}, \vec{\pi}, t) \right) \tag{5.7}$$

where the partition function, $Z_{\pi}$, sums over all possible sequences of argument pairs for each connective:

$$Z_{\vec{\pi}} = \sum_{\substack{\langle \alpha, \beta \rangle^{(t-1)} \\ \in G_K(\pi_{t-1})}} \sum_{\substack{\langle \alpha, \beta \rangle^{(t)} \\ \in G_K(\pi_t)}} \exp \left( \sum_{t=1}^{|T|} \sum_{k} \lambda_k f_k(\langle \alpha, \beta \rangle^{(t-1)}, \langle \alpha, \beta \rangle^{(t)}, \vec{\pi}, t) \right) \tag{5.8}$$

The main strength of the above formulation is that the feature functions, $f_k$, now have access to the candidate argument pairs at position $t$ *and* the candidate pairs at $t - 1$. Figure 5.1 provides an illustration of this. Additionally, feature functions have access to the connectives themselves at adjacent time steps within the sequence (i.e., $\pi_{t-1}$ and $\pi_t$). Accordingly, features can capture properties and relations between "nearby" arguments and connectives. Back in Example 29, features can capture, for example, that the Arg2 of the first connective in the discourse, *When*, is the same as the Arg1 of the subsequent connective, *and*.

### 5.5.3 Inference and Parameter Estimation

For any probabilistic model, two key questions are: 1) how can the model be used to make *inferences* (or predictions) given some observations, and 2) how can the model parameters be estimated from the data. In the context of identifying discourse arguments, the goal of inference is to solve:

$$\arg\max_{\vec{\langle\alpha,\beta\rangle}} (p(\vec{\langle\alpha,\beta\rangle}|\vec{\pi}) = \arg\max_{\vec{\langle\alpha,\beta\rangle}} \left( \sum_{t=1}^{|T|} \sum_{k} \lambda_k f_k(\langle\alpha,\beta\rangle^{(t-1)}, \langle\alpha,\beta\rangle^{(t)}, \vec{\pi}, t) \right) \tag{5.9}$$

This can be carried out with a slight variation of the Viterbi algorithm to identify the most likely sequence of argument pairs. The Viterbi algorithm provides a solution in time quadratic in the number of states and linear in the data size despite the fact that there are an exponential number of such sequences.

The problem of estimating the parameters of CRFs has received considerable attention over the last few years, not least of which is because training these models is often computationally expensive. Discriminative, conditional learning is generally more complicated than with generative models which frequently estimate parameters by maximum likelihood estimation that can be achieved by simply counting occurrences of events in the training data (and optionally smoothing). For conditional models, the optional parameters according to the model can not be identified in closed form and must be estimated in an iterative fashion by repeatedly carrying out inference. That is, at each iteration inference is applied to determine the difference between the model's predictions and the training data labels (along with the expected values of the various features) and this difference is used to update the parameters arriving at a model better able to predict the training data.

Most forms of parameters estimation employ *batch learning* whereby each iteration involves updating parameters only after performing inference over the entire data set. More recently, *online learning methods* in the form of stochastic gradient descent have gained attention as they offer much faster convergence rates. Many of these methods, such as Voted (or Averaged) Perceptron [Collins, 2002], however, tend to produce slightly less accurate models [Sha & Pereira, 2003] and are also very sensitive to the learning rate (speed at which parameters are adjusted).

Recently, Huang et al. [2007] have proposed a robust method for dynamically adjusting the learning rate separately for each parameter that works very well in practice. The method, called Periodic Step-size Adaptation (PSA), works by keeping a history of the degree to which adjustments to the parameter values for a particular parameter have fluctuated over the last $n$ updates. Parameters that have fluctuated more will have their learning rates slowed down more rapidly while parameters that appear to be "headed in one direction" will maintain higher learning rates for a longer period of time.

In this dissertation, we use online PSA training throughout for estimating the parameters of CRFs. In many cases, the number of parameters is well into the millions and there are tens of or hundreds of thousands of data instances. Accordingly, conventional, batch learning methods take considerable time (from days to even weeks) to converge, depending on the task. Furthermore, PSA training seems to consistently produce models that are as accurate as those obtained with batch learning methods based on convex optimization such as L-BFGS [Nocedal & Wright, 1999].

## 5.6 Results using Sequential Ranking for Argument Identification

Here we present results using the sequential ranking model to identify the arguments of explicit, and both explicit and implicit connectives.

### 5.6.1 Features

The sequential ranking model makes use of all the features described thus far for selecting arguments for each connective independently (as described in Section 4.1). In addition, Markov features are introduced to capture bi-connective dependences. Specifically we performed experiments with four feature classes CONSTRAINTS, PATTERNS and DOMINATES and SYNPATH which are predicate sets of the current connective $\pi^{(t)}$ and the previous connective, $\pi^{(t-1)}$, along with their candidate argument pairs.

The CONSTRAINTS features included a number of indicator features that checked for whether $\pi^{(t-1)}$ and $\pi^{(t)}$ were immediately adjacent to each other in the text, whether any of the candidate arguments for the two connectives are shared, along with a more specific feature that indicates which arguments were shared (e.g. ARG2-ARG1 would denote that the ARG2 of $\pi^{(t-1)}$ is shared with the ARG1 of $\pi^{(t)}$. Finally, we also introduced a feature when the adjacent connectives' arguments form crossing arcs.

The Patterns features take into account the order in which the connectives and each of their two arguments appear in the text. Re-using the shared argument example from Section 5.5.1:

(31) $\boxed{\text{When}}^{\pi^{(t-1)}}$ demand $\underline{\text{is}}^{\beta^{(t-1)}/\alpha^{(t)}}$ stronger than suppliers can handle $\boxed{\text{and}}^{\pi^{(t)}}$ delivery times $\underline{\text{lengthen}}^{\beta^{(t)}}$, prices $\underline{\text{tend}}^{\alpha^{(t-1)}}$ to rise.

The pattern would be $\langle \pi^{(t-1)} \prec \beta^{(t-1)}/\alpha^{(t)} \prec \pi^{(t)} \prec \beta^{(t)} \prec \alpha^{(t-1)} \rangle$ where $\alpha^{(t)}$ and $\beta^{(t)}$ denote the Arg1 and Arg2, respectively, of the connective at position $t$ within the sequence and $\beta^{(t-1)}/\alpha^{(t)}$ indicates that those two arguments share the same head.

The Dominates features include predicates that check for whether an argument of $\pi^{(t)}$ *syntactically dominates* an argument of $\pi^{(t-1)}$ or vice-versa. In the example above two features would fire indicating that $\alpha^{(t-1)}$ dominates both $\alpha^{(t)}$ and $\beta^{(t)}$.

The SynPath features include the syntactic dependency paths between all combinations of the arguments of adjacent connectives: $\alpha^{(t-1)} \rightsquigarrow \alpha^{(t)}$ , $\alpha^{(t-1)} \rightsquigarrow \beta^{(t)}$, $\beta^{(t-1)} \rightsquigarrow \alpha^{(t)}$ and $\beta^{(t-1)} \rightsquigarrow \beta^{(t)}$

### 5.6.2 Experiments

We performed a set of experiments to gain insight into the potential utility of these Markov features that capture dependencies between adjacent connectives and their arguments. The experiments look at combining these Markov features along with various *subsets* of the standard (non-Markov) connective and argument features. A simpler set of non-Markov

| | ConnectiveAccuracy | | | | | |
|---|---|---|---|---|---|---|
| Feature Set | None | Constraints | Patterns | Dominates | SynPath | All |
| Base | 54.36 | 55.76 | 57.35 | 56.00 | 57.12 | 58.63 |
| +Intervening | 63.47 | 63.61 | 64.78 | 64.26 | 65.02 | 65.67 |
| +Para | 65.25 | 65.29 | 66.60 | 66.74 | 67.16 | 67.21 |
| +ConstParse | 69.45 | 70.20 | 71.22 | 70.29 | 70.85 | 71.60 |
| +DepParse | 74.82 | 75.01 | 75.43 | 74.96 | 74.87 | 75.33 |
| All | 76.55 | 76.50 | 76.36 | 76.56 | 76.22 | 75.80 |
| Avg. Reduction | 0.0% | 1.1% | 3.6% | 2.2% | 3.2% | 4.5% |

Table 5.4: Effects on connective accuracy (i.e., the percentage of connectives with both arguments identified correctly) using different sets of discourse Markov features with various subsets of the original non-Markov argument selection features. Results here are reported on the Development Data. The Avg. Reduction is the average error reduction for the various Markov feature sets over None - i.e., no Markov features.

features allows us to see how useful the Markov features are "on their own" rather than in addition to an already elaborate set of features. We explore sequence models over only the explicit connectives as well as over both implicit and explicit connectives.

### Explicit Connectives

Table 5.4 considers various subsets of the argument identification features (cf. Section 4.1) along the rows with combinations of different Markov features along the columns. A clear pattern emerges showing that modeling Markov dependencies tends to help performance. However, when using all of the standard argument identification features, the Markov features provide no additional benefits and can actually hurt performance slightly.

Looking deeper into these results we can see that the SynPath Markov features, unsurprisingly, help considerably when the argument identification features do not include any features that make use of syntactic information. There is enough regularity

| | ConnectiveAccuracy | | | | | |
|---|---|---|---|---|---|---|
| Feature Set | NONE | CONSTRAINTS | PATTERNS | DOMINATES | SYNPATH | ALL |
| BASE | 50.81 | 51.88 | 53.62 | 51.83 | 54.64 | 55.77 |
| +INTERVENING | 56.74 | 57.88 | 58.41 | 57.99 | 60.12 | 60.52 |
| +PARA | 60.56 | 62.85 | 63.96 | 62.58 | 66.50 | 67.07 |
| +CONSTPARSE | 65.05 | 65.83 | 66.20 | 65.61 | 66.48 | 66.59 |
| +DEPPARSE | 75.12 | 75.28 | 75.01 | 75.32 | 75.19 | 75.28 |
| ALL | 75.79 | 75.93 | 75.64 | 75.90 | 75.82 | 75.82 |
| Avg. Reduction | 0.0% | 2.3% | 3.4% | 2.2% | 5.9% | 6.8% |

Table 5.5: Same results as in Table 5.4 but for both Explicit and Implicit connectives and discourse relations.

between the syntactic relationships of the arguments of adjacent connectives to provide some discriminative power for identifying arguments. However, when constituent and/or dependency parse features are added the Markov syntactic features between arguments may be somewhat redundant.

Another observation is that the PATTERNS features tend to be the most discriminative single source of Markov features. This is important because these features do *not* rely on syntactic information, but simply the natural orderings of the various arguments within the text, and might therefore be more robust when using automatic parses, as we explore in Chapter 7.

**All Connectives**

Considering *all* connectives in a sequence within a paragraph, both explicit and implicit, results in more elements per sequence with a greater potential for discourse constraints and tendencies to influence argument selection. Using the same set of features as for the sequential models considering only explicit connectives, the results for argu-

| | Predicate-ID F-measure | | | | | |
|---|---|---|---|---|---|---|
| Feature Set | NONE | CONSTRAINTS | PATTERNS | DOMINATES | SYNPATH | ALL |
| BASE | 53.91 | 54.85 | 54.84 | 55.03 | 55.40 | 55.76 |
| +INTERVENING | 62.71 | 62.75 | 62.75 | 62.50 | 62.87 | 63.14 |
| +PARA | 64.68 | 65.29 | 65.77 | 65.48 | 65.96 | 65.54 |
| +CONSTPARSE | 70.93 | 71.14 | 71.80 | 71.31 | 71.50 | 72.17 |
| +DEPPARSE | 74.48 | 74.41 | 74.56 | 74.40 | 74.46 | 74.83 |
| ALL | 75.31 | 75.35 | 75.16 | 75.33 | 75.52 | 75.28 |
| Avg. Reduction | 0.0% | 0.8% | 1.3% | 0.9% | 1.7% | 2.2% |

Table 5.6: Effects on Predicate ID F-measure for explicit connectives using different sets of discourse Markov features with various subsets of the original non-Markov argument selection features. Results here are reported on the Development Data. The Avg. Reduction is the average error reduction for the various Markov feature sets over NONE - i.e., no Markov features.

ment identification are shown in Table 5.5. In general, we can see that the accuracy improvements due to the various sets of Markov features are stronger when considering all connectives. This is not surprising given that the "density" of connectives is higher and there is more opportunity for bi-connective dependencies to influence the selection of arguments. In particular, the SYNPATH Markov features show a marked improvement in increasing argument identification accuracy in comparison to modeling only explicit discourse connectives.

## 5.7 Results using Sequential Ranking CRFs for Discourse Parsing

In this section we carry out an analysis of the sequential ranking model based on CRFs for the full discourse parsing task: connective identification and argument selection.

| | Predicate-ID F-measure | | | | | |
|---|---|---|---|---|---|---|
| Feature Set | None | Constraints | Patterns | Dominates | SynPath | All |
| Base | 51.56 | 52.05 | 52.97 | 51.76 | 53.22 | 54.24 |
| +Intervening | 57.12 | 57.53 | 56.82 | 57.13 | 59.07 | 59.60 |
| +Para | 60.58 | 61.91 | 64.02 | 61.86 | 64.90 | 65.04 |
| +ConstParse | 68.13 | 68.99 | 68.98 | 68.82 | 69.78 | 70.09 |
| +DepParse | 74.46 | 74.23 | 74.34 | 74.23 | 74.29 | 74.44 |
| All | 75.26 | 74.90 | 75.05 | 75.22 | 75.05 | 75.22 |
| Avg. Reduction | 0.0% | 1.0% | 2.1% | 0.8% | 3.8% | 4.8% |

Table 5.7: Same results as in Table 5.6 but using for *both* explicit and implicit connectives.

The key difference in the model compared with the previous section is that we need to consider all potential connectives within each paragraph and the model needs and additional *Null* alternative to designate a candidate connective as not acting in a discourse capacity. This is completely analogous to the difference between the non-sequential ranking models in Equation 5.1 and in Equation 4.1. As with the non-sequential model, the features associated with the *Null* outcome are those features used for connective identification. Markov features related to the *Null* outcome include just four features: 1) a feature capturing the case where the outcome for the connective $\pi_{t-1}$ is *Null* and the outcome for $\pi_t$ is not *Null*, 2) where the outcome for $\pi_{t-1}$ is not *Null* and the outcome for $\pi_t$ is, 3) where the outcomes for both $\pi_{t-1}$ and $\pi_t$ are *Null*, and 4) where the outcomes for both $\pi_{t-1}$ and $\pi_t$ are non-*Null*. Exactly one of these features will fire for each consecutive pair of potential connectives with a sequence.

Table 5.6 presents an analysis of the various contributions of Markov features for different groups of non-Markov features for the task of identifying explicit discourse predicates. Results in a corresponding table, Table 5.7, show results for the same set

| **Evaluation Data** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Connective-ID | | | Predicate-ID | | | ArgID Precision | | |
| | Rec | Prec | F1 | Rec | Prec | F1 | ARG1 | ARG2 | Conn |
| Cascade+Indep | 89.83 | 98.41 | 93.92 | 62.96 | 69.99 | 66.29 | 75.2 | 94.1 | 71.6 |
| Cascade+Joint | 89.83 | 98.41 | 93.92 | 65.22 | 72.51 | 68.67 | 76.3 | 95.2 | 74.1 |
| Full Joint | 89.52 | 98.21 | 93.66 | 67.23 | 73.76 | 70.34 | 77.6 | 95.0 | 75.1 |
| Full Joint Seq | 87.45 | 98.30 | 92.55 | 66.66 | 74.87 | 70.50 | 78.5 | 95.3 | 76.2 |
| **Development Data** | | | | | | | | | |
| | Connective-ID | | | Predicate-ID | | | ArgID Precision | | |
| | Rec | Prec | F1 | Rec | Prec | F1 | ARG1 | ARG2 | Conn |
| Cascade+Indep | 95.19 | 97.98 | 96.56 | 70.15 | 72.70 | 71.40 | 77.2 | 95.5 | 74.7 |
| Cascade+Joint | 95.19 | 97.98 | 96.56 | 72.26 | 74.87 | 73.54 | 78.9 | 95.1 | 76.9 |
| Full Joint | 94.82 | 97.32 | 96.05 | 73.56 | 75.50 | 74.52 | 79.5 | 94.9 | 77.6 |
| Full Joint Seq | 93.18 | 97.65 | 95.21 | 73.61 | 77.14 | 75.33 | 80.6 | 96.4 | 79.0 |

Table 5.8: Recapitulation of the discourse parsing results for explicit connectives in Table 5.2with the addition of results for identifying discourse predicates in sequence (Full Joint Seq).

of experiments using the exact same features but for *both* explicit and implicit connectives. As we saw with argument identification, the effect of the Markov features is most pronounced when considering both explicit and implicit connectives, presumably due to greater density of interaction between the arguments of nearby connectives.

### 5.7.1 Full Results for Discourse Parsing

The full results comparing the four different approaches for discourse parsing on both the evaluation data and development data are summarized in Table 5.8 for explicit connectives and in Table 5.9 for both explicit and implicit connectives.

| | **Evaluation Data** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Connective-ID | | | Predicate-ID | | | ArgID Precision | | |
| | Rec | Prec | F1 | Rec | Prec | F1 | ARG1 | ARG2 | Conn |
| Cascade+Indep | 94.64 | 99.17 | 96.85 | 68.53 | 71.82 | 70.14 | 78.9 | 90.7 | 72.4 |
| Cascade+Joint | 94.64 | 99.17 | 96.85 | 69.73 | 73.07 | 71.36 | 78.9 | 90.0 | 73.7 |
| Full Joint | 94.45 | 99.00 | 96.67 | 70.68 | 74.09 | 72.34 | 79.7 | 90.8 | 74.8 |
| Full Joint Seq | 93.15 | 99.26 | 96.11 | 70.71 | 75.34 | 72.96 | 80.7 | 91.5 | 75.9 |
| | **Development Data** | | | | | | | | |
| | Connective-ID | | | Predicate-ID | | | ArgID Precision | | |
| | Rec | Prec | F1 | Rec | Prec | F1 | ARG1 | ARG2 | Conn |
| Cascade+Indep | 97.41 | 98.98 | 98.19 | 71.62 | 72.78 | 72.19 | 79.7 | 91.0 | 73.5 |
| Cascade+Joint | 97.41 | 98.98 | 98.19 | 73.60 | 74.79 | 74.19 | 75.6 | 91.4 | 75.6 |
| Full Joint | 97.36 | 98.72 | 98.03 | 74.47 | 75.51 | 74.99 | 81.0 | 91.6 | 76.5 |
| Full Joint Seq | 96.04 | 98.85 | 97.42 | 74.18 | 76.37 | 75.26 | 81.3 | 92.4 | 77.2 |

Table 5.9: Discourse parsing results for both explicit and implicit connectives.

## 5.8 Summary

In this chapter, we have provided two novel problem formulations for parsing discourse. The first formulation, described in Section 5.3, involves jointly identifying whether a potential discourse connective phrase is, in fact, a discourse connective phrase *along with* its two arguments. We compared this approach to first identifying discourse connectives and subsequently identifying the arguments for those connectives (either jointly or independently). The fully joint model, which identifies both arguments and the connectives simultaneously, provided a 10-12% error reduction on both the development and evaluation data over the model which identified all three elements separately.

The second formulation captures additional global information by identifying the arguments for discourse connectives according to the *sequence of connectives* as they appear across each paragraph. This is carried out using a sequential ranking model based on

Conditional Random Fields with non-factored feature functions — this model is an augmentation of "standard" CRFs that parallels the difference between a conditional log-linear classifier and a log-linear ranker. Various feature sets that capture dependencies between the arguments of adjacent discourse connectives were developed. The results made clear that identifying the arguments of the connectives in sequence, offers the potential for improved performance, especially when identifying the arguments of explicit and implicit connectives within the same model. However, little improvement was made with Markov features when the full set of non-Markov argument identification features was used. We hypothesize that this is due to a combination of 1) a "saturated" model where the remaining errors are difficult cases not likely to be remedied by incorporating Markov features, 2) the use of gold-standard parses means that there is a large number of easy cases where arguments can be identified straightforwardly from the syntax and non-Markov features, and 3) the relative sparsity of discourse predicates — i.e., many discourse predicates in the PDTB have no overlap with nearby discourse predicates, especially when considering only explicit connectives.

Nevertheless, we will see better improvements with Markov features when utilizing automatic, rather than manual, parses in Chapter 7. We have established fairly clearly that Markov features and the proposed sequence model are beneficial. Additional feature engineering may be required, however, to achieve markedly better results over using a "local" model with rich features based on gold-standard syntactic analyses.

# Chapter 6

# Dependency Parsing

Earlier chapters have demonstrated the utility syntactic dependency parsing provides towards identifying the arguments of discourse connectives. So far, we have leveraged gold-standard dependency and parses derived from the Penn Treebank for discourse parsing. This chapter discusses a method for syntactic dependency parsing that again uses a CRF-based sequential ranking model similar to that discussed in the previous chapter for identifying the arguments of discourse connectives. Besides demonstrating the general applicability of our sequential ranking model, a robust syntactic dependency parser is required for fully automatic discourse parsing, which we explore in Chapter 7.

## 6.1 Background

Data-driven dependency parsing is an active area of research, being the subject of the last two CoNLL Shared Tasks [Nivre et al., 2007a; Buchholtz & Marsi, 2006] and a key component of the most recent CoNLL Shared Task [Surdeanu et al., 2008]. Dependency approaches to syntax benefit from extensibility across different languages due to their being somewhat easier to specify for free-word order languages. Additionally, for many NLP applications, a dependency representation is more appropriate or convenient than a phrase-structure representation; or, as demonstrated in earlier chapters, offer improved performance over constituent parses for a particular task.

Most approaches to data-driven dependency parsing fall into two camps: 1) *graph-based* approaches and 2) *transition-based* approaches. Briefly, graph-based methods learn a function that assigns scores to potential dependency parses for a given sentence. The process involves searching the space of possible dependency parses for the highest-scoring one. Scores are assigned to entire parses by decomposing the score into a sum of the scores for each edge between governor and dependent pairs. Maximum spanning tree-based algorithms can be used to efficiently find the maximum scoring parse tree given these edge scores. Typically, each edge score is computed based on the features that hold for that pair of words and the learned parameters associated with each feature. More expressive approaches exist that allow for larger factorizations beyond simple word pairs. For example, McDonald & Pereira [2006] provides a method to capture "second-order" dependencies over *pairs of edges* that have the same parent and Carreras [2007] provides an algorithm that considers $child \leftrightarrow parent$ and $parent \leftrightarrow grandparent$ edge-pairs.

In contrast to graph-based methods, transition-based approaches do not explore the entire space of possible parses during training or decoding. Instead, these methods build up a parse incrementally by performing a sequence of actions made over a single left-to-right pass over the words in a sentence. The approach is to train a classifier that decides which action to perform at each stage in the process. At decoding time, these methods generally operate using a deterministic (greedy) search in which parsing decisions are made by a classifier based on the current state of the parse and its history. The classifier has access to a much richer set of features than graph-based methods since it has access to the entire partial parse that has been constructed so far. The primary disadvantage with these approaches is that mistakes made early on can lead to a cascade of errors. This can be ameliorated, however, by using a beam-search rather than a deterministic search.

## 6.2 A Sequential CRF for Dependency Parsing

This section presents our approach to dependency parsing using a CRF-based sequential ranking approach. In the context of dependency parsing, this approach can be considered a *graph-based* method. However, rather than employing maximum spanning tree algorithms for inference, Viterbi inference is used within the context of sequential ranking.

### 6.2.1 Model Overview

Consider a sequence labeling problem where each word in a sentence, $w_i \in \vec{w}$, is an observation and the corresponding label $p_i \in \vec{p}$ for each word is an integer referencing the position of the $w_i$'s *parent* in the dependency graph. This structure defines a graph where the maximum incoming edge degree is 1; the graph can be viewed as a dependency parse, though it may be non-projective *and* may include cycles.

Note that the problem structure here is essentially the same as for sequentially re-ranking discourse argument pairs described in the previous chapter. With dependency parsing, however, rather than selecting an argument pair at each position in a sequence of discourse connectives, we are selecting a parent for each word in a sequence of words constituting a sentence. The set of candidate parents consists of all the other words in the sentence, which, of course, varies in number from sentence to sentence. This contrasts with the discourse parsing situation in which only the top $N$ argument pairs are ranked at each step in the sequence of discourse connectives and $N$ was fixed across the entire data set.

The model for dependency parsing is simply an instance of the general sequential re-ranking model described in Equation 5.7, which we re-write here for dependency parsing as:

$$p(\vec{p}|\vec{w}) = \frac{1}{Z} \exp\left(\sum_{t=1}^{|\vec{w}|} \sum_{k} f_k(p_{t-1}, p_t, \vec{w}, t)\right) \tag{6.1}$$

Each feature function, $f_k$, can consider features over the entire sentence, $\vec{w}$, and for a particular $w_t$ can consider features over its candidate parent $p_t$ as well as the candidate parent, $p_{t-1}$, for the previous word $w_{t-1}$.

Being a CRF, decoding with this model is known to have $O(s^2t)$ complexity using the Viterbi algorithm where $s$ is the number of states and $t$ is the length of the sequence. For the dependency parsing model, sequences are of length $n$ and the number of states is $n$ -

i.e, the number of words in the sentence. This amounts to $O(n^3)$ asymptotic complexity.

This framework has some advantages over maximum spanning tree methods in that it accommodates more expressive features without resorting to approximate inference [McDonald & Pereira, 2006] or methods with high computational complexity [Carreras, 2007] in order to incorporate higher-level information. The downside to our approach is that the higher-order features that can be considered are restricted to capturing the interactions between edges where the dependent words are adjacent in the original sentence. This doesn't allow the model to capture features over siblings in the graph where those siblings are far apart in the sentence, such as, for example, identifying compatibility between subject and objects (that both have the same parent verb).

Nevertheless, in section 23 of the WSJ, nearly 40% of siblings that are adjacent to each other in the dependency graph have dependents that are immediately adjacent to each other in the text. Thus, a sizeable percentage of the siblings are captured by our model, for English at least. Note also that the CRF-based method can capture *other* constraints/features that McDonald's second-order MST method can not model. Specifically, rich features over adjacent dependents that do not share a governor can be incorporated in our method that indicate, for example, that the adjacent dependents have their governors both to the left, both to the right or split. It's also possible to capture violations of projectivity between the two edges that have adjacent dependents. For two adjacent dependents, $w_{i-1}, w_i$ projectivity is violated if $w_i \prec p_{i-1} \prec p_i$ or if $p_{i-1} \prec p_i \prec w_{i-1}$ where $\prec$ denotes the natural ordering of words within the sentence.

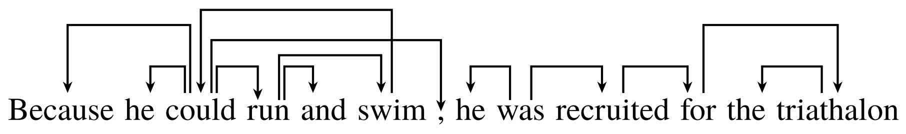

Because he could run and swim , he was recruited for the triathalon

Figure 6.1: A non-projective dependency graph with a cycle: {*could* → *run* → *swim* → *could*.}

One other potential downside with our approach is that it is much restricted that maximum spanning tree-based methods. Not only does it allow non-projective parses, it allows *cycles*. We describe below a simple algorithm for finding the approximate most likely parse that conforms to a tree structure - i.e., a single root word with all other words in the sentence having a single parent and no cycles.

### 6.2.2 Approximation Algorithm for Finding the Maximal Dependency Tree

At a high-level, the algorithm for finding a maximal dependency parse that corresponds to a *tree* works by: 1) identifying cycles, 2) considering all potential root words which consist of parent-less words or words within a cycle, 3) identifying a single parse for each potential root word using a simple heuristic (described below), and 4) scoring each of these candidate parses by identifying its likelihood with respect to the CRF model. Figure 6.1 provides an example parse containing containing a cycle, a pair of crossing arcs that introduce non-projectivity ( *could* → *,* and *swim* → *could*) as well as two disjoint sub-graphs (to the left and right of the ',').

### Cycle Identification

Given a set of learned parameters for dependency parsing, $\Lambda$, and a sentence (i.e., a sequence of words), $\mathbf{x}$, the probability of a dependency parse, $\mathbf{y}$ is $P_\Lambda(\mathbf{y}|\mathbf{x})$. The Viterbi algorithm is used to search for the dependency parse with the highest probability, $\mathbf{y}_\theta = argmax_{\mathbf{y}} P_\Lambda(\mathbf{y}|\mathbf{x})$ .

There is a one-to-one correspondence between cycles in $\mathbf{y}_\theta$ and its strongly connected components as made precise by the following proposition.

**Proposition 6.2.1** *Let $g$ be a directed graph where all vertices, $v_i \in g$ have a maximal incoming degree $\leq 1$. $c \in g$ is a strongly connected component if and only if the nodes in $c$ form a (simple) cycle.*

**Proof** Note that the left directed implication is trivial - a cycle clearly forms a strongly connected component. For the right directed implication, let us assume the implication is false and that there exists a strongly connected component, $c' \subseteq g$ where $c'$ does not form a simple cycle. Note that if $c'$ is not a simple cycle then there must exist $v_i, v_j \in c'$ such that there there are two (or more) distinct paths $p_1 : v_i \prec ... \prec v_j$, $p_2 : v_i \prec ... \prec v_j$ connecting $v_i$ and $v_j$. But clearly some shared vertex in the two paths - either $v_j$ or some $v_k \prec v_j$ must have an incoming degree $> 1$, violating an assumed condition on $g$.$\square$

The strongly connected components (and therefore cycles) can be detected in $O(|V|+|E|)$ time where $|V|$ is the number of vertices and $|E|$ is the number of edges in the graph using Tarjan's algorithm [Tarjan, 1972]. As our graphs are restricted such that there is one edge for each vertex, this step has complexity linear in the length of each sentence.

**Identifying Candidate Parses**

Consider first the case where the graph $g$ corresponding to the most likely parse $\mathbf{y}_\theta$ consists of a single cycle and no root. Note that each word corresponding to a vertex in the cycle is a candidate to be the root word of a sentence, as the cycle must be broken (by removing the incoming edge for one of its vertices) to produce a dependency tree. The word whose incoming edge is removed would be the root word. In the more general case, there may be multiple cycles in $g$ and/or multiple roots.

Let the graph $g$, contain $M$ disjoint sets of vertices, $d_1, ..., d_M$, where each $d_i$ contains a single cycle or a single root[1]. Let $R_i$ correspond to the set of candidate root words in $d_i$. If $d_i$ contains a single root, then $R_i$ will consist of only this root; if $d_i$ contains a cycle, the candidates in $R_i$ will consist of each element in the simply cycle within $d_i$. The set of all candidate parses can be formed from the sets $R_i$ in two steps:

1. First, the cross product $(R_1 \otimes ... \otimes R_M)$ is computed. This is simply the set of tuples representing combinations of candidate roots.

2. For each tuple in the cross product, $(r_1, ..., r_M) \in (R_1 \otimes ... \otimes R_M)$, $M$ candidate parses are formed where the root of the parse is $r_i$ and all $r_j$, where $j \neq i$ have $r_i$ as their parent. Put another way, the process works by selecting each word from the tuple, identifying it as the root and attaching all other words in the tuple to the root children. All combinations of roots and sub-roots are formed over the $R_i$.

Finally, the score for each candidate tree parse is produced. The time required to

[1]Note that a root vertex may be best viewed as a vertex whose parent is itself.

compute these scores is $O(n^3 + nC)$ where $C$ is the number of candidate parses. $O(n^3)$ time is required to compute the state transition lattice after which each candidate parse can be scored in linear time, resulting in $O(nC)$. In the worse case, the cardinality of $C$ is actually exponential in the length of the sentence.[2] Typically, however, the number of candidate parses using this method is small and less than $O(n^2)$ meaning that it does not affect the $O(n^3)$ asymptotic complexity of the sequential CRF-based parsing model. And in practice, we have noticed only very slight increases in parsing times, on the order of 1-3% when using the approximation algorithm to restrict parses to tree structures.

## 6.3 Feature Definitions

This section describes the space of features used by the CRF-based dependency parser. The features can be broken down into *single-edge features* which operate over the current word, $w_i$ and it's candidate parent $p_j$, and *Markov features* which involve features over the current word and the previous word along with their candidate parents. For computational reasons, as well as to avoid overfitting, these Markov features are restricted to operate over a fixed size *window* surrounding the current word, which we discuss below. Further, for the *single-edge* features we restricted the models to features that are *supported* in the training data. That is, only those features that "fire" for dependent-head links that actually belong to the *correct* parse for a sentence. There is some possibility that performance could be improve using *unsupported* features, however, the number of features is that

[2]The extreme worst-case is when the Viterbi parse, $\mathbf{y}_\theta$, consists of a series of cycles of length 3, in which case there would be $O(n^{\frac{n}{3}})$ candidate parses.

| Uni-gram | Bi-gram | Context |
|---|---|---|
| $p_i$-word | $w_i$-word, $w_i$-pos, $p_i$-word, $p_i$-pos | $w_i$-pos, $w_i$-pos+1, $p_i$-pos-1,$p_i$-pos |
| $p_i$-pos | $w_i$-pos, $p_i$-word, $p_i$-pos | $w_i$-pos-1, $w_i$-pos, $p_i$-pos-1,$p_i$-pos |
| $p_i$-word, $p_i$-pos | $w_i$-word, $p_i$-word, $p_i$-pos | $w_i$-pos, $w_i$-pos+1, $p_i$-pos,$p_i$-pos+1 |
| | $w_i$-word, $w_i$-pos, $p_i$-pos | $w_i$-pos-1, $w_i$-pos, $p_i$-pos,$p_i$-pos+1 |
| | $w_i$-word, $w_i$-pos, $p_i$-word | |
| | $w_i$-word, $p_i$-word | |
| | $w_i$-pos, $p_i$-pos | |

Table 6.1: Summary of the single-edge features. $w_i$ is the dependent word and $p_i$ is its governor.

case is on the order of 100 million which introduces computational difficulties.

For the most part, the features we used followed the features in McDonald [2006], which are based on uni-gram, bi-gram and other contextual features of tokens and part-of-speech tags. Table 6.1 summarizes the features.

### 6.3.1 Markov Features

The Markov features in our model can be divided into two types: 1) sibling features and 2) window features. Sibling features involve properties of a word $w_i$, the previous word $w_{i-1}$ and their *shared* candidate governor $p_i = p_{i-1}$ - i.e., these features *only* fire when both their candidate governors are the same token. Window features include properties of $w_i, w_{i-1}$ as well the candidate governors of these two tokens which need not be the same, $p_i, p_{i-1}$. Computing all such features requires time $O(n^3)$ where $n$ is the length of the sentence since for each word $w_i$ there are $n$ candidates to consider for it and $n$ candidates to consider for $w_{i-1}$. To avoid this, we restrict these features to only fire when both candidate parents for $w_i$ and $w_{i-1}$ are within a pre-specified window. We experimented

| Observation Properties | Sibling | Window |
|---|---|---|
| $w_i$-pos, $w_{i-1}$-pos, $p_i$-pos | | |
| $w_i$-pos, $p_i$-pos | | |
| $w_{i-1}$-pos, $p_i$-pos | | |
| $w_i$-pos, $w_{i-1}$-pos | | |
| ANY | | |
| NONPROJ | | |

Table 6.2: Summary of the Markov observation features and their inclusion as sibling and/or window features. The special features ANY and NONPROJ fire for all cases where $w_i$, and $w_{i-1}$ have the same parent and when their candidate parents result in crossing (i.e., non-projective) arcs, respectively.

with a few different window sizes and found a window size of 5 to be optimal on the development data, however the variance between different window sizes was small. The specific Markov features we employed are summarized in Table 6.2. For the Sibling features, each a version of each feature was constructed with the distance and direction to the parent. Similarly, for the Window features, a version of each feature was introduced that also considered the distance and direction between the two governor words, $p_i$ and $p_{i-1}$.

## 6.4 Experiments and Results

We performed a series of experiments to determine overall performance of the parser as compared with the state-of-the-art as well as to specifically ascertain the utility of features that predicate over pairs of edges with adjacent dependents - i.e., the *Markov features*. The ability to incorporate such features is what sets the CRF-based ranking model apart from a Maximum Entropy ranking-type model that doesn't take into account dependencies between adjacent elements.

For training a dependency parser, using the feature sets outlined above, there are typically on the order of 5-10 million features in the resulting model. There are nearly 40,000 instances (i.e., sentences) in the data set and each sentence requires $O(n^3)$ time to compute feature expectations (required for gradient-based learning methods). We found batch learning to be somewhat impractical with training times on the order of 5-10 days, even using the best available convex optimization methods. To improve training times here, as with discourse parsing, we used the stochastic gradient descent method, PSA, to learn the weights of the model [Huang et al., 2007].

## 6.4.1 Experimental Details

Dependency parsing systems have flourished in the last few years, largely driven by the CoNLL Shared Tasks. Two of the most competitive systems, which are also available (and quite usable) for research purposes are Ryan McDonald's MSTParser and the Malt-Parser [Nivre et al., 2007b]. We downloaded these two parsers and trained them on the same data as our CRF-based parser, CRFParser, sections 02-21 of the WSJ. Additionally, we trained Dan Bikel's implementation of the Collins parser and applied the same head-finding heuristics used to generate the dependency representation to its constituent output over the test data. Results are reported on section 23 of the WSJ.

We trained first and second-order MSTParser models using the default parameters. Second-order models introduce features over *pairs* of edges that share the same parent in the graph. This is achieved at the expense of exact inference/search - i.e., with the second order model, it is possible that the optimal parse will not be found. In practice, second-order or higher-order factorizations often improve performance.

The MaltParser implements a variety of parsing algorithms, though the parsing algorithm most widely used is Nivre's algorithm [Nivre, 2003]. The algorithm makes a single left-to-right pass over a sentence, maintaining a stack of partially processed tokens and performing one of four actions while incrementally building up a parse for the sentence: 1) SHIFT - push the next token onto the stack, 2) REDUCE - pop the stack, 3) RIGHT-ARC - add an arc from the top token on the stack to the next word and push the next word onto the stack and 4) LEFTARC - add an arc from the next token to the top token on the stack and then pop the stack. We trained the MaltParser on sections 02-21 using the same parameters as provided with the pre-trained model that comes with the parser trained on the English CoNLL 2007 Shared Task data.

For the CRFParser, we experimented with the full feature set, including Markov features with a window range of $[-5, 5]$, which we refer to as CRFParser-M and the system *without* any Markov features, CRFParser-Base.

The results comparing the MSTParser and MaltParser with the CRFParser are shown in Table 6.3. First, note that CRFParser-M improves performance by over three tenths a percent over CRFParser-Base, which is statistically significant. Note also that the CRFParser is competitive with existing state-of-the-art dependency parsers. The MSTParser performs slightly better than CRFParser-M when it is restricted to projective parses.

Since this particular dataset is projective in nature, this is to be expected. Note, however, the significant drop in performance for MSTParser using non-projective training and decoding methods. All the language datasets from the CoNLL 2007 Shared Task were non-projective (including English) except for Chinese [Nivre et al., 2007a].

Finally, we can see that the $2^{nd}$-order MSTParser marks a significant improvement over other methods on this dataset, including the CRFParser. This is not surprising since the $2^{nd}$-order MSTParser introduces considerably richer features capturing more global properties of potential parses. Carefully incorporating additional Markov features within the CRFParser beyond what we have introduced here may help to close this gap. Alternatively, parse re-ranking has the potential to greatly improve parsing accuracy by incorporating more global parse features. We discuss the potential for parse re-ranking below.

A final point to draw from Table 6.3 is that the CRFParser is very efficient, with processing times notably faster than MSTParser and MaltParser, the latter of which actually has linear asymptotic complexity. [3] The parsing times here amount to about 14 sentences/sec. (or 328 words/sec.) when using Markov features and over 16 sentences/sec. (or 380 words/sec.) without the more expressive features. This makes the CRFParser a viable choice as general-purpose parsing tool. Further investigation is required to look at performance of the CRFParser across languages besides English.

[3]MSTParser and MaltParser are both implemented in Java, while the CRFParser is implemented in Objective CAML, an ML dialect with static typing that tends to offer slightly better runtime performance than Java. The experiments were carried out on 1861MHz, 64-bit Linux machine with 6GB of memory.

| System | Attach. Acc. | Comp. Acc. | Root F1 | Time |
|---|---|---|---|---|
| MSTParser-NonProj | 90.85 | 34.0 | 95.1 | 397 sec. |
| MaltParser$^*$ | 91.30 | 39.7 | 88.0 | 1038 sec. |
| Collins | 91.30 | 38.4 | 96.9 | 5622 sec. |
| MSTParser-Proj$^\dagger$ | 91.58 | 39.4 | 94.9 | 451 sec. |
| MSTParser-NonProj ($2^{nd}$-ord)$^*$ | 92.13 | 41.7 | 96.2 | 549 sec. |
| CRFParser-Base | 91.11 | 34.6 | 94.7 | 149 sec. |
| CRFParser-M$^*$ | 91.46 | 36.8 | 95.1 | 173 sec. |

Table 6.3: Overall results on Section 23 of the WSJ comparing the MSTParser, MaltParser, Collins and CRFParser attachment accuracies (Attach Acc.), percentage of sentences parsed completely correctly (Comp. Acc.), balanced F-measure for finding the root word of the sentence (Root F1) and running times (Time). These results make use of the gold-standard part-of-speech tags. $^\dagger$ indicates the attachment accuracies are significantly different from those on the row above it using McNemar's test $p < 0.05$; $^*$ indicates the differences are significant at $p < 0.01$.

### 6.4.2 Labeled Dependency Parsing

The labeled dependency parsing task involves not only identifying the dependent-governor relations that constitute a dependency graph, but assigning grammatical functions to those relations. The set of labels for the target dependency representation here was presented in Table 4.2.

Some approaches to labeled dependency parsing try to jointly model the parsing and labeling tasks. This allows for shared information from the two decision sets to help resolve ambiguities. The downside is that to maintain tractability, just as with the parsing task, these models are restricted to looking at local aspects of the parse such as single edges or pairs of "nearby" edges in the case of $2^{nd}$-order MSTParser or the CRFParser, for example. A two-stage approach, however, in which the labeling task takes in an entire unlabeled parse as input can take advantage of richer features over the entire parse

| | Label Acc. | LAS |
|---|---|---|
| MaltParser | 91.57 | 88.58 |
| CRFParser-M | 92.60 | 89.05 |
| MSTParser-Proj | 92.51 | 89.17 |

Table 6.4: Label accuracy and Labeled Attachment Score (LAS) for the MaltParser, the CRFParser with Markov features and the projective, first-order MSTParser. Dependency labels for the CRFParser and MSTParser were derived using a second-stage CRF.

structure when making labeling decisions.

The approach to labeling edges in the CRFParser largely follows the approach outlined in [McDonald et al., 2006]. Rather than labeling each edge independently, however, using a classifier we use a CRF to assign edge labels that capture adjacent contextual dependencies. We used the following set of features:

**Dependent-Governor Features** These include the various combinations of the dependent and governor parts-of-speech and tokens along with information on the distance and direction of attachment.

**GrandParent Features** These features involve combinations of the grandparent's part-of-speech and token along with the properties of the dependent.

**Context Features** All the parts-of-speech of the words between the dependent and governor conjoined in various ways with parts-of-speech of the dependent and governor. Whether all the words between the dependent and governor have the same governor – i.e., the intervening parse structure is *flat*.

**Dependent Features** The number of children and the number of siblings the dependent has. Whether the dependent is the left-most or right-most sibling. Number of siblings to the right and left.

The results are shown in Table 6.4 for the CRFParser and MaltParser. The Label Accuracy metric captures the percentage of dependents whose incoming link is labeled with the correct dependency label regardless of whether it was attached to the correct governor. The Labeled Attachment Score measures the percentage of dependents who are attached to their correct governor *and* for which the label was assigned correctly. Labeling parse edges within the MSTParser can be carried out jointly with parsing, though McDonald et al. [2006] indicate that equal or better performance can be obtained using a separate labeling mechanism.

### 6.4.3 Towards Parse Re-ranking

One advantage of a sequence-model approach to parsing is that existing state-sequence decoding algorithms can be leveraged such as beam search decoding and $k$-best decoding. By using $k$-best decoding to identify the top $k$ parses, it is possible to employ re-ranking to select a parse from this list of $k$ parses. This technique has been exploited for phrase-structure parsing [Collins, 2000; Charniak & Johnson, 2005] and more recently with dependency parsing [Hall, 2007].

The latter work bears some similarity to our approach here. Briefly, a log-linear ranker model is used to develop an edge-factored model — that is, a model which for a particular word, $w_i$ in a sentence of length $n$ associates a score with each of the $n$ possible candidate parent/governors for $w_i$, including the possibility that $w_i$ is the root. Given the $n^2$ edge scores, the $k$-best maximum spanning trees can be calculated (cf. [Hall, 2007]). Given the $k$-best parses, a re-ranking model is trained which learns to select the best parse from the top $k$ parses. As the re-ranker takes entire parses as input, features may operate globally over the parse, capturing sub-categorization information, the branching factor for the parse, its depth, etc.

The CRFParser can be viewed as a generalization of the model put forth in [Hall, 2007]. Rather than a single-edge factorization, the CRF-based parser can take into account pairs of edges with adjacent dependents. In addition, it produces a proper probability distribution over all possible parses, rather than a score determined by the product of locally normalized probabilities over individual edges. As the CRF-based model explicitly estimates a distribution over parses, there is reason to believe that the probability scores will be well-calibrated - i.e., that the probability associated with a parse for a given sentence is positively correlated with a high attachment score for a given sentence. A well-calibrated model should produce higher quality $k$-best parse lists.

The results in Table 6.5 show the oracle accuracies for different values of $k$ for the CRFParser (with and without Markov features) and for the approach in [Hall, 2007].

Since these systems, the $k$-best MST system in particular, start from different baseline accuracies, it is perhaps useful to look at the *relative* error reductions in oracle accuracy scores for different pairs of values of $k$ as shown in Table 6.6. Relative reductions are highest with the CRFParser using Markov features, providing better potential for parse

| System | Oracle Accuracy | | | | | | |
|---|---|---|---|---|---|---|---|
| | $k = 1$ | $k = 2$ | $k = 5$ | $k = 10$ | $k = 50$ | $k = 100$ | $k = 500$ |
| $k$-best MST | 85.04 | | | 89.04 | 91.12 | 91.87 | 93.42 |
| CRFParser | 90.99 | 92.12 | 93.28 | 94.05 | 95.35 | 95.84 | 96.71 |
| CRFParser-M$_{(notree)}$ | 91.33 | 92.50 | 93.78 | 94.59 | 95.98 | 96.45 | 97.34 |
| CRFParser-M | 91.46 | 92.59 | 93.85 | 94.62 | 96.04 | 96.48 | 97.40 |

Table 6.5: Oracle attachment accuracies for different $k$-best lists of parses generated by the CRFParser and the results reported in [Hall, 2007].

| System | Relative Oracle Error Reductions | | | |
|---|---|---|---|---|
| | $k = 10 : 1$ | $k = 50 : 10$ | $k = 100 : 50$ | $k = 500 : 100$ |
| $k$-best MST | 26.74% | 18.98% | 8.45% | 19.04% |
| CRFParser | 33.96% | 21.85% | 10.54% | 20.91% |
| CRFParser-M | 37.00% | 26.39% | 11.11% | 26.14% |

Table 6.6: Relative error reductions comparing the MST $k$-best oracle parse scores of Hall [2007], the CRF-based $k$-best oracle parse scores with Markov features (CRFParser-M) and without (CRFParser).

re-ranking methods. The CRFParser without Markov features still arrives at better $k$-best lists than the $k$-best MST approach. While these systems use different features, it seems reasonable to hypothesize that the CRFParser is providing better $k$-best lists due to better calibration.

It is also worth noting that with approaches to re-ranking, typically the most important feature is the probability (or score) produced by the $k$-best generating model [Charniak & Johnson, 2005]. With well-calibrated probabilities, this important feature will have greater discriminative power for parse re-ranking.

## 6.5 Summary and Future Directions

In this chapter we demonstrated the applicability of the CRF-based sequential ranking model to dependency parsing, demonstrating its generality beyond discourse parsing. As the model does not preclude cycles, we provided an approximation algorithm for finding the most likely *tree*-structured parse which, besides generating tree-structured output also slightly improved the parser's accuracy. The overall results indicate that the model is competitive with the state-of-the-art, efficient and is qualitatively different than existing methods. Further, it allows for the construction of high-quality $k$-best parse lists using $k$-best Viterbi decoding, which leaves the door open for further accuracy improvements via parse re-ranking.

Besides parse re-ranking, an interesting area for future work is to explore the notion of *pruning* the set of candidate governors for each word in the sentence. Rather than considering *all* other words in the sentence as candidates, it would be possible to restrict the set of candidate governors to a fixed number $N$, just as we did for sequential discourse argument identification in Chapters 5 and 7. For many word classes, such as determiners, simple heuristics may be able to greatly reduce the number of candidate governors. Also, words of a particular part-of-speech are rarely or never governed by certain other parts-of-speech (e.g., verbs are not governed by determiners). A statistical ranker, as was employed for independent ARG1 and ARG2 identification could be used to generate a list of $N$ candidate governors for each word in the sentence. Note that this requires $O(n^2)$ time and could likely be done with very high oracle accuracies for a small $N$. Given a fixed

$N$, the CRF-based sequential ranking model in this context would actually have linear complexity. The entire process would thus have $O(n^2)$ complexity; further, the $O(n^2)$ process of generating candidate governors would likely require only a small feature set (much small than the full set of features for parsing), resulting in a small constant factor for that computation. This approach therefore has great potential for further improvements in runtime performance. Accuracy improvements may also be possible since a considerably smaller set of candidate governors for each dependent may reduce data sparsity.

# Chapter 7

# Fully Automatic Discourse Parsing

This chapter revisits the discourse parsing task using fully automatic systems rather than relying on gold-standard syntactic analyses. Additionally, we explore how syntactic dependency errors are correlated with drops in performance, provide learning curves indicating how performance correlates with the amount of training data, and perform a detailed analysis of the effect for different classes of Markov features using the CRF-based sequential ranking model.

## 7.1 Argument Identification

We have already presented some results for argument identification using automatic parses produced by the Charniak-Johnson parser in Sections 3.3 and 3.4. In this chapter we use automatic parses with the updated dependency representation described in Section 4.1.2. These parses are generated using the CRFParser, described in the previous chapter, while

| | Accuracy | | | |
|---|---|---|---|---|
| System | Arg1 | Arg2 | Conn | Conn (Gold) |
| Heuristics | 53.0 | 78.8 | 45.4 | 65.8 |
| Indep. ArgID | 71.2 | 92.2 | 66.8 | 72.9 |
| Re-rank ArgID | 72.2 | 91.9 | 68.9 | 75.6 |

Table 7.1: Argument identification results for explicit connectives using automatic parses from the CRFParser and Bikel parser. Results using gold-parses (reported in Section 4.2) are also provided for comparison.

the constituent parses are produced using Dan Bikel's implementation of the Collins parsing model [Bikel, 2004].

### 7.1.1 Parser Development

Chapter 6 described in detail the CRFParser which we use as a dependency parser. In order to apply the CRFParser as well as the Bikel parser for constituent parses we require that the discourse training data be parsed with automatic parses that are representative of parser performance on the test data. We again employ 2-fold jackknifing. The training data is partitioned into two portions; we train both the CRFParser and Bikel parser on one portion and apply each to the other portion and vice-versa to produce a fully automatically parsed training corpus. Since both parsers rely on part-of-speech tags, we used the same 2-fold jackknifing procedure to tag the entire training corpus with part-of-speech tags. We used the CRF-based part-of-speech tagger described in Wellner & Vilain [2006], which provides state-of-the art accuracies.

### 7.1.2 Results

Table 7.1 shows results for identifying the arguments of (given) connectives using automatic parses from the CRFParser and Bikel parser. Again the re-ranking model performs better than identifying the two arguments separately, providing 6.3% reduction in error from a Connective Accuracy of 66.8% up to 68.9%.

### 7.1.3 Effect of Automatic Parses and Comparison with Semantic Role Labeling

For the full re-ranking system, the error term for argument identification increases by over 21% when using automatic parses compared with the gold-standard parses. While considerable, the error term introduced by automatic parsing is actually smaller than for tasks such as semantic role labeling (SRL). SRL involves identifying shallow semantic arguments for verb predicates as annotated by a resource such as PropBank [Palmer et al., 2005] or for nouns with NomBank [Meyers et al., 2004]. A PropBank argument identification system quite similar to our discourse argument identification system is presented in Toutanova et al. [2005][1] in which the error rate from using automatic (Charniak) parses increases 57% for CORE arguments (subject, object, indirect object, etc.) and 47% for ARGM adjunct modifiers (e.g. temporal, causal modifying adjuncts). [2] Arguably, the CORE SRL arguments rely on aspects of the automatic parses that are more accurate than those aspects required for discourse. Identifying the arguments of discourse connectives

[1]The system presented in Toutanova et al. [2005] uses log-linear models to identify the arguments of verb predicates and also uses a log-linear re-ranking model to jointly identify all arguments of a given verb predicate.

[2]Note that these error rate increases are on section 23 of the WSJ and our results are carried out on sections 23 and 24.

(and identifying ArgM for SRL) relies more heavily on capturing the correct inter-clausal syntactic relationships such as subordination and coordination which are more prone to error than establishing many intra-clausal syntactic relationships.

The fact that SRL suffers more from automatic parses than discourse argument identification is probably due to the fact that for discourse parsing the syntax is less varied for the arguments of certain connectives - especially, subordinating and coordinating connectives. The primary non-alignment between discourse arguments and syntax has to do with attribution [Dinesh et al., 2005]. Thus, discourse arguments may be more aligned with syntax than most SRL arguments, but they rely on aspects of syntax that are harder for current statistical parsers to capture. This second point is made clear by the dramatic drop in performance for the heuristic-based system using automatic parses produced by the CRFParser: the error term introduced is 39% (over the independent argument system, which with which it is most comparable). One further insight is that when looking only at subordinating relations, the heuristic system with gold standard parses achieves 80% accuracy for Arg1 and 91% accuracy for Arg2 while with automatic parses the accuracies are 62% and 86% for Arg1 and Arg2, respectively. These differences correspond reasonably well with the results from Dinesh et al. [2005] where a tree-subtraction heuristic was used to identify the arguments of *subordinating* discourse relations. That work demonstrated that Arg1 accuracy drops from somewhere between 76-82% accuracy down to 65% and from 92-93% down to 84% for Arg2 when using automatic parses generated by the Bikel parser's emulation of the Collins parser.

| | Accuracy | | | | | |
|---|---|---|---|---|---|---|
| | AutoParses | | | Gold Parses | | |
| Connective Type | Arg1 | Arg2 | Conn | Arg1 | Arg2 | Conn |
| Subordinating | 83.6 | 95.7 | 82.1 | 89.2 | 98.4 | 88.5 |
| Coordinating | 73.4 | 89.3 | 68.0 | 82.7 | 93.0 | 78.1 |
| Adverbial | 53.7 | 90.9 | 51.7 | 54.8 | 92.2 | 53.0 |

Table 7.2: Argument identification performance by connective type for the re-ranking ArgID system using automatic CRFParser and Bikel parses and comparison with the connective type accuracies using gold standard parses.

| Path | Freq. | CRFParser | MSTParser | MaltParser | Collins |
|---|---|---|---|---|---|
| CONJ | 1170 | 79.3 | 80.4 | 79.7 | 77.8 |
| VMOD | 1044 | 92.2 | 92.3 | 92.0 | 93.5 |
| CONJ_COORD | 722 | 57.6 | 60.1 | 59.1 | 58.1 |
| ADV | 654 | 80.0 | 81.5 | 82.9 | 83.7 |
| VMOD_TMP | 386 | 71.8 | 71.5 | 72.2 | 75.9 |
| VMOD_ADV | 332 | 73.7 | 75.3 | 76.3 | 77.6 |
| TMP | 144 | 77.8 | 77.6 | 80.7 | 81.2 |

Table 7.3: Dependency parser attachment percentage accuracies for the most frequent dependency relation paths connecting discourse connectives to their arguments. Two-step paths here receive accuracies equal to the product of their constituent dependency label accuracies. The frequencies listed are the number of syntactic paths of the listed type that connective discourse connectives to their arguments in the evaluation data.

Table 7.2 shows the Arg1, Arg2 and connective (Conn) accuracies for the three connective types, comparing automatic parse results with those using gold-standard syntactic parses. Discourse adverbials are affected relatively little due to inaccuracies in the underlying syntactic parse; in particular, this is the case with Arg1s. Of course, these arguments are frequently anaphoric, often appearing in prior sentences and are generally less correlated with particular syntactic constructions. For both subordinating and coordinating connectives, the additional error term introduced by using automatic parses is around 30%.

To gain better insight into the errors introduced by the introduction of automatic parses, it is helpful to look at the syntactic parsing errors most relevant for identifying discourse arguments. Table 7.3 provides a breakdown of syntactic dependency attachment accuracies for various dependency label types for the four dependency parsers considered in Chapter 6. The accuracy scores for the CONJ row in the table, for example, are the percentages of words whose governor was identified correctly whose label was CONJ in *in the gold-standard data* for each of the different parsers.

The table provides some indication as to why performance degrades as it does. Poor handling of coordination, a known source of difficulty for parsers, contributes to the 30% increase in the error for identifying the arguments of coordinating connectives. The relevant syntactic paths are CONJ which is frequently the dependency relation between the connective and its Arg1 and the pair CONJ_COORD which links the connective to its Arg2. These paths have dependency accuracies in the 78-80% accuracy range for CONJ and in the range of 58-60% accuracy for the two step CONJ_COORD paths across the four different syntactic parsers. It is also worth noting that the accuracies here are across *all* instances of the stated syntactic paths, not only those instances involved in discourse relations; likely, the performances for those dependency links involved in discourse relations are *lower*, since they will tend to involve more difficult cases of clausal coordination rather than the easier instances of NP and VP coordination.

### 7.1.4 Feature Analysis

| | Accuracy | | |
|---|---|---|---|
| Feature Set | ARG1 | ARG2 | Conn |
| Full | 72.2 | 91.9 | 68.9 |
| NOCONSTPARSE | 70.8 | 91.0 | 67.4 |
| NODEPPARSE | 71.4 | 90.7 | 66.9 |
| NOLEXSYN | 70.6 | 91.3 | 66.9 |
| NOINTERVENING | 69.0 | 91.0 | 65.9 |
| NOPARA | 72.0 | 92.2 | 68.6 |
| NOBASE | 71.3 | 91.5 | 68.3 |
| NOSYNTACTICFEATURES | 67.2 | 87.7 | 62.0 |

Table 7.4: Argument identification results with various feature subsets.

In order to examine the effects of various feature classes, we again perform an ablation analysis carried out by removing various feature classes individually from the full feature set. The results of the ablation study are shown in Table 7.4. Note that these results mostly parallel the ablation analysis with gold-standard parses (cf. Table 4.5). It is also important to keep in mind that the argument candidates are selected based on the dependency relations. Accordingly, the candidate sets are of lower quality when automatic parses are used. That is, a greater number of candidates must be selected when automatic parses are employed to help ensure that the correct argument is included in the set of candidates since there may be errors in the dependency parse. Specifically, we expanded the number of path lengths traversed to 10 for selecting ARG1 candidates (up from 7 with gold-standard parses) and to 6 for selecting ARG2 candidates (up from 3 with gold-standard parses). This results in more candidate arguments, which help to prevent the correct argument from not being included in the candidate set, but makes the job more difficult for the statistical models. This is shown most clearly by comparing the results without any use of syntactic features: for gold-standard parses the connective accuracy

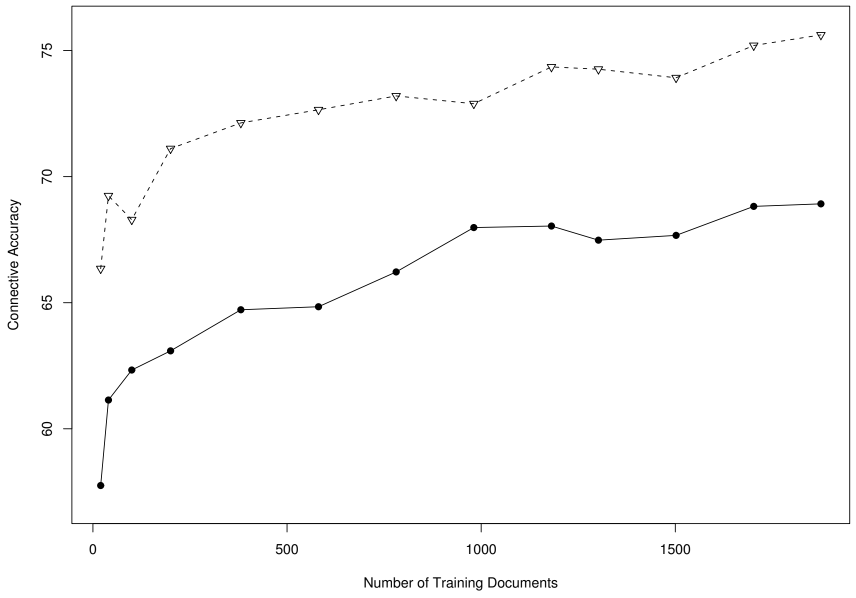


Figure 7.1: Argument identification connective accuracies for gold-standard parses (top curve) and automatic parses (bottom curve).

is 64.3, but drops to 62.0 with automatic parses. These systems are using the exact same set of features but the automatic system has a larger set of candidates to consider for each connective.

### 7.1.5 Learning Curve Analysis

Machine learning methods have many advantages over handcrafted grammars, heuristics or rules and have become a key component in the modern paradigm for approaching many tasks in NLP: 1) annotate 2) train, and 3) test. [3] This paradigm, however, requires annotated data which is often expensive to produce. Thus, an active area of computa-

[3]Note that formulation of NLP *tasks* is typically driven by some underlying *theory*. Thus, theoretical accounts of various linguistic phenomena have, for the time being, played a lesser role in the development of systems than in past years, but instead play a central role in defining what those components should do.

tional linguistics and machine learning, more generally, involves trying to learn with less data using semi-supervised learning (i.e., learning with some annotated data and lots of unannotated data) as well as areas such as transfer learning (i.e., learning with small amount of annotated data for a target task but a larger amount of data for a different, but related task).

While applying these ideas to discourse parsing is outside the scope of this dissertation, it is useful to examine how much annotated data is required to reach a certain level of performance. Figure 7.1 provides learning curves that track the connective accuracy performance with varying amounts of training data using automatic parses as well as gold-standard parses. These curves are relatively steep — using just over 10% of the full amount of training data provides accuracies within around 15% of the accuracies achieved with the full training data set. At the same time it is unclear if the learning curves have flattened out or whether additional training data will continue to provide substantive gains in performance.

Based on our intuitions of language and the nature of discourse, one might expect that discourse parsing may be very sensitive to genre and/or domain. That is, discourse parsing may have greater variance across different data sets than syntactic parsing, for example. Establishing this empirically will require annotation of additional data, such as the Brown corpus, for example. For syntactic parsing, McClosky et al. [2006] show that models tested on the Brown corpus perform at 87.4% when trained on Brown data and at 85.8% when trained on the WSJ portion of the Penn Treebank, despite the fact that the

**Evaluation Data**

| | Connective-ID | | | Predicate-ID | | | ArgID Precision | | |
|---|---|---|---|---|---|---|---|---|---|
| | Rec | Prec | F1 | Rec | Prec | F1 | ARG1 | ARG2 | Conn |
| Cascade+Indep | 85.94 | 97.29 | 91.26 | 57.68 | 65.31 | 61.26 | 71.1 | 93.1 | 67.1 |
| Cascade+Joint | 85.94 | 97.29 | 91.26 | 59.26 | 67.09 | 62.93 | 71.9 | 92.8 | 69.0 |
| Full Joint | 86.19 | 96.69 | 91.13 | 61.20 | 68.66 | 64.70 | 73.8 | 94.0 | 71.0 |
| Full Joint Seq | 85.62 | 97.22 | 91.05 | 61.51 | 69.85 | 65.42 | 74.3 | 94.3 | 71.8 |

**Development Data**

| | Connective-ID | | | Predicate-ID | | | ArgID Precision | | |
|---|---|---|---|---|---|---|---|---|---|
| | Rec | Prec | F1 | Rec | Prec | F1 | ARG1 | ARG2 | Conn |
| Cascade+Indep | 89.82 | 97.07 | 93.30 | 61.61 | 66.58 | 63.99 | 72.2 | 94.1 | 68.6 |
| Cascade+Joint | 89.82 | 97.07 | 93.30 | 63.29 | 68.40 | 65.74 | 73.2 | 93.3 | 70.5 |
| Full Joint | 91.87 | 96.80 | 94.27 | 67.95 | 71.60 | 69.73 | 76.3 | 94.7 | 74.0 |
| Full Joint Seq | 91.27 | 96.78 | 94.20 | 68.19 | 72.28 | 70.18 | 76.6 | 95.2 | 74.7 |

Table 7.5: Results using automatic parses comparing various methods for identifying explicit discourse connectives and their arguments.

latter training set was considerably larger. In semantic role labeling, the degradation is typically worse: in the 2005 Semantic Role Labeling shared task [Carreras & Marquez, 2005], all participating systems saw a roughly 30% increase in the error rate when testing on the Brown corpus vs. testing on the WSJ corpus (all were trained on just the WSJ data). The degree to which discourse parsing will degrade across genres is an interesting open research question.

## 7.2 Joint Connective and Argument Identification

In this section we turn to the full problem of identifying discourse connectives along with their arguments. This initially explored in Chapter 5. Here, however, we examine the results using automatic parses.

### 7.2.1 Explicit Connectives

We first take a look at identifying explicit discourse connectives and their arguments. Table 7.5 provides an overall picture of discourse parsing with fully automatic methods on the PDTB development and evaluation data. The Cascade+Indep method identifies the connective and both arguments separately; Cascade+Joint identifies the connectives in a first pass and then jointly identifies the two arguments for each connective; Full Joint identifies connectives and their arguments jointly; Full Joint Seq considers the sequence of connectives within a paragraph taking into account dependencies between the arguments of adjacent (candidate) discourse connectives.

The CRF-based sequential ranking model (Full Joint Seq) which identifies sequences of connectives and their arguments jointly provides the best overall performance on both the evaluation and development data, with Predicate-ID F-measure error reductions of 11% and 17%, respectively, over the Cascade+Indep approach on those data. Note that the Connective-ID F-measure is slightly degraded on the evaluation data with the Full Joint and Full Joint Seq models, but that the Predicate-ID scores are still improved due to better selection of the arguments of connectives that are identified. That is, the Full Joint and Full Joint Seq models appear to do a better job of selecting discourse connectives whose arguments are “easier” to identify, and avoiding the selection of a discourse connective when there is uncertainty about what the arguments of the connective would be were that connective to be selected.

Another point to be made is with regard to the differences between the evaluation

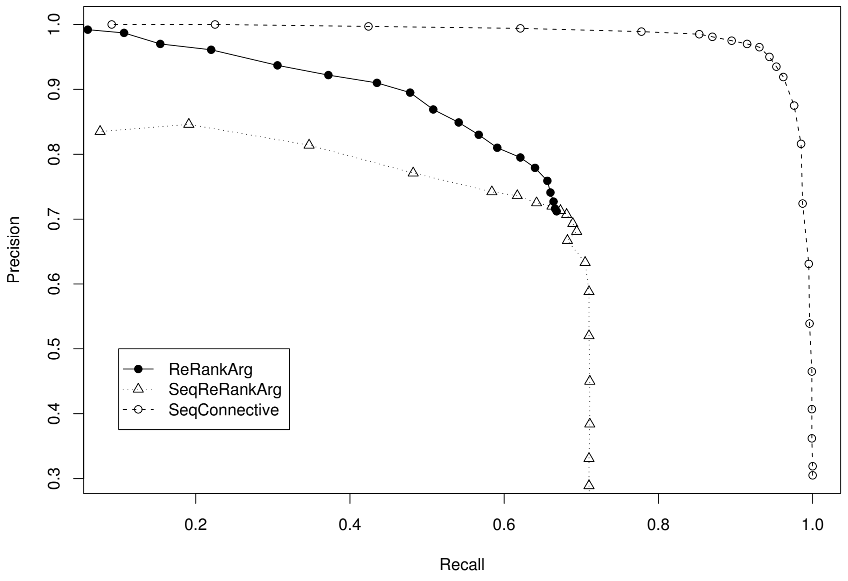


Figure 7.2: Precision and recall curves for Connective-ID (SeqConnective) and Predicate-ID with the sequential joint re-ranking model with a CRF (SeqReRankArg) and Predicate-ID using the non-sequential joint re-ranking model (ReRankArg).

and development results. Recall that the evaluation data includes annotation of discourse relations over VP coordination, which is generally not found in the training data. This explains the relatively low Connective-ID recall on the evaluation data. It would appear that this skew due to what appear to be annotation false positives in the data, removes some of the potential gains for the Full Joint and Full Joint Seq models. The results on the development data may in fact reflect a more realistic evaluation of the system, showing that the Full Joint and Full Joint Seq models can actually result in improved Connective-ID F-measure by considering the arguments jointly with the connective.

**Trading Off Recall for Precision**

One additional analysis we explore here is the ability of the different models to trade off recall for precision. This capability is useful since in certain application contexts, it may be preferable to have a system that operates at a low level of recall, but with high precision. That is, the model may return a relatively sparse set of discourse relations, but those returned have a high likelihood of being correct.

The non-sequential Full Joint re-ranking model can easily trade off precision and recall since for each candidate connective and its selected arguments, the model produces a probability score according to Equation 5.1. The Full Joint Seq model that uses a CRF, on the other hand, specifies a probability distribution over *sequences* of connectives (and their arguments). The confidence scores for individual discourse connectives can be obtained by marginalizing over the entire sequence. A simpler method to trade off recall for precision (or vice-versa) using this model is to follow a method outlined by Minkov et al. [2006] and adjust the weight of a single feature: the feature that fires only for the outcome *Null* - i.e, where the candidate connective is *not a connective*. If the weight of this feature is *increased*, then the probability for the connective not being a discourse connective goes down, whereas if it is *decreased* then the probability scores assigned to the outcomes where the connective is a discourse connective go up.

Figure 7.2 shows the results of adjusting the feature associated with *Null* as well as the results using the non-sequential model when precision and recall are computed by selecting different probability thresholds. The Full Joint model, despite not being able to quite match the Full Joint Seq model in terms of F-measure is able to achieve high Predicate-ID precision. For example, at a Predicate-ID recall of level of 40% (i.e., with 60% of discourse predicates failing to be identified) the precision is well over 90%. With even lower recall rates below 20% the precision is well into the upper 90's.

The precision rates for the sequential model fail to increase anywhere near as dramatically. This would appear to be due to the fact that simply modifying the value of the *Null* feature does not at all take into account the model's degree of certainty with regard to the connective's arguments. So, for example, the model may be somewhat uncertain about whether a particular candidate connective is indeed a connective but is confident about what its arguments should be. On the other hand, some connectives are easy to identify (e.g. "also") but there may be lots of uncertainty about its arguments. The Full Joint model curves, in contrast, are generated by ranking each connective based on the entire joint probability score, taking into account uncertainty about the connective itself and its arguments.

Note that there are other possibilities to produce precision recall curves with the sequential models. One possibility would be to rank entire sequences by their probability score [Wellner et al., 2007] which would capture the uncertainty over the connective arguments but over an entire sequence of connectives, rather than individually as in the Full Joint model.

### Analysis of Markov Features

The overall contributions of the CRF-based sequential ranking model, Full Joint Seq,

| | ConnectiveAccuracy | | | | | |
|---|---|---|---|---|---|---|
| Feature Set | None | Constraints | Patterns | Dominates | SynPath | All |
| Base | 47.83 | 48.72 | 49.88 | 48.76 | 49.65 | 51.10 |
| +Intervening | 59.04 | 58.88 | 59.27 | 59.17 | 59.51 | 59.77 |
| +Para | 62.26 | 62.85 | 63.70 | 62.82 | 63.42 | 63.80 |
| +ConstParse | 65.34 | 65.30 | 65.90 | 65.25 | 65.29 | 65.95 |
| +DepParse | 68.05 | 68.19 | 68.23 | 68.29 | 67.77 | 68.10 |
| All | 70.99 | 71.32 | 71.60 | 70.88 | 71.18 | 71.51 |
| Avg. Reduction | 0.0% | 0.7% | 2.1% | 0.6% | 1.2% | 2.6% |

Table 7.6: The effect of various Markov feature sets measured in terms of Connective Accuracy with different sets of argument identification features using automatic parses.

beyond the Full Joint model are rather modest when considering the full feature set. On the Evaluation data, the Predicate-ID F-measure improves from 64.70 up to 65.42. Despite this, being able to consider features that capture dependencies across the sequence of connectives opens up new possibilities for rich features which may very well lead to better improvements with additional careful feature engineering. We carry out the same kind of analysis here to ascertain the contributions of Markov features in the context of different non-Markov feature sets.

Table 7.6 shows the effect of the various Markov feature sets in combination with various local argument identification feature sets for the argument identification task (i.e., when the discourse connectives are provided) in terms of Connective Accuracy. We can see that on average the Patterns Markov features provide the best performance of any single set of Markov features rather than the syntactic based Markov features. This may well be due to high-levels of noise in the SynPath features due to parser errors and the fact that the Patterns features simply capture sequential information unaffected by parse quality. Also notable, is that the overall error reductions provided by Markov features are

| | Predicate-ID F-measure | | | | | |
|---|---|---|---|---|---|---|
| Feature Set | None | Constraints | Patterns | Dominates | SynPath | All |
| Base | 45.27 | 46.10 | 46.92 | 46.45 | 46.60 | 47.65 |
| +Intervening | 56.99 | 57.37 | 57.06 | 57.24 | 57.44 | 57.46 |
| +Para | 60.53 | 60.98 | 61.45 | 61.05 | 61.68 | 62.05 |
| +ConstParse | 66.74 | 66.82 | 67.22 | 67.00 | 66.36 | 67.50 |
| +DepParse | 66.29 | 66.53 | 65.58 | 66.47 | 66.23 | 66.36 |
| All | 69.66 | 69.77 | 70.01 | 70.02 | 69.87 | 70.18 |
| Avg. Reduction | 0.0% | 0.1% | 0.1% | 1.1% | 1.0% | 2.3% |

Table 7.7: The effect of various Markov feature sets with different argument identification features using automatic parses on the full task of identifying connectives and their arguments for explicit connectives.

lower than those reductions obtained with gold-standard parses (cf. Table 5.4).

Table 7.7 looks at the the full task of identifying connectives and their arguments in terms of Predicate-ID F-measure. The picture here is different, the Patterns features provide little discriminative power. This is likely due to the fact that the for the full connective identification and argument identification task, the sequences consist of all *candidate* connectives, not just the actual discourse connectives. Thus, there are many non-discourse connectives interspersed between actual discourse connectives. The Patterns features which include the relative order of adjacent (candidate) connectives and their arguments are thus diluted. For example, consider the sentence:

(32) Some dealers said the dollar was pressured slightly <u>because</u> a number of market participants had boosted their expectations in the past day $\boxed{\text{and}}$ were looking for an index above 50, which indicates an expanding manufacturing economy. <u>But</u> most said that the index had no more than a minimal effect on trade.

Between the actual discourse connectives, <u>because</u> and <u>But</u> lies a candidate connec-

**Evaluation Data**

| | Connective-ID | | | Predicate-ID | | | ArgID Precision | | |
|---|---|---|---|---|---|---|---|---|---|
| | Rec | Prec | F1 | Rec | Prec | F1 | ARG1 | ARG2 | Conn |
| Cascade+Indep | 93.03 | 98.56 | 95.71 | 63.14 | 66.90 | 64.96 | 76.4 | 87.8 | 67.9 |
| Cascade+Joint | 93.03 | 98.56 | 95.71 | 65.47 | 69.37 | 67.40 | 76.1 | 89.3 | 70.4 |
| Full Joint | 92.75 | 98.39 | 95.49 | 66.61 | 70.67 | 68.58 | 77.7 | 89.6 | 71.8 |
| Full Joint Seq | 92.06 | 98.85 | 95.33 | 67.09 | 72.04 | 69.47 | 78.0 | 91.1 | 72.9 |

**Development Data**

| | Connective-ID | | | Predicate-ID | | | ArgID Precision | | |
|---|---|---|---|---|---|---|---|---|---|
| | Rec | Prec | F1 | Rec | Prec | F1 | ARG1 | ARG2 | Conn |
| Cascade+Indep | 95.43 | 98.67 | 97.02 | 66.09 | 68.34 | 67.20 | 77.2 | 88.6 | 69.2 |
| Cascade+Joint | 95.43 | 98.67 | 97.02 | 68.85 | 71.19 | 70.00 | 77.4 | 89.9 | 72.1 |
| Full Joint | 95.77 | 98.36 | 97.05 | 69.35 | 71.23 | 70.28 | 77.5 | 90.0 | 72.4 |
| Full Joint Seq | 95.01 | 98.55 | 96.75 | 70.62 | 73.25 | 71.92 | 78.9 | 91.5 | 74.3 |

Table 7.8: Results using automatic parses comparing various methods for identifying discourse connectives and their arguments. The results here are over both explicit and implicit connectives.

tive $\boxed{\text{and}}$ which is not considered a discourse connective here due to its role in VP coordination. The PATTERN features for the argument identification task where connectives have already been identified or are given will operate over candidate arguments for $\langle\pi_{t-1}$:$\underline{\text{because}}$, $\pi_t$: $\underline{\text{But}}\rangle$, while for the joint task of connective and argument identification, those pattern features will operate over the for the pairs $\langle\pi_{t-1}$:$\underline{\text{Because}}$, $\pi_t$:$\boxed{\text{and}}\rangle$ and $\langle\pi_t$:$\boxed{\text{and}}$,$\pi_{t+1}$:$\underline{\text{But}}\rangle$.

This indicates that while we have shown that it is advantageous to identify connectives and their arguments jointly, it may be preferable to consider sequence effects only over *identified* discourse connectives, rather than all candidate connectives.

### 7.2.2 Incorporating Implicit Connectives

| | ConnectiveAccuracy | | | | | |
|---|---|---|---|---|---|---|
| Feature Set | NONE | CONSTRAINTS | PATTERNS | DOMINATES | SYNPATH | ALL |
| BASE | 43.18 | 43.22 | 44.72 | 42.69 | 44.94 | 46.44 |
| +INTERVENING | 50.64 | 51.57 | 51.69 | 51.53 | 53.38 | 53.55 |
| +PARA | 58.08 | 59.67 | 60.91 | 58.80 | 64.07 | 64.85 |
| +CONSTPARSE | 65.34 | 66.90 | 67.44 | 66.31 | 68.12 | 68.24 |
| +DEPPARSE | 68.05 | 69.27 | 69.53 | 69.35 | 69.33 | 69.70 |
| ALL | 71.82 | 72.14 | 72.18 | 72.42 | 72.10 | 72.12 |
| Avg. Reduction | 0.0% | 2.5% | 4.6% | 1.9% | 6.0% | 7.1% |

Table 7.9: The effect of various Markov feature sets with argument identification features using automatic parses.

In this section we provide results for identifying the arguments of all connectives, including implicit connectives, using automatic parses. The overall results are shown in Table 7.8. We see similar error reductions in Predicate-ID F-measure as was the case with only explicit connectives: a 14% reduction on the development data, and a 13% reduction on the evaluation data when comparing the Full Joint Seq model against the Cascade+Indep model.

**Markov Feature Analysis**

Again, we look at the the individual Markov feature classes with various local argument identification feature subsets. First, the affects of Markov dependencies for identifying the arguments of given discourse connectives are shown in Table 7.9. An interesting contrast is made with the connective accuracy results for only explicit connectives in Table 7.6 — there is considerably more benefit from the Markov features and sequential model when both explicit and implicit connectives are considered, just as was observed with the gold-standard parses.

| | Predicate-ID F-measure | | | | | |
|---|---|---|---|---|---|---|
| Feature Set | NONE | CONSTRAINTS | PATTERNS | DOMINATES | SYNPATH | ALL |
| BASE | 44.01 | 43.52 | 43.58 | 43.17 | 44.20 | 44.63 |
| +INTERVENING | 51.64 | 51.88 | 52.22 | 52.26 | 52.62 | 53.42 |
| +PARA | 57.75 | 58.43 | 59.00 | 58.29 | 61.62 | 61.92 |
| +CONSTPARSE | 64.68 | 65.46 | 66.07 | 65.51 | 66.79 | 67.11 |
| +DEPPARSE | 69.33 | 69.12 | 69.72 | 69.22 | 68.95 | 69.30 |
| ALL | 71.41 | 71.48 | 71.76 | 71.61 | 71.78 | 71.82 |
| Avg. Reduction | 0.0% | 0.5% | 1.6% | 0.6% | 2.9% | 3.8% |

Table 7.10: The effect of various Markov feature sets with different argument identification features using automatic parses on the full task of identifying connectives and their arguments for explicit and implicit connectives.

| | | Evaluation | | Development | |
|---|---|---|---|---|---|
| | Model | Gold Std | Auto | Gold Std | Auto |
| Explicit | Cascade+Indep | 66.29 | 61.26 | 71.40 | 63.99 |
| | Cascade+Joint | 68.67 | 62.93 | 73.54 | 65.74 |
| | Full Joint | 70.34 | 64.70 | 74.52 | 69.73 |
| | Full Joint Seq | 70.50 | 65.42 | 75.33 | 70.18 |
| Explicit+Implicit | Cascade+Indep | 70.14 | 64.96 | 72.19 | 67.20 |
| | Cascade+Joint | 71.36 | 67.40 | 74.19 | 70.00 |
| | Full Joint | 72.34 | 68.58 | 74.99 | 70.28 |
| | Full Joint Seq | 72.96 | 69.47 | 75.26 | 71.92 |

Table 7.11: Summary of Predicate-ID F-measure results for discourse parsing comparing systems based on gold-standard and automatic parses.

Table 7.10 shows the Predicate-ID F-measure results for various feature combinations for the task of identifying both connectives and their arguments. As before with only explicit connectives, the performance improvements with the introduction of Markov features are lower when considering joint connective and argument identification.

### 7.2.3 Summary Comparison of Automatic Versus Gold-standard Parses

In looking at the full discourse parsing task, we summarize here the primary results comparing performance with automatic parses versus performance using gold-standard parses. Table 7.11 summarizes Predicate-ID F-measure results for discourse parsing using the four different models for just explicit connectives as well as for both implicit and explicit connectives.[4] Clearly, the use of automatic parses degrades performance. However, when considering both explicit and implicit connectives on the evaluation data the Full Joint Seq model is able to hold the error introduced by automatic parses to 11.4% down from 14.8% when the simpler Cascade+Indep model is used with a similar reduction in the error term introduced on the development data from 15.2% with the Cascade+Indep model down to 11.9% with the Full Joint Seq model.

## 7.3 Summary

In this chapter we have explored fully automatic methods for discourse parsing. We have identified discourse connectives and their arguments for both explicit and implicit connectives. The resulting system is a fully-fledged low-level informational discourse parser. However, there is one subtle case not considered here: identifying *implicit* connectives. Determining whether an implicit connective exists involves identifying whether there is an explicit connective that connects the current sentence to a prior sentence in the discourse. Thus, a completely fully automatic parser aiming to generate all the PDTB connective and argument annotations would need to carry out this step.

The primary results of this chapter captured the degree to which automatic parsing

[4]This is a summary of Predicate-ID F-measure results found in Tables 5.8, 5.9, 7.5 and 7.8.

degrades the performance of the discourse parsing model. We saw error increases on the order of 21% for argument identification Connective Accuracy and error increases of over 15% for explicit connectives alone but under 12% for both explicit and implicit in terms of Predicate-ID F-measure on the evaluation data. We also established that jointly identifying explicit and implicit relations appears to help in the sense that Markov features are better able to model contextual dependencies. As in Chapter 5, we saw that jointly identifying connectives and their arguments, sequentially or not, improved performance over identifying those aspects independently. We also demonstrated how performance correlates with the quantity of training data and how recall can be traded for precision using both non-sequential and sequential joint models.

One interesting observation we noted is that it appears there are stronger contextual dependencies between adjacent discourse connectives rather than adjacent *candidate* connectives. The sequential model operating over *candidate* connectives appears to lose some of this contextual information. A simple way to correct for this would be to *not* identify discourse connectives jointly with their arguments and instead first identify the discourse connectives and subsequently identify their arguments using the sequential ranking model.

However, we have already established that it helps to carry these steps out jointly. An interesting alternative for future work would be to re-rank candidate discourse connective *sequences*. This would be yet another level of re-ranking that applies to sequences of connectives and their candidate arguments (rather than just re-ranking argument pairs of individual connectives as we have done). Such an approach would allow for additional features to capture properties of the entire sequence of candidate connectives and each of their candidate argument pairs.

# Chapter 8

# Relation Type Disambiguation

In Chapters 5 and 7, we addressed the problem of identifying the low-level discourse structure of a document by first identifying discourse connectives and then identifying their argument heads. The next important step in processing discourse is identifying the *type* of the rhetorical relation that holds between two abstract objects — i.e., the two arguments of a discourse connective, whether explicit or implicit. By analogy with verbal predicates, we can view this process as disambiguating the *sense* of the discourse connective that lexicalizes relation. Throughout this chapter, we will use the phrases "connective sense" and "relation type" interchangeably, with the former brining the notion of a discourse connective to the fore and the latter being more general, also applicable to discourse relations that do not involve a connective (such as those coherence relations based on *entities* and not abstract objects).

## 8.1 Background

The problem of assigning rhetorical types to discourse relations has received some attention in the literature. In previous work [Wellner et al., 2006] we looked at classifying discourse relations in the Discourse Graphbank into one of 11 different categories, such as *elaboration*, *cause*, *contrast* and *attribution* achieving 81% accuracy when provided the segments participating in the relations.

Sporleder & Lascarides [2005, 2008], following the framework and relation inventory of SDRT, and Marcu & Echihabi [2002] following RST, look at methods for identifying unmarked coherence relations (i.e., relations not signaled by a discourse connective) by creating large amounts of artificially labeled data in the following manner: 1) unambiguous discourse connectives (e.g. because, but) are identified in an unannotated corpus, 2) heuristics are applied to identify the argument spans for the connective, and 3) the two spans are presented *with the unambiguous discourse connective removed* as a training instance with the coherence type determined by the connective. These approaches are interesting since they directly address the more difficult problem of determining rhetorical types where there isn't an explicit discourse connective.

These unsupervised, or semi-supervised, methods are able to achieve scores ranging from 49.7% accuracy at 6-way classification [Marcu & Echihabi, 2002] to 57.6% accuracy at 5-way classification [Sporleder & Lascarides, 2005]. A problem with these results, however, is that the evaluations in these cases were carried out using an artificially imposed uniform distribution over the different relation types for both the training and the test data. Of course, the various relation types do not occur uniformly in real, found data. It therefore makes it difficult to extrapolate these results to more realistic evaluation settings, such as identifying the relation types in the PDTB. Further, as discussed in detail in Sporleder & Lascarides [2008], these evaluations were further biased in that the test samples used for evaluation were created by taking discourse relations signaled by an unambiguous discourse connective and artificially removing the connective, just as done for the training data. When testing such a model on naturally occurring discourse relations that do not have a discourse connective present, rather than on artificial examples that involved an explicit discourse connective that was removed, the results drop considerably (to only slightly better than random guessing).

Within the context of RST, Soricut & Marcu [2003] provide a full discourse parsing model for intra-sentential discourse relations. Their system, using gold-standard syntactic parses and discourse segments is able to identify which segments and segment groups are participating in a discourse relation with 96.2% accuracy, while performance at identifying those relations and their rhetorical type is 70.3%. This indicates that given a discourse relationship, their system can assign the correct rhetorical type 72.7% of the time.

Miltsakaki et al. [2005] present a MaxEnt classifier approach to disambiguating ambiguous explicit connectives such as *when*, *while* and *since* in the PDTB (using somewhat different sense categories as part of their classification than those that appear in PDTB 2.0). They are able to achieve disambiguation accuracies 15 to 20% higher than the majority class baseline using a set of features based on tense and aspect for these explicit connectives.

## 8.2 Discourse Connective Senses in the PDTB 2.0

For our work here, we again utilize the Penn Discourse Treebank 2.0. In addition to annotation of discourse connectives (explicit and implicit) and their arguments, the PDTB assigns one *sense* (and in some cases two) to each discourse connective. These senses can be broken down into four broad categories or classes:

**Comparison** This class includes relations that emphasize differences between the two arguments, such as different/contrasting predicates, different arguments to similar predicates or where one argument denies or refutes the other.

**Contingency** These relations hold when one argument causally influences the other in some manner.

**Temporal** A temporal relation holds when the two arguments are related temporally in some fashion.

**Expansion** This broad class of relations involve instances where one argument is expanding the discourse, adding additional relevant information or moving the narrative forward coherently.

In the PDTB, each of these four sense classes include a set of different relation *types*, some of which have one or more *sub-types*. The full hierarchy of senses is shown in Figure 8.1.

- **EXPANSION**
  - **Conjunction**
  - **Instantiation**
  - **Restatement**
    - specification
    - equivalence
    - generalization
  - **Alternative**
    - conjunctive
    - disjunctive
    - chosen alternative
  - **Exception**
  - **List**
- **COMPARISON**
  - **Contrast**
    - juxtaposition
    - opposition
  - **Pragmatic Contrast**
  - **Concession**
    - expectation
    - contra-expectation
  - **Pragmatic Concession**
- **CONTINGENCY**
  - **Cause**
    - reason
    - result
  - **Pragmatic Cause**
    - justification
  - **Condition**
    - hypothetical
    - general
    - unreal past
    - unreal present
    - factual past
    - factual present
  - **Pragmatic Condition**
    - relevance
    - implicit assertion
- **TEMPORAL**
  - **Synchronous**
  - **Asynchronous**
    - precedence
    - succession

Figure 8.1: Hierarchy of connective senses/types.

We provide here a few examples of a couple of ambiguous discourse connectives that have multiple senses. For a more detailed overview and description of the various sense distinctions, the reader is referred to [Miltsakaki et al., 2008; Prasad et al., 2008].

(33) (CONTINGENCY.CONDITION.GENERAL) *The Senate-House conference committee is used* <u>when</u> **a bill is passed by the House and Senate in different forms.**

(34) (TEMPORAL.SYNCHRONOUS) <u>When</u> *test booklets were passed out 48 hours ahead of time*, she says **she copied questions in the social studies section and gave the answers to students**.

Of course, senses are also attributed to discourse relations signaled by implicit connectives:

(35) (EXPANSION.CONJUNCTIVE) *The strong growth followed year-to-year increases of 21% in August and 12% in September.* <u>Implicit = In fact</u> **The monthly sales have been setting records every month since March.** .

Additionally, recall from Section 4.3 that besides explicit and implicit rhetorical relations, there are relations expressed via 1) alternative lexicalizations (ALTLEX) 2) coherence relations based on entity relations rather than relations between abstract objects (ENTREL) and 3) adjacent sentence units where no coherence relation is present (NOREL). Throughout this chapter, we will combine the ENTREL and NOREL types into a single type OTHER. ALTLEX relations will be treated just as other implicit relations. That is, the phrase denoting the alternative lexicalization is not provided as a discourse connective for computing features to determine the relation type, since identifying such alternative lexicalizations appears non-trivial and has not yet been automated.

## 8.3 Automatic Sense Disambiguation

In this section we present our approach towards automatically disambiguating discourse connectives and provide some experimental results.

### 8.3.1 Task Formulation

As shown in Figure 8.1 there are 29 fine-grained categories – i.e., those senses at the leaves in the hierarchy. In cases where two annotators disagreed about a particular fine-grained sense, the sense assigned to that item was the next consistent coarser-grained sense up the hierarchy consistent. For example, if one annotator assigned a connective the sense CONTINGENCY.Condition.hypothetical and another assigned CONTINGENCY.Condition.unreal-present the connective would be labeled simply CONTINGENCY.Condition. Due to this process of "backing-off", there are actually 42 fine-grained categories in the PDTB.

The inventory and level of granularity for sense distinctions remains open for debate. No doubt, the level of granularity is application-specific: some applications may need to make fine-grained distinctions whereas others do not. The degree to which performance degrades when making finer-grained distinctions is also something to consider and investigate. In our experiments we explore three different sets of sense categories: 1) the

| Semi-Coarse Categories | Fine Categories |
|---|---|
| EXPANSION.Restatement | EXPANSION.Restatement.* |
| EXPANSION.Conjunction | EXPANSION.Conjunction |
| EXPANSION.Instantiation | EXPANSION.Instantiation |
| EXPANSION.Other | EXPANSION.Exception<br>EXPANSION.List<br>EXPANSION.Alternative.* |
| COMPARISON.Contrast | COMPARISON.Contrast.*<br>COMPARISON.Pragmatic Contrast |
| COMPARISON.Concession | COMPARISON.Concession.*<br>COMPARISON.Pragmatic Concession |
| CONTINGENCY.Cause.Reason | CONTINGENCY.Cause.Reason<br>CONTINGENCY.Pragmatic Cause.Reason |
| CONTINGENCY.Cause.Result | CONTINGENCY.Cause.Result |
| CONTINGENCY.Condition | CONTINGENCY.Condition.*<br>CONTINGENCY.Pragmatic Condition.* |
| TEMPORAL.Synchronous | TEMPORAL.Synchronous |
| TEMPORAL.Asynchronous.Precedence | TEMPORAL.Asynchronous.Precedence |
| TEMPORAL.Asynchronous.Succession | TEMPORAL.Asynchronous.Succession |

Figure 8.2: The mapping from fine-grained categories to semi-coarse-grained categories.

four coarse-grained classes along with a fifth class OTHER (for EntRel and NoRel), 2) the fine-grained categorization with 42 categories (including OTHER) and 3) a semi-coarse-grained level of categorization including 13 senses which collapses some of the fine-grained categories that occur quite infrequently according to the mapping in Figure 8.2.

For each of these three sense label inventories, we consider the problems of assigning senses to both explicit and implicit connectives jointly as well as to explicit and implicit connectives, separately.

### Sense Disambiguation as a Sequence Labeling Problem

Recent work [Pitler et al., 2008] has shown that there are clear dependencies between adjacent discourse relations (i.e., adjacent connectives). That is, some pairs of relation types appear adjacent to each other in the text more frequently than they would by chance. Being able to leverage this context of the surrounding discourse could be especially important for implicit connectives since disambiguating these outright is very difficult; even weak contextual dependencies could help identify the senses for implicit connectives.

A naive way to handle this might be to identify the sense of each discourse connective, one at a time, starting from the beginning of each document or paragraph. This would allow the decision procedure for the $n$th discourse connective to use the previous $n - 1$ connectives and their assigned senses as features. Of course, a more principled approach is to use a global sequence model that jointly identifies the best sequence of connective senses or relation types for the corresponding sequence of discourse connectives.

To model these dependencies between adjacent discourse relations, we once again return to using a first-order, sequential CRF. We treat each connective/relation as an element in a sequence where sequences are delimited by paragraph boundaries, just as was done for argument identification with CRFs. As we have a fixed set of labels for each discourse relation (i.e. the set of coarse-grained or fine-grained senses, depending on the task) and each connective is assigned a single label [1], we can use a standard factored CRF, rather than the non-factored CRF that we have developed in this dissertation for

[1] In this work, we only consider the problem of assigning the *first* sense to each connective. The problem of identifying the second sense for those connectives with multiple sense is left for future work.

argument identification.

### 8.3.2 Features

For our experiments we consider the following set of features:

**Connective** This is a single feature which is the connective phrase itself. In the case of implicit connectives, this feature simply denotes that the connective is, in fact, implicit (vs. explicit).

**Argument Heads** These features include the Arg1 semantic head conjoined with the connective and the Arg2 semantic head similarly conjoined.

**Discourse Context** The connectives before and after the connective as well as the bigram including the previous and current connective.

**Syntactic Context** These features include the path from the Arg1 head to the Arg2 head as well as the same path, but conjoined with the connective.

**Tense and Aspect** The tense of the two arguments conjoined with the connective with the following categories: present-continuous, past-continuous, copula, present-perfect, present, past-perfect-progressive, present-perfect, present-perfect-progressive, past-perfect, aorist (simple-past), future-perfect, future, conditional-perfect, and conditional.

**Part-of-speech Context** The part-of-speech of the previous and succeeding words to the connective along with various conjunctions including the connective.

**Modal/Adverbial Context** Negation adverbials as well as basic "manner" adverbs of the Arg1 and Arg2 heads.

**Discourse-Syntactic Context** This feature includes the syntactic dependency path between the current connective and the previous discourse connective.

**Argument Coherence** Whether the discourse arguments for the connective have the same syntactic subject head string, syntactic object string.

**LexPos** These features included 1) content *lexemes* from both argument spans along with their part-of-speech, 2) the positions of the sentences containing the two arguments relative to paragraph boundaries (with value of: beginning, middle, end), and 3) features capturing the length of the two argument spans in terms of tokens. The length features were binned at 3,6,9,12,15,20 and 30. So, a span with 11 words would trigger the feature `len=LessThan12`, while a span with 26 words would trigger `len=LessThan30`.

The LexPos features were inspired by those introduced in Sporleder & Lascarides [2005]. Following that work, we also considered various overlap features that measured the degree to which various words overlapped in the two argument spans, including features that captured the degree to which WordNet senses overlap (by following the hypernym links to the "root" senses for each word). These features did not improve performance, however, when used together with the lexical, positional and span length features. This may have been due to the fact that we did not perform word-sense disam-

| | Sense Granularity | | |
|---|---|---|---|
| Feature Set | Coarse | Semi-coarse | Fine |
| Connective | 64.63 | 54.57 | 52.36 |
| Argument Heads | 66.58 | 54.41 | 50.69 |
| Connective Context | 64.47 | 54.82 | 51.92 |
| Syntactic Context | 66.80 | 57.85 | 54.13 |
| Tense | 65.07 | 54.98 | 51.32 |
| Part-of-speech Context | 65.07 | 55.42 | 50.76 |
| Modal Context | 64.66 | 54.67 | 52.43 |
| Discourse Context | 64.79 | 54.67 | 52.14 |
| Argument Coherence | 64.38 | 55.20 | 52.99 |
| LexPos | 65.76 | 55.63 | 53.15 |
| Syntactic+Discourse+Argument | 67.47 | 57.16 | 53.59 |
| Syntactic+Discourse+Argument+LexPos | 68.38 | 56.72 | 52.89 |
| All Features | 68.88 | 57.29 | 51.36 |

Table 8.1: Accuracies on the evaluation data for labeling explicit and implicit connectives using the four coarse-grained categories with different feature sets. All feature sets include the "Connective" feature.

biguation, and simply selected the most frequent, first sense of each word in WordNet, following Sporleder & Lascarides [2005].

### 8.3.3 Results

Table 8.1 provides results for the various feature sets described for the task of labeling both explicit and implicit discourse connectives with their coarse-grained category, which includes the four connective classes as well as the class Other for NoRel and EntRel cases. Note that the majority baseline for this task provides 36.6% accuracy, which is achieved by labeling all connectives with type **EXPANSION**. The Syntactic Context features provide the most help, individually, but here the entire set of features works additively to provide a modest, but notable improvement over the Connective baseline,

| Relation Type | Prec. | Rec. | F1 | Count |
|---|---|---|---|---|
| Comparison | 88.41 | 67.57 | 76.60 | 666 |
| Contingency | 65.90 | 55.68 | 60.02 | 616 |
| Expansion | 66.62 | 78.66 | 72.14 | 1162 |
| Temporal | 81.48 | 77.81 | 79.61 | 311 |
| Other | 50.54 | 56.59 | 53.99 | 417 |

Table 8.2: Precision, recall and f-measure scores for assigning relation types to explicit and implicit discourse connectives using the best performing feature configuration (All Features).

improving results from 64.6% to 68.9%. That the Connective feature, by itself, provides a very competitive baseline has been demonstrated in previous work [Wellner et al., 2006].

The results for labeling connectives with semi-coarse and fine sense categories are also shown in Table 8.1. For both these granularities of categorization, using all the features or combinations of different feature sets does not perform as well as using the Syntactic Context features on their own. This appears to be due to data sparsity and overfitting. The accuracies on the *training data* (not shown) are considerably higher using greater numbers of features, but those results to not generalize well to the evaluation data. With the fine categorization the class sparsity is considerable with 9 categories of them having fewer than 15 occurrences in the *training* data.

While there remains considerable room for improvement, the inter-annotator agreement (among two annotators) within the PDTB was 92% at the coarse level and just 77% at the fine level of classification [Miltsakaki et al., 2008]. Further, these agreement numbers do not include agreement on identifying relations as Other (i.e., NoRel or EntRel), thus the agreement numbers are likely somewhat lower for the classification tasks put forward here.

| Relation Type | Prec. | Rec. | F1 | Count |
|---|---|---|---|---|
| Comparison.Contrast | 90.75 | 60.65 | 72.71 | 615 |
| Comparison.Other | 32.76 | 37.25 | 34.86 | 51 |
| Contingency.Cause.Reason | 74.17 | 30.58 | 43.31 | 291 |
| Contingency.Cause.Result | 94.23 | 23.44 | 37.55 | 209 |
| Contingency.Condition | 94.85 | 79.31 | 86.38 | 116 |
| Expansion.Conjunction | 76.02 | 72.14 | 74.03 | 646 |
| Expansion.Instantiation | 96.88 | 21.53 | 35.23 | 144 |
| Expansion.Other | 92.31 | 24.24 | 38.40 | 99 |
| Expansion.Restatement | 22.81 | 26.74 | 24.62 | 273 |
| Temporal.Asynchronous.Precedence | 89.86 | 72.09 | 80.00 | 86 |
| Temporal.Asynchronous.Succession | 98.33 | 54.63 | 70.24 | 108 |
| Temporal.Synchronous | 54.26 | 87.18 | 66.89 | 117 |
| Other | 35.17 | 94.96 | 51.33 | 417 |

Table 8.3: Precision, recall and f-measure scores for assigning semi-coarse relation types to explicit and implicit connectives using the Syntactic Context features.

Beyond looking at overall classification accuracies it is insightful to examine precision, recall and f-measure results for each individual category. Tables 8.2, 8.3 and 8.4 show the per-category results for coarse, semi-coarse and fine categorizations, respectively. The counts in the test data are also provided. Clearly, the fine categorization suffers from class sparsity and performance for each category tends to correlate with the number of instances of that category in the data. Another clear difficulty apparent from these results is that the model is having a difficult time distinguishing Other relations — i.e., NoRel and EntRel — from actual discourse relations.

## 8.3.4 Explicit vs. Implicit Connectives

Identifying the rhetorical type of a relation is very difficult when an explicit discourse connective is not present. Table 8.5 provides a breakdown of results for labeling explicit

| Relation Type | Prec. | Rec. | F1 | Count |
|---|---|---|---|---|
| Comparison | 100.0 | 0.0 | 0.0 | 4 |
| Comparison.Conecssion.Contra-expection | 20.69 | 26.09 | 23.08 | 23 |
| Comparison.Concession.Expectation | 13.16 | 26.32 | 17.54 | 19 |
| Comparison.Contrast | 87.96 | 59.47 | 70.96 | 528 |
| Comparison.Contrast.Juxtaposition | 31.58 | 18.75 | 23.53 | 64 |
| Comparison.Contrast.Opposition | 0.0 | 0.0 | 0.0 | 23 |
| Comparison.Contrast.Pragmatic-concession | 100.0 | 0.0 | 0.0 | 5 |
| Contingency.Cause.Reason | 38.11 | 48.45 | 42.66 | 291 |
| Contingency.Cause.Result | 94.23 | 23.44 | 37.55 | 209 |
| Contingency.Condition.Factual-past | 100.0 | 0.0 | 0.0 | 2 |
| Contingency.Condition.Factual-present | 100.0 | 0.0 | 0.0 | 8 |
| Contingency.Condition.General | 100.0 | 0.0 | 0.0 | 24 |
| Contingency.Condition.Hypothetical | 52.58 | 98.08 | 68.46 | 52 |
| Contingency.Condition.Unreal-past | 100.0 | 0.0 | 0.0 | 2 |
| Contingency.Condition.Unreal-present | 100.0 | 0.0 | 0.0 | 14 |
| Contingency.Pragmatic-cause.Justification | 100.0 | 0.0 | 0.0 | 6 |
| Contingency.Pragmatic-cond.Implicit-assertion | 100.0 | 0.0 | 0.0 | 3 |
| Contingency.Pragmatic-cond.Relevance | 100.0 | 0.0 | 0.0 | 5 |
| Expansion | 100.0 | 0.0 | 0.0 | 4 |
| Expansion.Alternative | 100.0 | 0.0 | 0.0 | 9 |
| Expansion.Alternative.Chosen-alternative | 100.0 | 0.0 | 0.0 | 18 |
| Expansion.Alternative.Conjunctive | 13.33 | 100.0 | 23.53 | 2 |
| Expansion.Alternative.Disjunctive | 100.0 | 20.0 | 33.33 | 5 |
| Expansion.Conjunction | 73.86 | 72.60 | 73.22 | 646 |
| Expansion.Exception | 100.0 | 100.0 | 100.0 | 1 |
| Expansion.Instantiation | 96.88 | 21.53 | 35.23 | 144 |
| Expansion.List | 66.67 | 3.33 | 6.35 | 60 |
| Expansion.Restatement | 100.0 | 0.0 | 0.0 | 22 |
| Expansion.Restatement.Equivalence | 100.0 | 0.0 | 0.0 | 21 |
| Expansion.Restatement.Generalization | 100.0 | 0.0 | 0.0 | 12 |
| Expansion.Restatement.Specification | 100.0 | 0.0 | 0.0 | 218 |
| Temporal.Asynchronous.Precedence | 89.86 | 72.09 | 80.00 | 86 |
| Temporal.Asynchronous.Succession | 96.72 | 54.63 | 69.82 | 108 |
| Temporal.Synchronous | 54.26 | 87.18 | 66.89 | 117 |
| Other | 35.17 | 94.96 | 51.33 | 417 |

Table 8.4: Precision, recall and f-measure scores for assigning fine relation types to explicit and implicit connectives using the Syntactic Context features.

| Connectives | Coarse | Semi-coarse | Fine |
|---|---|---|---|
| Explicit Only | 92.40 (35.84) | 84.18 (29.44) | 76.59 (29.44) |
| Implicit Only | 45.85 (37.40) | 31.03 (26.41) | 30.40 (26.41) |
| Explicit+Implicit (Separate) | 69.22 (36.63) | 57.77 (20.37) | 53.59 (20.37) |
| Explicit+Implicit (Joint) | 68.88 (36.63) | 57.85 (20.37) | 54.13 (20.37) |

Table 8.5: Coarse, semi-coarse and fine categorization results comparing separate classification of explicit and implicit connectives with joint classification using the best feature configuration for each categorization. Baseline class-majority results are indicated in parentheses.

and implicit connectives independently as well as jointly for both the coarse-grain and fine-grain categorizations. The Explicit Only and Implicit Only results were obtained by training separate CRFs over only explicit and only implicit connectives, respectively. Combining these accuracies arrives at the results in row labeled: Explicit+Implicit (Separate). The final row in the table shows the results using a single CRF across both explicit and implicit connectives.

Modeling explicit and implicit connectives separately seems to be beneficial since performance is roughly the same as modeling both connective types jointly, which would have the advantage of better contextual dependencies. For example, the most important dependencies identified in Pitler et al. [2008] were those where an implicit connective of type Contingency was found following an explicit connective of type Comparison. Further, 9 of the 10 most significant dependencies involved one explicit connective with an adjacent implicit one. This would indicate that contextual advantages can be best exploited in a model that handles both connective types jointly.

Handling the connective types separately, however, could benefit from further improvements provided by separate, customized feature sets. For example, the lexical features are likely to be much more useful for implicit connectives, whereas syntactic features are more useful for explicit ones. Also, however, having separate models allows for each to capture the intrinsic differences between explicit and implicit relations. Even though the features carefully conjoin the connective phrase with other features (e.g. an explicit connective or simply *implicit* for implicit connectives is conjoined with the syntactic path from the head of Arg1 to the head of Arg2), there are different class distributions associated with explicit and implicit connectives.

## 8.4 Future Directions

We have demonstrated some benefits to identifying rhetorical types for discourse relations *in context* — that is, taking into account how adjacent discourse relation types influence each other. However, as made clear by the results in the previous section, performance for identifying rhetorical types of implicit connectives, in particular, leaves much room for improvement.

As mentioned earlier, two previous approaches [Marcu & Echihabi, 2002; Sporleder & Lascarides, 2008] looked at bootstrapping classifiers to identify implicit discourse relations (i.e., those without a discourse "marker") by identifying unambiguous connectives in a large body of text, identifying the argument spans of each connective and creating a classification instance from the two spans with the label of the instance determined by the connective. Crucially, the discourse connective is not used as a source of features, so as to simulate an implicit discourse relation. As demonstrated in Sporleder & Lascarides

[2008], however, this approach falls down when one applies a classifier trained in this manner to *found* implicit discourse relations. Naturally occurring implicit relations are qualitatively different from explicit relations minus the discourse connective.

Despite these results, semi-supervised approaches seem key to improving performance for identifying implicit relations since performance is still poor even when training on the entire Discourse Treebank training set. Rather than labeling data using the trick above, however, an alternative approach in the context of the discourse parser in this dissertation would be to use self-training. This could work by applying the discourse parser in Chapter 7 along with the sense disambiguator discussed in this chapter to unannotated text. Since it is possible to extract probability scores for each extracted discourse relation and its assigned sense/type, it is possible to rank the extracted relations by the models' confidence. In the case of self-training, the extracted (and labeled) relations with high confidence scores can be used as new training exemplars. This process can be iterated with the goal of improving performance until some local optimum is converged upon. Self-training has been employed successfully in syntactic constituent parsing [McClosky et al., 2006], and could perhaps be employed not only for sense disambiguation, but for argument selection. Indeed, it has been shown that re-ranking models as we have utilized in this work, are especially able to benefit from self-training [McClosky et al., 2008]. Co-training [Blum & Mitchell, 1998] could be employed in a similar fashion. These methods could prove more successful than the unsupervised approaches described earlier since they avoid modifying the actual data, however they may be susceptible to other pitfalls associated with semi-supervised learning such as failing to identify a good local optimum.

Other future work could look at carefully constructing separate feature sets for explicit and implicit connectives. Further along these lines would be to introduce separate features for different explicit connectives. For example, the features most useful for disambiguating *when* may be different than those for disambiguating *indeed*.

Another avenue worth exploring is modeling the actual connective phrase for implicit connectives. Since the relation type can be identified quite reliably with a discourse connective, trying to either explicitly generate the discourse connective or modeling it as a hidden variable in a generative model may be one way to provide better performance. Experimenting with other discriminative classifiers such as Support Vector Machines may also improve performance. Kernel-based learning methods, in particular, could be beneficial since designing features and appropriate feature combinations for this task is especially problematic; methods that implicitly explore the space of features may do a better job at discrimination.

Finally, jointly modeling relation type classification together with connective and argument identification may provide additional improvements. For example, certain relation types may be more or less likely to select the complement of an attribution-denoting verb. These regularities, to the extent that they exist, could jointly benefit both relation type and argument identification.

# Chapter 9

# Conclusions and Future Work

## 9.1 Summary

The main goal of this dissertation has been the development of a robust, machine-learning based system for identifying informational discourse relations in text as annotated in the Penn Discourse Treebank. The most novel contribution we have made is the development of a sequential ranking model based on Conditional Random Fields (CRFs) that makes use of non-factored feature functions. This model was applied to both discourse and syntactic parsing with promising results. The model is general, applicable to many NLP tasks that can be modeled sequentially (as discussed in more detail below) and is able to leverage existing and future developments within the growing body of work on sequential CRFs.

Additional accomplishments of this dissertation include: a dependency-based framework for parsing discourse; a novel approach to syntactic dependency parsing based on the sequential ranking model; analysis of increasingly sophisticated log-linear statistical models for deciding various sub-problems of the discourse parsing task; and a comprehensive analysis of various features and knowledge sources for the various parsing sub-tasks and their relative contributions to performance.

Along the way we have built a fully automated discourse parser based on the annotations found in the Penn Discourse Treebank, capable of identifying discourse relations and disambiguating their types. Our plan is to include the discourse parser as part of the Carafe open source toolkit[1] for statistical modeling of natural language in the near future to help accelerate the progress of research in discourse analysis as well as to enable applications that could make use of discourse parsing.

The results we have presented here further a trend within the computational linguistics community towards *global models* that 1) jointly model various sub-problems within a complex, multi-staged process, and 2) consider larger contexts within a single sub-problem in which decisions are made jointly with other "nearby" decisions. In this work, we have explored identifying both arguments jointly and also jointly identifying discourse connectives and their arguments as an instance of (1). We considered sequential dependencies between argument (and connective) decisions as an instance of (2). The results fairly clearly demonstrated that modeling the discourse parsing sub-tasks of connective and (joint) argument identification improves performance over carrying out such steps separately. Identifying discourse relations in sequence appears to offer promise for further improving performance, however the improvement over non-sequential methods using

[1]Carafe is available at: http://www.sourceforge.net/project/carafe.

the full set of features was only slight when automatic parses were used and non-existent with gold-standard parses.

Beyond discourse parsing, we established that sequential ranking can be applied in a novel fashion to the problem of syntactic dependency parsing. Ranking the candidate governors for each word in the sentence *sequentially* through the introduction of appropriate Markov feature improved performance, at a statistically significant level, over a model not capturing sequential dependencies.

Throughout this work we have paid careful attention to designing various feature sets and providing analysis as which sets of features are most promising. Feature engineering is a difficult and somewhat arduous process. Beyond the statistical models put forth, it is hoped that the features developed herein will provide useful starting points for additional work in discourse parsing. For example, Elwell & Baldridge [2008] have provided some improved results at argument identification building upon the features and approach in this work.

## 9.2 Future Directions

There are a number of directions for future work that we outline below.

### 9.2.1 Global Modeling

Whether and the degree to which global modeling helps, is something that will always require empirical verification. We demonstrated, for the most part, advantages to global modeling in the context of discourse parsing. However, as our work here has explored, there are many different architectures with various advantages and disadvantages along the spectrum of local-cascaded models to fully global-joint models. For example, we noticed here that the advantages of modeling sequential dependencies was greater when operating over sequences of “given” (or previously identified) discourse connectives rather than the sequence of candidate connectives. This indicates that it may be beneficial, in fact, to identify connectives up-front in a separate process so as to enable better modeling of argument dependencies between adjacent connectives, interesting area of future work to explore further.

A related idea is to move beyond sequential Markov type models for discourse as we have looked at here and consider longer-range dependencies. For example, the Arg1 for a given connective may exist in the neighborhood not of the immediately preceding connective (and its arguments), but of a connective occurring much earlier in the discourse. Interesting future research might consider methods for capturing these longer distance dependencies, such as skip-chain CRFs [Galley, 2006; Sutton & McCallum, 2007] or two-pass CRFs [Krishnan & Manning, 2006] — an advantage of our formulation of discourse parsing within CRFs, is the ability to leverage the considerable amount of existing and future work with CRFs for improvements in capturing such global dependencies. Another approach would be to avoid sequential models and explicitly consider richer discriminative models that capture arbitrary dependencies. Markov Logic Networks (MLNs) [Richardson & Domingos, 2006] are an interesting instance of such models, which allow the specification of dependencies between output variables as well as specification of features via first-order logic statements. One difficulty, however, is that selecting the arguments of discourse connectives seems more naturally formulated as a ranking problem, rather than multinomial classification as would be required with MLNs. Thus, extensions to the MLNs to handle ranking problems would be an interesting area to explore in this vein.

Rather than abandoning sequence methods altogether, yet another option would be to move to higher-order sequential CRFs that would allow features not over pairs of connectives (and their arguments), but triples or larger groupings of connectives. Data sparsity then becomes an issue. This could be mitigated, however, with careful feature engineering.

### 9.2.2 Sequential Ranking and Additional Applications

A final interesting extension, more in line with our current approach, would be to again use *re-ranking* — given the $k$-best *sequences* of connectives and their selected arguments, a re-ranking approach could consider features over entire sequences of connectives and their arguments with the aim of learning to identify the best sequence. Indeed, carrying our notion of sequential (re-)ranking to the next level, it would be possible to re-rank the arguments for sequences of connectives in sequence. That is, re-ranking the sequences of connectives and their arguments for a given paragraph[2] could be dependent on the

[2]Recall that we have identified the paragraph as the delimiting unit for sequences of connectives in this work.

candidate sequences in the previous and subsequent paragraph, allowing for even greater contextual dependencies to be modeled.

Many other complex NLP tasks could be modeled with the sequential ranking approach we have developed. For example, PropBank parsing could be approached by 1) ranking the arguments for each predicate *in sequence* and 2) re-ranking the arguments for a sequence of predicates (within a sentence, for example) taking into account dependencies between the arguments of adjacent predicates. This would amount to applying the same sequential ranking generalization in this work to previous work on re-ranking PropBank arguments [Toutanova et al., 2005]. Other semantic predicate-argument parsing tasks would be amenable to this architecture, such as FrameNet and NomBank parsing.

Co-reference is another problem that may be well-suited to this framework. For example, Denis & Baldridge [2007] approach the problem of anaphora resolution as a ranking problem — i.e., the potential antecedents are ranked (as we have done here for discourse connective argument identification). A natural extension of their approach would be to rank the antecedents for the *sequence* of pronouns (or other entity mentions). This model would be able to capture dependencies between adjacent pronouns and their candidate antecedents. For example, a subjective and objective pronoun within the same sentence (e.g. "He did not like him") would not refer to the same antecedent. These types of dependencies/constraints could be easily captured using the appropriate Markov features. This approach would occupy a middle ground between "local" approaches to co-reference that resolve each referent independently and approaches that consider the full transitive closure during inference and/or learning [McCallum & Wellner, 2003; Richardson & Domingos, 2006].

### 9.2.3 Argument Extents

In this dissertation, we have argued for a dependency representation of discourse as a way to avoid the difficult and less well-defined aspect of identifying discourse argument extents. While in many cases the extent can be retrieved accurately by following the syntactic descendents of the lexical head (in a dependency parse), there are cases where the extent is difficult to determine (e.g. whether to include a free adjunct). Further, syntactic parser errors will thwart accurate extraction of the extent. Measuring this is important, especially since the extents of discourse arguments are larger than those for the arguments in semantic role labeling, for example. Most importantly, requiring systems to identify the extent would provide a better means to compare different systems, since the argument extents are established in the PDTB (while lexical heads are derived and may vary depending on the heuristics used).

One additional point is that identifying extents may be useful as a source of additional features for argument selection. Recall that we introduced a class of Markov features called DOMINATES that determined whether the candidate argument for a connective $\pi_i$ was dominated by the candidate argument for an adjacent connective. As some discourse extents don't follow exactly from syntactic dominance, better modeling of argument extents could improve the quality of Markov features such as those in the

DOMINATES feature class. Relatedly, there may be other features of a candidate argument's proposed extent (e.g. its size) that could prove useful for argument selection.

Finally, as we observed earlier, identifying only the lexical head of a discourse argument is ambiguous in some cases. In particular, in cases of subordination and coordination, identifying the lexical head of the matrix clause and first conjunct, respectively, does not distinguish between whether the argument is *only* the matrix clause or first conjunct or whether the argument includes the entire sentence. Relatedly, some arguments span over multiple sentences and identifying the head only indicates where such argument extents *begin*.

### Towards Identifying Argument Extents

We outline here a few possible approaches for automatically identifying the arguments of discourse extents.

**Heuristic post-processing** The simplest method for identifying argument extents would be to use the existing system(s) put forth in this dissertation to recover the head-based discourse structure and identify the extents with a simple, heuristic post-process. For many ARG2 arguments, the argument extent is retrieved reliably by identifying all descendents of the identified argument head. The ARG1 argument is then identified by taking the descendents of the identified head *excluding the already identified* ARG2 *extent*. This is analogous to the tree-subtraction heuristic employed in Dinesh et al. [2005]. Manual inspection of a part of the development data indicates that the vast majority of argument extents for intra-sentential relations can be identified using this approach.

**Machine learning post-processing** The tree-subtraction approach breaks down when considering Arg1 arguments that lie in prior sentences within the document and when supplementary text is found (typically as parentheticals, appositional phrases and clauses, or free adjuncts) that is deemed unnecessary as a realization of the abstract object denoted by the argument extent. Supplementary text associated with arguments is annotated in the PDTB when it is relevant to the discourse but not "minimally necessary"; in other cases such supplementary text is not included at all as with the free adjunct below beginning with "which was..":

(36) *But a Soviet bank here would be crippled* unless **Moscow found a way to settle the $188 million debt**, which was lent to the country's short-lived democratic Kerensky government before the Communists seized power in 1917.

While some of these cases might be handled heuristically, identifying what to include as part of the argument in cases like the above appears to be a major source of inter-annotator disagreement, as mentioned earlier and noted in Miltsakaki et al. [2004b].

There are at least two ways to form the argument identification task as a machine learning problem. The first approach would be to identify, through heuristic means, a set of reasonable candidate argument extents (again, assuming the existing head-based discourse structure is provided as an input) and use a ranking or classification approach to select the best extent[3]. An alternative framework would be to view the argument extraction task as a *text compression* task [Clarke & Lapata, 2006]. Such an approach could work

[3]This kind of approach has been applied in the context of Semantic Role Labeling.

by first identifying the maximum possible (or maximum most likely) argument extent, which would typically consist of all descendents of the head in the dependency graph ( this could include subsequent sentences when considering the augmented discourse dependency graph that forms a directed link between the root word of a sentence and the subsequent sentence). Given the maximal extent, the argument extent could be extracted using sentence/text compression techniques. Briefly, these techniques take a sentence (or other unit of text) and determine which portions to elide such that the remaining text is both coherent and contains relevant information. This approach would allow for identification of disjoint argument extents and likely be more robust than the candidate selection approach in the face of syntactic parser errors.

**Joint identification of discourse Arguments, rhetorical types and extents** Identifying argument extents jointly with the other aspects of discourse parsing we have addressed in this dissertation may offer improved performance over carrying out this stage independently. This could be carried out by identifying for each candidate head, the top $k$ candidate extents for that head which could be achieved with the candidate selection method above or text compression methods that can identify the top $k$ compressions. The set of candidates for each argument would thus be $Nk$ where $N$ is the number of candidate heads and $k$ the number of candidate extents for each head. The argument extent appears to be useful for identifying rhetorical types (e.g. certain rhetorical types are correlated with larger or smaller argument extents). Considering extents may also help identify the arguments for certain adverbial connectives such that are strongly correlated with particular rhetorical types. For example, *instead* frequently has a negated predicate or quantifier over the scope of it's argument that could be identified by determining the scope of the candidate argument extent. The extent to which having access to the candidate arguments potential extents improves argument selection is the prime determining factor for favoring a fully joint model over a cascaded model in this context.

### 9.2.4 Additional Features and Finer-grained Analyses

In this work, we have approached the discourse parsing problem by employing large numbers of features whose weights are determined by machine learning methods. For a given task (e.g. argument identification), we have, for the most part, employed the same set of features for all instances of that task. Many of these instances can be grouped into different classes, such as the arguments of different connective types (e.g. coordinating connectives vs. adverbials). Finer-grained classifications are also possible. For example, adverbial connectives such as *instead* or *in fact* behave quite differently from connectives such as *nevertheless* or *moreover*. A systematic analysis that groups these different categories and then leverages *different* sets of features for each group may provide accuracy improvements for identifying discourse arguments. Similarly, this approach could benefit rhetorical type identification.

Beyond different sets of features, a more detailed set of heuristics could benefit argument identification or rhetorical type identification since, in some cases, high precision rules are fairly obvious. These heuristics could be utilized in a hybrid system that identifies certain cases using heuristics and others using machine learning methods; they could also serve as a new set of richer features; and finally, their evaluation provides a clearer indication of the phenomena involved for identifying discourse relations than using machine learning methods in which it is somewhat difficult to determine the contribution of linguistic features due to the lack of statistical independence among the features.

### 9.2.5 Employing Richer Semantics

We have explored discourse parsing from the task of connective identification to the task of relation type disambiguation using a number of linguistically motivated, yet largely surface-based, features. As mentioned in Chapter 8, we made some use of WordNet to help reduce lexical sparsity for identifying relation types. We found this did not improve performance, however, quite possibly due to naïvely using just the most frequent WordNet sense for each lexical item, instead of carrying out word sense disambiguation.

Clearly, however, lexical semantics is important for properly identifying discourse relation types and their semantics [Asher & Lascarides, 2003]. Beyond using lexical knowledge to simply reduce data sparsity, explicitly taking into account lexical aspect and *event structure* [Pustejovsky, 1995] also seems important for constraining the discourse. For example, causatives and resultatives may provide strong cues for causation discourse relations. Also, logically polysemous constructions [Pustejovsky & Bouillon, 1995] with aspectual predicates such as *begin* as in

(37) He began the trip.

may, for example, prime the discourse for subsequent Elaboration-type relations since the predicate selected by the aspectual predicate is elided. Indeed, coercions in general may be correlated with particular types of coherence relations.

Another important source of information about coherence relations involves entity-level co-reference. There appear to be clear correlations between entity-level coherence and particular relation types such as Elaboration. Such information may help to not only disambiguate relation types but also serve to help identify long-distance ARG1 arguments. For example, the ARG1 of the connective *also* frequently involves the same entity as described in its ARG2.

An interesting area for future work would be to leverage existing and emerging linguistic resources such as Ontonotes [Hovy et al., 2006] which annotates the Penn Treebank with word sense information and co-reference. The PropBank and NomBank resources could also be leveraged. NomBank could provide useful information about nominalizations while certain PropBank ARGM arguments, in particular, overlap with the discourse annotations in the PDTB. Beyond using these as sources of features, jointly modeling PropBank and/or NomBank arguments and discourse arguments may be fruitful. The higher density of arguments for these tasks combined might provide a better setting for the sequence models we have put forward here to improve performance. Annotations conforming to the emerging Generative Lexicon Mark-up Language (GLML) [Pustejovsky et al., 2008] could provide important information regarding lexical aspect and coercions that aids in parsing discourse. As automating these levels of analyses in these corpora are difficult tasks themselves, establishing which analyses have the greatest utility for discourse parsing using gold-standard annotations is important for making additional progress in automated discourse parsing by focusing development on important enabling components.

Integrating and modeling the interaction of multiple levels of linguistic annotation and resulting automatic analyses is a promising trend in computational linguistics. Rich learning architectures, such as those advocated in this dissertation, will be required to exploit these interactions to their fullest.

# Appendix A

# Argument Identification Errors Per Connective

This appendix provides a detailed listing of the the accuracies per explicit connective phrase on the evaluation data for argument identification. These accuracies correspond to the aggregate results in Table 4.3 .

| | | | |
|---|---|---|---|
| 1.000000 | 38 / 38 | If | (subord) |
| 1.000000 | 10 / 10 | or | (coord) |
| 1.000000 | 5 / 5 | After | (subord) |
| 1.000000 | 5 / 5 | thus | (disadv) |
| 1.000000 | 4 / 4 | still | (disadv) |
| 1.000000 | 4 / 4 | Or | (coord) |
| 1.000000 | 3 / 3 | largely because | (subord) |
| 1.000000 | 3 / 3 | even after | (subord) |
| 1.000000 | 3 / 3 | just as | (subord) |
| 1.000000 | 3 / 3 | Even if | (subord) |
| 1.000000 | 3 / 3 | Since | (subord) |
| 1.000000 | 2 / 2 | on the other hand | (disadv) |
| 1.000000 | 2 / 2 | in part because | (subord) |
| 1.000000 | 2 / 2 | specifically | (disadv) |
| 1.000000 | 2 / 2 | Nonetheless | (disadv) |
| 1.000000 | 2 / 2 | now that | (disadv) |
| 1.000000 | 2 / 2 | so that | (disadv) |
| 1.000000 | 2 / 2 | if then | (coord) |
| 1.000000 | 2 / 2 | Besides | (disadv) |
| 1.000000 | 2 / 2 | Before | (subord) |

| | | | | |
|---|---|---|---|---|
| 1.000000 | 2 / 2 | once | (subord) |
| 1.000000 | 1 / 1 | just eight days before | (subord) |
| 1.000000 | 1 / 1 | presumably because | (subord) |
| 1.000000 | 1 / 1 | particularly after | (subord) |
| 1.000000 | 1 / 1 | just a month after | (subord) |
| 1.000000 | 1 / 1 | especially because | (subord) |
| 1.000000 | 1 / 1 | A few hours after | (subord) |
| 1.000000 | 1 / 1 | especially after | (subord) |
| 1.000000 | 1 / 1 | perhaps because | (subord) |
| 1.000000 | 1 / 1 | especially when | (subord) |
| 1.000000 | 1 / 1 | On the contrary | (disadv) |
| 1.000000 | 1 / 1 | Simultaneously | (disadv) |
| 1.000000 | 1 / 1 | In other words | (disadv) |
| 1.000000 | 1 / 1 | shortly after | (subord) |
| 1.000000 | 1 / 1 | especially if | (subord) |
| 1.000000 | 1 / 1 | nevertheless | (disadv) |
| 1.000000 | 1 / 1 | consequently | (disadv) |
| 1.000000 | 1 / 1 | Additionally | (disadv) |
| 1.000000 | 1 / 1 | not because | (subord) |
| 1.000000 | 1 / 1 | nonetheless | (disadv) |
| 1.000000 | 1 / 1 | long before | (subord) |
| 1.000000 | 1 / 1 | if and when | (subord) |
| 1.000000 | 1 / 1 | by contrast | (disadv) |
| 1.000000 | 1 / 1 | Even though | (subord) |
| 1.000000 | 1 / 1 | as long as | (subord) |
| 1.000000 | 1 / 1 | otherwise | (disadv) |
| 1.000000 | 1 / 1 | only when | (subord) |
| 1.000000 | 1 / 1 | even when | (subord) |
| 1.000000 | 1 / 1 | either or | (coord) |
| 1.000000 | 1 / 1 | as though | (subord) |
| 1.000000 | 1 / 1 | Similarly | (disadv) |
| 1.000000 | 1 / 1 | Just when | (subord) |
| 1.000000 | 1 / 1 | moreover | (disadv) |
| 1.000000 | 1 / 1 | whereas | (subord) |
| 1.000000 | 1 / 1 | only if | (subord) |
| 1.000000 | 1 / 1 | much as | (subord) |
| 1.000000 | 1 / 1 | in fact | (disadv) |
| 1.000000 | 1 / 1 | even if | (subord) |
| 1.000000 | 1 / 1 | even as | (subord) |
| 1.000000 | 1 / 1 | by then | (coord) |
| 1.000000 | 1 / 1 | If only | (disadv) |
| 1.000000 | 1 / 1 | Even as | (subord) |
| 1.000000 | 1 / 1 | unless | (subord) |
| 1.000000 | 1 / 1 | rather | (disadv) |
| 1.000000 | 1 / 1 | except | (disadv) |
| 1.000000 | 1 / 1 | Rather | (disadv) |
| 1.000000 | 1 / 1 | hence | (disadv) |
| 1.000000 | 1 / 1 | as if | (subord) |
| 1.000000 | 1 / 1 | Later | (disadv) |
| 1.000000 | 1 / 1 | Hence | (disadv) |
| 1.000000 | 1 / 1 | yet | (coord) |
| 0.972222 | 35 / 36 | while | (subord) |
| 0.967742 | 30 / 31 | When | (subord) |
| 0.954545 | 21 / 22 | before | (subord) |
| 0.928571 | 26 / 28 | after | (subord) |
| 0.916667 | 11 / 12 | since | (subord) |
| 0.909804 | 232 /255 | and | (coord) |
| 0.909091 | 10 / 11 | for instance | (disadv) |
| 0.909091 | 10 / 11 | Although | (subord) |
| 0.900000 | 9 / 10 | As | (subord) |
| 0.892857 | 50 / 56 | as | (subord) |
| 0.877551 | 43 / 49 | if | (subord) |
| 0.866667 | 52 / 60 | because | (subord) |
| 0.858407 | 97 /113 | but | (coord) |

```
0.857143            6  /  7          even though     (subord)
0.851064           40  / 47                 when     (subord)
0.846154           11  / 13             although     (subord)
0.800000           12  / 15                While     (subord)
0.800000           12  / 15                   so     (subord)
0.800000            4  /  5          As a result     (disadv)
0.800000            4  /  5               Though     (subord)
0.800000            4  /  5                  nor     (coord)
0.750000            3  /  4           previously     (disadv)
0.750000            3  /  4              Because     (subord)
0.714286            5  /  7                until     (subord)
0.692308            9  / 13                Still     (disadv)
0.684211           13  / 19                 then     (coord)
0.666667            4  /  6              In fact     (disadv)
0.666667            2  /  3              earlier     (disadv)
0.625000            5  /  8                 Then     (coord)
0.611940           41  / 67                  And     (coord)
0.605714          106  /175                  But     (disadv)
0.570175           65  /114                 also     (disadv)
0.545455            6  / 11               though     (subord)
0.533333            8  / 15                 Also     (disadv)
0.500000            7  / 14              However     (disadv)
0.500000            4  /  8          for example     (disadv)
0.500000            4  /  8          For example     (disadv)
0.500000            2  /  4         For instance     (disadv)
0.500000            2  /  4             Moreover     (disadv)
0.500000            2  /  4                later     (disadv)
0.500000            1  /  2       partly because     (subord)
0.500000            1  /  2          as a result     (disadv)
0.500000            1  /  2           as soon as     (subord)
0.500000            1  /  2            therefore     (disadv)
0.500000            1  /  2              Instead     (disadv)
0.500000            1  /  2                 Once     (subord)
0.500000            1  /  2                  Yet     (coord)
0.400000            8  / 20          In addition     (disadv)
0.363636            4  / 11         Nevertheless     (disadv)
0.363636            4  / 11            Meanwhile     (disadv)
0.333333            3  /  9                 Thus     (disadv)
0.333333            2  /  6            meanwhile     (disadv)
0.323529           11  / 34              however     (disadv)
0.250000            2  /  8               Indeed     (disadv)
0.250000            2  /  8                   So     (subord)
0.250000            1  /  4           Separately     (disadv)
0.000000            0  /  3              instead     (disadv)
0.000000            0  /  2          Accordingly     (disadv)
0.000000            0  /  1On the one hand On the other hand     (disadv)
0.000000            0  /  1     two months before     (subord)
0.000000            0  /  1      On the other hand     (disadv)
0.000000            0  /  1       in the meantime     (disadv)
0.000000            0  /  1       In the meantime     (disadv)
0.000000            0  /  1       simultaneously     (disadv)
0.000000            0  /  1        By comparison     (disadv)
0.000000            0  /  1         Specifically     (disadv)
0.000000            0  /  1          accordingly     (disadv)
0.000000            0  /  1          Furthermore     (disadv)
0.000000            0  /  1          By contrast     (disadv)
0.000000            0  /  1             Meantime     (disadv)
0.000000            0  /  1              thereby     (disadv)
0.000000            0  /  1              in turn     (disadv)
0.000000            0  /  1              finally     (disadv)
0.000000            0  /  1              Overall     (disadv)
0.000000            0  /  1              Earlier     (disadv)
0.000000            0  /  1               indeed     (disadv)
0.000000            0  /  1                 Next     (disadv)
```

# Appendix B

# Log-linear Ranking Model Derivations

Let $Y_i = \{y_{i,1}, y_{i,2}, ..., y_{i,m}\}$ be a set of instances to rank. The conditional probability for an instance, $y_{i,j} \in Y_i$ is:

$$P_\Lambda(y_{i,j}|Y_i) = \frac{\exp(\sum_k \lambda_k f_k(y_{i,j}, Y_i))}{\sum_{y_{i,j} \in Y_i} \exp(\sum_k \lambda_k f_k(y_{i,j}, Y_i))}$$

Our training set consists of a set of instance sets:

$$\mathcal{D} = \{\langle Y_1, \tilde{P}_1(Y_1)\rangle, \langle Y_2, \tilde{P}_2(Y_2)\rangle, ..., \langle Y_n, \tilde{P}_n(Y_n)\rangle\}$$

where associated with each instance set, $Y_i$, is a probability distribution over that set, $\tilde{P}_i$.

The weights in a ranking model are learned by maximizing the conditional log-likelihood:

$$L_\Lambda(\mathcal{D}) = \sum_{i=1}^{|\mathcal{D}|} \sum_{j=1}^{|Y_i|} \tilde{P}_i(y_{i,j}) \log P_\Lambda(y_{i,j}|Y_i) \tag{B.1}$$

This equation is maximized when the conditional probabilities according to the model, $P_\Lambda$ most closely match the empirical distributions, $\tilde{P}_i$.

Unfortunately, finding the set of parameters, $\Lambda$, that maximize the conditional log-likelihood can't be solved in closed form. Moreover, for many practical problems (especially in NLP) the number of features can be in the hundreds of thousands or even millions. The conditional log-likelihood function is, however, convex and thus amenable to a variety of convex optimization methods. In general, these methods require at each iteration an evaluation of the function being optimized (with the current set of input values), in this case the conditional log-likelihood B.1. Also required is the gradient of the function at the input values — i.e., at the current parameter values, $\Lambda$.

For the conditional log-likelihood, the gradient takes on an intuitive form: each component $\frac{\partial L_\Lambda \mathcal{D}}{\partial \lambda_i}$ is simply the difference between the empirically measured expected value of the corresponding feature, $f_i$, according to the training data and the expected value according to the model, $P_\Lambda$.

$$\frac{\partial L_\Lambda \mathcal{D}}{\partial \lambda_i} = \tilde{E}(f_i) - E_\Lambda(f_i) = \sum_{i=1}^{|\mathcal{D}|} \sum_{j=1}^{|Y_i|} (\tilde{P}_i(y_{i,j}) - P_\Lambda(y_{i,j}|Y_i)) f_i \tag{B.2}$$

To derive the gradient of the conditional log-likelihood, let us first rewrite the conditional log-likelihood function as follows:

$$\begin{aligned}
L_\Lambda(\mathcal{D}) &= \sum_{i=1}^{|\mathcal{D}|}\sum_{j=1}^{|Y_i|} \tilde{P}_i(y_{i,j}, Y_i) \log P_\Lambda(y_{i,j}|Y_i) \\
&= \sum_{i=1}^{|\mathcal{D}|}\sum_{j=1}^{|Y_i|} \tilde{P}_i(y_{i,j}, Y_i) \log\left(\frac{\exp\left(\sum_k \lambda_k f_k(y_{i,j}, Y_i)\right)}{\sum_m \exp\left(\sum_k \lambda_k f_k(y_{i,j}, Y_i)\right)}\right) \\
&= \sum_{i=1}^{|\mathcal{D}|}\sum_{j=1}^{|Y_i|} \tilde{P}_i(y_{i,j}, Y_i) \left(\sum_k \lambda_k f_k(y_{i,j}, Y_i) - \log\sum_{m=1}^{|Y_i|} \exp\left(\sum_k \lambda_k f_k(y_{i,m}, Y_i)\right)\right)
\end{aligned}$$

The gradient is obtained by taking the partial derivative of the conditional log-likelihood function with respect to the parameters $\lambda_i$.

$$\frac{\partial L_\Lambda(\mathcal{D})}{\partial \lambda_k} = \sum_{i=1}^{|\mathcal{D}|}\sum_{j=1}^{|Y_i|} \tilde{P}_i(y_{i,j}, Y_i) \left(f_k(y_{i,j}, Y_i) - \sum_{m=1}^{|Y_i|} \frac{\exp\left(\sum_k \lambda_k f_k(y_{i,m}, Y_i)\right) f_k(y_{i,m}, Y_i)}{\sum_n \exp\left(\sum_k \lambda_k f_k(y_{i,n}, Y_i)\right)}\right)$$

Note that the latter part of the second term in the above equation is arrived at by applying the chain rule twice, first with $\log(x)$ and then with $\exp(x)$ as follows:

$$\begin{aligned}
\frac{\partial}{\partial \lambda_k} \log\sum_m \exp\sum_k \lambda_k f_k(y_{i,m}, Y_i) &= \frac{\frac{\partial}{\partial \lambda_k}\sum_m \exp\left(\sum_k \lambda_k f_k(y_{i,m}, Y_i)\right)}{\sum_n \exp\left(\sum_k \lambda_k f_k(y_{i,n}, Y_i)\right)} \\
&= \frac{\sum_m \exp\left(\sum_k \lambda_k f_k(y_{i,m}, Y_i)\right) \frac{\partial}{\partial \lambda_k}\sum_k \lambda_k f_k(y_{i,m}, Y_i)}{\sum_n \exp\left(\sum_k \lambda_k f_k(y_{i,n}, Y_i)\right)} \\
&= \sum_{m=1}^{|Y_i|} \frac{\exp\left(\sum_k \lambda_k f_k(y_{i,m}, Y_i)\right) f_k(y_{i,m}, Y_i)}{\sum_n \exp\left(\sum_k \lambda_k f_k(y_{i,n}, Y_i)\right)}
\end{aligned}$$

Continuing now with the original gradient derivation, we have:

$$
\begin{aligned}
L_\Lambda(\mathcal{D}) &= \sum_{i=1}^{|\mathcal{D}|} \sum_{j=1}^{|Y_i|} \tilde{P}_i(y_{i,j}, Y_i) \left( f_k(y_{i,j}, Y_i) - \sum_{m=1}^{|Y_i|} f_k(y_{i,m}, Y_i) P_\Lambda(y_{i,m}|Y_i) \right) \\
&= \sum_{i=1}^{|\mathcal{D}|} \sum_{j=1}^{|Y_i|} \tilde{P}_i(y_{i,j}, Y_i) f_k(y_{i,j}, Y_i) - \sum_{i=1}^{|\mathcal{D}|} \sum_{j=1}^{|Y_i|} \tilde{P}_i(y_{i,j}, Y_i) \sum_{m=1}^{|Y_i|} f_k(y_{i,m}, Y_i) P_\Lambda(y_{i,m}|Y_i) \\
&= \sum_{i=1}^{|\mathcal{D}|} \sum_{j=1}^{|Y_i|} \tilde{P}_i(y_{i,j}, Y_i) f_k(y_{i,j}, Y_i) - \sum_{i=1}^{|\mathcal{D}|} \sum_{m=1}^{|Y_i|} f_k(y_{i,m}, Y_i) P_\Lambda(y_{i,m}|Y_i) \\
&= \tilde{E}(f_k) - E_\Lambda(f_k)
\end{aligned}
$$

Note that the term $\left(\sum_{j=1}^{|Y_i|} \tilde{P}_i(y_{i,j}, Y_i)\right)$ above sums to one and can be factored out since the index $j$ only appears within that term.

# Bibliography

Asher, N. (1993). *Reference to Abstract Objects in Discourse*. Kluwer Academic Publishers.

Asher, N., & Lascarides, A. (1995). Lexical disambiguation in a discourse context. *Journal of Semantics*, *12*(1), 69–108.

Asher, N., & Lascarides, A. (2003). *Logics of Conversation*. Cambridge University Press.

Baldridge, J., Asher, N., & Hunter, J. (2007). Annotation for a robust parsing of discourse structure on unrestricted texts. *Zeitschrift fur Sprachvissenschaft*, *26*, 213–239.

Baldridge, J., & Lascarides, A. (2005). Probabilistic head-driven parsing for discourse structure. In *Proceedings of the Ninth Conference on Computational Natural Language Learning CoNLL-2005*.

Bethard, S., Martin, J. H., & Klingenstein, S. (2007). Timelines from text: Identification of syntactic temporal relations. In *Proceedings of the International Conference on Semantic Computing*.

Bikel, D. (2004). Intricacies of Collins' parsing model. *Computational Linguistics*, *30*(4), 479–511.

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *COLT: Proceedings of the Workshop on Computational Learning Theory, Morgan Kaufmann Publishers*, (pp. 92–100).
URL `citeseer.comp.nus.edu.sg/73282.html`

Blunsom, P., & Cohn, T. (2006). Discriminative word alignment with conditional random fields. In *ACL '06: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, (pp. 65–72). Morristown, NJ, USA: Association for Computational Linguistics.

Buchholtz, S., & Marsi, E. (2006). The CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL)*.

Carlson, L., Marcu, D., & Okurowski, M. E. (2003). Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Current Directions in Discourse and Dialogue*, (pp. 85–112). Kluwer Academic Publishers.

Carreras, X. (2007). Experiments with a higher-order projective dependency parser. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, (pp. 957–961).
URL `http://www.aclweb.org/anthology/D/D07/D07-1101`

Carreras, X., & Marquez, L. (2005). Introduction to the conll-2005 shared task: Semantic role labeling.

Charniak, E., & Johnson, M. (2005). Coarse-to-fine n-best parsing and maxent reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*.

Clarke, J., & Lapata, M. (2006). Models for sentence compression: A comparison across domains, training requirements and evaluation measures. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, (pp. 377–384). Sydney, Australia.

Collins, M. (1999). *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.
URL `citeseer.ist.psu.edu/collins03headdriven.htmlc`

Collins, M. (2000). Discriminative reranking for natural language parsing. In *Proceedings of ICML-2000*.

Collins, M. (2002). Ranking algorithms for named-entity extraction: Boosting and the voted perceptron. In *In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL-02*, (pp. 489–496).

Danlos, L. (2004). Discourse dependency structures as constrained dags. In M. Strube, & C. Sidner (Eds.) *Proceedings of the 5th SIGdial Workshop on Discourse and Dialogue*, (pp. 127–135). Cambridge, Massachusetts, USA: Association for Computational Linguistics.

Danlos, L. (2006). Discourse verbs. In *Workshop on Constraints in Discourse*. Maynooth, Ireland.

Danlos, L. (2007). Integrating discourse relations into lexical semantics. In *Proceedings of the Fourth International Workshop on Generative Approaches to the Lexicon*.

Danlos, L. (2008). A discourse formalism using synchronous tag. In M. Aunargue, K. Korta, & J. Lazzarabal (Eds.) *Language, Representation and Reasoning*. University of Basque country Press.

de Marneffe, M.-C., MacCartney, B., & Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *In Proceedings of the 5th International Conference on Language Resources and Evaluation*. Genoa, Italy.

Denis, P., & Baldridge, J. (2007). A ranking approach to pronoun resolution. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*.

Dinesh, N., Lee, A., Miltsakaki, E., Prasad, R., Joshi, A., & Webber, B. (2005). Attribution and the (non)-alignment of syntactic and discourse arguments of connectives. In *Proceedings of the ACL Workshop on Frontiers in Corpus Annotation II: Pie in the Sky*. Ann Arbor, Michigan, USA.

Dowty, D. R., Wall, R. E., & Peters, S. (1981). Introduction to montague semantics. *Reidel*.

Elwell, R., & Baldridge, J. (2008). Discourse connective argument identification with connective specific rankers. In *The IEEE Conference on Semantic Computing*.

Forbes, K., Miltsakaki, E., Prasad, R., Sarkar, A., Joshi, A. K., & Webber, B. L. (2003). D-ltag system: Discourse parsing with a lexicalized tree-adjoining grammar. *Journal of Logic, Language and Information*, *12*(3), 261–279.

Galley, M. (2006). A skip-chain conditional random field for ranking meeting utterances by importance. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, (pp. 364–372). Sydney, Australia: Association for Computational Linguistics.

Gardent, C. (1997). Discourse tree adjoining grammars. Tech. Rep. 89, Saarbrücken, Saarbrücken, Germany.
URL `citeseer.ist.psu.edu/gardent98discourse.html`

Grimes, J. (1975). *The Thread of Discourse*. The Hague: Mouton.

Groenendijk, J., & Stokhof, M. (1991). Dynamic predicate logic. *Linguistics and Philosophy*, *14*, 39–100.

Hall, K. (2007). k-best spanning tree parsing. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*.

Halliday, M., & Hasan, R. (1976). *Cohesion in English*. Longman.

Hirst, G. (1981). Discourse-oriented anaphora resolution in natural language understanding: A review. *American Journal of Computational Linguistics*, *7*(2), 85–98.
URL `citeseer.ist.psu.edu/569719.html`

Hobbs, J. (1979). Coherence and coreference. *Cognitive Science*, *3*(1), 67–90.

Hobbs, J. (1985). On the coherence and structure of discourse. Tech. rep., Stanford University.

Hobbs, J. R., Stickel, M., Martin, P., & Edwards, D. D. (1988). Interpretation as abduction. In *26th Annual Meeting of the Association for Computational Linguistics: Proceedings of the Conference*, (pp. 95–103). Buffalo, New York.
URL `citeseer.ist.psu.edu/hobbs90interpretation.html`

Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., & Weischedel, R. (2006). Ontonotes: The 90% solution. In *Proceedings of HLT-NAACL 2006*, (pp. 57–60).

Hovy, E. H. (1993). Automated discourse generation using discourse structure relations. *Artificial Intelligence*, *63*(1-2), 341–385.
URL `citeseer.ist.psu.edu/hovy93automated.html`

Huang, H., Chang, Y., & Hsu, C. (2007). Training conditional random fields by periodic step size adaptation for large-scale text mining. In *Proceedings of the Seventh IEEE Conference on Data Mining*.

Jiang, Z. P., & Ng, H. T. (2006). Semantic role labeling of nombank: A maximum entropy approach. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006)*. Sydney, Australia.

Johansson, R., & Nugues, P. (2007). Extended constituent-to-dependency conversion for english. In *Proceedings of NODALIDA 2007*.

Kamp, H. (1981). A theory of truth and semantic representation. In J. Groenendijk, T. Janssen, & M. Stockhof (Eds.) *Formal Methods in the Study of Language*, (pp. 277–322). Amsterdam.

Kamp, H., & Reyle, U. (1993). *From Discourse to Logic*. Kluwer, Dordrecht.

Knott, A. (1996). *A Data-Driven Methodology for Motivating a Set of Coherence Relations*. Ph.D. thesis, University of Edinburgh.

Krishnan, V., & Manning, C. D. (2006). An effective two-stage model for exploiting non-local dependencies in named entity recognition. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, (pp. 1121–1128). Morristown, NJ, USA: Association for Computational Linguistics.

Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, (pp. 282–289). Morgan Kaufmann, San Francisco, CA.
URL `citeseer.ist.psu.edu/lafferty01conditional.html`

Lapata, M., & Lascarides, A. (2006). Learning sentence-internal temporal relations. *Journal of Artificial Intelligence Research*, *27*, 85–117.

Lee, A., Prasad, R., Joshi, A., Dinesh, N., & Webber, B. (2006). Complexity of dependencies in discourse: Are dependencies in discourse more complex than in syntax. In *Proceedings of the 5th International Workshop on Treebanks and Linguistic Theories*. Prague, Czech Republic.

Lee, A., Prasad, R., Joshi, A., & Webber, B. (2008). Departures from tree structures in discourse: Shared arguments in the penn discourse treebank. In *Proceedings of the Constraints in Discourse III Workshop*. Potsdam, Germany.

Litman, D. (1996). Cue phrase classification using machine learning. *Journal of Artificial Intelligence Research*, *5*, 53–94.
URL `http://www.cs.washington.edu/research/jair/abstracts/litman96a.html`

Magerman, D. M. (1994). *Natural Language Parsing as Statistical Pattern Recognition*. Ph.D. thesis, Stanford University.

Mani, I., & Pustejovsky, J. (2004). Temporal discourse models for narrative structure. In *In: Proceedings of the ACL Workshop on Discourse Annotation. East Stroudsburg (PA), Association for Computational Linguistics*.

Mani, I., Verhagen, M., Wellner, B., Lee, C. M., & Pustejovsky, J. (2006). Machine learning of temporal relations. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, (pp. 753–760). Morristown, NJ, USA: Association for Computational Linguistics.

Mann, W. C., & Thompson, S. A. (1988). Rhetorical structure theory: Toward a functional theory of text organization. *Text*, *8*(3), 243–281.

Marcu, D. (1998). Improving summarization through rhetorical parsing tuning. In *The Sixth Workshop on Very Large Corpora*, (pp. 206–215). Montreal, Canada.

Marcu, D. (1999a). A decision-based approach to rhetorical parsing. In *In the 37th Meeting of the Association for Computational Linguistics (ACL '99)*. Maryland.

Marcu, D. (1999b). Discourse trees are good indicators of importance in text. In *Advances in Automatic Text Summarization*, (pp. 123–136). The MIT Press.

Marcu, D., & Echihabi, A. (2002). An unsupervised approach to recognizing discourse relations. In *Proceedings of the Association for Computational Linguistics (ACL-2002)*.

Marcus, M., Santorini, B., & Marcinkiewicz, M. A. (1993). Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, *19*, 313–330.

Martin, J. R. (1992). *English Text: System and Structure*. Amsterdam and Philadelphia: John Benjamins.

McCallum, A., Bellare, K., & Pereira, F. C. N. (2005). A conditional random field for discriminatively-trained finite-state string edit distance. In *UAI*, (pp. 388–395). AUAI Press.

McCallum, A., & Wellner, B. (2003). Toward conditional models of identity uncertainty with application to proper noun coreference. In *Proceedings of the IJCAI Workshop on Information Integration and the Web*, (pp. 79–84).

McClosky, D., Charniak, E., & Johnson, M. (2006). Reranking and self-training for parser adaptation. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, (pp. 337–344). Morristown, NJ, USA: Association for Computational Linguistics.

McClosky, D., Charniak, E., & Johnson, M. (2008). When is self-training effective for parsing? In *Proceedings of the International Conference on Computational Linguistics (COLING 2008)*. Manchester, UK.

McDonald, R. (2006). *Discriminative Learning and Spanning Tree Algorithms for Dependency Parsing*. Ph.D. thesis, University of Pennsylvania.

McDonald, R., Lerman, K., & Pereira, F. (2006). Multilingual dependency analysis with a two-stage discriminative parser. In *Proceedings of the Conference on Natural Language Learning (CoNLL)*.

McDonald, R., & Pereira, F. (2006). Online learning of approximate dependency parsing algorithms. In *Proceedings of the European Association for Compuational Lingistics*.

Meyers, A., Reeves, R., Macleod, C., Szekely, R., Zielinska, V., Young, B., & Grishman, R. (2004). Annotating Noun Argument Structure for NomBank. In *Proceedings of LREC-2004*. Lisbon, Portugal.

Miltsakaki, E., Dinesh, N., Prasad, R., Joshi, A., & Webber, B. (2005). Experiments on sense annotations and sense disambiguation of discourse connectives. In *Proceedings of the Fourth Workshop on Treebanks and Linguistic Theories*. Barcelona, Spain.

Miltsakaki, E., Prasad, R., Joshi, A., & Webber, B. (2004a). Annotating discourse connectives and their arguments. In *Proceedings of the HLT/NAACL Workshop on Frontiers in Corpus Annotation*.

Miltsakaki, E., Prasad, R., Joshi, A., & Webber, B. (2004b). The penn discourse treebank. In *LREC 2004*.

Miltsakaki, E., Robaldo, L., Lee, A., & Joshi, A. (2008). Sense annotation in the penn discourse treebank. In A. Gelbukh (Ed.) *Computational Linguistics and Intelligent Text Processing*, (pp. 275–286). Springer Berlin / Heidelberg.

Minkov, E., Wang, R. C., Tomasic, A., & Cohen, W. W. (2006). Ner systems that suit user's preferences: Adjusting the recall-precision trade-off for entity extraction. In *Proceedings of HLT-NAACL*.

Montague, R. (1974). Formal philosophy: Selected papers of Richard Montague. Edited with an introduction by R. H. Thomason.

Moore, J. D., & Pollack, M. E. (1992). A problem for RST: The need for multi-level discourse analysis. *Computational Linguistics*, *18*(4), 537–544.
URL `citeseer.ist.psu.edu/article/moore92problem.html`

Nivre, J. (2003). An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT 03)*, (pp. 149–160).

Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., & Yuret, D. (2007a). The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, (pp. 915–932).
URL `http://www.aclweb.org/anthology/D/D07/D07-1096`

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., & Marsi, E. (2007b). Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, *13*(2), 95–135.

Nocedal, J., & Wright, S. J. (1999). *Numerical Optimization*. Springer.

Palmer, M., Kingsbury, P., & Gildea, D. (2005). The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, *31*(1), 71–106.

Peng, F., & McCallum, A. (2004). Accurate information extraction from research papers using conditional random fields. In *HLT-NAACL*, (pp. 329–336).

Pitler, E., Raghupathy, M., Mehta, H., Nenkova, A., Lee, A., & Joshi, A. (2008). Easily identifiable discourse relations. In *Proceedings of the 22nd International Conference on Computational Linguistics*.

Polanyi, L., & Scha, R. (1984). A syntactic approach to discourse semantics. In *Proceedings of the 10th international conference on Computational linguistics*, (pp. 413–419). Morristown, NJ, USA: Association for Computational Linguistics.

Polanyi, L., & van der Berg, M. (1999). Logical structure and discourse anaphora resolution. In *Proceedings of the Workshop on The Relation of Discourse/Dialogue Structure and Reference*.

Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A., & Webber, B. (2008). The penn discourse treebank 2.0. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*.

Prasad, R., Miltsakaki, E., Dinesh, N., Lee, A., Joshi, A., & Webber, B. (2006). The penn discourse treebank 1.0 annotation manual. Tech. rep., Institute for Research in Cognitive Science, University of Pennsylvania. IRCS Technical Report IRCS-06-01.

Pustejovsky, J. (1995). *The Generative Lexicon*. Cambridge, MA: MIT Press.

Pustejovsky, J., & Bouillon, P. (1995). Logical polysemy and aspectual coercison. *Journal of Semantics*, *12*, 133–162.

Pustejovsky, J., Hanks, P., Saurí, R., See, A., Gaizauskas, R., Setzer, A., Radev, D., Sundheim, B., Day, D., Ferro, L., & Lazo, M. (2003). The timebank corpus. In *Proceedings of Corpus Linguistics*, (pp. 647–656).

Pustejovsky, J., Moszkowicz, J. L., Batuikova, O., & Rumshisky, A. (2008). Glml: Annotating argument selection and coercion. In *Proceedings of the Eighth International Conference on Computational Semantics*.

Ravichandran, D., Hovy, E., & Och, F. J. (2003). Statistical qa - classifier vs. re-ranker: What's the difference? In *Proceedings of the ACL Workshop on Multilingual Summarization and Question Answering-Machine Learning and Beyond*.

Richardson, M., & Domingos, P. (2006). Markov logic networks. *Machine Learning*, *62*, 107–136.

Scha, R., & Polanyi, L. (1988). An augmented context free grammar for discourse. In *Proceedings of the 12th conference on Computational linguistics*, (pp. 573–577). Morristown, NJ, USA: Association for Computational Linguistics.

Sha, F., & Pereira, O. (2003). Shallow parsing with conditional random fields. In *In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL-03*, (pp. 213–220).

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

Sporleder, C., & Lapata, M. (2005). Discourse chunking and its application to sentence compression. In *Proceedings of the HLT/EMNLP*, (pp. 257–264). Vancouver.

Sporleder, C., & Lascarides, A. (2005). Exploiting linguistic cues to classify rhetorical relations. In *Proceedings of Recent Advances in Natural Langauge Processing (RANLP)*. Bulgaria.

Sporleder, C., & Lascarides, A. (2008). Using automatically labelled examples to classify rhetorical relations: A critical assessment. *Natural Language Engineering*, *14*(03).

Sundheim, B. M. (1992). Overview of the fourth message understanding evaluation and conference. In *MUC4 '92: Proceedings of the 4th conference on Message understanding*, (pp. 3–21). Morristown, NJ, USA: Association for Computational Linguistics.

Surdeanu, M., Johansson, R., Meyers, A., Marquez, L., & Nivre, J. (2008). The conll 2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of CoNLL 2008*.

Sutton, C., & McCallum, A. (2007). An introduction to conditional random fields for relational learning. In L. Getoor, & B. Taskar (Eds.) *Introduction to Statistical Relational Learning*. MIT Press.

Tarjan, R. (1972). Depth-first search and linear graph algorithms. *SIAM Journal on Computing*, *1*(2), 146–160.

Toutanova, K., Haghighi, A., & Manning, C. D. (2005). Joint learning improves semantic role labeling. In *Proceedings of the Association for Computational Linguistics(ACL)*. Ann Arbor, Michigan, USA.

Webber, B. (2006). Accounting for discourse relations: Constituency and dependency. *Intelligent Linguistic Architectures*, (pp. 339–360).

Webber, B., Joshi, A., Knott, A., & Stone, M. (2003). Anaphora and discourse structure. *Computational Linguistics*.

Wellner, B., Castaño, J., & Pustejovsky, J. (2005). Adaptive string similarity metrics for biomedical reference resolution. In *Proceedings of the ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics*, (pp. 9–16). Detroit: Association for Computational Linguistics.
URL `http://www.aclweb.org/anthology/W/W05/W05-1302`

Wellner, B., Huyck, M., Mardis, S., Aberdeen, J., Morgan, A., Peshkin, L., Yeh, A., Hitzeman, J., & Hirschman, L. (2007). Rapidly retargetable approaches to de-identification in medical records. *Journal of the American Medical Informatics Association*.
URL `http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=1975794`

Wellner, B., & Pustejovsky, J. (2007). Automatically identifying the arguments of discourse connectives. In *Proceedings of Emprical Methods in Natural Language Processing and the Conference on Natural Language Learning*. Prague, Czech Republic.

Wellner, B., Pustejovsky, J., Havasi, C., Sauri, R., & Rumshisky, A. (2006). Classification of discourse coherence relations: An exploratory study using multiple knowledge sources. In *SIGDIAL 2006*. Sydney, Australia.

Wellner, B., & Vilain, M. (2006). Leveraging machine-readable dictionaries in discriminative sequence models. In *In Proceedings of the Language Resources and Evaluation Conference (LREC)*. Genoa, Italy.

Wolf, F., & Gibson, E. (2005). Representing discourse coherence: A corpus-based analysis. *Computational Linguistics*, *31*(2), 249–287.