THE SEMANTICS OF ROLE LABELING

BY

VIVEK SRIKUMAR

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2013

Urbana, Illinois

Doctoral Committee:

Professor Dan Roth, Chair
Professor Gerald Dejong
Assistant Professor Julia Hockenmaier
Professor Martha Palmer, University of Colorado

Abstract

The problem of ascribing a semantic representation to text is an important one that can help text understanding problems like textual entailment. In this thesis, we address the problem of assigning a shallow semantic representation to text. This problem is traditionally studied in the context of verbs and their nominalizations. We propose to extend the task to go beyond verbs and nominalizations to include other linguistic constructions such as commas and prepositions

We develop an ontology of predicate-argument relations that commas and prepositions express in text. Just like the verb and nominal semantic role labeling schemes, the relations we propose are domain independent. For these two classes of phenomena, we introduce new corpora where these relations are annotated.

From the machine learning perspective, learning to predicting these relations is a structured learning problem. However, we only have the small (for commas) or partially annotated (for prepositions) data sets. To predict the new relations, we show that using linguistic knowledge and information about output structure can bias the learning to build robust models.

Finally, we observe that the relations expressed by the various phenomena interact with each other by constraining each others' output. We show that we can take advantage of these inter-dependencies by enforcing coherence between their predictions. By constraining inference using linguistic knowledge, we can improve relation prediction performance.

To my parents.

Acknowledgments

I should have started writing this part of my thesis when I started my research – I would have had the time to express my thanks to everyone who has helped me over the course of my Ph.D. What follows is a feeble attempt at doing so.

Dan Roth has been a perfect advisor and he never fails to amaze me. He has been played a big role shaping my thoughts as a researcher and a person. For the duration of my stay in grad school, Dan has steered my research and yet given me the intellectual freedom to pursue my interests.

I would also like to thank the other members of my doctoral committee. I am indebted to Julia Hockenmaier for the countless discussions about language, graduate school and research in general. I admire Jerry DeJong's ability to constantly keep the big picture in mind without missing the details and have enjoyed my several-hour-long "five minute discussions" with him. Martha Palmer has been a big influence in my research, not only through her work, but also by providing encouragement and detailed comments about my thesis.

Over the years, I have interacted and worked closely with many people belonging to the Cognitive Computation Group. No part of my research would have been possible without help from Mark Sammons. I thank him for enthusiastic and deeply thought-provoking discussions about NLP and his willingness to provide honest critiques of my research. I have worked extensively with Ming-Wei Chang and would like to hope that at least some of his obsession to get it right and his infectious excitement for problem solving rubbed off on me. Dan Goldwasser has always provided sage advice – both research-wise and in general. I have also benefited immensely from discussions and collaborations with Kai-Wei Chang, James Clarke, Michael Connor, Rodrigo de Salvo Braz, Quang Do, Jacob Eisenstein, Prateek Jindal, Alexandre Klementiev, Gourab Kundu, Wei Lu, Stephen Mayhew, Jeff Pasternack, Lev Ratinov, Nick Rizzolo, Alla Rozovskaya, Rajhans Samdhani, Kevin Small, Chen-Tse Tai, Ivan Titov, Yuancheng Tu, Vinod Vydiswaran, Shankar

Vembu and John Weiting.

The Computer Science department at UIUC provides a stimulating community for research. In particular, I would like to thank Derek Hoiem, David Forsyth, Sariel Har-Peled, Yonathan Bisk, Leonardo Bobadilla and Ian Endres for several great discussions. I would also like to acknowledge the support staff in the CS department – specifically, Eric Horn, for his cheerful help in all administrative matters.

Sebastian Riedel and Roi Reichart have given me much food for thought in all our interactions. I spent a wonderful summer at Microsoft Research and learned much from Chris Quirk and the entire NLP team.

I would like to acknowledge several friends in Champaign-Urbana who helped me keep my sanity through grad school and made the cornfields feel like home – Anshu, Aravind, Avanija, Chandu, Gayatri, Gowri, Harini, JP, Kaushik, Navaneethan, Padmaja, Pradeep, Preeti, Raghu, Swetha, Vidisha, Vikhram, Vinay, Vinod, Urmi, Yamini and especially Milu.

Finally, and most importantly, my family, particularly my parents, my brother Deepak and my grandparents. Where do even I begin? They convinced a three-year old me that scientists have all the fun in the world. I thank them for that, for everything else and a little more.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation: The End

Understanding text is a complex cognitive endeavor which requires the combination of linguistic cues with background knowledge to create a coherent, structured interpretation of the text. Yet, people are surprisingly adept at this task. Whether computer programs can mimic this human ability is an important open question, the resolution of which can lead to natural language processing (NLP) applications that can reason about natural text.

Recognizing textual entailment and information extraction are two such examples of natural language understanding applications that push research boundaries. We will briefly discuss these applications, which have seen recent research, and revisit them in the final chapter.

Recognizing Textual Entailment The task of recognizing textual entailment (TE) has been the focus of the PASCAL challenges (Dagan et al. (2006) and many others). Given two text fragments, called the *text* and the *hypothesis*, the goal is to identify whether the hypothesis can be inferred from the text.

For example, the following sentences are taken from the TE corpus annotated for the first PASCAL challenge:

(1) a. Latvia, for instance, is the lowest-ranked team in the field but defeated World Cup semifinalist Turkey in a playoff to qualify for the final 16 of Euro 2004.

b. Turkey was defeated by Latvia.

c. Latvia was defeated by Turkey.

Here, sentence (a) entails sentence (b) and not sentence (c). It has been pointed out, e.g. by Vanderwende and Dolan (2006), that to correctly make these decisions we need the ability to automatically identify predicates in the text along with their arguments. In the worst case, textual entailment can be arbitrarily difficult because it requires complex chains of reasoning involving the entities mentioned and background knowledge about them.

Information Extraction Information extraction (IE) is broadly defined as the task of populating a structured database with information from natural language text. We will use event extraction as an example here, which has been the focus of the Message Understanding Conferences (Grishman and Sundheim (1996) summarizes the first six MUC conferences) and the Automatic Content Extraction (ACE) program (Doddington et al., 2004).

With an information extraction system, we could convert complex text into records in a database. For example, suppose we have the following news report:

(2) A police officer in the restive southern province of Ingushetia, neighboring Chechnya, has been wounded by unidentified assailants, officials said Sunday. The assailants ambushed the officer's car and sprayed it with automatic gunfire on Saturday, said Roman Shchekotin, a spokesman for Russia's Interior Ministry branch in southern Russia. In Kabardino-Balkariya, another nearby North Caucasus province, police arrested a suspected militant Saturday, Shchekotin said.

An event extraction system that is trained to identify instances of attacks and extract meta-data about them could identify the following information from this text.

Predicate	`Attack`
`Victim`	A police officer
`Perpetrator`	unidentified assailants
`MeansOfAttack`	automatic gunfire
`Location`	the restive southern province of Ingushetia, neighboring Chechnya
`Time`	Saturday

Semantics plays a crucial role in making entailment decisions and information extraction. In this dissertation, we investigate the problem of representing sentences in a predicate-argument form that conveys shallow semantics. To do so, we propose an extension to the NLP task of **semantic role labeling** (Palmer et al., 2010) and address the problem of automatically predicting this structure using statistical methods.

The rest of this chapter gives an overview of this dissertation. Section 1.2 discusses the need for a semantic representation of text. In Section 1.3, we outline the contributions of this thesis, and in section 1.4, we state the scope of this work. Finally, section 1.5 gives a road map for the rest of this thesis.

1.2 The Need for a Semantic Representation

Over the last couple of decades, we have seen a significant headway in statistical syntactic parsing starting with the work of Charniak (1997) and Collins (1997). In recent years, we have also seen syntactic parse trees used to build applications like relation extraction (for example, Culotta and Sorenson (2004) and others) and textual entailment (for example, Bar-Haim et al. (2007b); Haghighi et al. (2005) and others). However, syntactic parse structures do not provide the right abstraction for such applications for two reasons:

1. Alternations of sentences, with different parse trees, can mean the same. We will refer to this as the problem of **variability**.

2. Two parse trees with the same structure could have different meanings. We will call this the problem of **ambiguity**.

Variability leads to two sentences conveying the same information despite having different surface forms, and hence different syntactic structures. The entailment examples in (1) exhibit the *active/passive alternation.* Consider the simplified version below:

(3) a. Latvia defeated Turkey.

b. Turkey was defeated by Latvia.

The parse trees for the two sentences are shown in Figure 1.1. While the two sentences convey the same information, the parse trees are not similar, thus making problem of determining entailment difficult. However, both sentences can be represented by the same predicate argument structure, shown below[1]:

[1] In this dissertation, we adopt the following representation to show a predicate argument structure:

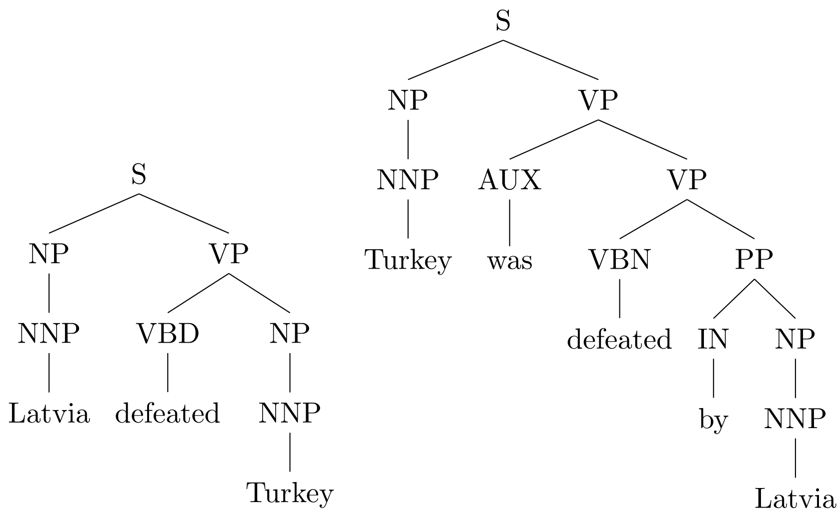


Figure 1.1: Parse trees for sentences in the entailment pair in (3). Compare to the predicate-argument structure that abstracts away the difference between the active and passive voice.

Predicate	`defeat`
`A0`	Latvia *(entity victorious)*
`A1`	Turkey *(entity defeated)*

Indeed, for several alternations, entailment is preserved even though the syntactic structure changes. The sentences in (4) shows an example of the *causative/inchoative alternation*,

(4) a. John boiled the water.

b. The water boiled.

Here, *the water* is the object of the verb *boil* in the first sentence, while it is the subject in the second one, as shown in Figure 1.2. However, in both cases, *the water* is the entity that boils. These examples show the problem of **variability** and highlight the need to build a representation that abstracts away the different syntactic representations of the text.

On the other hand, because of ambiguity, the syntactic structure of a sentence does not always reveal its meaning. We see this in the case of syntactically identical prepositional phrases that have different meanings, as in the examples in figure 1.3. The first prepositional phrase expresses

---

Predicate	`predicate-name`
`arg-label-1`	Argument text *(explanation)*
`arg-label-2`	Argument text *(explanation)*

A predicate can have an arbitrary number of arguments, depending on the definition of the predicate.

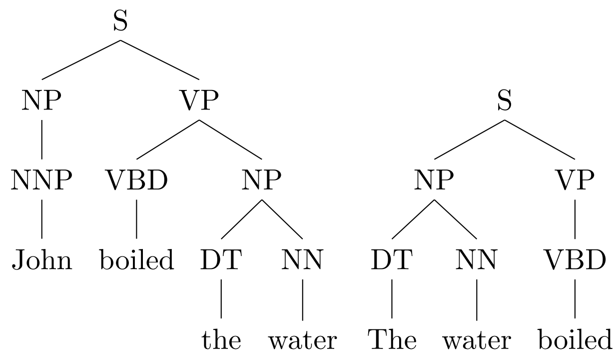


Figure 1.2: Parse trees for the sentences in example (4). We see that the phrase *the water* is the object of the verb in the first example, while it is the subject in the second one.

a locative meaning, while the second indicates a time point. However, the structure of the non-terminals does not give this information. This shows that the syntactic structure *alone* is insufficient to address the **ambiguity** of syntactic structures and capture the meaning of text.

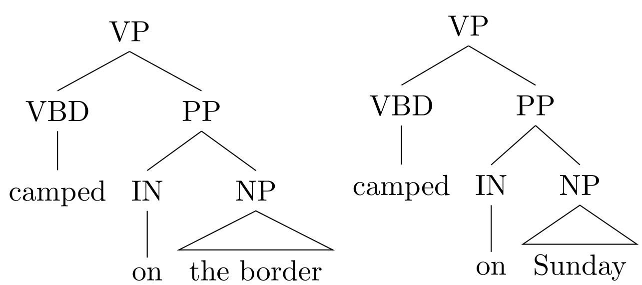


Figure 1.3: Two prepositional phrases where the syntactic structure does not reveal the meaning of the phrase. The phrases are syntactically identical, except for the words in the sentence. However, the syntax does not expose the difference in their meanings.

The problems of variability and ambiguity motivate the need for a shallow semantic representation that goes beyond the syntactic structure.

1.3 Contributions of This Thesis

In this thesis, we investigate the problem of ascribing a semantic representation to text that abstracts away from syntactic alterations *and* exposes the differences between certain syntactically identical constructions. In particular, we make the following claims in this thesis:

> *Semantic relations are triggered not only by verbs and nominalizations, but also by other tokens in sentences. These relations can be analyzed systematically, in a syntax-*

driven manner, just like verbs and nominalizations. To reduce dependence on annotated data, we can take advantage of structure, even when it is not explicitly annotated. We can exploit the inherent constraints that stem from the fact that the input is a natural language sentence.

The primary contributions of this thesis are summarized below:

1. We propose an extension to the NLP task of *Semantic Role Labeling* (SRL) (Palmer et al., 2010). Research in this task has focused on semantic relations that are triggered by verbs and their nominalizations in sentences. We argue that analyzing verbs and nominalizations alone is insufficient for natural language understanding applications and extend the definition of the task to include relations triggered by other tokens like prepositions and commas.

2. We define an inventory of relations that can be expressed by commas and prepositions and introduce new corpora that annotates these relations.

3. Using the new and existing corpora, we build statistical models that predict the the semantic relations expressed by verbs, nominalizations, prepositions and commas. In order to effectively model these relations using the limited available data, we bias the learning and prediction using linguistic knowledge.

   (a) For preposition relations, we model the relations, their arguments and the types of the arguments jointly by treating the arguments and types as latent variables.

   (b) We introduce an algorithm that uses the annotated corpus to learn sentence transformation rules based on the parse trees. We show that this is an effective way to identify comma relations.

   (c) Finally, we point out that the different extensions to SRL are not independent. As a case study, we show that jointly modeling preposition relations along with the arguments of verbs can lead to an improvement in the performance of both.

1.4 Scope and Assumptions

We make two scope-limiting assumptions in this thesis.

First, the representation constructed here does not account for phenomena like quantification, coreference or entity resolution. The study of these higher order phenomena is indeed important and there has been work in the literature which deals with these as separate problems. For example, Bengtson and Roth (2008); Haghighi and Klein (2010) and others study the problem of coreference resolution. Ratinov and Roth (2009) and Ratinov et al. (2011) address the problem of identifying named entities and linking them to a reference collection.

Second, we do not consider cross-sentence discourse relations. This restriction allows us to use syntactic structure of sentences as a base to build the shallow semantic representation. The Penn Discourse TreeBank of Miltsakaki et al. (2004) and subsequent work study discourse relations that are not necessarily constrained by sentence boundaries.

In order to assign a semantic interpretation to text, we will indeed have to address these questions. We will return to this issue in the conclusion (Chapter 9).

From a procedural point of view, we assume that our input consists not only of raw text, but is also syntactically processed with part-of-speech and parse annotation. The interaction between the syntactic and semantic analysis of text has been studied via the CoNLL shared tasks of 2008 and 2009 (Surdeanu et al., 2008; Hajič et al., 2009). While the question of jointly predicting a syntactic and semantic structure for a sentence is an interesting one, most of the participants in the shared task (including the best performing system) used the pipeline from syntax to semantic relations.

1.5 A Road Map

This section presents a readers guide for the rest of the thesis, which can be divided into three logical segments.

Part I: Background

The first part of this thesis surveys background work in the areas of semantic role labeling, structured learning and inference. These chapters do not provide exhaustive surveys of these research directions and we will only go over relevant material here and include additional pointers as necessary. A reader familiar with these areas can skim over this part of the thesis, noting only the summary of notation at the end of the machine learning survey.

- Chapter 2 discusses techniques for statistical machine learning and structure prediction. It also introduces the notation for the learning algorithms used throughout the thesis.

- Chapter 3 briefly traces the history of thematic relations from a linguistic perspective. It also presents an overview of statistical techniques for semantic role labeling.

Part II: Extending Semantic Role Labeling

The second part of the thesis motivates the need to extend the definition of semantic role labeling, and presents specific extensions along with machine learning approaches to predict these structures.

- Chapter 4 shows, using examples, that for text understanding applications, analyzing only verbs and nominalizations is insufficient and we need to consider other linguistic constructions. We identify the three sub-problems that need to be solved for semantic role labeling, irrespective of the construction that triggers the relations.

- Chapter 5 presents an approach for identifying predicate-argument structures triggered by verbs and nominalizations. This approach extends the work of Punyakanok et al. (2008) to identify verb and nominal triggers, their senses and arguments.

- Chapter 6 proposes a set of relations that are expressed by prepositions and presents an approach for identifying preposition relations jointly with their arguments and the types of the arguments.

- Chapter 7 identifies a set of relations expressed by commas along with an algorithm that learns to predict these relations.

- Chapter 8 addresses the interactions between the preposition and verb relations and presents an approach that can learn to predict a coherent structure for a sentence using only a limited amount of jointly labeled data.

Part III: Conclusion

The final part of the thesis offers concluding remarks.

- Chapter 9 summarizes the contributions of this dissertation and identifies directions for future research that can be built on top of this work.

Chapter 2

Predicting Structures in NLP

This chapter presents a survey of the machine learning techniques employed in this thesis. In Section 2.1, we will define the notion of a structure as an assignment to a set of binary variables. Using this definition, Section 2.2 surveys inference and learning algorithms. Finally, section 2.4 summarizes the notation introduced in this chapter.

2.1 What is a Structure?

The concept of a structure is an important one in computational linguistics and machine learning. Burton-Roberts (1997) informally defines a structure as a complex object that is divisible into component parts, each of which can belong to different categories, have specific functions in the complete object and are arranged in a specifiable way.

We will use a simplified version of the part-of-speech tagging task as a running example in this section. We will use the following sentence as an illustrative example:

(5) Trouble dogs John.

Consider a simplified language with only three parts of speech: `Verb`, `Noun` and `Punctuation`. Under this scheme, the output structure for the sentence (5) is `Noun`, `Verb`, `Noun`, `Punctuation`, each label corresponding to one token in the sentence. Here, the assignment of labels for each word corresponds to the notion of parts in the informal definition above. The sequence of the tokens in the sentence specifies the arrangement of the parts.

2.1.1 Structures as Binary Vectors

Many NLP tasks can be phrased as structure prediction problems, where the goal is to convert (possibly pre-processed) text into a structured form. In this thesis, our goal is to convert input text into a structured shallow semantic representation. We will denote pre-processed input sentences by $\mathbf{x}$. Without loss of generality, we can define the structure to be a collection of binary decisions, each corresponding to a part of the structure. We will denote this output by $\mathbf{y}$, which is comprised of individual binary decisions $\{y_1, y_2, \cdots\}$.[1]

In the running example of part-of-speech tagging, the task is typically modeled using a hidden Markov model, where each label is associated with a preference for generating a word (called the emission) and also the next label in the sequence (called the transition). Since, in our toy setting, there are 3 different part-of-speech categories, we have to choose from among one of three labels for each word. To do so, we will introduce three binary variables for each token, only one of which can be set to 1. We will refer to these variables as $y_{i,j,E}$ to indicate the decision that the j^{th} label emits the i^{th} word. To capture the transition decisions, we will introduce a decision for each possible transition, giving us 9 variables for each word. We will use the notation $y_{i,j,k,T}$ to indicate the decision that the i^{th} token is labeled with the j^{th} tag and the next word is labeled with the k^{th} tag. For the sentence (5), we will thus create 12 emission variables and 39 transition variables (using the convention that the transition for the first label is from a dummy start state).

In general, the complete structure for a sentence is determined by an assignment to all the inference variables. We will refer to variables of a structure that are set to 1 as the **active** elements or active variables of the structure and the others as the **inactive** ones.

2.1.2 The Two Search Problems of Structured Learning

From the machine learning perspective, there are two related problems: inference and learning.

Inference is the problem of predicting the best feasible structure for a given input. The notion of "best" is defined using a scoring function that assigns a real-valued score to each possible structure. The structure that has the highest score is considered to be the best one. Thus, inference is the search problem of finding the highest scoring *feasible* set of active variables for a given input.

Learning is the problem of finding the scoring function that can drive inference. In the general setting for supervised learning, nature provides a collection of *labeled* examples – that is, examples $\mathbf{x}^i \in \mathcal{X}$ that are paired with a structure $\mathbf{y}^i \in \mathcal{Y}$, forming a dataset $\mathcal{D} = \{(\mathbf{x}^i, \mathbf{y}^i)\}$. The goal of

[1] Throughout this thesis, we will use subscripts to denote components of vectors and superscripts to enumerate examples of a dataset.

learning is to find a function $f : \mathcal{X} \times \mathcal{Y} \rightarrow \Re$ using this dataset, which when coupled with an inference algorithm can label previously unseen examples as $\arg\max_{\mathbf{y}} f(\mathbf{x}, \mathbf{y})$. Thus, learning is also a search problem, where the search space (or the *hypothesis space*) is the space of functions that assign scores to structures.

2.2 Learning and Inference with Linear Models

In Natural Language Processing, the most widely used hypothesis space is the linear model (Roth, 1998; Collins, 2002). Under this model, for an input $\mathbf{x}$, a structure $\mathbf{y}$ is scored as follows[2]:

$$f(\mathbf{x}, \mathbf{y}) = \mathbf{w}^T \Phi(\mathbf{x}, \mathbf{y}) \tag{2.1}$$

Here, $\mathbf{w} \in \Re^n$ is a dense high dimensional **weight vector**, while $\Phi(\mathbf{x}, \mathbf{y})$ is a n dimensional **feature vector**. The dimensionality of the feature space (denoted by n here) is usually very high. The feature function, Φ, is a deterministic function that captures attributes of the input and output. Feature functions are further described in Section 2.2.1. With a linear model, the goal of learning is to use the labeled dataset $\mathcal{D}$ to train a weight vector $\mathbf{w}$.

2.2.1 Feature Representation for Structures

For an input $\mathbf{x}$ and a structure $\mathbf{y}$, features encode properties associated with the input and output and also inter-dependencies between parts of the output.

In our running example, for the model described in Section 2.1.1, we can define a feature representation that captures (a) token-tag dependencies for the emissions, and (b) tag-tag dependencies for the transitions. For token-tag dependencies, we will use indicators properties of the token and also its neighbors. For example, we could use the following set of properties:

1. The lower cased token,

2. Three character long prefixes and suffixes of the token,

3. Lower cased previous and next tokens, and

[2] Note that this includes log-linear models. While log-linear models are trained with a probabilistic interpretation, at prediction time, the scoring function is the same as Equation 2.1.

4. Indicators for whether the token is the first/last token in the sentence.

These properties, which are string valued, can be mapped to integers using a dictionary called the **lexicon**. This gives us a sparse feature vector[3] that corresponds to the binary decision variable $y_{i,j,E}$ defined in Section 2.1.

To capture tag-tag dependencies, for a pair of consecutive tags j and k in the structure, we can introduce a feature "$j \rightarrow k$" corresponding to the binary decision variable $y_{i,j,k,T}$. For the example structure for sentence (5), this procedure will give us the following features: `Start`→`Noun`, `Noun`→`Verb`, `Verb`→`Noun` and `Noun`→`Punctuation`. Each of these can be converted to a sparse vector (in this case, with one non-zero element each) using the lexicon.

In general, for a decision variable y_i of a structure $\mathbf{y}$, we will define a feature generating function $\phi_i : \mathcal{X} \rightarrow \Re^n$, which is typically a sparse vector. This function extracts features for the structure $\mathbf{y}$ if the corresponding variable is active. The feature representation for the entire structure will then be defined to be the summation of the feature vectors for each of its parts. That is, we can define the feature generating function for the full structure as

$$\Phi\left(\mathbf{x}, \mathbf{y}\right) = \sum_i y_i \phi_i\left(\mathbf{x}\right) \tag{2.2}$$

In the rest of this thesis, we will assume the existence of a lexicon and specify features as a list of properties of the input and output, as above. Using the definition of feature vectors allows us to define the learning and inference procedures.

2.2.2 Inference

The objective of inference is to find the highest scoring feasible structure. Since the feature vector for a structure decomposes over its constituent parts, as in Equation 2.2, so does its score. From Equation (2.1), we can write inference as the following problem:

$$\arg\max_{\mathbf{y}} \mathbf{w}^T \Phi\left(\mathbf{x}, \mathbf{y}\right) = \arg\max_{\mathbf{y}} \sum_i y_i \ \mathbf{w}^T \phi_i\left(\mathbf{x}\right) \tag{2.3}$$

[3]The feature vector is typically sparse in comparison to the underlying dimensionality. That is, the number of non-zero values in the vector is much smaller than the size of the weight vector. However, there is no technical reason for the feature vector to be sparse. In general, the feature vector can be any element of $\Re^n$ for a n dimensional feature space.

Here, all possible assignments to the inference variables $\mathbf{y}$ are not allowed. In our running example, since each word can have only one part-of-speech tag, we can write the following constraint:

$$\forall i,\ y_{i,Verb,E} + y_{i,Noun,E} + y_{i,Punctuation,E} = 1 \tag{2.4}$$

Similarly, we can also write constraints that say that if the transition from a label j to label k is active for any word i, then the i^{th} label *should* be j and the $(i+1)^{th}$ one should be k and vice versa. This constraint can also be stated using linear equations over the inference variables.

In general, we can state linear (in)equalities that allow only a certain subset of possible structures. The work of Rizzolo and Roth (2007) presents a general scheme for converting logical statements into such linear inequalities. In this setting, inference is the combinatorial optimization problem of finding a score-maximizing feasible solution for this 0-1 **integer linear program (ILP)**.

Note that though we can represent inference as an integer linear program, we do not necessarily have to solve it as one. For some special structures (such as the hidden Markov model for the part-of-speech tagging example), inference can be solved using specialized dynamic programming algorithms that run in polynomial time. In general, however, the problem of solving an integer linear program can be computationally intractable.

Many approaches for inference have been explored in the literature for cases when inference is not provably tractable. One approach is to use an off-the-shelf integer linear program solver as a black box to perform inference. In recent years, we have seen the use of other techniques such as beam search (Russell and Norvig, 2003), cutting plane inference (Sontag and Jaakkola, 2007; Riedel, 2008), dual decomposition (Rush et al., 2010; Koo et al., 2010) and Lagrangian relaxation (Rush and Collins, 2011; Chang and Collins, 2011). We refer the reader to these papers for details about the inference algorithms.

In this thesis, unless otherwise specified, we have used cutting plane inference that uses the Gurobi solver as its underlying inference mechanism.[4]. The cutting plane method, listed as Algorithm 1 assumes the existence of a base ILP solver that can solve any integer linear program

[4]We used the state-of-the-art Gurobi engine Gurobi Optimization, Inc. (2012) for all the experiments described in this thesis.

instance. A subset of constraints are marked as the cutting plane constraints and are incrementally added to the solver only if they are violated. First, the problem is solved without any of the cutting plane constraints. If any cutting plane constraints are violated, these are added to the ILP and the base solver is called again. Note that this is not an expensive step because the solver can re-use state from its previous solution. This process is repeated till no constraint is violated, in which case we have a solution, or until the number of iterations crosses a threshold, in which case, we can return the best solution so far. In all the experiments described in this thesis, we allow the loop to iterate 100 times.

Algorithm 1 The cutting plane method for solving an ILP

Input: An ILP P, a set of additional constraints C called the cutting plane constraints, total number of iterations T

Output: The solution of P in the presence of the cutting plane constraints C

Initialize ILP p to be the same as P
for $t = 1, 2, \cdots, T$ **do**
    $\mathbf{y} \leftarrow$ solve p using the base solver
    Let v be the set of constraints in C that are violated by $\mathbf{y}$
    if $v = \emptyset$ **then**
        return $\mathbf{y}$
    else
        Add v to the constraints of ILP p
    end if
end for
Return $\mathbf{y}$, the best solution so far.

2.2.3 Learning

The goal of learning is to use the dataset $\mathcal{D}$ to determine a weight vector that can score structures. In this section, we will see two structured learning algorithms that are used in this thesis: Perceptron and Support Vector Machine (SVM).

Structured Perceptron

The structured Perceptron (Collins, 2000), listed as Algorithm 2, is based on the Perceptron algorithm of Rosenblatt (1958). This algorithm iterates over the examples in $\mathcal{D}$ to find a weight vector that gives a higher score to annotated structures than others. To do so, for an example $(\mathbf{x}^i, \mathbf{y}^i)$, it uses the current estimate of the weight vector to find the highest scoring structure (line 6). If this structure is not the same as the annotated one, then it updates the weight vector in the direction of the features for the annotated structure and away from those of the predicted one (line 9).

It can be shown that if the data is separable (that is, if there is some weight vector that can correctly predict all the annotated examples), then this algorithm will eventually converge to a separating hyperplane that correctly predicts the structure for all the training examples.

Algorithm 2 The averaged structured Perceptron algorithm

Input: Dataset $\mathcal{D} = \{\mathbf{x}^i, \mathbf{y}^i\}$, number of epochs T
Output: The averaged weight vector

1: Initialize $\mathbf{w} \leftarrow \mathbf{0}, \mathbf{w}_A \leftarrow \mathbf{0}$
2: Initialize $n \leftarrow 1$
3: **for** $t = 1, 2, \cdots, T$ **do**
4:   Shuffle the examples in the dataset $\mathcal{D}$
5:   **for all** $\{\mathbf{x}^i, \mathbf{y}^i\} \in \mathcal{D}$ **do**
6:     Predict $\hat{\mathbf{y}} \leftarrow \arg\max_{\mathbf{y}} \mathbf{w}^T \phi(\mathbf{x}^i, \mathbf{y})$
7:     **if** $\hat{\mathbf{y}} \neq \mathbf{y}^i$ **then**
8:       Set $\mathbf{u} \leftarrow \Phi\left(\mathbf{x}^i, \mathbf{y}^i\right) - \Phi\left(\mathbf{x}^i, \hat{\mathbf{y}}\right)$
9:       Update $\mathbf{w} \leftarrow \mathbf{w} + \mathbf{u}$
10:       Update $\mathbf{w}_A \leftarrow \mathbf{w}_A + n\mathbf{u}$
11:     **end if**
12:     Update $n \leftarrow n + 1$
13:   **end for**
14: **end for**
15: **return** $\mathbf{w}_0 - \frac{\mathbf{w}_A}{n}$

One important addition to the algorithm described above is the use of **averaging**: The final output of the algorithm is the average of the weight vectors over every iteration. This helps the algorithm to generalize well even when the data set is not separable (Freund and Schapire, 1999; Collins and Duffy, 2002).

Structural Support Vector Machines

The other learning algorithm we will use in this thesis is the structural support vector machine (SVM). This is a generalization of the binary SVM algorithm (Cortes and Vapnik, 1995) to structured outputs (Tsochantaridis et al., 2005).

The goal of learning is to solve the following minimization problem:

$$\min_{\mathbf{w}} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_i L(\mathbf{x}^i, \mathbf{y}^i, \mathbf{w}) \tag{2.5}$$

where $L(\mathbf{x}^i, \mathbf{y}^i, \mathbf{w})$ is the loss function for the structured prediction. The loss function is written as

$$L(\mathbf{x}^i, \mathbf{y}^i, \mathbf{w}) = \ell(\max_{\mathbf{y}}(\Delta(\mathbf{y}^i, \mathbf{y}) - \mathbf{w}^T\Phi\left(\mathbf{x}^i, \mathbf{y}^i\right) + \mathbf{w}^T\Phi\left(\mathbf{x}^i, \mathbf{y}\right) \tag{2.6}$$

Here, Δ quantifies the quality of a structure $\mathbf{y}$ with respect to the annotated structure $\mathbf{y}^i$ and is typically defined to be the Hamming distance between their binary representations. The function $\ell : \Re \to \Re$ can be instantiated by loss functions like the hinge-loss, with $\ell(x) = \max(0, x)$ or the squared-hinge loss, with $\ell(x) = \max(0, x)^2$.

Different algorithms have been proposed in the literature to solve the optimization problem (for example, Joachims et al. (2009), Shalev-Shwartz et al. (2011), Chang et al. (2010b)). In this thesis, we use the solver based on the work of Chang et al. (2010b), which uses the squared-hinge loss and adopts the cutting plane strategy of Joachims et al. (2009) and uses dual-coordinate descent to solve the resulting series of optimization problems.

All these algorithms require us to define a procedure for **loss-augmented inference**, which solves the following modified inference problem for a training example $(\mathbf{x}^i, \mathbf{y}^i)$:

$$\max_{\mathbf{y}} \mathbf{w}^T\Phi\left(\mathbf{x}^i, \mathbf{y}\right) + \Delta(\mathbf{y}, \mathbf{y}^i) \tag{2.7}$$

Since Δ is the Hamming loss, this can be written as an ILP instance and, if necessary, can be solved as one.

2.3 Discussion and Special Cases

For learning and inference with structured models, we need to specify the following:

1. The feature representation for a structure $\mathbf{y}$ for an input $\mathbf{x}$, $\Phi(\mathbf{x}, \mathbf{y})$
2. An inference algorithm that finds the best structure for an input $\mathbf{x}$ using a weight vector $\mathbf{w}$.
3. A learning algorithm.

We will now see an important special case: multiclass classification. Historically, binary and multiclass classification were developed before the structured variants. However, mathematically, structured learning generalizes them.

For multiclass classification, the goal of prediction is to assign one of a fixed set of N labels. In other words, the structure $\mathbf{y}$ can be written as a collection of N binary indicators exactly one of which can be 1. Using a feature representation of the input alone $\Phi(\mathbf{x})$, we define $\Phi(\mathbf{x}, \mathbf{y})$ as $\sum_i y_i \Phi(\mathbf{x})$. This corresponds to the standard Kessler construction for multiclass classification (Duda et al., 2001).

2.4 Summary of Notation

Symbol	**Definition**
$\mathbf{x}$	Input text examples, defined in Section 2.1.1.
$\mathbf{y}$	Structured output, represented as a binary vector, defined in Section 2.1.1.
y_i	i^{th} element of the structure $\mathbf{y}$, defined in Section 2.1.1. We will use descriptive subscripts that indicate the provenance of the sub-structure.
$\mathbf{w}$	High dimensional dense weight vector, defined in Section 2.2.
$\Phi(\mathbf{x}, \mathbf{y})$	Feature representation for the structure $\mathbf{y}$ for the input $\mathbf{x}$, defined in Section 2.2.
$\phi_i(\mathbf{x})$	Features for the y_i, defined in Section 2.2.1.
$\mathcal{D}$	A dataset consisting of annotated examples $(\mathbf{x}^i, \mathbf{y}^i)$.
$\Delta(\mathbf{y}^i, \mathbf{y})$	The Hamming distance between structures $\mathbf{y}^i$ and $\mathbf{y}$, defined in Section 2.2.3.

Chapter 3

Semantic Role Labeling: The Linguistics

In the introduction, we saw the need for a shallow semantic representation of text. This chapter is a survey of the linguistic roots of the problem of semantic role labeling. Section 3.1 traces the history of frame semantics. Section 3.2 summarizes recent machine learning based approaches for addressing the problem of ascribing a semantic structure to sentences.

3.1 Thematic Roles and Frames

3.1.1 Thematic Roles

The thematic structure of a verb is a representation of its semantics, consisting of predicate, corresponding to event or state expressed by the verb, and its arguments, which identify its participants. The arguments are also called the **thematic roles** or the **semantic roles** for the predicate and have been seen as an important abstraction for both language understanding and acquisition (Chierchia and McConnell-Ginet, 2000).

Perhaps the oldest known theory of thematic structure comes from the work of the Sanskrit grammarian Panini in the 6^{th} century BCE, who described a set of grammatical rules that identifies the correspondence between semantic representations of sentences their phonetic representations. His work describes thematic roles, called *karakas*, that act as an intermediary between a morphosyntactic expression and its meaning.

In modern generative grammar, semantic roles were introduced by the thesis of Gruber (1965) (and consequently taken up by Jackendoff (1976) and others) and the case grammar of Fillmore (1968). Fillmore observed that verb-argument relations exist across languages and described thematic role types (which he called *deep cases*) as comprising “a set of universal, presumably innate, concepts which identify certain types of judgments human beings are capable of making about the events that are going on around them, judgments about such matters as who did it, who it happened to, and what got changed."

To illustrate thematic roles in English, consider the two sentences in example (6) below.

(6) (a) John baked the cake.

(b) The cake is baking.

Both sentences express the predicate *bake*. In sentence (a), *John* serves as the `Agent`, which can be defined as the animate instigator of the action and *the cake* is the `Theme`, which indicates the entity undergoing change in the event. Sentence (b) only identifies the `Theme` of the same relation. Note that the `Theme` of the verb *bake* is expressed as the object in the sentence (a) while it is the subject of sentence (b). Early work in thematic roles focused on the problem of **argument selection**, that is, identifying such mappings between syntactic positions to semantic roles.

Fillmore's original work posited an initial list of thematic role types[1]. However, since the early work on case grammar, it is understood that the set of role types is not completely known and additional roles types will be needed. Over the years, many role types have been proposed in the literature, leading to a proliferation of role types with fine semantic distinctions between them.

To counter this problem, Dowty (1991) suggested that the set of roles be collapsed to two cluster concepts – proto-agent and proto-patient – which are defined as a only set of properties. Barker and Dowty (1993) extended this argument to non-verbal thematic roles.

3.1.2 Frames

The theory of case grammar led to the development of **frame semantics** by Fillmore (1976). Frames, as defined by Minsky (1975), are structures that represent a stereotyped intuition. In the linguistic context, frames refer to cognitive structures that are evoked by words associated with it and comprise of frame elements that correspond to the notion of semantic roles. For example, the `Commercial Transaction` frame can be evoked by words like *buy* and *sell* and consists of frame elements `Buyer`, `Seller`, `Goods` and `Money`. Thus, the sentence "John bought a car from Mary." could be represented as:

[1] Fillmore suggested the following preliminary list of cases: `Agent`, `Instrument`, `Dative`, `Factitive`, `Locative` and `Objective`.

Predicate	`Commercial Transaction`
`Seller`	John
`Buyer`	Mary
`Goods`	a car

3.2 Statistical Approaches

In this section, we will take a brief look at recent corpus-driven approaches that address the problem of semantic role labeling.

3.2.1 Corpora

FrameNet (Fillmore et al., 2003; Ruppenhofer et al., 2006) is a knowledge base of semantic frames along with annotated text. This annotation scheme follows the frame semantics specified by Fillmore.

The study of Levin (1993) argues that the underlying semantics of a verb manifests itself via its syntactic frame. Levin also defined classes of verbs that share syntactic frames and, consequently, their semantic meaning. Following this work, PropBank (Kingsbury and Palmer, 2002, 2003; Palmer et al., 2005) takes a syntax-driven approach and annotates semantic roles associated with verbs in the Penn Treebank (Marcus et al., 1994). The PropBank annotation scheme identifies a set core arguments (`A0-A5` and `AA`) that are common to all verbs. The interpretation of these labels is dependent on the verb and its **frameset** which corresponds to the sense of the verb. PropBank defines the senses of the verbs and the interpretations of the core arguments in frame files, which consist of one frameset per sense of the verb. Apart from the core arguments, there are 13 modifier arguments which act as adjuncts and whose meaning is fixed across verbs: `AM-ADV`, `AM-DIR`, `AM-DIS`, `AM-EXT`, `AM-LOC`, `AM-MNR`, `AM-MOD`, `AM-NEG`, `AM-PRD`, `AM-PRP`, `AM-REC`, `AM-TMP`. The two other classes of arguments are references and continuations. A reference to a core argument or a modifier `X`, denoted by `R-X`, indicates that the span of text is a relative pronoun that refers to the actual argument `X` that is outside that clause that contains the verb. A continuation of an argument `X`, denoted by `C-X`, indicates continuations of non-contiguous arguments.

While FrameNet defines frame-specific argument labels, PropBank uses a small set of abstract semantic roles that can be thought of as extensions to the proto-roles approach proposed by Dowty (1991). The two most frequent argument `A0` and `A1` generally correspond to the notions of proto-agent and proto-patient respectively, while the other core arguments `A2-5` are overloaded and are not semantically meaningful labels.

The NomBank project (Meyers et al., 2004) annotated the Treebank with similarly abstract semantic roles associated with nominalizations of verbs.

The goal of verb and nominal semantic role labeling is to identify the predicate-argument structure defined by verbs and nominalizations in sentences. For example, in *John gave a rose to Mary*, the verb *give* takes three arguments: [`A0` *John*], [`A1` *a rose*], [`A2` *to Mary*], denoting the giver, the thing given and the recipient respectively. The final interpretation of the abstract roles comes from the definition of the arguments for the specific verb.

3.2.2 Predicting Semantic Roles

Gildea and Jurafsky (2002) was one of the earliest works that addressed the problem of predicting semantic roles. Their work used the FrameNet annotation scheme. In recent years, FrameNet semantic role labeling has seen increased interest via the SemEval 2007 shared task (Baker et al., 2007). The work of Das et al. (2010) represents the current state-of-the-art for this task.

The CoNLL shared tasks of 2005 and 2005 (Carreras and Màrquez, 2004, 2005) used the PropBank corpus to predict argument structures for verbs. The systems of Punyakanok et al. (2004); Koomen et al. (2005); Punyakanok et al. (2008) and Toutanova et al. (2005, 2008) were the best performing ones in these shared tasks and employed inference for each verb to predict a globally coherent assignment to its arguments. We describe the system of Punyakanok et al. (2008) and extend it in Chapter 5. Since then there have been several other works that deal with verb semantic role labeling. Palmer et al. (2010) gives an overview of recent work in this area.

3.2.3 Other Work

VerbNet (Kipper, 2005) is a domain-independent verb lexicon that extends the work of Levin (1993). It consists of classes of verbs that are syntactically and semantically alike along with a description of their thematic roles and semantic restrictions of arguments. In addition, it also includes mappings to other resources like WordNet (Fellbaum, 1998), PropBank and FrameNet. Kipper et al. (2004) used preposition semantic roles to significantly extend the VerbNet lexicon.

SemLink[2] is an ongoing project that whose aim is to unify the various lexical resources described here. It provides a mapping between PropBank, FrameNet, VerbNet and Wordnet.

[2] http://verbs.colorado.edu/semlink

Chapter 4

Extending Semantic Role Labeling

This chapter provides the motivation for the work in this thesis and presents the need to extend PropBank-style Semantic Role Labeling. Using several text understanding tasks, in Section 4.1 we will see why the standard task of semantic role labeling can be extended to serve as a better foundation for building NLP-driven applications. Section 4.2 provides a generic definition of the problem of semantic role labeling and identifies the common sub-tasks that need to be solved for each of the components of the extended SRL problem. In section 4.3, we will look at why such an extension can be challenging, not only because of the paucity of annotated data and also due to the increased computational complexity of the prediction task.

4.1 Introduction: The 'Why'

We saw in Chapter 3 that semantic role extraction has been studied in the literature in the context of verb and nominal predicates via the PropBank and NomBank corpora. However, sentences also express semantic relations via other linguistic constructions and in order to build NLP-driven applications that rely on the meaning of the text, we need a structured representation of the text that can account for such relations. To highlight this, we will look at textual entailment, event extraction and question answering examples where verb-centric SRL is not sufficient.

4.1.1 Textual Entailment

In Section 1.1, we identified the task of recognizing textual entailment as an application where linguistic cues and background knowledge about the world need to be brought together to make a coherent decision.

Sammons et al. (2009) and Chang et al. (2010a) present an inference-driven approach for textual entailment that depends on finding an alignment between the verb SRL representations of the text and the hypothesis. However, verb analysis alone is insufficient, as shown by Sammons et al. (2010), which studied the impact of various linguistic phenomena using a subset of RTE-5 test examples. For each entailment pair, this work identified linguistic phenomena whose analysis is needed to make the entailment decision and found that **implicit relations** account for 23.3% of the pairs annotated[1].

The following entailment pairs are taken from the annotated RTE-5 data, which highlight the need to analyze such implicit relations. In each pair, the text and hypothesis are represented by **T** and **H** respectively.

(7) **T** A Soyuz capsule carrying a Russian cosmonaut, an American astronaut and U.S. billionaire tourist Charles Simonyi has docked at the international space station.

H Charles Simonyi is a Russian cosmonaut.

Label: Not entailed.

(8) **T** The Wildlife Conservation Society, which manages the city's zoos and the New York Aquarium in Coney Island, is among the many organizations in the city facing steep budget cuts.

H The New York Aquarium is located in Coney Island.

Label: Entailed.

In the entailment pair (7), the hypothesis is not entailed by the text because *Charles Simonyi* is not in apposition with *a Russian cosmonaut* and is in fact a member of a three element comma list. This information is expressed in the text via the comma structure. In the pair (8), the phrases *the New York Aquarium* and *Coney Island* are connected by the preposition *in* in the text and the hypothesis asks whether the preposition expresses a locative relation.

While the merits of an alignment-based-entailment approach have been demonstrated in the literature, this approach can not make valid entailment decisions for the examples above using only

[1]Sammons et al. (2010) define an implicit relation as one that is not expressed by verbs and nominalizations in text and is, instead, inferred by analyzing the internal structure of noun phrases.

verb and nominal semantic role labeling. In order to better capture the entailment decision, we need a representation that accounts for implicit relations like the ones described above.

4.1.2 Event Extraction

Our second motivating example is that of event extraction, which is the problem of mapping text into an ontology of events. Suppose we have an ontology of sport-related events that includes a SCORE event, whose arguments include SCORER, which identifies the agent responsible for the event and NUMBEROFPOINTS, which is the number of points (an abstraction over points, goals and runs) that were accumulated in that event.

Each of the following sentences contains a SCORE event with both arguments realized.

(9) Babe Ruth hit a home run.

(10) The home run by Babe Ruth changed the game.

(11) The visitors lost the match despite Sachin Tendulkar's unbeaten century.

In the sentence (9), verb-centered semantic role labeling can identify the arguments of the verb *hit* as

Predicate	`hit.01`
`A0`	Babe Ruth *(agent, hitter)*
`A1`	a home run *(thing hit)*

Using this representation, the SCORE event and its arguments can be identified by mapping the SRL arguments to those of the event, to give

Predicate	SCORE
`Scorer`	Babe Ruth
`NumberOfPoints`	home run

However, in sentences (10) and (11), the verbs arguments do not align with the score event. The preposition *by* in the former identifies the scorer and the number of points, while the possessive form does so in the latter.

On Feb. 20, 1962 – 50 years ago next Monday – a Marine Corps fighter pilot from small-town America stepped forward in response to the country's need. The astronaut was John Glenn, whom the author Tom Wolfe has called "the last true national hero America has ever had." On Monday and Tuesday, he will be honored with a dinner and a spaceflight forum at Ohio State University, home of the John Glenn School of Public Affairs.

Questions:

a. Who called John Glenn "the last true national hero America has ever had"?

b. Where will the spaceflight forum be held?

Figure 4.1: A reading comprehension example

4.1.3 Question Answering

Our final example is the problem of question answering, which is closely related to the textual entailment and information extraction problems. It is analogous to a reading comprehension task, where the program is given a passage of text and is asked questions about it.

Figure 4.1 shows an example containing a paragraph of text along with two questions that can be answered by reading it. The first question can be directly answered by analyzing the verb *called* as

Predicate	`call.01`
`A0`	the author Tom Wolfe *(Caller)*
`A1`	John Glenn *(Item being labeled)*
`A2`	the last true national hero America has ever had *(Attribute)*

We see that the answer to the question is the `A0` of the verb. However, the second question asks for the the location of the spaceflight forum. This is is only implied by the preposition *at* in the phrase *spaceflight forum at Ohio State University*. In order to answer the question, we need to understand that the preposition *at* expresses a locative meaning.

4.1.4 Discussion

Under the standard definition, thematic relations act as mediators between the syntax of text and its meaning by specifying the manner in which arguments of a verb interact with the verb. In the examples seen so far, we have seen that events or states can be realized in a sentence without the use of verbs (or their nominalizations). Indeed, quite often, we can reformulate such an event or state using a verb, as in sentences (9) and (10). Human readers can easily see the entailment between such reformulations, suggesting the existence of implicit predicates.

The task of semantic role labeling, as defined by PropBank and NomBank, provides a foundation for building natural language understanding applications. Since verbs and nominalizations may not always point us towards the right answer, there is a need to extend the semantic role labeling task to include other relations.

4.2 A Unified Framework

Conceptually, we can unify the task of identifying semantic relations expressed by the different lexical items in a sentence by defining three related sub-tasks:

1. **Relation trigger detection** identifies one or more tokens in the sentence that evoke the relation.

2. **Relation labeling** assigns a name to the relation that abstracts the surface forms of the triggers.

3. **Argument labeling** identifies and assigns labels to the arguments of the relation.

We group the types of relations by the part-of-speech of the words that trigger them. Thus, we have verb relations, nominal relations, preposition relations and so on. Predicting each of these can be seen as a sub-problem of the broader problem comprising of the three tasks enumerated above. The main advantage of separating the SRL problem in this fashion is that it simplifies the first step, namely trigger detection, especially for relations triggered by tokens belonging to a closed set of word classes like prepositions.

Note that the sub-division of problems into the three components listed above is applicable not only to the semantic role labeling task discussed in this dissertation, but also to FrameNet parsing, as in the work of Das et al. (2010), and in general for relation and event extraction tasks. The main difference is in the nature of the argument labels. While FrameNet annotates relations expressed by verbs, nouns and adjectives, the frame element names (which are the equivalent of argument labels) are specific to each frame (which corresponds to the relation label). For example the Commercial Transaction frame has the frame elements Buyer, Seller, Goods and Money. Such frame-specific labels makes the learning problem more difficult. In contrast, the argument labels in a PropBank style annotation are generic, this making learning easier. We seek this crucial property for the expanded set of relations and their arguments.

The forthcoming chapters will look at semantic relations triggered by several linguistic phenomena. In each case, we will map the three sub-problems to the sub-components of the predictor that converts text to predicate-argument relations.

4.3 Challenges

The problem of extending semantic role labeling to include predicate-argument relations triggered by various phenomena is a challenging one from many perspectives.

The first difficulty concerns the definition of the tasks. Predicate argument structures are well defined for verbs and nominalizations via the PropBank and NomBank annotation schemes. There is no analogous definition for the other phenomena. In this work, we define ontologies of comma and preposition relations.

Second, for the verb and nominal relations, the existence of annotated corpora has led to the use of supervised machine learning approaches for training classifiers that predict the structures. For the newly proposed relations, there are no large hand-annotated corpora. We alleviate this problem by (1) hand-annotating a small corpus for the newly defined comma relations, and (2) taking advantage of the related task of preposition sense disambiguation to induce training and test data for preposition relations. The development and annotation of these corpora are further explored in the relevant chapters.

Third, the annotated corpora for prepositions and commas are small. Only 1000 sentences are annotated for comma structures (compared to over 40,000 PropBank and NomBank annotations). For the preposition relations, the data only provides annotation for the predicate labels. From the machine learning perspective, this necessitates the development of new learning schemes where linguistic intuition heavily biases learning. In the case of comma resolution, the learning algorithm only considers arguments that align with parse constituents. For preposition relations, we use a latent variable model to identify the arguments and also show that joint inference with other phenomena can give us better performance.

Chapter 5

Semantic Role Labeling of Verbs and Nominalizations

This chapter gives an overview of the problem of predicting predicate-argument structures triggered by verbs and nominalizations in text. We instantiate and extend the work of Punyakanok et al. (2008), which only labels arguments for verb predicates. For this purpose, we will use the PropBank (Palmer et al., 2005) and NomBank (Meyers et al., 2004) annotations for verb and nominal SRL respectively. The three sub-tasks identified in Section 4.2, in the context of verbs and nominalizations are:

1. **Trigger detection**: Identifying verbs and nominalizations in a sentence that trigger a predicate-argument structure,
2. **Relation labeling**: Identifying the *sense* of the trigger word, and,
3. **Argument labeling**: Identifying and labeling the arguments of the relation.

In the first three sections of this chapter, we will describe the components of the prediction task. Section 5.4 describes the full inference problem by describing the interactions between the sense and argument labeling components. We will conclude this chapter with an empirical evaluation of the sense and argument labelers.

5.1 Trigger Identification

Both PropBank and NomBank annotations define semantic frames for a fixed set of verbs and nominalizations. This suggests that we can identify predicate-defining triggers as all words whose lemma is annotated, filtered according to the part-of-speech tag: PropBank relations can only be triggered by verbs and NomBank relations by nouns.

In practice, this approach is not sufficient for two reasons. First, we may encounter verb or nominal triggers that are yet to be annotated in PropBank or NomBank. Nonetheless, we may want to identify these words as triggers, even though their semantic frame is not defined. The following example illustrates such a case:

(12) An online rebel tweeted allegations of corruption within the ruling family.

The verb *tweet* (i.e "to post an update to Twitter") is not a verb annotated by PropBank. However, from the perspective of a higher level text understanding task, analyzing the verb is crucial. Of course, without a frame entry defined, the labels of arguments for this predicate will not have a rigorously defined meaning. Yet, since PropBank and NomBank argument labels are proto-arguments as described by Dowty (1991), predicting them for unseen triggers can not only prove to be beneficial for subsequent tasks, but also serve as a starting point for extending existing definitions to include the new triggers.

Second, in the case of NomBank, not all words with the appropriate lemma define predicates. Consider, for example, the following sentence, where a word with a valid lemma occurs within a named entity:

(13) Acme Construction Corp. oversaw the renovation of the city park.

In this sentence, while the words *oversaw* and *renovation* define predicate-argument structures, *Construction* does not. Identifying all words with the appropriate lemma as triggers does not distinguish between the two cases and will include spurious predicates.

5.1.1 Verb Trigger Identification

We treat all verbs, except modals and auxiliaries[1], as triggers. In addition, we also include words that start with *re-* if the rest of the word is a known PropBank lemma. Note that this includes words like *tweeted* which are not annotated by PropBank, if they are tagged by the part-of-speech tagger[2].

[1]We ignore words with the part-of-speech `MD` and all instances of *do*, *be* or *have* followed by another verb, possibly with a negation between them.

[2]The verb heuristic will not include words like *re-tweeted* because the root word is not a known PropBank lemma. Alternatively, we could have considered all *re-* words whose suffix is a verb. However, to identify that the suffix is a verb, we have to re-annotate the perturbed sentence with a part-of-speech tagger. Since we assume that the pre-processing is fixed, we do not re-annotate the perturbed sentence. Note that such a perturbation is a specific instance of the approaches proposed by Kundu and Roth (2011), which can be used to identify triggers like *re-tweeted* using any SRL system as a black box.

One important class of tokens included by the heuristic, even though it is not annotated by the current release of PropBank is the verb *to be* in sentences like *He is the president.*

This heuristic achieves a 99.9% recall on the Penn Treebank test set with a precision of 84.8%[3]. The excess predicates comprises of three classes of words: part-of-speech errors, auxiliary verbs and legitimate verb predicates that were missed in the original data (for example, the word *reinvent* in *Rosie reinvented the man*).

5.1.2 Nominal Trigger Identification: Heuristics and Features

For NomBank SRL, the problem of trigger identification is more difficult. Any simple part-of-speech tag based heuristic (e.g., every noun) that can identify triggers will significantly over-generate candidates because not every noun is relation inducing. To address this, we define a high-recall heuristic whose output is filtered by a learned classifier.

The heuristic only considers tokens that are tagged as nouns by the part-of-speech tagger. If any of the following perturbations of the lower cased token are present in the NomLex dictionary (Meyers, 2007), we consider the word to be a possible trigger: 1. The token itself, or 2. its lemma, or 3. the singular form of a plural token, or 4. the second part of a hyphenated token. Additionally, we also include tokens with non-standard lemmas, shown in table 5.1 with the lemma in the right column. We generated this list using the NomBank training set.

Word → Lemma	**Word → Lemma**
bondholder → holder	supressor → suppressor
earthquake → quake	buyout → buy-out
spokesperson → person	hookup → hook-up
reelection → election	ceasefire → cease-fire
coauthor → author	startup → start-up
people → person	eurobond → bond

Table 5.1: List of non-standard lemmas for nominal trigger identification

To train a classifier for filtering the heuristically generated triggers, we extract the following features for a candidate words that have been identified as possible predicates[4]:

1. The lower-cased word, lemma and an indicator for whether the word is capitalized.

[3]While computing performance, we do not account for the *to be* verbs predicted by the heuristic.
[4]For pre-processing, we use the Illinois part-of-speech tagger and chunker from Clarke et al. (2012).

2. The part-of-speech tag of the word.

3. The sub-categorization frame of the lowest phrase that covers the word, according to the Charniak and Johnson (2005) parser.

4. The phrase label of the chunk that contains the word along with indicators for the word being the first or last word of the chunk conjoined with the chunk label.

5. The type of the named entity that contains the word, if one exists, using the named entity recognizer of Ratinov and Roth (2009).

6. Word context: Lower cased words within a window of size 5 centered at the word.

7. POS context: Part-of-speech tags of words within a window of size 5 centered at the word.

In addition, we conjoin these features with all the NomLex classes in which the candidate trigger word occurs. The NomLex dictionary classifies nominal triggers into one of 22 types[5]. The same predicate lemma might belong to multiple classes. By conjoining the class of the trigger to the features, we can capture class-specific aspects that define a trigger. Since the actual class label is not available, we conjoin the features with all the classes to which the word may belong.

Using these features, we trained a binary classifier over the Penn Treebank training and development sets using the SVM algorithm with five-fold cross validation to identify the parameter C. We evaluated the performance of the trigger identifier by computing its precision, recall and F-score over the Penn Treebank test set.

We compare against the underlying high-recall heuristic defined earlier in Table 5.2. The top row shows the performance of the heuristic trigger extractor and the second row shows the performance of the two-stage system where the output of the heuristic has been filtered by the learned classifier. We see that even though recall decreases by about 8%, precision almost doubles. To our knowledge, this represents the first system for identifying NomBank triggers.

[5]NomLex identifies 22 types of relation-inducing nouns. These are ABILITY, ABLE-NOM, ATTRIBUTE, CRISSCROSS, ENVIRONMENT EVENT, FIELD, GROUP, HALLMARK, ISSUE, JOB, NOM, NOMADJ, NOMADJLIKE, NOMING, NOMLIKE, PARTITIVE, RELATIONAL, SHARE, TYPE, VERSION, WORK-OF-ART.

Method	P	R	F1
Heuristic	49.30	99.67	65.97
+ filter	89.35	91.28	90.30

Table 5.2: Performance of the nominal trigger identifier.

5.2 Predicate Sense Identification

The second sub-task of semantic role labeling is that of assigning a name to the relation. In the case of PropBank and NomBank relations, this is accomplished by predicting the sense of the trigger word. Knowing the sense of a predicate allows us to assign semantic meanings to the abstract labels of its argument using the corresponding roleset in the frame files. To illustrate the need for predicate senses, consider the following examples:

(14) The Fed *banked* billions in 2011.

(15) Churchill *banked* the little Cessna sharply.

Both instances of the verb *to bank* take two core arguments, labeled A0 and A1. For sentence (14), the arguments of the verb *to bank* are:

Predicate	`bank.01`
`A0`	The fed
`A1`	billions
`AM-TMP`	in 2011

The arguments of the verb in sentence (15) are

Predicate	`bank.02`
`A0`	Churchill
`A1`	the little Cessna
`AM-MNR`	sharply

For this verb, the first sense of the verb indicates the activity of "to put (money) in the bank", with A0 indicating the agent and A1 indicating the money. The second sense indicates the event of "turning like an airplane". In this context, A0 indicates the pilot and A1 the airplane. Note that without knowing the sense label of the verb (**01** and **02** in the examples), it is impossible to distinguish between the two meanings of the core arguments.

5.2.1 Modeling Predicate Senses

By convention, the senses of the words are ordered according to their frequency, with sense **01** being the most frequent. For verbs, the sense labels range from **01** to **21**, and for nominalizations, they range from **01** to **14**. In both cases, the distribution of senses is highly skewed, with the most frequent sense accounting for 82.72% of PropBank predicates and 90.43% NomBank predicates over the entire Penn Treebank[6].

We frame the sense classification problem as a multiclass classification problem. The problem of predicting the sense of the predicate depends not only on the local context of the predicate, but also its syntactic context. To capture this, we define the following feature set for the task:

1. The lower cased token, part-of-speech tag and lemma for all words within a window of size seven centered at the trigger word (and ignoring the trigger itself).
2. Bigrams and trigrams centered at the previous and next word.
3. The following features for all words dominated by the trigger word and the word which immediately dominates it (using the head percolation rules from Magerman (1995) to identify word dependencies):
   (a) The lower cased word, lemma, part-of-speech, indicators for capitalization and the conflated POS label[7].
   (b) Indicators for common suffixes that indicate de-verbal nouns, de-nominal nouns, and de-adjectival abstract nouns.
   (c) Indicators for prefixes related to number, degree or order.
   (d) Indicator for existence of the word in WordNet.
   (e) WordNet synsets and hypernyms for the first and all senses.
   (f) WordNet part, member and substance holonyms.
   (g) Brown cluster prefixes as used by Ratinov and Roth (2009).

[6]In the PropBank 1.0 data, 1485 predicates assigned the sense label **XX**. In this work, we will consider these to belong to the most frequent sense.

[7]Following Tratz and Hovy (2009), we define the *conflated* part-of-speech tag by mapping the part-of-speech labels to the following set: `Noun`, `Verb`, `Adjective`, `Adverb`, `Punctuation`, `Pronoun` and `Other`.

(h) Indicators for existence of the word in NomLex.

(i) Indicator for whether it ends with an *-ing*.

(j) Named entity label and indicator for whether the word starts or ends a named entity of a specified type.

In addition to using the features as is, we also conjoined them with the lemma of the trigger word. The syntactic features (3a)-(3f) are from Hovy et al. (2010). See Appendix B for implementation details. With these features , we trained the sense predictor using the SVM algorithm with five-fold cross-validation to determine the learning parameter.

5.3 Verb and Nominal Arguments

The third aspect of the relation extraction problem is that of identifying and labeling arguments of the predicates. As mentioned in Section 3.2.1, PropBank and NomBank follows the proto-argument convention. Consequently, not only are the labels generic, but also small in number. The approach described below is a generalization of the one of Punyakanok et al. (2008) to include nominalizations.

Following the standard approach for argument labeling, we decompose the problem into three parts: candidate generation, argument identification and argument type classification. For a given predicate, the candidate generation stage proposes an over-complete set of argument candidates which is filtered by a binary classifier called the **argument identifier**. The argument type is predicted by a multi-class classifier called the **argument labeler** that assigns labels to each filtered candidate. In addition to the labels defined in PropBank/NomBank, the argument type classifier can also assign the special label $\emptyset$, to indicate the decision that the candidate is, in fact, *not* an argument of the predicate. To fully specify these three steps, we need to define the heuristic for candidate generation, and features for the argument identifier and labeler.

In the following discussion, we describe the argument identifier and labeler as classifiers that pick one label for each candidate. In the actual system, we will combine the predictions across all candidates using inference, as described in Section 5.4.

5.3.1 Candidate Generation

The objective of argument candidate generation is to maximize recall. However, we do not consider all possible spans in the input sentence as argument candidates in order to reduce the computational complexity of prediction. In the literature this step is also referred to as **pruning**.

PropBank SRL For generating verb argument candidates, we use the now standard approach that was first described in Xue and Palmer (2004), which starts at the predicate in the parse tree and iterates up the tree to the root. For each node, we include its siblings as argument candidates (except the ones that are coordinated with it). Note that this approach was originally defined for gold parse trees. For parse trees predicted by the Charniak and Johnson (2005) parser, we observe a recall of 86.8%.

NomBank SRL The Xue and Palmer (2004) approach does not work for nominal SRL because arguments do not always align with parse constituents. We propose a new argument generation heuristic for nominal predicates that includes:

1. All non-terminal nodes in the parse tree.
2. All siblings of predicate in the parse tree (since the Penn Treebank annotation does not annotate the structure of noun phrases).
3. The predicate token itself (because unlike PropBank, the predicate itself can be an argument in NomBank).
4. Pronouns in all noun phrases within the same clause that dominate the predicate.
5. All verbs that dominate the predicate.
6. The head of the prepositional phrase that immediately dominates the predicate, if one exists.

This argument generation heuristic gets a recall of 91.8% for arguments of nominalizations using predicted parse trees.

5.3.2 Argument Identification

The argument identifier filters the candidates generated by the candidate generation step. To do so, we train a binary classifier that predicts whether a candidate for a predicate is, in fact, an argument or not.

The features for the verb and nominal argument identifier are listed in Appendix A. We trained the verb (nominal) argument classifier on the output of the verb (nominal) candidate generator with the SVM algorithm using five-fold cross-validation to find the best learning parameter.

The goal of the argument identifier is only to filter candidates that are highly unlikely to be arguments. Since it is only a filter, we desire high recall from the identifier. To find an optimum balance of high recall and reasonable precision, we tune the threshold of the binary classifier using held out data. For verb SRL, we use the F_2 measure to select the threshold, and for nominal SRL, we use the F_3 measure[8].

5.3.3 Argument Type Classification

The third phase of argument labeling is argument type classification, where the candidates are assigned an argument label. This is modeled as a multi-class classifier that takes a argument candidate for a specific predicate as the input. The output label can be one of the argument labels or Ø to indicate that the candidate is not an argument.

Appendix A lists the feature representation for the argument candidates. As before, we trained the model using the SVM algorithm with five-fold cross validation to identify the learning parameter.

5.4 Combining Predictions with Inference

The argument classifier assigns labels to each argument candidate independently. However, doing so violates many natural constraints. Instead of using the prediction of the multiclass classifiers, we combine their predictions using inference over all the candidates. We do not include the trigger identification component in the inference.

[8]If the precision and recall are denoted by P and R respectively, the F_β measure is defined as $(1+\beta^2)\frac{P\cdot R}{\beta^2 P+R}$. A higher value of β emphasizes recall more. We choose $\beta = 3$ for nominal SRL because the number of candidates generated by the heuristic is very large and we wish to filter most of them out.

Inference variables

For both nominal and verb SRL, we have two kinds of decisions that we can predict jointly: the predicate sense and argument labels. We will introduce inference variables for each of these decisions. For a predicate p, for each possible sense label s that it can take, we will define an inference variable $y_{p,s}$. Using the model defined in Section 5.2, this decision is associated with a score, say $\Theta_{p,s}$. For every argument a for this predicate, we define an inference variable $y_{p,a,l}$ for the decision that it is labeled with the label l and the corresponding score from the argument classifier (Section 5.3.3) as $\Theta_{p,a,l}$.

Let $\mathbf{y}$ denote the vector comprising of all the inference variables. The objective of inference is to find an assignment to the inference variables that maximizes the score. That is,

$$\max_{\mathbf{y}\in\{0,1\}^M} \sum_p \left(\sum_s y_{p,s}\Theta_{p,s} + \sum_{a,l} y_{p,a,l}\Theta_{p,a,l} \right) \tag{5.1}$$

Here, M denotes the total number of inference variables.

Constraints

Not all assignments to the variables $\mathbf{y}$ are valid and constraints allow us to restrict the assignments to the variables. We will consider several constraints that inject structural and linguistic knowledge into the inference. The final prediction will be the solution to the integer linear program defined by 5.1 and the constraints.

We will now see several possible constraints, each of which can be converted into linear inequalities using the procedure of Rizzolo and Roth (2007).

1. **Unique label**: Each argument candidate must have exactly one label.

2. **No overlapping arguments**: There cannot be any non-null arguments that overlap. That is, for every token, at most one candidate that covers that token can have a non-null label.

3. **No duplicate arguments**: For each predicate, at most one argument candidate can be labeled with certain arguments. Punyakanok et al. (2008) applied this constraint for core arguments (that is `A0`-`A5` and `AA`). We found that in addition to core arguments, in the verb

training data, it also holds for the labels `AM-REC`, `AM-NEG`, `AM-PRD`. For nominalizations, this constraint holds for the core arguments and also the labels `Support`, `AM-ADV`, `AM-DIR`, `AM-DIS`, `AM-EXT`, `AM-LOC`, `AM-MNR`, `AM-NEG`, `AM-PNC` and `AM-PRD`.

4. **Continuations**: For every predicate, if an argument candidate a is labeled `C-X` for some label `X`, then there must be some candidate that appears *before* a in the sentence with the label `X`.

5. **References**: For every predicate, if an argument candidate is labeled `R-X` for some label `X`, then there must be some other candidate in the sentence with the label `X`.

6. **Sense-argument constraints**: For each predicate, for each sense label, only certain core argument labels are allowed. This information is available in the frame definitions of the predicates. This constraint is new in this work and supersedes the legal arguments constraints that is used in the literature.

7. **Support verb constraint**: For a nominal predicate, an argument can not be labeled `Support` or `C-Support` if it does not coincide with either a nominal or verb predicate in the sentence. This constraint connects the verb and nominal SRL and follows from the definition of the argument `Support`. However, in practice, as seen in Section 5.4.1, this constraint is violated even in the training data and does not lead to an improvement in performance.

8. **No cross-predicate overlap**: Arguments for two different predicates cannot overlap, but they can embed. This constraint was introduced in Surdeanu et al. (2007).

9. **Cross-predicate modifiers**: If an argument candidate is labeled as a modifier for one predicate, then the semantics must be retained across other predicates. This constraint is new to this work.

   We enforce this as follows: If a candidate is labeled as a modifier m for one predicate and is also an argument candidate for a different predicate, then for the second predicate the candidate can assume only one of the following labels: (a) the same label m, or (b) $\emptyset$, or (c) One of a small set of predicate-specific core arguments, obtained from the training corpus, or (d) `C-X` for some X. For nominal SRL, we found that in the training set, `AM-MNR` and `AM-ADV` are frequently confused. So we allow these two labels to co-occur with each other as well.

As a technical point, note that the constraints define one inference problem per sentence, rather than per predicate as in the verb SRL system of Punyakanok et al. (2008).

5.4.1 Constraint Effectiveness

One important question regarding introducing new constraints concerns their effectiveness. We can measure performance on the standard test set with and without the constraints. The drawback of this measurement is that if the model is good enough that the constraint is not violated in the prediction (which might be the case for an in-domain test set), then the improvement can be very small.

Alternatively, we can measure the intrinsic correctness of the constraint by counting the number of violations of the constraints in the training set. If a constraint is never (or almost never) violated in a large training set, then enforcing it at test time can only help performance. Table 5.3 shows the percentage of arguments in the training set for which each constraint holds. Note that some constraints are violated even though the definition of the task mandates them (for example, the references constraint for verbs and the support verb constraint). This is because of noise in the training set. We enforce these constraints despite their being violated to ensure that the output of the system is semantically coherent.

Constraint Name	**Percentage of arguments where the constraint holds**	
	Verb	**Nominal**
Unique label	100	100
No overlapping arguments	100	100
No duplicate arguments	100[9]	100
Continuations	100	100
References	99.9	100
Sense-argument constraints	100	100
Support verb constraint	N/A	97.6
No cross-predicate overlap	99.9	99.1
Cross-predicate modifiers	99.9	99.8

Table 5.3: Constraint violation statistics for verb and nominal SRL on the Penn Treebank training set. See text for details.

5.5 Experiments and Results

This section reports the performance of the SRL systems described in this chapter. We trained the verb SRL models on the Proposition Bank I annotation and the nominal SRL on NomBank 1.0. We trained the sense classifier and the argument classifier on all sections of the data. For the argument identifier, we held out section 22 of the training set to tune the threshold. We pruned away features that occur infrequently in the training examples – we set the pruning threshold to six for the argument classifiers and four for the other models.

We report the average accuracy of predicate sense prediction and the average F1 for the argument labels on section 23. To isolate the effect of incorrectly identified triggers, we use the triggers from the annotation as part of the input. This setting corresponds to the standard evaluation in the literature, where the input is a sentence and a trigger.

5.5.1 PropBank Results

For verb SRL, the system described here is a modification of the work of Punyakanok et al. (2008). Our baseline is to use this system along with an independent predicate sense predictor. That is, we only use the first five constraints and legal argument constraints from Section 5.4. The full system includes joint constraints that connect the predicate and argument along with the additional constraints defined in the previous section.

Table 5.4 describes the performance of these two settings. Predicting the most frequent sense gets an accuracy of only 82.72%. So even the independent sense predictor improves significantly upon this. The additional constraints give a improvement in performance of both the argument labeler and the predicate detector.

5.5.2 NomBank Results

In the literature, NomBank SRL has been studied by the work of Jiang and Ng (2006). Their approach differs from the one described here in the following ways:

1. We use a broader feature set with all possible feature conjunctions.

[9] The **No duplicate argument** constraint holds for all verb arguments mentioned except `AM-LOC` and `AM-MNR`, where it holds for 99.9% of the arguments.

Setting	Argument labels (F1)	Sense (Accuracy)
Most frequent sense	–	82.72
Baseline	75.87	92.91
Full system	76.14	93.25

Table 5.4: Performance of verb semantic role labeling. The baseline predicts the arguments and predicate sense independently. The full system includes additional constraints from Section 5.4. The improvement in sense accuracy is statistically significant at $p < 0.01$ using Dan Bikel's stratified shuffling test.

2. Jiang and Ng (2006) only enforce the non-overlapping argument constraints. We experiment with additional constraints that depend on the semantics of the output labels.

3. We also predict the sense of the predicate, which allows us to assign meanings to the core arguments.

Our goal is to study whether adding the additional constraints (particularly the joint argument and sense and the `Support` argument constraint) helps. Table 5.5 shows the performance of these settings. We make four observations from the results. First, we see that the most frequent sense for nominalizations has an accuracy 90.43% and the classifier defined here gives a significant improvement to 96.38%. Second, the additional constraints do not give a significant improvement for either argument or sense prediction. The performance of argument labeling improves over the setting of Jiang and Ng (2006) but not significantly. Third, as suggested by Table 5.3, the `Support` verb constraint does not help the overall performance, even though its presence produces an output structure that is more consistent with the definition of the label. Finally, unlike the case of verbs, jointly predicting predicate sense and argument labels does not give any change in the sense accuracy.

Setting	Argument labels (F1)	Sense (Accuracy)
Most frequent sense	–	90.43
Only non-overlapping arguments	70.42	–
Independent sense & arguments	70.65	96.38
All constraints (except `Support`)	70.84	96.38
+ `Support` constraints	70.56	96.38

Table 5.5: Performance of nominal semantic role labeling

5.6 Discussion and Related Work

Semantic Role Labeling for verbs and nominalizations have been studied over the last decade, largely driven by the PropBank and the NomBank annotation. The problems of trigger and sense detection for SRL have only recently been studied in the NLP community. The CoNLL shared task of 2008 (summarized in Surdeanu et al. (2008)) focused on the problem of identifying both syntactic and semantic dependencies. For the latter, the task included all three components, namely trigger detection, predicate identification and argument labeling. However, the trigger detection component was not considered the 2009 shared task (Hajič et al., 2009).

Dang and Palmer (2005) studied the interactions between verb senses and their semantic roles and pointed out that semantic role information can help verb sense prediction. Meza-Ruiz and Riedel (2009) proposed a joint learning and inference solution for all three problems for verb SRL using the CoNLL-2008 shared task data (Surdeanu et al., 2008). The approach presented here differs from that work in three important ways:

1. We do not *learn* a joint model for all the tasks and only combine the sense and argument prediction tasks via inference. Doing so significantly reduces the learning complexity.

2. We enforce the coherence between the predicate sense and argument predictions via constraints that are derived from the frame files that define the roles. In contrast, Meza-Ruiz and Riedel (2009) learn weights for this interaction and do not use any background knowledge for training or prediction.

One variant of the approach described here is to include the argument identifier in the inference by enforcing agreement between the identifier and classifier at prediction time. Doing so has the advantage that we do not need to tune the threshold of the identifier for recall. However, it makes the inference problem more difficult from a computational perspective. We will use this setting in Chapter 8, where we jointly predict preposition and verb relations, in order to ensure that no argument candidates are pruned away.

Chapter 6

Preposition Relations

In this chapter, we will consider the extension of the semantic role labeling task to address the problem of predicting semantic relations conveyed by prepositions in sentences[1]. In a survey of the role of prepositions in applications, Baldwin et al. (2009) points out that in the computational linguistics literature, there are three views to the semantics of prepositions:

1. Prepositions are "semantically vacuous and unworthy of semantic representation" (usually in the information retrieval community), or
2. It is impossible to devise a standalone characterization of preposition semantics because it is entirely a function of the words they select (Old, 2003) , or
3. Preposition semantics is complex, but can be captured in a standalone way (see Kipper et al. (2004) and Litkowski and Hargraves (2005)).

In this work, we posit that while preposition semantics is complex, it needs to be studied and can be modeled as a function of the words they select. This chapter presents a standalone approach for modeling preposition semantics. In Chapter 8, we will see that going past the standalone characterization and accounting for interactions between different linguistic phenomena can help. Prepositions express many semantic relations between their governor and object and modeling these complex relationships can lead to a robust text understanding resource (Kipper et al., 2004). To motivate the sub-problems of this task, consider the sentence:

(16) The book of Prof. Alexander on primary school methods is a valuable teaching resource.

Here, the preposition *on* indicates that *the book* and *primary school methods* are connected by the relation `Topic` and *of* indicates the `Creator-Creation` relation between *Prof. Alexander* and

[1]This chapter is based on Srikumar and Roth (2011) and Srikumar and Roth (2013).

the book. Predicting these relations can help answer questions about the subject of the book and also recognize the entailment of sentences like *Prof. Alexander has written about primary school methods.*

Being highly polysemous, the same preposition can indicate different kinds of relations, depending on its governor and object. Furthermore, several prepositions can indicate the same semantic relation. For example, consider the sentence:

(17) Poor care led to her death from pneumonia.

The preposition *from* in this sentence expresses the relation `Cause`(*death*, *pneumonia*). In a different context, it can denote other relations, as in the phrases *copied from the film* (`Source`) and *recognized from the start* (`Temporal`). On the other hand, the relation `Cause` can be expressed by several prepositions; for example, the following phrases express a `Cause` relation: *died of pneumonia* and *tired after the surgery*.

We characterize the semantic relations expressed by transitive prepositions and develop accurate models for predicting the relations, identifying their arguments and recognizing the types of the arguments. Building on the word sense disambiguation task for prepositions, we collapse semantically related senses across prepositions to derive our preposition relation inventory. These relations act as predicates in a predicate-argument representation, where the arguments are the governor and the object of the preposition. While ascertaining the arguments is a largely syntactic decision, we point out that syntactic parsers do not always make this prediction correctly. However, as illustrated in the examples above, identifying the relation depends on the governor and object of the preposition.

The three sub-tasks identified in Section 4.2, from the preposition perspective, are

1. **Trigger detection**: Identifying a preposition in a sentence, which can be done using the part-of-speech tags.
2. **Relation labeling**: Identifying the predicate implicitly expressed by the predicate.
3. **Argument labeling**: Identifying the governor and the object of the preposition.

Given a sentence and a preposition, our goal is to model the predicate (i.e. the preposition relation) and its arguments (i.e. the governor and object). Very often, the relation label is not influenced by the surface form of the arguments but rather by their *semantic types*. In sentence (2) above, we want the predicate to be `Cause` when the object of the preposition is any illness. We thus suggest modeling the argument types along with the preposition relations and arguments, using different notions of types. These three related aspects of the relation prediction task are further explained in Section 6.2 leading up to the problem definition.

Though we wish to predict relations, arguments and types, there is no corpus which annotates all three. The SemEval 2007 shared task of word sense disambiguation for prepositions provides sense annotations for prepositions. We use this data to induce training and test corpora for the relation labels. In Section 6.3, we present two models for the prepositional relation identification problem. The first model considers all possible argument candidates from various sources along with all argument types to predict the preposition relation label. The second model treats the arguments and types as latent variables during learning using a generalization of the latent structural SVM of Yu and Joachims (2009). We show in Section 6.4 that this model not only predicts the arguments and types, but also improves relation prediction performance.

The primary contributions of this work are:

1. We introduce a new inventory of preposition relations that covers the 34 prepositions that formed the basis of the SemEval 2007 task of preposition sense disambiguation.

2. We model preposition relations, arguments and their types jointly and propose a learning algorithm that learns to predict all three using training data that annotates only relation labels.

3. We show that jointly predicting relations with word sense not only improves the relation predictor, but also gives a significant improvement in sense prediction.

6.1 Prepositions & Predicate-Argument Semantics

Though prepositions are attached to verbs, nouns or adjectives, not all relations expressed by them are covered by the PropBank and NomBank annotation schemes. For example, the prepositions in sentence (16) in the previous section express relations that are not captured by verbs or nominalizations. FrameNet covers some prepositional relations, but allows only temporal, locative and directional senses of prepositions to evoke frames, accounting for only 3% of the targets in the SemEval 2007 shared task of FrameNet parsing. In fact, the state-of-the-art FrameNet parser of Das et al. (2010) does not consider any frame inducing prepositions.

Baldwin et al. (2009) highlights the importance of studying prepositions for a complete linguistic analysis of sentences and surveys work in the NLP literature that addresses the syntax and semantics of prepositions. One line of work Ye and Baldwin (2006) addressed the problem of preposition semantic role labeling by considering prepositional phrases that act as arguments of verbs according to the PropBank annotation. They built a system that predicts the labels of these prepositional phrases alone. However, by definition, this covered only verb-attached prepositions. Zapirain et al. (2012) studied the impact of automatically learned selectional preferences for predicting arguments of verbs and showed that modeling prepositional phrases separately improves the performance of argument prediction.

Preposition semantics has also been studied via the Preposition Project of Litkowski and Hargraves (2005) and the related SemEval 2007 shared task of word sense disambiguation of prepositions (Litkowski and Hargraves, 2007). The Preposition Project identifies preposition senses based on their definitions in the Oxford Dictionary of English. There are 332 different labels to be predicted with a wide variance in the number of senses per preposition ranging from 2 (*during* and *as*) to 25 (*on*). For example, according to the preposition sense inventory, the preposition *from* in sentence (2) above will be labeled with the sense **from:12(9)** to indicate a cause. Dahlmeier et al. (2009) added sense annotation to seven prepositions in four sections of the Penn Treebank with the goal of studying their interaction with verb arguments.

Using the SemEval data, Tratz and Hovy (2009) and Hovy et al. (2010) showed that the arguments offer an important cue for identifying the sense of the preposition. However, though these works used a dependency parser to identify arguments, in order to overcome parsing errors, they augment the parser's predictions using part-of-speech based heuristics.

We argue that, while disambiguating the sense of a preposition does indeed reveal nuances of its meaning, it leads to a proliferation of labels to be predicted. Most importantly, sense labels do not transfer to other prepositions that express the same meaning. For example, both *finish lunch before noon* and *finish lunch by noon* express a `Temporal` relation. According to the preposition project, the sense label for the first preposition is **before:1(1)**, and that for the second is **by:17(4)**. This both defeats the purpose of identifying the relations to aid natural language understanding and makes the prediction task harder than it should be: using the standard word sense classification approach, we need to train a separate classifier for each word because the labels are defined per-preposition. In other words, we cannot share features across the different prepositions. This motivates the need to combine such senses of prepositions into the same class label.

In this direction, Srikumar and Roth (2011) merged preposition senses of seven prepositions into relation labels. Litkowski (2012) also suggests collapsing the definitions of prepositions into a smaller set of semantic classes. To aid better generalization and to reduce the label complexity, we follow these works to define a set of relation labels which abstract word senses across prepositions[2].

6.2 Preposition-triggered Relations

This section describes the inventory of preposition relations introduced in this thesis and then identifies the components of the preposition relation extraction problem.

6.2.1 Preposition Relation Inventory

We build our relation inventory using the sense annotation in the Preposition Project, focusing on the 34 prepositions[3] annotated for the SemEval-2007 shared task of preposition sense disambiguation. As discussed in Section 6.1, we construct the inventory of preposition relations by collapsing semantically related preposition senses across different prepositions. For each sense that is defined, the Preposition Project also specifies related prepositions. These definitions and related prepositions provide a starting point to identify senses that can be merged across prepositions. We followed this with a manual cleanup phase. Some senses do not cleanly align with a single relation because the definitions include idiomatic or figurative usage. For example, the sense **in:7(5)** of the preposition *in*, according to the definition, includes both spatial and figurative notions of the spatial sense

[2]Since the preposition sense data is annotated over FrameNet sentences, sense annotation can be used to extend FrameNet (Litkowski, 2012). We believe that the abstract labels proposed in this thesis can further help in this effort.

[3]We consider the following prepositions: about, above, across, after, against, along, among, around, as, at, before, behind, beneath, beside, between, by, down, during, for, from, in, inside, into, like, of, off, on, onto, over, round, through, to, towards, and with. This does not include multi-word prepositions such as *because of* and *due to*. Note that we do include phrasal verbs like *suffer from* if the Preposition Project assigns senses to the preposition. The preposition *from* in this case is assigned the sense **from:12(9)**.

(that is, both *in London* and *in a film*). In such cases, we sampled 20 examples from the SemEval 2007 training set and assigned the relation label based on majority. If sufficient examples could not be sampled, these senses were added to the label `Other`, which is not a semantically coherent category and represents the 'overflow' case.

Overall, we have 32 relation labels[4]. Appendix C provides detailed definitions of each relation and the senses that were merged to create each label. Since we define relations to be groups of preposition sense labels, each sense can be uniquely mapped to a relation label. Hence, we can use the annotated sense data from SemEval 2007 to obtain a corpus of relation-labeled sentences.

To validate the labeling scheme, two native speakers of English annotated 200 sentences from the SemEval training corpus using only the definitions of the labels as the annotation guidelines.We measured Cohen's kappa coefficient (Cohen, 1960) between the annotators to be 0.75 and also between each annotator and the original corpus to be 0.76 and 0.74 respectively.

6.2.2 Preposition Relation Extraction

The input to the prediction problem consists of a preposition in a sentence and the goal is to jointly model the following: (i) The relation expressed by the preposition, and (ii) The arguments of the relation, namely the governor and the object.

We use the sentence (17) in the introduction of this chapter as our running example in further discussion. In our running example, the relation label is `Cause`. We represent the predicted relation label by r.

Arguments The relation label crucially depends on correctly identifying the arguments of the preposition, which are *death* and *pneumonia* in our running example. While a parser can identify the arguments of a preposition, simply relying on the parser may impose an upper limit on relation prediction.

We build an oracle experiment to highlight this limitation. Table 6.1 shows the recall of the easy-first dependency parser of Goldberg and Elhadad (2010) on Section 23 of the Penn Treebank for identifying the governor and object of prepositions.

[4]Note that, even though we do not consider intransitive prepositions, the definitions of some relations could be extended to apply to prepositional particles such *drive down* (`Direction`) and *run about* (`Manner`).

We define heuristics that generate candidate governors and objects for a preposition. For the governor, this set includes the previous verb or noun and for the object, it includes only the next noun. The row labeled Best(Parser, Heuristics) shows the performance of an oracle predictor which selects the true governor/object if present among the parser's prediction and the heuristics. We see that, even for the in-domain case, if we are able to re-rank the candidates, we could achieve a big improvement in argument identification.

	Recall	
	Governor	**Object**
Parser	88.88	92.37
Best(Parser, Heuristics)	92.50	93.06

Table 6.1: Identifying governor and object of prepositions in the Penn Treebank data. Here, Best(Parser, Heuristics) reports the performance of an oracle that picks the true governor and object, if present among the candidates presented by the parser and the heuristic. This presents an in-domain upper bound for governor and object detection. See text for further details.

To overcome erroneous parser decisions, given a preposition, we entertain governor and object candidates proposed both by the parser and the heuristics. In further discussion, we denote the chosen governor and object by g and o respectively.

Argument types While the primary purpose of this work is to model preposition relations and their arguments, the relation prediction is strongly dependent on the *semantic type* of the arguments. To illustrate this, consider the following incomplete sentence: *The message was delivered at* $\cdots$. This preposition can express both a `Temporal` or a `Location` relation depending on the object (for example, *noon* vs. *the doorstep*).

Agirre et al. (2008) show that modeling the semantic type of the arguments jointly with attachment can improve PP attachment accuracy. In this work, we point out that argument types should be modeled jointly with both aspects of the problem of preposition relation labeling.

Types are an abstraction that capture common properties of groups of entities. For example, WordNet provides generalizations of words in the form of their hypernyms. In our running example, we wish to generalize the relation label for *death from pneumonia* to include cases such as *suffering from flu*. Figure 6.1 shows the hypernym hierarchy for the word *pneumonia*. In this case, synsets in the hypernym hierarchy, like *pathological state* or *physical condition*, would also include ailments like flu.

```
pneumonia
  => respiratory disease
    => disease
      => illness
        => ill health
          => pathological state
            => physical condition
              => condition
                => state
                  => attribute
                    => abstraction
                      => entity
```

Figure 6.1: Hypernym hierarchy for the word *pneumonia*

We define a semantic type to be a cluster of words. In addition to WordNet hypernyms, we also cluster verbs, nouns and adjectives using the dependency-based word similarity of Lin (1998) and treat cluster membership as types. These are described in detail in Section 6.4.1.

Relation prediction involves not only identifying the arguments, but also selecting the right semantic type for them, which together, help in predicting the relation label. Given an argument candidate and a collection of possible types (given by WordNet or the similarity based clusters), we need to select one of the types. For example, in the WordNet case, we need to pick one of the hypernyms in the hypernym hierarchy. Thus, for the governor and object, we have a vector of type labels, comprising of one element for each type category. We denote this by $\mathbf{t}^g$ (governor type) and $\mathbf{t}^o$ (object type) respectively.

6.2.3 Problem Definition

The input to our prediction task is a preposition in a sentence. Our goal is to jointly model the relation it expresses, the governor and the object of the relation and the types of each argument (both WordNet hypernyms and cluster membership). We denote the input by $\mathbf{x}$, which consists not only of the preposition but also a set of candidates for the governor and the object and, for each type category, the list of types for the governor and candidate.

The prediction, which we denote by $\mathbf{y}$, consists of the relation r, which can be one of the valid relation labels from Appendix C and the governor and object, denoted by g and o, each of which is one of text segments proposed by the parser or the heuristics. Additionally, $\mathbf{y}$ also consists of type predictions for the governor and object, denoted by $\mathbf{t}^g$ and $\mathbf{t}^o$ respectively, each of which is a vector of labels, one for each type category. Table 6.2 summarizes the notation described above. We refer to the i^{th} element of vectors using subscripts and use the superscript * to denote gold labels. Recall that we have gold labels only for the relation labels and not for arguments and their types.

Symbol	Meaning
$\mathbf{x}$	Input (pre-processed sentence and preposition)
r	relation label for the preposition
g, o	governor and object of the relation
$\mathbf{t}^g, \mathbf{t}^o$	vectors of type assignments for governor and object respectively
$\mathbf{y}$	Full structure $(r, g, o, \mathbf{t}^g, \mathbf{t}^o)$

Table 6.2: Summary of notation

6.3 Learning Preposition Relations

A key challenge in modeling preposition relations is that our training data only annotates the relations labels and not the arguments and types. In this section, we introduce two approaches for predicting preposition relations using this data.

6.3.1 Feature Representation

We use the notation $\Phi(\mathbf{x}, \mathbf{y})$ to indicate the feature function for an input $\mathbf{x}$ and the full output $\mathbf{y}$. We build Φ using the features of the components of $\mathbf{y}$:

1. **Arguments**: For g and o, which represent an assignment to the governor and object, we denote the features extracted from the arguments as $\phi_A(\mathbf{x}, g)$ and $\phi_A(\mathbf{x}, o)$ respectively.

2. **Types**: Given a type assignment t_i^g to the i^{th} type category of the governor, we define features $\phi_T(\mathbf{x}, g, t_i^g)$. Similarly, we define features $\phi_T(\mathbf{x}, o, t_i^o)$ for the types of the object.

We combine the argument and type features to define the features for classifying the relation, which we denote by $\phi(\mathbf{x}, g, o, \mathbf{t}^g, \mathbf{t}^o)$:

$$\phi = \sum_{a \in \{g,o\}} \left(\phi_A(\mathbf{x}, a) + \sum_i \phi_T(\mathbf{x}, a, t_i^a) \right) \tag{6.1}$$

Section 6.4 describes the actual features used in our experiments.

Observe that given the arguments and their types, the task of predicting relations is simply a multiclass classification problem. Thus, following the standard convention for multiclass classification, the overall feature representation for the relation *and* argument prediction is defined by conjoining the relation r with features for the corresponding arguments and types, ϕ. This gives us the full feature representation, $\Phi(\mathbf{x}, \mathbf{y})$.

6.3.2 Model 1: Predicting only Relations

The first model aims at predicting only the relation labels and not the arguments and types. This falls into the standard multiclass classification setting, where we wish to predict one of 32 labels. To do so, we sum over all the possible assignments to the rest of the structure and define features for the inputs as

$$\phi_1(\mathbf{x}) = \sum_{g,o,\mathbf{t}^g,\mathbf{t}^o} \phi(\mathbf{x}, g, o, \mathbf{t}^g, \mathbf{t}^o) \tag{6.2}$$

Once again, for a relation label r, the overall feature representation is defined by conjoining the relation r with the features for that relation ϕ_1, which we write as $\phi_R(\mathbf{x}, r)$. Note that this summation is computationally inexpensive in our case because the sum decomposes according to equation (6.1). With a learned weight vector $\mathbf{w}$, the relation label is predicted as

$$r = \arg\max_{r'} \mathbf{w}^T \phi_R(\mathbf{x}, r') \tag{6.3}$$

We use a structural SVM to train a weight vector $\mathbf{w}$ that predicts the relation label as above. The training is parameterized by C, which represents the tradeoff between generalization and the hinge loss.

6.3.3 Model 2: Learning from Partial Annotations

In the second model, even though our annotation does not provide gold labels for arguments and types, our goal is to predict them. At inference time, if we had a weight vector $\mathbf{w}$, we could predict the full structure using inference as follows:

$$\mathbf{y} = \arg\max_{\mathbf{y}'} \mathbf{w}^T \Phi(\mathbf{x}, \mathbf{y}) \tag{6.4}$$

We propose an iterative learning algorithm to learn this weight vector.

In the following discussion, for a labeled example $(\mathbf{x}, \mathbf{y}^*)$, we refer to the missing part of its structure as $h(\mathbf{y}^*)$. That is, $h(\mathbf{y}^*)$ is the assignment to the arguments of the relation and their types. We use the notation $r(\mathbf{y})$ to denote the relation label specified by a structure $\mathbf{y}$.

Our learning procedure is a variant of the Latent Structural SVM algorithm of Yu and Joachims (2009). We begin by defining two additional inference procedures:

1. **Latent Inference**: Given a weight vector $\mathbf{w}$ and a partially labeled example $(\mathbf{x}, \mathbf{y}^*)$, we can 'complete' the rest of the structure by inferring the highest scoring assignment to the missing parts. In the algorithm, we call this procedure $LatentInf(\mathbf{w}, \mathbf{x}, \mathbf{y}^*)$, which solves the following maximization problem:

$$\begin{aligned} \hat{\mathbf{y}} = \quad & \arg\max\nolimits_{\mathbf{y}} \quad \mathbf{w}^T \Phi(\mathbf{x}, \mathbf{y}), \\ & \text{s.t} \qquad\quad r(\mathbf{y}) = r(\mathbf{y}^*). \end{aligned} \tag{6.5}$$

2. **Loss augmented inference**: This is a variant of the the standard loss augmented inference for structural SVMs, which solves the following maximization problem for a given $\mathbf{x}$ and *fully labeled* $\mathbf{y}^*$:

$$\arg\max_{\mathbf{y}} \mathbf{w}^T \Phi(\mathbf{x}, \mathbf{y}) + \Delta(\mathbf{y}, \mathbf{y}^*) \tag{6.6}$$

   Here, $\Delta(\mathbf{y}, \mathbf{y}^*)$ denotes the loss function. In the Latent Structural SVM formulation, the loss is defined only over the part of the structure with the gold label. In the standard structural SVMs, the loss is over the entire structure. In this work, we use the standard Hamming loss over the entire structure, but scale the loss for the elements of $h(\mathbf{y})$ by a parameter

$\alpha < 1$. This is a generalization of the latent structural SVM, which corresponds to the setting $\alpha = 0$. The intuition behind having a non-zero α is that in addition to penalizing the learning algorithm if it violates the annotated part of the structure, we also incorporate a small penalty for the rest of the structure.

Using these two inference procedures, we define the learning algorithm as Algorithm 3. The weight vector is initialized using Model 1. The algorithm then finds the best arguments and types for all examples in the training set (steps 3-5). Doing so gives an estimate of the arguments and types for each example, giving us 'fully labeled' structured data. The algorithm then proceeds to use this data to train a new weight vector using the standard structural SVM with the loss augmented inference listed above (step 6). These two steps are repeated several times. Note that as with the summation in Model 1, solving the inference problems described above is computationally inexpensive.

Algorithm 3 Algorithm for learning Model 2

Input: Examples $D = \{\mathbf{x}_i, r(\mathbf{y}_i^*)\}$, where examples are labeled only with the relation labels.

1: Initialize weight vector $\mathbf{w}$ using Model 1
2: **for** $t = 1, 2, \cdots$ **do**
3:     **for** $(\mathbf{x}_i, \mathbf{y}_i^*) \in D$ **do**
4:         $\hat{\mathbf{y}}_i \leftarrow \mathit{LatentInf}(\mathbf{w}, \mathbf{x}_i, \mathbf{y}_i^*)$ (Eq. 6.5)
5:     **end for**
6:     $\mathbf{w} \leftarrow \mathit{LearnSSVM}(\{\mathbf{x}_i, \hat{\mathbf{y}}_i\})$ with the loss augmented inference of Eq. 6.6
7: **end for**
8: **return** $\mathbf{w}$

Algorithm 3 is parameterized by C and α. The parameter α controls the extent to which the hypothesized labels according to the previous iteration's weight vector influence the learning.

6.3.4 Joint Inference between Preposition Sense and Relations

By defining preposition relations as disjoint sets of preposition senses, we effectively have a hierarchical relationship between senses and relations. This suggests that joint inference can be employed between sense and relation predictions with a validity constraint connecting the two. We use the features defined by Hovy et al. (2010), which we write as $\phi_s(\mathbf{x}, s)$ for a given input $\mathbf{x}$ and sense label s, and train a separate preposition sense model on the SemEval data with features $\phi_s(\mathbf{x}, s)$ using structural SVM. Thus, we have two weight vectors – the one for predicting preposition relations described earlier, and the preposition sense weight vector. At prediction time, for a given input, we find the highest scoring *joint* assignment to the relation, arguments and types and the sense, subject to the constraint that the sense and the relation agree according to the definition of the relations.

6.4 Experiments and Results

The primary research goal of our experiments is to evaluate the different models (Model 1, Model 2 and joint relation-sense inference) for predicting preposition relations. In additional analysis experiments, we also show that the definition of preposition relations indeed captures cross-preposition semantics by taking advantage of shared features and also highlight the need for going beyond the syntactic parser.

6.4.1 Features and Types

Features Our argument features, denoted by ϕ_A in Section 6.3.1, are derived from the preposition sense feature set of Hovy et al. (2010) and extract the following from the argument: 1. Word, part-of-speech, lemma and capitalization indicator, 2. Conflated part-of-speech (one of Noun, Verb, Adjective, Adverb, and Other), 3. Indicator for existence in WordNet, 4. WordNet synsets for the first and all senses, 5. WordNet lemma, lexicographer file names and part, member and substance holonyms, 6. Roget thesaurus divisions for the word. 7. The first and last two and three letters, and 8. Indicators for known affixes. Our type features (ϕ_T) are simply indicators for the type label, conjoined with the type category.

One advantage of abstracting word senses into relations is that we can share features across different prepositions. The base feature set (for both types and arguments) defined above does not encode information about the preposition to be classified. We do so by conjoining the features with the preposition. In addition, since the relation labels are shared across all prepositions, we include the base features as a shared representation between prepositions.

We consider two variants of our feature sets. We refer to the features described above as the **basic** features. In addition, we define the **basic+gen** features by conjoining argument and type features of **basic** with the generator that proposes the argument. Recall that governor candidates are proposed by the dependency parser, or by the heuristics described earlier. Hence, for a governor, the **basic+gen** features would conjoin the corresponding **basic** features with one of *parser*, *previous-verb*, *previous-noun*, *previous-adjective*, or *previous-word*.

Types As described in Section 6.2, we use WordNet hypernyms as one of the type categories. We use all hypernyms within four levels in the hypernym hierarchy for all senses.

The second type category is defined by word-similarity driven clusters. We briefly describe the clustering process here. The thesaurus of Lin (1998) specifies similar lexical items for a given word along with a similarity score from 0 to 1. It treats nouns, verbs and adjectives separately. We use the score to cluster groups of similar words using a greedy set-covering approach described in Algorithm 4.

This algorithm proceeds by randomly selecting a word which is not yet in a cluster as the center of a new cluster and add all words whose score is greater than σ to it. We repeat this process till all words are in some cluster. A word can appear in more than one cluster because *all* words similar to the cluster center are added to the cluster. We repeat this process for $\sigma \in \{0.1, 0.125, 0.15, 0.175, 0.2, 0.25\}$. By increasing the value of σ, the clusters become more selective and hence smaller. Table 6.3 shows example noun clusters created using $\sigma = 0.15$. For a given word, identifiers for clusters to which the word belongs serve as type label candidates for this type category.

Algorithm 4 Greedy set-covering based approach for clustering words

Input: A list of words $w_1, w_2, \cdots$ and their pairwise similarities, a real valued σ
Output: A set of clusters of the words

```
Mark all words as unvisited
C ← ∅
while some word is unvisited do
    Pick a random unvisited word w
    Add w to a new cluster c
    Add all words whose similarity to w is higher than σ to c
    C ← C ∪ {c}
    Mark all words in c as visited
end while
return C
```

Jimmy Carter; Ronald Reagan; richard nixon; George Bush; Lyndon Johnson; Richard M. Nixon; Gerald Ford
metalwork; porcelain; handicraft; jade; bronzeware; carving; pottery; ceramic; earthenware; jewelry; lacquerware; stoneware
degradation; erosion; pollution; logging; desertification; siltation; urbanization; felling; poaching; soil erosion; water pollution; deforestation; depletion;
expert; Wall Street analyst; analyst; economist; telecommunications analyst; strategist; media analyst
fox news channel; NBC News; MSNBC; Fox News; CNBC; CNNfn; C-Span
Tuesdays; Wednesdays; weekday; Mondays; Fridays; Thursdays; sundays; Saturdays

Table 6.3: Examples of noun clusters generated using the set-covering approach for $\sigma = 0.15$

6.4.2 Experimental Setup and Data

All our experiments are based on the SemEval 2007 data for preposition sense disambiguation (Litkowski and Hargraves, 2007) comprising of word sense annotation over 16176 training and 8058 examples of prepositions labeled with their senses. We pre-processed sentences with part-of-speech tags using the Illinois POS tagger and dependency graphs using the parser of Goldberg and Elhadad (2010). For the experiments described below, we used the relation-annotated training set to train the models and evaluate accuracy of prediction on the test set.

We chose the structural SVM parameter C using five-fold cross-validation on a 1000 random examples chosen from the training set. For Model 2, we picked α using a validation set consisting of a separate set of 1000 training examples. We ran the Algorithm 3 for 20 rounds.

Predicting the most frequent relation for a preposition gives an accuracy of 21.18%. Even though the performance of most-frequent relation label is poor, it does not represent the problem's difficulty and is not a good baseline. To compare, for preposition senses, using features from the neighboring words, Ye and Baldwin (2007) obtained an accuracy of 69.3% , and with features designed for the preposition sense task, Hovy et al. (2010) get up to 84.8% accuracy for the task. Our re-implementation of the latter system using a different set of pre-processing tools gets an accuracy of 83.53%.

For preposition relations, our baseline system for relation labeling uses the **basic** feature set, but without any type information. This produces an accuracy of 88.01% with Model 1 and 88.64% with Model 2. We report statistical significance of results using our implementation of Dan Bikel's stratified-shuffling based statistical significance tester[5].

[5] http://www.cis.upenn.edu/~dbikel/software.html

6.4.3 Main Results: Relation Prediction

Our main result, presented in Table 6.4, compares the baseline model (without types) against other systems, using both models described in Section 6.3. First, we see that adding type information (**basic**) improves performance over the baseline. Expanding the feature space (**basic+gen**) gives further improvements. Finally, jointly predicting the relations with preposition sense gives another improvement.

Setting	Accuracy	
	Model 1	Model 2
Baseline	88.01	88.64
basic	**88.77**	89.14
basic+gen	**89.90***	**89.43***
Joint **basic+gen** & sense	**89.99***	**90.26***†

Table 6.4: Accuracy of relation labeling. Results in bold are statistically significant ($p < 0.01$) improvements over the baseline. Superscripts $*$ and $\dagger$ indicate significant improvements ($p < 0.01$) over **basic** and **basic+gen** respectively. For Model 2, the improvement of **basic** over the baseline is significant with $p < 0.05$.

With Model 2, jointly predicting the sense and relations improves not only the performance of relation identification, but via joint inference between relations and senses also leads to a large improvement in sense prediction accuracy. Table 6.5 shows the accuracy for sense prediction. We see that while Model 1 does not lead to a significant improvement in the accuracy, Model 2 gives an absolute improvement of over 1%.

Setting	Sense accuracy
Baseline	83.53
Joint + Model 1	83.78
Joint + Model 2	**84.78**

Table 6.5: Sense prediction performance. Joint inference with Model 1, while improving relation performance, does not help sense accuracy. However, with Model 2, the improvement over the baseline sense classifier is statistically significant at $p < 0.01$.

6.4.4 Ablation Experiments

Feature Sharing across Prepositions In our first analysis experiment, we seek to highlight the utility of sharing features between different prepositions. To do so, we compare the performance of a system trained without shared features against the baseline system, which uses shared features. To discount the influence of other factors, we use Model 1 in the **basic** setting without any types. Table 6.6 reports the accuracy of relation extraction for these two feature sets. We observed a similar improvement in performance even when type features are added or the setting is changed to **basic+gen** or with Model 2.

Setting	**Accuracy**
Independent	87.17
+ Shared	**88.01**

Table 6.6: Comparing the effect of feature sharing across prepositions. We see that having a shared representation that goes across prepositions improves accuracy of relation prediction ($p < 0.01$).

Different Argument Candidate Generators Our second ablation study looks at the effect of the various argument candidate generators. Recall that in addition to the dependency governor and object, our models also use the previous word, the previous noun, adjective and verb as governor candidates and the next noun as object candidate. We refer to the candidates generated by the parser as *Parser only* and the others as *Heuristics only*. Table 6.7 compares the performance of these two argument candidate generators against the full set using Model 1 in both the **basic** and **basic+gen** settings.

We see that the heuristics give a better accuracy than the parser based system. This is because the heuristics often contain the governor/object predicted by the dependency. This is not always the case, though, because using all generators gives a slightly better performing system (not statistically significant). In the overall system, we retain the dependency parser as one of the generators in order to capture long-range governor/object candidates that may not be in the set selected by the heuristics.

	Feature sets	
Generator	**basic**	**basic+gen**
Parser only	87.12	87.12
Heuristics only	87.63	88.84
All	88.01	89.12

Table 6.7: The performance of different argument candidate generators. We see that considering a larger set of candidate generators gives a big accuracy improvement.

6.5 Discussion

There are two key differences between Model 1 and 2. First, the former predicts only the relation label, while the latter predicts the entire structure. Table 6.8 shows example predictions of Model 2 for relation label and WordNet argument types. These examples show how the argument types can be thought of as an explanation or a certificate for the choice of relation label.

Input	**Relation**	**Hypernyms**	
		governor	**object**
died of pneumonia	`Cause`	experience	disease
suffered from flu	`Cause`	experience	disease
recovered from flu	`StartState`	change	disease

Table 6.8: Example predictions according to Model 2. The hypernyms column shows an example from the synset chosen for the WordNet types.

The main difference between the two models is in the treatment of the unlabeled (or latent) parts of the structure (namely, the arguments and the types) during training and inference. During training, for each example, Model 1 aggregates features from all governors and objects even if they are possibly irrelevant, which may lead to a much bigger model in terms of the number of active weights. On the other hand, for Model 2, algorithm 3 uses the *single* highest scoring prediction of the latent variables, according to the current parameters, to refine the parameters. Indeed, in our experiments, we observed that the number of non-zero weights in the weight vector of Model 2 is much smaller than that of Model 1. For instance, in the **basic** setting, the weight vector for Model 1 had 2.57 million elements while that for Model 2 had only 1.0 million weights. Similarly, for the **basic+gen** setting, Model had 1 5.41 million non-zero elements in the weight vector while Model 2 had only 2.21 million non-zero elements.

The learning algorithm itself is a generalization of the latent structural SVM of Yu and Joachims (2009). By setting α to zero, we get the latent structure SVM. However, we found via cross-validation that this is not the best setting of the parameter. A theoretical understanding of the sparsity of weights learned by the algorithm and a study of its convergence properties is an avenue of future research.

6.6 Conclusion

In this chapter, we addressed the problem of modeling semantic relations expressed by prepositions. We approached this task by defining a set of relations that prepositions express by combining preposition senses across prepositions. Doing so allowed us to leverage existing annotated preposition sense data to induce a corpus for preposition labels. We modeled preposition relations in terms of their arguments, namely the governor and object of the preposition *and* the semantic types of the arguments. Using a generalization of the latent structural SVM, we trained a relation, argument and type predictor using only annotated relation labels. This allowed us to get an accuracy of 89.43% on relation prediction. By employing joint inference with a preposition sense predictor, we further improved the relation accuracy to 90.23%.

Chapter 7

Comma Resolution

In this chapter, we address the question of understanding the role of the commas in a sentence[1]. Consider the following sentences, which illustrate the importance of understanding commas for text comprehension:

(18) Authorities have arrested John Smith, a retired police officer.

(19) Authorities have arrested John Smith, his friend and his brother.

(20) Authorities have arrested John Smith, a retired police officer announced this morning.

Sentence (18) states that John Smith is a retired police officer. The comma and surrounding sentence structure represent the `Is-A` relation. In sentence (19), the commas signify a list; so the sentence states that three people – (i) John Smith, (ii) his friend, and (iii) his brother – were arrested. In sentence (20), a retired police officer announced that John Smith has been arrested. Here, the comma structure indicates the clause boundary.

In all three sentences, the comma and the surrounding sentence structure signify relations essential to comprehending the meaning of the sentence, in a way that is not easily captured using lexical information.

We define the problem of **comma resolution** to be that of solving the three sub-tasks for relation extraction in the context of commas. Trigger identification for commas is the problem of finding groups of commas in the text that, together, express a relation. Comma resolution consists of (1) disambiguating the comma type and (2) determining the relations entailed from the sentence given the commas' interpretation. Specifically, for (1) we assign each comma to one of five possible types, and for (2) we generate a set of natural language sentences that express the relations, if any, signified by each comma structure. The algorithm uses information extracted from parse trees.

[1]This chapter is based on Srikumar et al. (2008)

To evaluate the algorithm, we have developed annotation guidelines, and manually annotated sentences from the WSJ PennTreebank corpus. We present a range of experiments showing the good performance of the system, using gold-standard and parser-generated parse trees.

In Section 7.1 we motivate comma resolution through Textual Entailment examples. Section 7.2 describes related work. Sections 7.3 and 7.4 present our corpus annotation and learning algorithm. Results are given in Section 7.5.

7.1 Motivating Comma Resolution

Comma resolution involves not only comma disambiguation but also inference of the arguments (and argument boundaries) of the relationship represented by the comma structure, and the relationships holding between these arguments and the sentence as a whole. Consider the following three sentences:

(21) T Parviz Davudi was representing Iran at a meeting of the Shanghai Co-operation Organization (SCO), the fledgling association that binds two former Soviet republics of central Asia, Russia and China to fight terrorism.

H1 SCO is the fledgling association that binds several countries.

H2 Russia and China are two former Soviet republics of central Asia .

To see that T entails H1, one must understand that the first comma in sentence T is an apposition structure, and does not indicate the beginning of a list. The second comma marks a boundary between entities in a list. To make the correct inference one must determine that the second comma is a list separator, not an apposition marker. Misclassifying the second comma in T as an apposition leads to the conclusion that T entails H2.

Note that even to an educated native speaker of English, sentence T may be initially confusing; during the first reading, one might interpret the first comma as indicating a list, and that *the Shanghai Co-operation Organization* and *the fledgling association that binds...* are two separate entities that are meeting, rather than two representations of the same entity.

From these examples, we can conclude that comma resolution is essential for comprehending natural language text. Further, comma structures might be highly ambiguous, nested and overlapping, and consequently their interpretation is a difficult task. The argument boundaries of the corresponding extracted relations are also not easy to detect.

7.2 Related Work

Since we focus on extracting the relations represented by commas, there are two main strands of research with similar goals: 1) systems that directly analyze commas, whether labeling them with syntactic information or correcting inappropriate use in text; and 2) systems that extract relations from text, typically by trying to identify paraphrases.

The significance of interpreting the role of commas in sentences has already been identified by Doran (1998); van Delden and Gomez (2002); Bayraktar et al. (1998); White and Rajkumar (2008) and others. A review of this line of research is given in Say and Akman (1997).

Doran (1998) points out that punctuations help us understand text and studies the role of punctuation in grammar by incorporating punctuation marks into the framework of Lexicalized Tree Adjoining Grammar (Joshi and Schabes, 1997). White and Rajkumar (2008) incorporates a precise analysis of punctuations into the CCGbank corpus (Hockenmaier and Steedman, 2007).

Bayraktar et al. (1998) analyzed the Wall Street Journal portion of the Penn Treebank corpus and created a very detailed list of syntactic patterns that correspond to different roles of commas. However, they do not study the extraction of entailed relations as a function of the comma's interpretation. Furthermore, the syntactic patterns they identify are unlexicalized and would not support the level of semantic relations that we show here. Finally, theirs is a manual process dependent only on syntactic patterns. While our comma resolution system uses syntactic parse information as its primary source of information, the approach we have developed is both data-driven and not limited to using only syntactic information.

The most directly comparable prior work is that of van Delden and Gomez (2002), who use finite state automata and a greedy algorithm to learn comma syntactic roles. However, their approach differs from ours in a number of critical ways. First, their comma annotation scheme does not identify arguments of predicates, and therefore cannot be used to extract complete relations. Second, for each comma type they identify, a new Finite State Automaton must be hand-encoded; the learning component of their work simply constrains which FSAs that accept a given, comma containing, text span may co-occur. Third, their corpus is preprocessed by hand to identify specialized phrase types needed by their FSAs; once our system has been trained, it can be applied directly to raw text. Fourth, they exclude from their analysis and evaluation any comma they deem to have been incorrectly used in the source text. We include all commas that are present in the text in our annotation and evaluation.

There is a large body of NLP literature on punctuation. Most of it, however, is concerned with aiding syntactic analysis of sentences and developing comma checkers, much based on Nunberg (1990). Pattern-based relation extraction methods (Davidov and Rappoport, 2008; Davidov et al., 2007; Banko et al., 2007; Pasca et al., 2006; Sekine, 2006) could in theory be used to extract relations represented by commas. However, the types of patterns used in web-scale lexical approaches currently constrain discovered patterns to relatively short spans of text, so will most likely fail on structures whose arguments cover large spans (for example, appositional clauses containing relative clauses). Relation extraction approaches such as Roth and Yih (2004, 2007); Hirano et al. (2007); Culotta and Sorenson (2004); Zelenko et al. (2003) focus on relations between named entities; such approaches miss the more general apposition and list relations we recognize in this work, as the arguments in these relations are not confined to Named Entities. Paraphrase acquisition work such as Lin and Pantel (2001), Pantel and Pennacchiotti (2006) and Szpektor et al. (2004) is not constrained to named entities, and by using dependency trees, avoids the locality problems of lexical methods. However, these approaches have so far achieved limited accuracy, and are therefore hard to use to augment existing NLP systems.

7.3 Corpus Annotation

For our corpus, we selected 1,000 sentences containing at least one comma from the Penn Treebank WSJ section 00, and manually annotated them with comma information[2]. This annotated corpus served as both training and test datasets (using cross-validation).

By studying a number of sentences from WSJ (not among the 1,000 selected), we identified four significant types of relations expressed through commas: `Substitute`, `Attribute`, `Location`, and `List`. Each of these types can in principle be expressed using more than a single comma. We

[2]The guidelines and annotations are available at http://cogcomp.cs.illinois.edu/page/resources/data.

define a **comma structure** to be a set of one or more commas that all relate to the same relation in the sentence. Identifying the comma structure represents the trigger identification stage. We write the predicate and the arguments of the comma structure as a collection of entailed sentences.

`Substitute` indicates an `Is-A` relation. An example is *John Smith, a Renaissance artist, was famous.* By removing the relation expressed by the commas, we can derive three sentences: *John Smith is a Renaissance artist*, *John Smith was famous*, and *a Renaissance artist was famous.* Note that in theory, the third relation will not be valid: one example is *The brothers, all honest men, testified at the trial*, which does not entail *all honest men testified at the trial.* However, we encountered no examples of this kind in the corpus, and leave this refinement to future work.

`Attribute` indicates a relation where one argument describes an attribute of the other. For example, from *John, who loved chocolate, ate with gusto*, we can derive *John loved chocolate* and *John ate with gusto.*

`Location` indicates a LOCATED-IN relation. For example, from *Chicago, Illinois saw some heavy snow today* we can derive *Chicago is located in Illinois* and *Chicago saw some heavy snow today.*

`List` indicates that some predicate or property is applied to multiple entities. In our annotation, the list does not generate explicit relations; instead, the boundaries of the units comprising the list are marked so that they can be treated as a single unit, and are considered to be related by the single relation `Group`. For example, the derivation of *John, James and Kelly all left last week* is written as *[John, James, and Kelly] [all left last week].*

Any commas not fitting one of the descriptions above are designated as `Other`. This does not indicate that the comma signifies no relations, only that it does not signify a relation of interest in this work. Analysis of 120 `Other` commas show that approximately half signify clause boundaries, which may occur when sentence constituents are reordered for emphasis, but may also encode implicit temporal, conditional, and other relation types (for example, *Opening the drawer, he found the gun.*). The remainder comprises mainly coordination structures (for example, *Although he won, he was sad*) and discourse markers indicating inter-sentence relations (such as *However, he soon cheered up.*). Such relations are beyond the scope of this work.

Four annotators annotated the same 10% of the WSJ sentences in order to evaluate inter-annotator agreement. The remaining sentences were divided among the four annotators. The resulting corpus was checked by two judges and the annotation corrected where appropriate; if the two judges disagreed, a third judge was consulted and consensus reached. Our annotators were asked to identify comma structures, and for each structure to write its relation type, its arguments, and all possible simplified version(s) of the original sentence in which the relation implied by the comma has been removed. Arguments must be contiguous units of the sentence and will be referred to as *chunks* hereafter. Agreement statistics and the number of commas and relations of each type are shown in Table 7.1. The accuracy closely approximates the Cohen's Kappa score (Cohen, 1960) in this case, since the baseline probability of chance agreement is close to zero.

Rel. Type	**Avg. Agreement**	**# of Commas**	**# of Rel.s**
`Substitute`	0.808	243	729
`Attribute`	0.687	193	386
`Location`	0.929	71	140
`List`	0.803	230	230
`Other`	0.949	909	0
Combined	0.869	1646	1485

Table 7.1: Average inter-annotator agreement for comma relations.

7.4 A Sentence Transformation Rule Learner

In this section, we describe an algorithm that learns Sentence Transformation Rules (STRs) for comma resolution. We will first define the hypothesis space (i.e., STRs) and two operations – substitution and introduction. Finally, we describe the learning algorithm.

In Section 7.1, we discussed the example which presents an ambiguity between a list and an apposition structure in the fragment *two former Soviet republics, Russia and China*. In addition, simple surface examination of the sentence could also identify the noun phrases *Shanghai Co-operation Organization (SCO)*, *the fledgling association that binds two former Soviet Republics*, *Russia* and *China* as the four members of a list. To resolve such ambiguities, we need a nested representation of the sentence. This motivates the use of syntactic parse trees to represent sentences for comma resolution. (Note, however, that semantic and pragmatic ambiguities might still remain.)

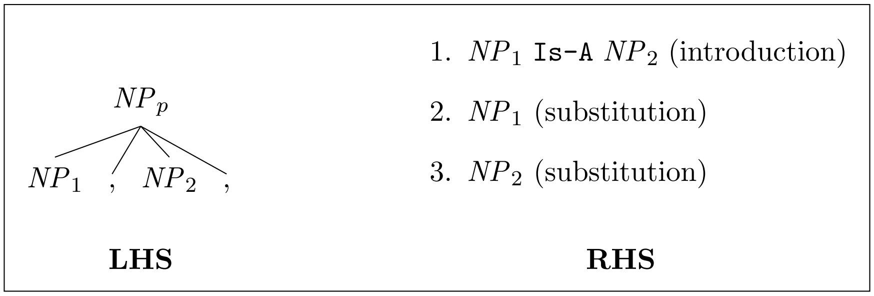


Figure 7.1: Example of a Sentence Transformation Rule. If the *LHS* matches a part of a given parse tree, then the *RHS* will generate three relations.

7.4.1 Sentence Transformation Rules

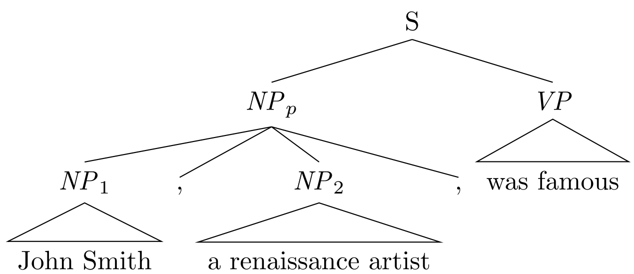
A **Sentence Transformation Rule** (STR) takes a parse tree as input and generates new sentences. We formalize an STR as the pair $l \rightarrow r$, whose left hand side l is a tree fragment that can consist of non-terminals, part-of-speech tags and lexical items. The right hand side, r, is a set of templates, each denoted by r_i, that generate the output sentences. Each r_i can consist of the non-terminals of l and, possibly new tokens that are not present in the input sentence. In the forthcoming discussion, we will refer to the output sentences as relations.

Figure 7.1 shows an example of an STR. The left hand side consists of three noun phrases, labeled as NP_1, NP_2 and NP_p and the RHS consists of three templates. Each output relation is marked either as an **introduction** or a **substitution**. This dictates how the output is generated according the template.

The process of applying an STR $l \rightarrow r$ to a parse tree of a sentence begins with finding a match for l in the parse tree. If matched, the non-terminals of each template r_i are instantiated with the terminals that they cover in T. This is followed by generation of the output relations for each template, based on whether it is marked as an introduction or a substitution.

If a relation is marked as an introductory one, then the output is obtained by replacing the non-terminals in r_i with the terminals they cover in the input sentence. In other words, introduction creates a new output sentence. Figure 7.2 shows the application of the STR from Figure 7.1 to a sentence. The first output relation, NP_1 `Is-A` NP_2, is marked as an introduction. For the input tree, the first noun phrase corresponds to the phrase *John Smith* and the second one corresponds to *a renaissance artist*. Binding these instantiations to the output, we get *John Smith* `Is-A` *a* *renaissance artist.*

If a relation is marked as a substitution, then the output is obtained by replacing the terminals covered by the node that matches with the root of the pattern with the output. In the example STR, the second and third relations are marked as substitutions. When applied to the tree in Figure 7.2, the phrase *John Smith*, corresponding to NP_1, replaces the phrase *John Smith, a renaissance artist,* that corresponds to the root of the pattern NP_p to obtain the second output relation, namely *John Smith was famous.* Effectively, substitution simplifies the input sentence by replacing a non-terminal with one of its descendants. The notions of introduction and substitution were motivated by ideas introduced in Bar-Haim et al. (2007a).



1. [John Smith]/NP_1 `Is-A` [a renaissance artist]/NP_2
2. [John Smith] /NP_1 [was famous]
3. [a renaissance artist]/NP_2 [was famous]

Figure 7.2: Example of application of the STR in Figure 7.1. In the first relation, an introduction, we use the relation `Is-A`, without dealing with tense. NP_1 and NP_2 are both substitutions, each replacing NP_p to generate the last two relations.

7.4.2 Algorithm Overview

In our corpus annotation, the relations and their argument boundaries (chunks) are explicitly marked. For each training example, our learning algorithm first finds the smallest valid STR – the STR with the smallest *LHS* in terms of depth. Then it refines the *LHS* by specializing it using statistics taken from the entire data set.

7.4.3 Generating the Smallest Valid STR

To transform an example into the smallest valid STR, we use the parse tree[3] of the sentence. For a given relation, for each argument in the annotation, we find the lowest node in the parse tree that covers it and no other arguments (even partially). It may, however, cover words that do not belong to any argument. We refer to such a node as a **chunk root**. We then find the lowest node that covers all the chunk roots, referring to it as the **pattern root**. The initial *LHS* consists of the parse sub-tree rooted at the pattern root whose leaf nodes are all either chunk roots or other nodes from the parse tree that do not belong to any chunk. All the nodes are labeled with the corresponding labels in the augmented parse tree. For example, if we consider the parse tree and relations shown in Figure 7.2, then doing the above procedure gives us the initial *LHS* as $S\ (NP_p(NP_1, NP_2,)\ VP)$. The three relations gives us the *RHS* with three elements 'NP_1 `Is-A` NP_2', '$NP_1\ VP$' and '$NP_2\ VP$', all three being introduction.

This initial *LHS* need not be the smallest one that explains the example. So, we proceed by pruning the initial *LHS* to give a new STR that explains the example using both introduction *and* substitution. In our example, the initial *LHS* has a sub-tree, $NP_p(NP_1, NP_2,)$ that can cover all the relations with the *RHS* consisting of 'NP_1 `Is-A` NP_2', NP_1 and NP_2. The first *RHS* is an introduction, while the second and the third are both substitutions. Since no sub-tree of this *LHS* can generate all three relations even with substitution, this is the required STR. This final pruning step ensures that we have the smallest valid STR at this stage.

7.4.4 Statistical Refinement

The STR generated using the procedure outlined above explains the relations generated by a single example. In addition to covering the relations generated by the example, we wish to ensure that it does not cover erroneous relations by matching any of the other comma types in the annotated data.

For this purpose, we specialize the *LHS* so that it covers as few examples from the other comma types as possible, while covering as many examples from the current comma type as possible. The complete algorithm to learn a set of STRs is listed as Algorithm 5.

[3] We augment parse tree nodes with named entity information. This is explained in Section 7.5.

Algorithm 5 ASTRL: A Sentence Transformation Rule Learner.

1: **for all** t: Comma type **do**
2: Initialize $STRList[t] = \emptyset$
3: $\mathbf{p}$ = Set of annotated examples of type t
4: $\mathbf{n}$ = Annotated examples of all other types
5: **for all** $x \in \mathbf{p}$ **do**
6: r = Smallest Valid STR that covers x
7: Get fringe of $r.LHS$ using the parse tree
8: $S = Score(r, \mathbf{p}, \mathbf{n})$
9: $S_{prev} = -\infty$
10: **while** $S \neq S_{prev}$ **do**
11: **if** adding some fringe node to $r.LHS$ causes a significant change in score **then**
12: Set r = New rule that includes that fringe node
13: $S_{prev} = S$
14: $S = Score(r, \mathbf{p}, \mathbf{n})$
15: Recompute new fringe nodes
16: **end if**
17: **end while**
18: Add r to $STRList[t]$
19: Remove all examples from $\mathbf{p}$ that are covered by r
20: **end for**
21: **end for**

Given the most general STR, we generate a sequence of additional, more detailed, candidate rules. Each of these is obtained from the original rule by adding a single node to the tree pattern in the rule's *LHS*, and updating the rule's *RHS* accordingly. We then score each of the candidates (including the original rule). If there is a clear winner, we continue with it using the same procedure (i.e., specialize it). If there isn't a clear winner, we stop and use the current winner. After finishing with a rule (line 18), we remove from the set of positive examples of its comma type all examples that are covered by it (line 19).

To generate the additional candidate rules that we add, we define the *fringe* of a rule as the siblings and children of the nodes in its *LHS* in the original parse tree. Each fringe node defines an additional candidate rule, whose *LHS* is obtained by adding the fringe node to the rule's *LHS* tree. We refer to the set of these candidate rules, plus the original one, as the rule's *fringe rules*. We define the score of an STR as

$$Score(Rule, \mathbf{p}, \mathbf{n}) = \frac{R_p}{|\mathbf{p}|} - \frac{R_n}{|\mathbf{n}|} \tag{7.1}$$

where **p** and **n** are the set of positive and negative examples for this comma type, and R_p and R_n are the number of positive and negative examples that are covered by the STR. For each example, all examples annotated with the same comma type are positive while all examples of all other comma types are negative. The score is used to select the winner among the fringe rules. The stopping criterion in line 11 checks whether any fringe rule has a significantly better score than the rule it was derived from, and exits the specialization loop if there is none.

Since we start with the smallest STR, we only need to add nodes to it to refine it and never have to delete any nodes from the tree. Also note that the algorithm is essentially a greedy algorithm that performs a single pass over the examples; other, more complex, search strategies could also be used.

7.5 Evaluation

7.5.1 Experimental Setup

To evaluate ASTRL, we used the WSJ derived corpus. We experimented with three scenarios; in two of them we trained using the gold standard trees and then tested on gold standard parse trees (*Gold-Gold*), and text annotated using a state-of-the-art statistical parser of Charniak and Johnson (2005) (*Gold-Charniak*), respectively. In the third, we trained and tested on the Charniak Parser (*Charniak-Charniak*).

In gold standard parse trees the syntactic categories are annotated with functional tags. Since current statistical parsers do not annotate sentences with such tags, we augment the syntactic trees with the output of the Illinois Named Entity tagger. We evaluate our system from different points of view, as described below. For all the evaluation methods, we performed five-fold cross validation and report the average precision, recall and F-scores.

7.5.2 Relation Extraction Performance

First, we present the evaluation of the performance of ASTRL for generating the arguments without considering of the type of relation. This metric, called the *Relation metric* in further discussion, is the most relevant one from the point of view of the textual entailment task. After learning the STRs for the different comma types using the gold standard parses, we generated entailed sentences by applying the STRs on the test set. Table 7.2 shows the precision, recall and F-score of the entailed sentences, without accounting for the comma type of the STR that was used to generate them. Since a list does not generate any relations in our annotation scheme, we use the commas to identify the list elements. Treating each list in a sentence as a single relation, we score the list with the fraction of its correctly identified elements.

In addition to the Gold-Gold and Gold-Charniak settings described above, for this metric, we also present the results of the Charniak-Charniak setting, where both the train and test sets were annotated with the output of the Charniak parser. The improvement in recall in this setting over the Gold-Charniak case indicates that the parser makes systematic errors with respect to the phenomena considered.

Setting	**P**	**R**	**F**
Gold-Gold	86.1	75.4	80.2
Gold-Charniak	77.3	60.1	68.1
Charniak-Charniak	77.2	64.8	70.4

Table 7.2: Performance of ASTRL (precision, recall and F-score) for generating entailed sentences. The comma types were used only to learn the rules.

7.5.3 Comma Resolution Performance

Type	**Gold-Gold Setting**						**Gold-Charniak Setting**					
	Relation-Type metric											
	Smallest Valid STRs			**ASTRL**			**Smallest Valid STRs**			**ASTRL**		
	P	R	F	P	R	F	P	R	F	P	R	F
Total	66.2	76.1	70.7	81.8	73.9	**77.6**	61.0	58.4	59.5	72.2	59.5	**65.1**
	Relations Metric, Per Comma Type											
`Attribute`	40.4	68.2	50.4	70.6	59.4	**64.1**	35.5	39.7	36.2	56.6	37.7	**44.9**
`Substitute`	80.0	84.3	81.9	87.9	84.8	**86.1**	75.8	72.9	74.3	78.0	76.1	**76.9**
`List`	70.9	58.1	63.5	76.2	57.8	**65.5**	58.7	53.4	55.6	65.2	53.3	**58.5**
`Location`	93.8	86.4	**89.1**	93.8	86.4	**89.1**	70.3	37.2	**47.2**	70.3	37.2	**47.2**

Table 7.3: Performance of STRs learned by ASTRL and the smallest valid STRs in identifying comma types *and* generating entailed sentences.

We now present a detailed analysis of the performance of the algorithm for the full task of comma resolution. Since this is the first work that deals with the task, we could not compare our results to previous work. Also, there is no clear baseline to use. We tried a variant of the most frequent baseline common in other disambiguation tasks, in which we labeled all commas as `Other` (the most frequent type) except when there are list indicators like *and*, *or* and *but* in adjacent chunks (which are obtained using a shallow parser), in which case the commas are labeled `List`. This gives an average precision 0.85 and an average recall of 0.36 for identifying the comma type. However, this baseline does not help in identifying the other relations.

We use the following approach to evaluate the comma type resolution *and* relation extraction performance – a relation extracted by the system is considered correct only if both the type of the comma structure *and* the set of entailed sentences are correctly identified. We call this metric the *Relation-Type metric*. Another way of measuring the performance of comma resolution is to measure the correctness of the entailed sentences per comma type. In both cases, lists are scored as in the Relation metric. The performance of our system with respect to these two metrics are presented in Table 7.3. In this table, we also compare the performance of the STRs learned by ASTRL with the smallest valid STRs without further specialization (i.e., using just the procedure outlined in Section 7.4.3).

There is an important difference between the Relation metric (Table 7.2) and the Relation-type metric (top part of Table 7.3) that depends on the semantic interpretation of the comma types. For example, consider the sentence *John Smith, 59, went home.* If the system labels the commas in this as both `Attribute` and `Substitute`, then, both will generate the output "John Smith is 59." According to the Relation metric, there is no difference between them. However, there is a semantic difference between the two sentences – the `Attribute` relation says that being 59 is an attribute of John Smith while the `Substitute` relation says that John Smith is the number 59. This difference is accounted for by the Relation-Type metric.

From this standpoint, we can see that the specialization step performed in the full ASTRL algorithm greatly helps in disambiguating between the `Attribute` and `Substitute` types and consequently, the Relation-Type metric shows an error reduction of 23.5% and 13.8% in the Gold-Gold and Gold-Charniak settings respectively. In the Gold-Gold scenario the performance of ASTRL is much better than in the Gold-Charniak scenario. This reflects the non-perfect performance of the parser in annotating these sentences (parser F-score of 90%).

Another key evaluation question is the performance of the method in identification of the `Other` category. A comma is judged to be as `Other` if no STR in the system applies to it. The performance of ASTRL in this aspect is presented in Table 7.4. The categorization of this category is important if we wish to further classify the `Other` commas into finer categories.

Setting	P	R	F
Gold-Gold	78.9	92.8	85.2
Gold-Charniak	72.5	92.2	81.2

Table 7.4: ASTRL performance (precision, recall and F-score) for `Other` identification.

7.6 Conclusion

In this chapter, we defined the task of comma resolution and designed a comma annotation scheme, where each comma structure is assigned to one of four types. We described a learning algorithm that learns Sentence Transformation Rules to perform this task. Though the size of the annotated corpus is small, we are able to learn to predict the comma relations by biasing the learning procedure to consider the smallest explanations possible that can effectively account for each annotated example. This work, in addition to having immediate significance for natural language processing systems that use semantic content, has potential applications in improving a range of automated analysis by decomposing complex sentences into a set of simpler sentences that capture the same meaning.

Chapter 8

Jointly Predicting Verb and Preposition Relations

So far, we have seen that semantic relations can be triggered by verbs, nominalizations, commas and prepositions. For each type of relation, we developed independent systems that identify and predict relations. In this chapter[1], we argue that the different phenomena interact with each other by constraining the possible interpretations that they can take, suggesting that there can be a benefit in predicting them together. However, modeling the different phenomena jointly is challenging because large jointly labeled corpora do not exist. This motivates the need for novel learning and inference schemes that address the data problem and can still benefit from the interactions among the phenomena.

We propose a joint inference scheme that combines *existing* structure predictors for multiple linguistic phenomena. We do so using hard constraints that involve only the labels of the phenomena. The strength of this model is that is easily extensible, since adding new phenomena does not require fully retraining the joint model from scratch. Furthermore, our approach minimizes the need for extensive jointly labeled corpora and, instead, uses existing predictors as black boxes. Using a small portion of the Penn Treebank that is annotated with preposition relations, we report the results of a case study where we jointly predict verb and preposition relations, improving both.

The rest of this chapter is organized as follows: We motivate the need for joint inference in Section 8.1 and define the global model in Section 8.2. Section 8.3 provides details on the coherence constraints we use and demonstrates the effectiveness of the joint model through experiments. Section 8.4 discusses our approach in comparison to existing work.

[1]This chapter is based on Srikumar and Roth (2011).

8.1 The Need for Joint Inference

Relations expressed by different linguistic phenomena often overlap. For example, consider the following sentence:

(22) Construction of the library began in 1968.

The relation expressed by the nominalization *construction* recognizes *the library* as the argument of the predicate *construct*. We can arrive at the same conclusion using the preposition *of*, which triggers the relation `ObjectOfVerb`. We see from the definition of this relation (from Appendix C) that "the object of the preposition is an object of the verb or the nominalization that is the governor of the preposition." A similar redundancy can be observed with analyses of the verb *began* and the preposition *in*. The above example motivates the following intuition: *The correct interpretation of a sentence is the one that gives a consistent analysis across all the linguistic phenomena expressed in it.*

An inference mechanism that simultaneously predicts the structure for different phenomena should account for consistency between the phenomena. A model designed to address this has the following desiderata:

1. It should account for the dependencies between phenomena.

2. It should be extensible to allow easy addition of new linguistic phenomena.

3. It should be able to leverage existing state-of-the-art models with minimal use of jointly labeled data, which is expensive to obtain.

The systems that we have seen thus far are trained on each task independently and do not account for the interplay between them. One approach for accounting for the dependencies is to define pipelines, where the predictions for one of the tasks acts as the input for another. However, a pipeline does not capture the two-way dependency between the tasks. Training a fully joint model from scratch is unrealistic from a data perspective because it requires text that is annotated for all the tasks, thus making joint training implausible. Another alternative is to use the latent learning algorithm described in Chapter 6 by treating the interactions as latent variables. However, this significantly increases the complexity of learning.

8.2 A Joint Model for Verbs and Prepositions

We seek to jointly model verb SRL and preposition relations, using the independently trained systems as black boxes. Our starting point is the verb SRL system from Chapter 5 that is trained on the PropBank corpus. We only consider the argument labeling aspect of the verb SRL problem here, but the approach presented here can be extended to include verb sense disambiguation as well. For prepositions, we train a preposition relation predictor using the Model 1 from 6 with the **basic+gen** setting.

We will use a preposition relation corpus derived from the preposition sense annotation of Dahlmeier et al. (2009) on sections 2-4 and 23 of the Wall Street Journal portion of the Penn TreeBank[2].

Our approach extends the strategy of training individual models and combining them at inference time, proposed by Roth and Yih (2004). Our joint model is a Constrained Conditional Model (Chang et al., 2012), which allows us to build upon existing learned models using declarative constraints.

The starting point is the observation that inference problems can be represented as instances of integer linear programs (ILPs). We saw in Chapter 5 that inference for SRL is instantiated as an ILP problem. The problem of predicting preposition relation labels using Model 1 (from Chapter 6) is a multiclass classification problem that is easily transformed into an ILP instance. Let $y_{p,r,R}$ denote the decision variable that encodes the prediction that the preposition p is assigned a relation r and let $\Theta_{p,r,R}$ denote its score. Let $\mathbf{y}_R$ denote all the relation variables for a sentence. Then preposition relation prediction is equivalent to the following maximization problem:

$$\max_{\mathbf{y}_R} \quad \sum_{p,r} \Theta_{p,r,R} \cdot y_{p,r,R} \tag{8.1}$$

$$\text{subj. to} \quad \sum_{r} y_{p,r,R} = 1, \quad \forall p \tag{8.2}$$

$$y_{p,r,R} \in \{0,1\}, \quad \forall p, r. \tag{8.3}$$

In general, let p denote a linguistic structure prediction task of interest and let $\mathcal{P}$ denote all such

[2]This dataset does not annotate all prepositions and restricts itself mainly to prepositions that start a Propbank argument. The data annotates seven prepositions (*at, for, in, of, on, to* and *with*) and is available at http://nlp.comp.nus.edu.sg/corpora. These seven prepositions produce 22 relation labels.

tasks. Let $\mathcal{Z}_p$ denote the set of labels that the parts of the structure associated with phenomenon p can take. For example, for the SRL argument classification component, the parts of the structure are all the candidates that need to be labeled for a given sentence and the set $\mathcal{Z}_p$ is the set of all argument labels. For each phenomenon $p \in \mathcal{P}$, we use $\mathbf{y}_p$ to denote its set of inference variables for a given sentence. Each inference variable $y_{u,z,p} \in \mathbf{y}_p$ corresponds to the prediction that the part u has the label $z \in \mathcal{Z}_p$ in the final structure. This variable is associated with a score $\Theta_{u,z,p}$ that is obtained from a learned score predictor. Let $\mathcal{C}_p$ denote the structural constraints that are "local" to the phenomenon. Thus, for verb SRL, these would be the constraints defined in the previous section, and for preposition relation, the only local constraint would be the constraint (8.2) defined above.

With this notation, the independent inference problem for the phenomenon p is the following integer program:

$$\max_{\mathbf{y}_p} \quad \sum_{z \in \mathcal{Z}_p} \sum_{u} y_{u,z,p} \cdot \Theta_{u,z,p}, \tag{8.4}$$

$$\text{subj. to} \quad \mathcal{C}_p(\mathbf{y}_p), \tag{8.5}$$

$$y_{u,z,p} \in \{0,1\}, \qquad \forall z, u. \tag{8.6}$$

8.2.1 Joint Inference

We consider the problem of jointly predicting several phenomena incorporating linguistic knowledge that enforces consistency between the output labels. Suppose p_1 and p_2 are two phenomena. If z_{p_1} is a label associated with the former and $z_{1,p_2}, z_{2,p_2}, \cdots$ are labels associated with the latter, we consider constraints of the form

$$z_{p_1} \rightarrow z_{1,p_2} \vee z_{2,p_2} \vee \cdots \tag{8.7}$$

We expand this language of constraints by allowing the specification of pre-conditions for a constraint to apply. This allows us to enforce constraints of the form *"If an argument that starts with the preposition 'at' is labeled `AM-TMP`, then the preposition can be labeled either `Numeric` or `Temporal`."* This constraint is universally quantified for all arguments that satisfy the precondition of starting with the preposition *at*.

Given a first-order constraint in this form and an input sentence, suppose the inference variable y_{p_1} is a grounding of z_{p_1} and $y_{1,p_2}, y_{2,p_2}, \cdots$ are groundings of the right hand labels such that the preconditions are satisfied, then the constraint can be phrased as the following linear inequality.

$$-y_{p_1} + \sum_i y_{i,p_2} \geq 0. \tag{8.8}$$

In the context of the preposition relations and verb SRL, we consider constraints between labels for a preposition and SRL argument candidates that begin with that preposition. This restriction forms the pre-condition for all the joint constraints considered in this chapter. Since the joint constraints involve only the labels, they can be derived either manually from the definition of the tasks or using statistical relation learning techniques. In addition to mining constraints of the form (8.7), we also use manually specified joint constraints. The constraints used in our experiments are described further in Section 8.3.

In general, let J denote a set of pairwise joint constraints. The joint inference problem can be phrased as that of maximizing the score of the assignment subject to the structural constraints of each phenomenon ($\mathcal{C}_p$) *and* the joint linguistic constraints (J). However, since, the individual tasks were not trained on the same datasets, the scoring functions need not be in the same numeric scale. In our model, each label z for a phenomenon p is associated with a scoring function $\Theta_{u,z,p}$ for a part y. To scale the scoring functions, we associate each label with a parameter $\lambda_{z,p}$. This gives us the following integer linear program for joint inference:

$$\begin{aligned}
\max_{\mathbf{y}} \quad & \sum_{p \in \mathcal{P}} \sum_{z \in \mathcal{Z}_p} \lambda_{z,p} \left(\sum_u y_{u,z,p} \cdot \Theta_{u,z,p} \right), && && (8.9)\\
\text{subj. to} \quad & \mathcal{C}_p(\mathbf{y}_p), && \forall p \in \mathcal{P}, && (8.10)\\
& J(\mathbf{y}), && && (8.11)\\
& y_{u,z,p} \in \{0,1\}, && \forall z, u, p. && (8.12)
\end{aligned}$$

Here, $\mathbf{y}$ is the vector of inference variables which is obtained by stacking all the inference variables of each phenomena.

For our experiments, we use a cutting plane solver to solve the integer linear program. (See Chapter 2 for a description of this method.) This allows us to solve the inference problem without explicitly having to instantiate all the joint constraints.

8.2.2 Learning to Rescale the Individual Systems

Given the individual models and the constraints, we only need to learn the scaling parameters $\lambda_{z,p}$. Note that the number of scaling parameters is the total number of labels. When we jointly predict verb SRL and preposition relations, we have 22 preposition relations (for the seven prepositions annotated in the Treebank data), one SRL identifier label and 54 SRL argument classifier labels. Thus we learn only 77 parameters for our joint model. This means that we only need a very small dataset that is jointly annotated with all the phenomena.

We use the structured Perceptron algorithm of Collins (2002) to learn the scaling weights. Note that for learning the scaling weights, we need each label to be associated with a real-valued feature. Given an assignment of the inference variables $\mathbf{y}$, the value of the feature corresponding to the label z of task p is given by the sum of scores of all parts in the structure for p that have been assigned this label, i.e. $\sum_{u_p} y_{u,z,p} \cdot \Theta_{u,z,p}$. This feature is computed for the gold and the predicted structures and is used for updating the weights.

8.3 Experiments and Results

In this section, we describe our experimental setup and evaluate the performance of our approach. The research question addressed by the experiments is the following: *Given independently trained systems for verb SRL and preposition relations, can their performance be improved using joint inference between the two tasks?* To address this, we report the results of the following two experiments:

1. First, we compare the joint system against the baseline systems and with pipelines in both directions. In this setting, both base systems are trained on the Penn Treebank data.

2. Second, we show that using joint inference can provide a strong performance gain even when the underlying systems are trained on different domains.

In all experiments, we report the F1 measure for the verb SRL performance using the CoNLL 2005 evaluation metric and the accuracy for the preposition relation labeling task.

8.3.1 Data and Constraints

For both the verb SRL and preposition relations, we used the first 500 sentences of section 2 of the Penn Treebank corpus to train our scaling parameters. For the first set of experiments, we trained our underlying systems on the rest of the available Penn Treebank training data for each task. For the adaptation experiment, we train the relation classifier on the SemEval data (restricted to the same Treebank prepositions). In both cases, we report performance on section 23 of the Treebank.

We mined consistency constraints from the sections 2, 3 and 4 of the Treebank data. As mentioned in Section 8.2.1, we considered joint constraints relating preposition relation to verb argument candidates that start with the preposition. We identified the following types of constraints: (1) For each preposition, the set of invalid verb arguments and preposition relations. (2) For each preposition relation, the set of allowed verb argument labels if the role occurred more than ten times in the data, and (3) For each verb argument, the set of allowed preposition relations, similarly with a support of ten. Note that, while the constraints were obtained from jointly labeled data, the constraints could be written down because they encode linguistic intuitions about the labels.

$$
\begin{aligned}
\text{srl-arg}(\texttt{A2}) \quad \rightarrow \quad & \text{prep-relation}(\texttt{Attribute}) \\
\vee \quad & \text{prep-relation}(\texttt{Cause}) \\
\vee \quad & \text{prep-relation}(\texttt{Instrument}) \\
\vee \quad & \text{prep-relation}(\texttt{ObjectOfVerb}) \\
\vee \quad & \text{prep-relation}(\texttt{PartWhole}) \\
\vee \quad & \text{prep-relation}(\texttt{Participant/Accompanier}) \\
\vee \quad & \text{prep-relation}(\texttt{ProfessionalAspect}).
\end{aligned}
$$

Figure 8.1: Example of a constraint that captures the interaction between verb arguments and preposition relations.

Figure 8.1 shows a constraint extracted from the data, which applies to the preposition *with*. This constraint says that if any candidate that starts with *with* is labeled as an A2, then the preposition can be labeled only with one of the relations on the right hand side.

Some of the mined constraints have negated variables to enforce that a preposition relation or a verb argument label should not be allowed. These can be similarly converted to linear inequalities.

See Rizzolo and Roth (2007) for a further discussion about converting logical expressions into linear constraints.

In addition to these constraints that were mined from data, we also enforce the following hand-written constraints: (1) If the relation label of a verb attached preposition is labeled TEMPORAL, then there should be a verb predicate for which this prepositional phrase is labeled AM-TMP. (2) For verb attached prepositions, if the preposition is labeled with one of ACTIVITY, ENDCONDITION, INSTRUMENT or PROFESSIONALASPECT, there should be at least one predicate for which the corresponding prepositional phrase is not labeled Ø.

The conversion of the first constraint to a linear inequality is similar to the earlier cases. For each of the relations in the second constraint, let r denote a relation variable that assigns the label to some preposition. Suppose there are n SRL candidates across all verb predicates that begin with that preposition, and let $s_1, s_2, \cdots, s_n$ denote the SRL variables that assign these candidates to the label Ø. Then the second constraint corresponds to the following inequality:

$$r + \sum_{i=1}^{n} s_i \leq n. \tag{8.13}$$

8.3.2 Results of Joint Learning

First, we compare our approach to the performance of the baseline independent systems and to pipelines in both directions in Table 8.1. For one pipeline, we added the prediction of the baseline preposition relation system as an additional feature to both the identifier and the argument classifier for argument candidates that start with a preposition. Similarly, for the second pipeline, we added the SRL predictions as features for prepositions that were the first word of an SRL argument. In all cases, we performed five-fold cross validation to train the classifiers.

As discussed in section 5.6, for verb SRL, we treated the identifier and classifier as two separate phenomena and trained re-rescaling weights for them separately. By doing so, we do not prune away any argument candidates that might eventually be labeled as an argument because of the joint inference.

The results show that both pipelines improve performance. This justifies the need for a joint system because the pipeline can improve only one of the tasks. The last line of the table shows

Setting	SRL (F1)	Preposition Role (Accuracy)
Baseline SRL	76.22	–
Baseline Prep.	–	67.82
Prep. → SRL	**76.84**	–
SRL → Prep.	–	**68.55**
Joint inference	**77.07**	**68.39**

Table 8.1: Performance of the joint system, compared to the individual systems and the pipelines. All performance measures are reported on Section 23 of the Penn Treebank. The verb SRL systems were trained on sections 2-21, while the preposition relation classifiers were trained on sections 2-4. For the joint inference system, the scaling parameters were trained on the first 500 sentences of section 2, which were held out. All the improvements in this table are statistically significant at the 0.05 level.

that the joint inference system improves upon both the baselines. We achieve this improvement without retraining the underlying models, as done in the case of the pipelines.

On analyzing the output of the systems, we found that the SRL precision improved by 2.75% but the recall decreased by 0.98%, contributing to the overall F1 improvement. The decrease in recall is due to the joint hard constraints that prohibit certain assignments to the variables which would have otherwise been possible. Note that, for a given sentence, even if the joint constraints affect only a few argument candidates directly, they can alter the labels of the other candidates via the "local" SRL constraints.

Consider the following example of the system output which highlights the effect of the constraints.

(23) Weatherford said market conditions led to the cancellation of the planned exchange.

The independent preposition relation system incorrectly identifies the *to* as a Location. The semantic role labeling component identifies the phrase *to the cancellation of the planned exchange* as the A2 of the verb *led*. One of the constraints, mined from the data, prohibits the label Location for the preposition *to* if the argument it starts is labeled A2. This forces the system to change the preposition label to the correct one, namely EndCondition. Both the independent and the joint systems also label the preposition *of* as ObjectOfVerb, which indicates that the phrase *the planned exchange* is the object of the deverbal noun *cancellation*.

8.3.3 Effect of Constraints on Adaptation

Our second experiment compares the performance of the preposition relation classifier that has been trained on the SemEval dataset with and without joint constraints. Table 8.2 shows these results.

We see that the SemEval-trained preposition relation classifier (baseline in the table) gets an accuracy of 53.29% when tested on the Treebank dataset. This drop in performance is because of a change in vocabulary between the two domains. Using this classifier jointly with the verb SRL classifier via joint constraints gets an improvement of almost 3 percent in accuracy. The primary reason for this improvement, even without re-training the classifier, is that the constraints are defined using only the labels of the systems. This avoids the standard adaptation problems of differing vocabularies and unseen features.

Setting	**Preposition Role** (Accuracy)
Baseline	53.29
Joint inference	**56.22**

Table 8.2: Performance of the SemEval-trained preposition relation classifier, when tested on the Treebank dataset with and without joint inference with the verb SRL system. The improvement, in this case is statistically significant at the 0.01 level using the sign test.

8.4 Discussion

Roth and Yih (2004) formulated the problem of extracting entities and relations as an integer linear program, allowing them to use global structural constraints at inference time even though the component classifiers were trained independently. In this chapter, we use this idea to combine classifiers that were trained for two different tasks on different datasets using constraints to encode linguistic knowledge.

In the recent years, we have seen several joint models that combine two or more NLP tasks . Andrew et al. (2004) studied verb subcategorization and sense disambiguation of verbs by treating it as a problem of learning with partially labeled structures and proposed to use EM to train the joint model. Finkel and Manning (2009) modeled the task of named entity recognition together with parsing. Dahlmeier et al. (2009) addressed the problem of jointly learning verb SRL and preposition sense using the Penn Treebank annotation that was introduced in that work. The key difference between these and the model presented here lies in the simplicity of our model and its easy extensibility because it leverages existing trained systems. Moreover, our model has the advantage that the complexity of the joint parameters is small, hence does not require a large jointly labeled dataset to train the re-scaling parameters.

Our approach is conceptually similar to that of Rush et al. (2010), which combined separately trained models by enforcing agreement using global inference and solving its linear programming relaxation. They applied this idea to jointly predict dependency and phrase structure parse trees and on the task of predicting full parses together with part-of-speech tags. The main difference in our approach is that we treat the re-scaling problem as a separate learning problem in itself and train a joint model specifically for re-scaling the output of the trained systems.

The SRL combination system of Surdeanu et al. (2007) studied the combination of three different SRL systems using constraints and also by training secondary scoring functions over the individual systems. Their approach is similar to the one presented in this chapter in that, unlike standard reranking, as in Collins (2000), we entertain all possible solutions during inference, while reranking approaches train a discriminative scorer for the top-K solutions of an underlying system. Unlike the SRL combination system, however, our approach spans multiple phenomena. Moreover, in contrast to their re-scoring approaches, we do not define joint features drawn from the predictions of the underlying components to define our global model.

8.5 Conclusion

In this chapter, we argued that the different relation-inducing linguistic phenomena interact with each other by constraining the possible structures for each other. We demonstrated the validity of this argument using a case study that shows that jointly predicting the various linguistic phenomena can help improve the quality of prediction. The approach suggested here does not require extensive re-training or annotation and can be easily extended to include additional linguistic constructions.

Chapter 9

Conclusions and Future Directions

Much of the world's knowledge is in the form of text and the problem of building an automated system that understands and processes text has been an important open problem in Computer Science. Making progress towards solving this problem can not only help in providing easier access to information, but can also lead to a more natural interaction with computers. A structured predicate-argument representation of text provides a foundation to build such text understanding applications. In the literature, this problem has been studied as the NLP task of semantic role labeling. However, traditionally, only predicate-argument relations expressed by verbs and nominalizations have been considered.

In this thesis, we pointed out that semantic relations in sentences can be triggered not only by verbs and nominalizations, but also by other linguistic phenomena, such as prepositions and commas. As a human reader, we understand these structures easily, but automated systems for Information Retrieval, Question Answering, and Textual Entailment are likely to encounter problems with these structures which can be structurally similar, but express different meanings.

For a complete semantic understanding of the sentence, we need to analyze this broader set of semantic role relations. In order to systematically analyze this larger class of relations, we identified three sub-problems that are common to all types of relations: trigger detection, predicate identification and argument labeling.

We defined an ontology of relations expressed by prepositions and commas. The definitions of these corpora may themselves be the starting point for a broader annotation scheme for relations. For these two phenomena, we proposed data-driven approaches for predicting them. In both cases, we took advantage of the structure of the output to bias the learning algorithms. For the commas, the strong learning bias in the form of restrictions in the hypothesis space helped us to learn a predictor from the small set of annotated examples. For the preposition relations, knowledge of the output structure gives us better performing and sparser models despite partial annotation.

In the final part of the thesis, we pointed out that though we can think of the different phenomena as being independent, since these phenomena are, in fact, expressing information about the same sentence, there is an implicit requirement of consistency across their predictions. We show that taking advantage of this intuition and enforcing coherence across its parts can help the performance of the predictors.

9.1 Future Research Directions

The work described in this thesis opens up several interesting avenues for future research.

9.1.1 Applications of Extended Semantic Role Labeling

In Chapter 1 of this thesis, we saw two examples of text understanding problems, namely textual entailment and event extraction, and motivated the need for a semantic representation of text by observing the problems of **variability** and **ambiguity**. We showed that these problems manifest themselves not just in the case of verbs and nominalizations but also other phenomena. Can these extensions help in addressing the text understanding problems? We believe that the answer is likely affirmative.

For the textual entailment problem, Braz et al. (2005) shows that entailment can be helped by verb semantic role labeling. In Chang et al. (2010a), we used an SRL driven representation to model the TE problem and achieved promising results. In the latter case, for relations that are not captured by verbs, we backed off to the syntactic representation of the sentences. However, as the analysis of Sammons et al. (2010) showed, a significant fraction of the textual entailment decisions are dependent on uncovering the meaning of precisely such *implicit* relations, which are discussed in this thesis.

For the event extraction problem, a preliminary analysis shows that many event arguments align with arguments of the various linguistic phenomena discussed in this thesis. Indeed, by taking advantage of the representation proposed in this thesis, the learning problem for event extraction can be made easier because it can reduce the complexity of learning the predictors.

Another related line of work is that of directly converting text into a logical representation, typically lambda calculus (see, for example, Chen and Mooney (2008), Zettlemoyer and Collins (2009), Goldwasser and Roth (2011) and related work). Such work is typically restricted to a closed domain. In contrast, the representation considered in this thesis is domain independent and can, hence, serve as a starting point for these problems.

These applications of the extended SRL proposed in this thesis are indeed exciting directions. However, as noted in Section 1.4, we only predict a *shallow* semantic representation of sentences. This raises the question about what it means to assign a semantic representation to text.

9.1.2 The Question of Semantics

Assigning a semantic interpretation to text is a complex endeavor that requires not only an understanding of the thematic structure of sentences, but also knowledge about discourse, modality, quantification and entity and coreference resolution. Fillmore (1968) hypothesized that the semantics of a sentence consists of two parts: a tenseless proposition that relates verbs and their arguments, and a modality constituent that introduces negation, tense, mood and aspect.

In this thesis, we focus on only building the tenseless proposition (albeit not just from verbs). In recent years, we have seen statistical approaches for solving the other aspects of the problem of understanding – coreference resolution (e.g., Bengtson and Roth (2008) and Haghighi and Klein (2010)), entity resolution to a knowledge base (e.g., Ratinov et al. (2011)) and discourse (e.g., work based on the Penn Discourse Treebank (Miltsakaki et al., 2004)). These approaches can serve as foil to each other to build higher level text understanding components.

An interesting question about these aspects of the problem of semantics addresses the interactions between these higher level components. In this thesis, we saw that within the realm of shallow semantics interactions can help improve predictions. Can we do the same for the higher level tasks? This question motivates several machine learning challenges.

9.1.3 Structured Learning and Prediction

As the complexity of the predicted structures increases, we face two challenges from the perspective of structured learning and prediction. First, it is impossible to annotate all possible structures of interest. Second, the computational complexity of inference becomes an important question.

From the learning perspective, we need to bias the learner so that we can train robust models despite limited data. We have seen some examples of such biases in this thesis – restricting the hypothesis space, restricting the space of structures and enforcing coherence. One area of future research is to investigate other such cues that can help learning structured predictors efficiently, especially in the presence of partial, or even missing annotation (as in Chang et al. (2010b) and Yu and Joachims (2009)).

The computational complexity of inference increases as we make the output structure larger. In this thesis, we used the cutting plane inference algorithm for joint inference. One important area of future work is to investigate efficient inference algorithms. Recent work has pointed in several interesting directions such as amortizing inference costs over the lifetime of the predictor (Srikumar et al., 2012) and decomposing inference over its parts. Advances in the areas of structured learning and inference will be applicable, not only to NLP tasks, but to a broad spectrum of structure prediction problems where annotated data is hard to obtain.

Appendix A

Identifier and Classifier Features for Verb and Nominal SRL

This section lists the feature set used for training the argument identifiers and argument classifiers for verb and nominal predicates. The input to the feature extractors is a span of text.

For extracting features, we pre-processed the sentences with part-of-speech tags and chunks from the Illinois NLP system (Clarke et al., 2012), parse trees from the Charniak and Johnson (2005) parser and named entities from the tagger of Ratinov and Roth (2009). We identified head words using the head percolation table of Magerman (1995) and Collins (1999).

Verb SRL

The Verb SRL features are based on those defined in Punyakanok et al. (2008). We only list the extra features here and refer the reader to the original paper for details.

Argument Identifier

1. All the features from Punyakanok et al. (2008).

2. **Named entity embedding**: tells if a constituent *is*, *embeds*, *exclusively overlaps* or *is embedded in* a named entity with its type. This is comparable to the **Chunk** feature in Punyakanok et al. (2008), using named entities.

3. The **projected path** feature defined in Section 4.1 of Toutanova et al. (2008).

4. The phrase label, head word and part-of-speech tag of the left and right siblings of the candidate in the parse tree. These features are from Pradhan et al. (2004).

5. If the parent is a prepositional phrase, its head word and the head word and part-of-speech tag of its right most noun phrase child. These features are also from Pradhan et al. (2004).

In addition, we also included feature conjunctions of all the predicate-specific features from Punyakanok et al. (2008) with the path features, the linear position (both from Punyakanok et al. (2008)) and the prepositional phrase specific features described above.

Argument Classifier

For the verb argument classifier, we included possible conjunctions of the identifier features (instead of specific conjunctions as described above).

Nominal SRL

Nominal Argument Identifier

1. **Head word features**: From the head word of the input constituent, we extract the lower-cased word, its part-of-speech tag, an indicator for whether it is a number, an indicator for whether it ends with *-ing*, an indicator for whether it is a known nominalization and indicators for whether it is a day of the week a month or a valid date.

2. The **linear position** of the argument, in comparison to the predicate (one of *before*, *after* or *contained*).

3. **Parse phrase**: The label of the lowest parse tree node that contains the argument.

4. Chunk path, chunk embedding and parse path, as described in Punyakanok et al. (2008).

5. Named entity embedding, described above

6. All lower cased words and their part-of-speech tags withing a window of size 5, centered at the head word, conjoined with their location relative to the head.

7. The subcategorization frame as extracted for verbs.

8. **Predicate features**: From the predicate, the lower-cased word, its part-of-speech tag, lemma, indicator for capitalization, all NomLex classes to which it belongs

In addition, we also included conjoined the predicate and the subcategorization features with the parse path, the linear position and the parse phrase.

Nominal Argument Classifier

For the nominal argument classifier, we added all the features (except the specific conjunctions) described for the argument identifier. We modified the head word features to include indicators for the most frequent WordNet synset and all its hypernyms. In addition, we also added all possible feature conjunctions, as in the verb case.

Appendix B

Feature Extraction Details

This chapter provides details about about features used in this thesis. All the features were implemented using the Edison API[1] using the NLP Curator (Clarke et al., 2012) to provide annotations.

List-based Features

These features, used for sense disambiguation of verbs, nominalizations and prepositions (Chapters 5, 6), are identifiers for whether an input constituent starts or ends with a known prefix or suffix.

- **Deverbal suffixes**: The input word ends with one of *-ate, -en, -fy, -ise, -ize.*
- **Denominal noun producing suffixes**: The input word ends with one of *-ade, -age, -ana, -arian, -ary, -dom, -eer, -er, -ery, -ese, ess, -ette, -ette, -ful, -hood, -ian, -ing, -ing, -ism, -ism, -ist, -ist, ite, -itis, -let, -ling, -nik, -oid, -scape, -ship, -ster.*
- **De-adjectival suffixes**: The input word ends with one of *ability, -ancy, -hood, -ity, ness, -ship.*
- **Prefixes indicating number, degree or order**: The word starts with one of *poly, ultra, post, multi, pre, fore, ante, pro, meta, out*

Conflated Part-of-speech Tags

Based on Tratz and Hovy (2009); Hovy et al. (2010), we collapsed the part-of-speech tagset into one of `Noun`, `Verb`, `Adjective`, `Adverb`, `Punctuation`, `Pronoun` and `Other`. These features provide an abstraction over the finer distinctions provided by the full set of Penn Treebank part-of-speech tags for training the preposition relation predictor (Chapter 6). An alternative is to use the universal part-of-speech tagset proposed by Petrov et al. (2011).

[1] We used the version 0.2.9 available at http://cogcomp.cs.illinois.edu/software/edison.

Negation and Modality Indicator

This feature is useful for verb SRL. For identifying negations, we check if the shallow parse chunk containing the verb predicate also contains one of *not* or *n't*. To identify if the verb predicate has is associated with a modal verb, we check if the chunk contains the part-of-speech tag `MD`.

Appendix C

Preposition Relation Definitions

This chapter lists a taxonomy of relations expressed by prepositions. These relations are defined by collapsing related preposition senses from the inventory defined by the Preposition Project Litkowski and Hargraves (2005)[1]. The sense inventory, which is based on the definitions of prepositions in the Oxford Dictionary of English, treats each dictionary definition of the preposition as a separate sense and has been used by the 2007 shared task of preposition sense disambiguation Litkowski and Hargraves (2007).

By grouping semantically related senses across prepositions into relations, we define 32 relation labels covering the 34 prepositions[2] considered by the Preposition Project. One of the labels is a catch-all category, `Other`, which consists of many infrequent senses of prepositions and is not expected to be a semantically coherent class. Section C.1 defines all the relations and provides examples of prepositional phrases that exhibit them. Section C.2 lists the number of training and test examples obtained by converting the SemEval 2007 preposition sense data using the mapping defined in Section C.1.

C.1 Preposition Relations

For each relation that is defined in this section, we list the set of prepositions and their senses that are included as part of this relation. The sense identifiers refer to those used by the the SemEval 2007 shared task. Senses that are infrequent according to the training data for the shared task (that is, seen fewer than five times) are marked with an asterisk.

[1]The preposition sense inventory can be accessed online at http://www.clres.com/cgi-bin/onlineTPP/find_prep.cgi.

[2]We cover the transitive usages of the following prepositions: about, above, across, after, against, along, among, around, as, at, before, behind, beneath, beside, between, by, down, during, for, from, in, inside, into, like, of, off, on, onto, over, round, through, to, towards, with.

Preposition	Senses	Examples		
at	8(4a)	good	at	boxing
in	11(8)	prudent	in	planning
into	9(9)*	he is	into	jet-skiing
on	19(9)*	out	on	errands
through	8(2a)*	struggle	through	it
to	10(4a)	prelude	to	disaster

Table C.1: The list of preposition senses that define the relation `Activity`

Preposition	Senses	Examples		
by	1(1)* 2(1a) 3(1b)	understood	by	the customers
		the address	by	the officer
from	12(9)-1	allegations	from	the FDA

Table C.2: The list of preposition senses that define the relation `Agent`

C.1.1 Activity

Definition: Describes the relationship between some entity and an activity, an ordeal or a process that can be a verbal noun or a gerund. Note that the object of the preposition is the activity here. (See Table C.1 for the mapping from preposition senses to `Activity`.)

Comments: If there is a change of state and the object of the preposition is the final state, the preposition indicates the relation `EndState`.

C.1.2 Agent

Definition: The object of the preposition is the agent of the action indicated by the attachment point. This primarily covers the use of the preposition *by* in a passive construction. (See Table C.2 for the mapping from preposition senses to `Agent`.)

Comments: This relation does not include prepositional phrases like *sonnet by Shakespeare* because *Shakespeare* is the agent of a verb such as write or compose, and not the word *sonnet*. This type of creator-creation relation will be labeled as `Source`.

Preposition	Senses	Examples		
as	1(1)	interpreted their action	as	betrayal
at	7(4)*	sell	at	a loss
by	15(3c) 6(2a)* 7(2b)*	breakdown	by	age
		what is meant	by	"fair"?
		call	by	last name
from	9(6)	paint made	from	resin
in	1(1)-1 10(7a)* 6(4a) 9(7)-1	blinked	in	astonishment
		dressed	in	a sweater
		Mozart's Piano Concerto	in	E flat
of	17(8) 6(3)-1	house built	of	bricks
		smile	of	delight
with	2(2) 3(2a)	man	with	glasses
		blouse	with	white collar

Table C.3: The list of preposition senses that define the relation `Attribute`

Preposition	Senses	Examples		
after	4(1c)*	clean up	after	him
behind	5(3)*	throw weight	behind	candidate
for	1(1) 3(3)	fought	for	Napoleon
		vote	for	independence
		present	for	you
to	8(3)-1	be charming	to	them

Table C.4: The list of preposition senses that define the relation `Beneficiary`

C.1.3 Attribute

Definition: The object of the preposition indicates some attribute of either the governor or the subject/object of the governor in case of verb-attached prepositions. (See Table C.3 for the mapping from preposition senses to `Attribute`.)

Comments:

1. Prepositions where the governor indicates an attribute are classified as `Possessor`.

2. If the object of the preposition does not indicate an attribute, this relation does not apply.

C.1.4 Beneficiary

Definition: The object of the preposition is a beneficiary of the action indicated by the preposition's governor. (See Table C.4 for the mapping from preposition senses to `Beneficiary`.)

Preposition	**Senses**	**Examples**		
after	1(1)-1	tired	after	work
at	11(6)-1	disapproval	at	the behavior
for	6(5)	admire	for	bravery
from	12(9)	suffering	from	asthma
of	16(7b)	died	of	cancer
with	10(7a)* 11(7b) 12(7c)	in bed	with	flu
		shaking	with	anger
		wisdom comes	with	age

Table C.5: The list of preposition senses that define the relation `Cause`

Preposition	**Senses**	**Examples**		
among	3(3) 4(4)*	drop in tooth decay	among	children
		divide his kingdom	among	them
between	4(4) 6(4b) 8(5)* 9(5a)*	links	between	science and industry
		the difference	between	income and expenditure
		choose	between	two options

Table C.6: The list of preposition senses that define the relation `Co-Participants`

Comments: This relation includes examples like *the power behind the throne*, where the phrase indicates that the object of the preposition gains some advantage.

C.1.5 Cause

Definition: The object indicates a cause for the governor. (See Table C.5 for the mapping from preposition senses to `Cause`.)

C.1.6 Co-Participants

Definition: The preposition indicates a choice, sharing or differentiation and involves multiple participants, represented by the object. (See Table C.6 for the mapping from preposition senses to `Co-Participants`.)

Comments: This is different from `Participant/Accompanier` where the governor and the object indicate the two entities that are associated. Here, the object specifies all the participants.

Preposition	Senses	Examples		
for	7(6)*	leaving	for	London tomorrow
in	2(1a)	put coal	in	the bath
inside	2(1a)	tucked the books	inside	his coat
into	1(1) 2(2) 3(3)	Sara got	into	her car
		crashed	into	a parked car
		the road led	into	the village
to	1(1)	walking	to	the shops

Table C.7: The list of preposition senses that define the relation `Destination`

C.1.7 Destination

Definition: The object of the preposition is the destination of the motion indicated by the governor. (See Table C.7 for the mapping from preposition senses to `Destination`.)

Comments:

1. One boundary case which is not a `Destination` is that of a motion that indicates that two things are attached or lined. Such cases indicate the relation `Participant/Accompanier`. For example, *fasten the insulator to the frame* might indicate motion of the insulator to the frame, but Participant/Accompanier is a better choice because it indicates that the two objects are linked to each other.

2. If there is not physical motion, but merely a change of state, the relation `EndState` is preferred.

C.1.8 Direction

Definition: The prepositional phrase (that is, the preposition along with the object) indicates a direction that modifies the action which is expressed by the governor. (See Table C.8 for the mapping from preposition senses to `Direction`.)

Comments:

1. This relation sometimes shares meaning with the relation `Location`.

2. This relation does not include metaphorical meaning and almost always has a spatial sense

Preposition	Senses	Examples		
after	5(2) 6(2a)*	shut the door	after	her
along	1(1)	driving	along	the road
behind	4(2a)	crept up	behind	him
by	19(5a)*	drive	by	the house
down	1(1) 2(1a) 3(1b)	tears streaming	down	her face
		wander	down	the road
off	1(1)	roll	off	the bed
towards	1(1)	drive	towards	the house
with	15(9)*	swim	with	the current

Table C.8: The list of preposition senses that define the relation `Direction`

Preposition	Senses	Examples		
into	6(6) 7(7)	protest turned	into	a violent confrontation
		forced the club	into	a special meeting
to	3(1b)* 5(2)* 6(2a)	profits dropped	to	$75 million
		she was moved	to	tears
		smashed	to	smithereens
towards	2(1a)*	return	towards	traditional Keynesianism

Table C.9: The list of preposition senses that define the relation `EndState`

C.1.9 EndState

Definition: The object of the preposition indicates the resultant state of an entity that is undergoing change. The governor of the preposition is usually a verb or noun denoting transformation. (See Table C.9 for the mapping from preposition senses to `EndState`.)

Comments:

1. This relation requires some change of state, where the object of the preposition indicates the final state at the end of the transformation.

2. This relation includes cases like *level fell to 3 feet*, where the object of the preposition is a numeric quantity. See note for `Numeric`.

C.1.10 Experiencer

Definition: The object of the preposition denotes the entity that is target or subject of the action, thought or feeling that is usually indicated by the governor. (See Table C.10 for the mapping from preposition senses to `Experiencer`.)

Preposition	Senses	Examples		
on	11(5) 11(5)-1	focus attention	on	her
		he blamed it	on	her
towards	4(2) 4(2)-1	felt angry	towards	him

Table C.10: The list of preposition senses that define the relation `Experiencer`

Preposition	Senses	Examples		
at	11(6)*	hold	at	knifepoint
by	5(2)	provide capital	by	borrowing
on	3(1b)	banged his head	on	the beam
over	15(6)* 15(6)-1*	voice	over	the loudspeaker
through	13(5a)	heard	through	the grapevine
with	4(3) 5(3a)	cut the fish	with	a knife
		fill the bowl	with	water

Table C.11: The list of preposition senses that define the relation `Instrument`

Comments: There are some examples where the relation might overlap with `Topic`. The sense on:11(5) includes both phrases like *focus attention on the her*, which conform to the definition of `Experiencer`, and also *deliberate on the matter*, which are better suited to be labeled `Topic`. However, in this annotation, the sense is labeled as `Experiencer`.

C.1.11 Instrument

Definition: The object of the preposition indicates the means or the instrument for performing an action that is typically the governor. (See Table C.11 for the mapping from preposition senses to `Instrument`.)

Comments:

1. This relation includes both the means of performing actions and instruments. Though there are subtle differences between them, they are collapsed into one category to avoid sparse labels.

2. The sense at:11(6) refers to idiomatic uses, where the object of the preposition can be words *knifepoint* or *gunpoint*. These are marked as `Instrument` because they indicate that the action was performed with a knife or a gun.

Preposition	Senses	Examples		
about	3(2) 3(2)-1 4(3)*	look	about	the room
above	1(1)* 2(1a) 3(1b)* 4(2)	the cable runs	above	the duct
		the hills	above	the capital
		bruises	above	both eyes
across	1(1) 2(2)	travel	across	Europe
along	3(2)	parked	along	the grass
among	1(1)	hidden	among	the roots
around	1(1) 3(2) 4(3) 4(3)-1 5(4)	large depots	around	the country
at	1(1)	live	at	Conway house
before	2(2) 3(2a)	she stood	before	her
behind	1(1) 3(2)	kept	behind	the screens
beneath	1(1) 2(1a) 3(2)	labyrinths	beneath	Moscow
beside	1(1)	sat	beside	her
between	1(1)	border	between	the countries
by	18(5)	discovered	by	the roadside
down	4(1c)*	going	down	the pub
from	8(5)	see them	from	here
in	1(1) 7(5)	living	in	London
inside	1(1) 3(1b)	waiting	inside	the house
into	4(4)	wind blowing	into	your face
of	8(4)*	north	of	Watford
off	2(2)* 3(2a)	street	off	Whitehall
on	2(1a)* 7(2)	camp	on	the island
onto	1(1)	copy	onto	a disk
over	1(1)* 11(4) 12(4a)* 13(4b) 2(1a) 3(1b) 4(2)	flames	over	the city
round	1(1) 3(2) 4(2a) 5(3) 6(3a) 8(4)	went	round	the house
through	1(1) 12(5)-1* 2(1a) 3(1b) 4(1c) 5(1d) 6(1e)*	stepped	through	the doorway
to	2(1a)			

Table C.12: The list of preposition senses that define the relation `Location`

C.1.12 Location

Definition: The prepositional phrase indicates a locative meaning. This relation includes both physical and figurative aspects of location. That is, both *left it in the cupboard* and *left it in her will* are treated as `Location`s. (See note below.) (See Table C.12 for the mapping from preposition senses to `Location`.)

Comments: Categorizing preposition senses as `Location`s is not always easy. Some senses may indicate multiple relations based on the specific sentence and the sense could have ideally been split into two different labels. For example, at:1(1) includes both destinations (e.g. *they stopped at a small trattoria*) and locations (e.g. *they live at Conway House*). In such ambiguous cases, the sense has been categorized as `Location`. The preposition *in*, in the sense in:7(5) includes figurative

Preposition	Senses	Examples		
along	4(3)*	frame the definition	along	those lines
in	5(4) 6(4a)-1	blinked	in	astonishment
		disappear	in	a flash
like	1(1) 2(1a) 3(1b) 4(1c)* 5(1d) 6(2)*	plummet	like	a dive-bomber
through	12(5)	obtained	through	fraudulent means
with	7(5)	shout	with	pleasure

Table C.13: The list of preposition senses that define the relation `Manner`

meanings like *read it in a book* and has been included here because this is the closest meaning. The same holds for inside:3(1b), which includes phrases like *anger simmered inside me*. An alternative relation could be `PartWhole`. Some examples that are not a `Location` relation are:

1. when there is a comparison to a norm or a number (eg. *the food was above average*), which is an `Other` relation;

2. when there is a notion of something being attached to or coming into physical contact with something else, where `Participant/Accompanier` is a better fit;

3. when one argument of the preposition is indicated to be a member of the other, where `PartWhole` is a better fit;

4. when one object is physically supported by a surface or another object, where `PhysicalSupport` is a better choice.

C.1.13 Manner

Definition: The prepositional phrase indicates the manner in which an action is performed. The action is typically the governor of the preposition. (See Table C.13 for the mapping from preposition senses to `Manner`.)

Comments: There may be some overlap of meaning between `Manner`, `Attribute` and `Cause`, stemming from the preposition sense in:5(4). For example, consider the following phrases that are included in this sense: *pursed her lips in a silent whistle*, *shrugged in embarrassment* and *seething in outrage*.

Preposition	Senses	Examples		
in	9(7)	say it	in	French
on	12(6) 13(6a)	put your idea down	on	paper
		saw the new series	on	TV

Table C.14: The list of preposition senses that define the relation `MediumOfCommunication`

Preposition	Senses	Examples		
at	5(3) 6(3a)*	driving	at	50mph
between	3(3)			
by	12(3) 13(3a)* 16(3d)*	missed the shot	by	miles
for	10(8a) 13(11) 14(12)*	crawled	for	300 yards
into	8(8)*			
of	4(2) 5(2a)*	a boy	of	15
on	23(13)*			
to	11(4b)* 12(4c)*			

Table C.15: The list of preposition senses that define the relation `Numeric`

C.1.14 MediumOfCommunication

Definition: The prepositional phrase indicates the medium or language of some form of communication or idea. The object is, in a general sense, a 'mode of communication'. This includes languages (eg. say it in French), media like TV or the Internet (eg. saw it on a website), or specific instances of these (eg. saw it on the Sopranos). (See Table C.14 for the mapping from preposition senses to `MediumOfCommunication`.)

Comments: In boundary cases between `MediumOfCommunication` and `Attribute` (as in *Shakespeare's plays in comic book form*), the former is a better choice if it is a specific case of `Attribute`. The same argument holds for the boundary cases with `Manner`.

C.1.15 Numeric

Definition: The object of the preposition indicates a numeric quantity (age, price, percentage, etc). (See Table C.15 for the mapping from preposition senses to `Numeric`.)

Comments:

1. Several senses included in this relation do not have any examples in the training set.

Preposition	Senses	Examples		
after	7(3)	inquired	after	him
		chase	after	something
at	10(5a) 9(5)	sipped	at	his coffee
for	2(2) 2(2)-1	considerations	for	the future
from	11(8)	saved	from	death
of	11(6) 12(6a) 13(6b) 14(7) 15(7a)	the wedding	of	his daughter
		it was kind	of	you
		she tells	of	her marriage
on	9(3a)			
over	6(2b)	presided	over	the meeting
round	7(3b)*			
through	10(3) 10(3)-1*	scan	through	document
to	14(6)	a threat	to	world peace
towards	5(3)*	a grant	towards	the cost
with	15(9)-1 4(3)-1 9(7),	cross	with	her.

Table C.16: The list of preposition senses that define the relation `ObjectOfVerb`

2. The prepositions in phrases such as *drop from $105 million to $75 million* are not labeled as `Numeric`. Instead, the *from* is labeled as `StartState` and the *to* is labeled as `EndState`.

C.1.16 ObjectOfVerb

Definition: The object of the preposition is an object of the verb or the nominalization that is the governor of the preposition. This includes cases like *construction of the library*, where the object of the preposition is an object of the underlying verb. (See Table C.16 for the mapping from preposition senses to `ObjectOfVerb`.)

Comments:

1. Many of the other relations can be considered to be an object of the verb in question. However, if the sense of a preposition was better suited to a different relation, that relation has been chosen. With this caveat, we expect the object of the preposition to be a core argument for the governor according to the PropBank or the NomBank schemes.

2. This relation includes the senses for:2(2), of:13(6b), to:14(6) and with:9(7) which are very vaguely defined in the Preposition Project. This contributes to some noise in the data.

3. It is possible that some of the senses from the `Other` category could be grouped into this

Preposition	Senses	Examples		
against	1(1) 2(1a) 3(1b) 6(2b)	fight	against	crime
		gave evidence	against	him
		the match	against	Somerset
		turned up his collar	against	the wind
between	5(4a) 7(4c)	the wars	between	Russia and Poland
from	14(11)	fees are distinct	from	expenses
with	6(4)	fought	with	another man

Table C.17: The list of preposition senses that define the relation `Opponent/Contrast`

relation, if there is more evidence in the data.

Some examples which are not `ObjectOfVerb` are: 1. when there is a notion of connection or physical contact between entities, `Participant/Accompanier` is a better fit, and 2. when the object of the preposition is the target of an action or is affected by it, `Recipient` or `Beneficiary` is preferred over this label.

C.1.17 Opponent/Contrast

Definition: This relation indicates a collision, conflict or contrast and the object of the preposition refers to one or more entities involved. (See Table C.17 for the mapping from preposition senses to `Opponent/Contrast`.)

Comments: The main distinguishing criterion is that there are one or more antagonists in a conflict or in opposition. This includes cases like *pulled up the collar against the wind*. Some near misses include:

1. when there is no explicit conflict, the relation `Co-Participants` is a better fit, and,

2. when there is a hint of a conflict, without mention of the entities involved (like *turn away from appeasement* and *stagger from crisis to crisis*), the relation `Source` would be a better fit.

C.1.18 Other

Definition: This is the catch-all category, which includes infrequent senses of prepositions that do not fit any other category. A preposition being labeled as `Other` does not mean that there is no

Preposition	Senses	Examples		
above	5(2a)* 6(2b)* 7(2c)* 8(2d) 9(3)	at a level	above	the people
		married	above	her
		heard	above	the din
after	10(5a)* 8(4)* 9(5)*	a drawing	after	Millet's The Reapers
		health comes	after	housing
		named her Pauline	after	her mother
against	4(2) 5(2a)* 7(2c)* 8(3)* 9(3a)*	gritted his teeth	against	the pain
		odds were 5-1	against	them
		money loaned	against	the property
		benefits weighed	against	the costs
along	2(1a)*	picked up tips	along	the way
as	2(2)*	been ill	as	a child
before	4(3)*	placed duty	before	everything
behind	2(1a)* 6(3a)* 8(5)* 9(6)*	years	behind	them
beneath	4(2a)* 5(2b)* 6(2c)*	he was rather	beneath	the princess
		tragic life	beneath	the gloss
beside	2(1a)* 3(2)*	felt clumsy	beside	her
by	21(7)* 22(8)*	right	by	me
		swear	by	God
for	11(9) 8(7) 9(8)	'F' is	for	fascinating
		swap	for	that
		tall	for	her age
		married	for	over a year
inside	4(1c)*			
like	7(3)*	works	like	Animal Farm
on	20(10)* 21(11)* 22(12)* 9(3a)-1*	he is	on	morphine
		drinks are	on	me
over	10(3)* 5(2a)* 7(2c)* 8(2d)* 9(2e)*	married for	over	a year
		the director is	over	him
to	15(7)* 16(8)* 7(2b)*	he smiled	to	her astonishment
with	8(6)*	leave it	with	me

Table C.18: The list of preposition senses that define the relation `Other`

relation expressed by it. Neither does it indicate that there is no shared meaning across prepositions that can be abstracted. In most cases, there is such little support from the training set for this label that either adding it to an existing relation or creating a new one cannot be justified. (See Table C.18 for the mapping from preposition senses to `Other`.)

Comments: Even though this label includes many senses, as we see in C.33, this relation is has a very small representation in the data. Being a very infrequent category, this label would apply only when none of the other relations hold.

Preposition	Senses	Examples		
among	2(2)	see a friend	among	them
in	12(9)*			
of	1(1)* 2(1a) 3(1b) 3(1b)-1	sleeve	of	the coat
		a slice	of	the cake
		group	of	monks
		cup	of	soup

Table C.19: The list of preposition senses that define the relation `PartWhole`

C.1.19 PartWhole

Definition: This relation indicates that one argument is a part or member of another. It includes two distinct cases: (1) the governor of the preposition (or the subject of the governor, if the governor is a verb) is a part of the object, *and* (2) the governor is a number, a partitive noun or a container and the object is a group or substance that is modified by the governor. (See Table C.19 for the mapping from preposition senses to `PartWhole`.)

Comments:

1. The two cases were clubbed together because both indicate a notion of an object or substance being divided into parts or groups. This is distinct from the `Possessor` relation, where neither argument is divided to define the other.

2. This label indicates all quantifiers and partitive nouns that are connected to entities via the preposition. For example, *hundreds of people*, *a piece of cake*, *many of the protesters*.

3. This label also includes cases where the object of the preposition is a set of entities (or a container) and the governor is an element of that set (or the contents of the container), as in the example, snakes are among the most feared animals.

C.1.20 Participant/Accompanier

Definition: The object of the preposition indicates an entity which accompanies another entity or participates in a relation with another entity, which is typically indicated by either the governor of the preposition or the subject of the governor. (See Table C.20 for the mapping from preposition senses to `Participant/Accompanier`.)

Preposition	Senses	Examples		
by	11(2f)*			
onto	3(3)	stick the drawings	onto	a large map
to	13(5) 9(4)	he is married	to	Emma
		a map pinned	to	the wall
with	1(1)	his marriage	with	Emma

Table C.20: The list of preposition senses that define the relation `Participant/Accompanier`

Preposition	Senses	Examples		
against	10(4)	stood with her back	against	the wall
on	1(1) 4(1c) 5(1d)	a water jug	on	the table

Table C.21: The list of preposition senses that define the relation `PhysicalSupport`

Comments:

1. In this relation, the object indicates only one of the participating entities. In contrast, in the `Co-Participants` relation, the object defines all the participating entities, typically in the form of a plural or a conjunction.

2. The sense by:11(2f) is included here based on the notes on the treatment of the preposition in the Preposition Project. However, there are no examples to support this inclusion, and this may be moved to the relation `Other`.

3. All cases where something is joined to something else should be treated as instances of the relation `Participant/Accompanier`. There might be some confusion between this label and `ObjectOfVerb`, `Destination` or `Location`. Whenever there is the notion of connection between entities, this label is a better choice.

C.1.21 PhysicalSupport

Definition: The object of the preposition is physically in contact with and supports the governor (if it is an entity) or the subject of the governor (if it is a verb or a nominalization). (See Table C.21 for the mapping from preposition senses to `PhysicalSupport`.)

Comments: This may often overlap with `Location`. It is the preferred label if there is the notion of one object being in physical contact with and supported by an surface or another object.

Preposition	Senses	Examples		
about	5(3a)*	a look	about	her
by	10(2e)* 9(2d)*	a black filly	by	Guldfuerst
		his son	by	his third wife
of	6(3)	son	of	a friend
		a photograph	of	a bride
on	6(1e)*	a few pounds	on	her.

Table C.22: The list of preposition senses that define the relation `Possessor`

Examples include *knelt on the cold stone floor* and *placed the jug on the table.*

C.1.22 Possessor

Definition: The governor of the preposition is something belonging to the object or an inherent quality of the object . This relation includes familial relations. (See Table C.22 for the mapping from preposition senses to `Possessor`.)

Comments:

1. Note that the `Possessor` relation is largely represented in the training set via the preposition *of* in the sense of:6(3).

2. This relation is different from the `Attribute` relation, where the governor specifies the attribute or quality of the object.

C.1.23 ProfessionalAspect

Definition: This relation signifies a professional relationship between the governor (or the subject of the governor, if the governor is a verb) and the object of preposition, which is an employer, a profession, an institution or a business establishment. (See Table C.23 for the mapping from preposition senses to `ProfessionalAspect`.)

Comments: This is an infrequent relation that could have been merged into the `Other` category, but has been separated because it forms a distinct cluster.

Preposition	Senses	Examples		
at	4(2b)*	began performing	at	the university
for	4(3a)	tutor	for	the University
in	8(6)	works	in	publishing
on	10(4)*	serve	on	committees
with	13(8) 14(8a)*	she is	with	Inland Revenue
		bank	with	TSB

Table C.23: The list of preposition senses that define the relation `ProfessionalAspect`

Preposition	Senses	Examples		
for	5(4)	networks	for	the exchange of information
		tools	for	making the picture frame.

Table C.24: The list of preposition senses that define the relation `Purpose`

C.1.24 Purpose

Definition: The object of the preposition specifies the purpose (i.e., a result that is desired, intention or reason for existence) of the governor. (See Table C.24 for the mapping from preposition senses to `Purpose`.)

C.1.25 Recipient

Definition: The object of the preposition identifies the person or thing receiving something. (See Table C.25 for the mapping from preposition senses to `Recipient`.)

Comments: The difference between a `Recipient` and the `Beneficiary` is that in the latter case, the recipient gets some advantage from the action or event.

C.1.26 Separation

Definition: The relation indicates separation or removal. The object of the preposition is the entity that is removed. (See Table C.26 for the mapping from preposition senses to `Separation`.)

Preposition	Senses	Examples		
to	8(3)	unkind	to	her
		donated	to	the hospital

Table C.25: The list of preposition senses that define the relation `Recipient`

Preposition	Senses	Examples		
from	10(7)	the party was ousted	from	power
off	4(3) 5(3a)* 6(3b)* 7(4)*	tear the door	off	its hinges
		burden	off	my shoulders
		I stay	off	alcohol
with	16(10)	part	with	possessions

Table C.26: The list of preposition senses that define the relation `Separation`

Preposition	Senses	Examples		
by	4(1c)	book	by	Hemmingway
from	1(1) 13(10) 2(1a) 4(3)	I am	from	Hackeney
		information	from	books
of	7(3a)	paintings	of	Rembrandt

Table C.27: The list of preposition senses that define the relation `Source`

C.1.27 Source

Definition: The object of the preposition indicates the provenance of the governor or the subject of the governor. This includes cases where the object is the place of origin, the source of information or the creator of an artifact. (See Table C.27 for the mapping from preposition senses to `Source`.)

Comments:

1. While it might be possible to split this into multiple sub-relations (location, source of information and creator-creation), all but the first will end up being infrequent.

2. Some figurative uses are included into this category, such as *turning them away from appeasement* and *stagger from crisis to crisis.*

C.1.28 Species

Definition: This expresses the relationship between a general category or type and the thing being specified which belongs to the category. The governor is a noun indicating the general category and the object is an instance of that category. (See Table C.28 for the mapping from preposition senses to `Species`.)

Preposition	Senses	Examples		
of	10(5a) 9(5)	the city	of	Prague
		this type	of	book

Table C.28: The list of preposition senses that define the relation `Species`

Preposition	Senses	Examples		
from	10(7)-1 6(4) 7(4a)*	recovered	from	the disease
		a growth	from	$2.2 billion to $2.4 billion

Table C.29: The list of preposition senses that define the relation `StartState`

C.1.29 StartState

Definition: The object of the preposition indicates the state or condition that an entity has left. (See Table C.29 for the mapping from preposition senses to `StartState`.)

Comments:

1. This relation requires some entity to change state, with the object of the preposition indicating the initial state before the transformation.

2. This relation also includes cases where the preposition indicates the first element in a numeric or conceptual range.

Some examples that are not `StartState` are: 1. *the ball fell from his hands*, which is a `Source` relation because the object indicates the original physical *location* of the ball, 2. *came back from a holiday*, which is also a `Source` relation, and 3. *a steady increase from June*, which is a `Temporal` relation because the object of the preposition is a temporal expression.

C.1.30 Temporal

Definition: The *object* of the preposition specifies the time of when an event occurs, either as an explicit temporal expression or by indicating another event as a reference. In some prepositional phrases, the governor may indicate a temporal expression. These do not express a `Temporal` relation. (See Table C.30 for the mapping from preposition senses to `Temporal`.)

Preposition	Senses	Examples		
across	1(1)-1*			
after	1(1) 2(1a)* 3(1b)*	shortly	after	Christmas
at	2(2) 3(2a)*	sleep	at	nine o'clock
		cooler	at	night
before	1(1)	rest	before	dinner
behind	7(4)*			
between	2(2)*			
by	14(3b)* 17(4)* 20(6)*			
down	5(2)*			
during	1(1) 2(1a)	open	during	the party
for	12(10)	jailed	for	12 years
from	3(2) 5(3a)	run	from	ten to two
in	3(2) 4(3)*	met	in	1985
inside	5(2)*			
into	1(1)-1*			
of	18(9)*			
on	17(8) 18(8a)*	reported	on	September 26
over	14(5)*			
through	11(4)* 7(2)* 9(2b)			
to	4(1c)*	run from ten	to	two
towards	3(1b)*			

Table C.30: The list of preposition senses that define the relation `Temporal`

Preposition	Senses	Examples		
about	1(1) 2(1a)*	thinking	about	you
around	2(1a)			
into	5(5)*	insight	into	what was involved
on	8(3)	book	on	careers
over	16(7)	debate	over	unemployment
round	2(1a)*			

Table C.31: The list of preposition senses that define the relation `Topic`

C.1.31 Topic

Definition: The object of the preposition denotes the subject or topic under consideration. In many cases, the preposition can be replaced by the phrase *on the subject of*. The governor of the preposition can be a verb or nominalization that implies an action or analysis or a noun that indicates an information store. (See Table C.31 for the mapping from preposition senses to `Topic`.)

Preposition	Senses	Examples		
by	8(2c)	traveling	by	bus
on	14(7) 15(7a) 16(7b)*	he is	on	his way
		sleep	on	the plane
		got	on	the train
onto	2(2)*			
through	5(1d)-1	go	through	the tube

Table C.32: The list of preposition senses that define the relation `Via`

C.1.32 Via

Definition: This is an infrequent relation where the object of the preposition indicates a mode of transportation or a path for travel. The governor can be action indicating movement or travel or a noun denoting passengers. (See Table C.32 for the mapping from preposition senses to `Via`.)

Comments: This relation applies only when there is evidence of a physical travel or a mode of conveyance.

C.2 Statistics

Table C.33 shows the number of training and test examples for each relation, obtained by applying the mapping defined in Section C.1 to the training and test data of the SemEval-2007 shared task. Note that even though the label `Other` is associated with the highest number of senses, it has a comparatively small number of examples associated with it. The relations that have fewer examples than `Other` have considerably fewer senses that define it, indicating that they are more coherent classes.

Table C.34 shows the confusion matrix between one of our annotators and the SemEval-based annotation. (Section 6.2 gives more details about the annotation.) It is instructive to compare this confusion matrix with the one for learned models. Table C.35 shows the confusion matrix for the relation labels predicted by Model 2 from Chapter 6. By comparing these matrices, we see that the pattern of errors of the annotators and the learned system is not the same. For example, we see that the label `ObjectOfVerb` is confused with `Direction` by the annotators but never by the model. Similarly, while `Source` is confused with `StartState` by the annotators, it is a very rare error for the system. We also see pairs of labels that confuse the system, but not the annotators, such as Destination and Location.

Relation	**Train**	**Test**
Activity	63	39
Agent	367	159
Attribute	510	266
Beneficiary	205	105
Cause	591	289
Co-Particiants	112	58
Destination	1054	526
Direction	909	441
EndState	188	104
Experiencer	116	54
Instrument	565	290
Location	3096	1531
Manner	457	245
MediumOfCommunication	57	30
Numeric	113	48
ObjectOfVerb	1801	882
Opponent/Contrast	233	131
Other	72	42
PartWhole	958	471
Participant/Accompanier	292	142
PhysicalSupport	399	202
Possessor	508	269
ProfessionalAspect	45	22
Purpose	261	113
Recipient	378	190
Separation	345	172
Source	740	357
Species	394	198
StartState	69	37
Temporal	331	157
Topic	886	462
Via	61	26
Total	16176	8058

Table C.33: Number of training and test examples for preposition relations, obtained by applying the mapping defined in this thesis to the SemEval-2007 training and test data.

	Activity	Agent	Attribute	Beneficiary	Cause	Co-Participants	Destination	Direction	EndState	Experiencer	Instrument	Location	Manner	MediumOfCommunication	Numeric	ObjectOfVerb	Opponent/Contrast	Other	PartWhole	Participant/Accompanier	PhysicalSupport	Possessor	ProfessionalAspect	Purpose	Recipient	Separation	Source	Species	StartState	Temporal	Topic	Via
Activity	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1
Agent	.	5	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Attribute	.	.	6	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	.	.	.	.	.	.	.	.	.	.	.	.
Beneficiary	.	.	.	2	.	.	.	.	.	.	.	.	.	.	.	.	.	1	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Cause	.	.	.	.	4	.	.	.	.	.	.	.	1	.	.	2	.	.	.	1	.	.	.	.	.	.	.	.	.	.	.	.
Co-Participants	.	.	.	.	.	.	.	.	.	.	.	2	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Destination	.	.	.	.	.	.	14	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Direction	.	.	.	.	.	.	.	12	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
EndState	.	.	.	.	.	.	1	.	2	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Experiencer	.	.	.	.	.	.	.	.	.	2	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Instrument	.	.	.	.	.	.	.	.	.	.	5	.	.	.	.	.	.	.	.	2	.	.	.	.	.	.	.	.	.	.	.	.
Location	.	.	.	.	.	.	.	.	.	.	.	39	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Manner	.	.	.	.	.	.	.	.	.	.	.	.	6	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
MediumOfCommunication	.	.	.	.	.	.	.	.	.	.	.	.	.	1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Numeric	.	.	.	.	.	.	.	.	.	.	.	.	.	.	2	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
ObjectOfVerb	.	.	.	.	.	.	.	5	.	.	.	.	.	.	.	18	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Opponent/Contrast	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	3	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Other	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	.	.	.	.	.	.	.	.	.	.	.	.	.	.
PartWhole	.	.	.	.	.	.	.	.	.	.	.	1	.	.	.	.	.	.	11	.	.	.	.	.	.	.	.	.	.	.	.	.
Participant/Accompanier	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	4	.	.	.	.	.	.	.	.	.	.	.	.
PhysicalSupport	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	6	.	.	.	.	.	.	.	.	.	.	.
Possessor	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	5	.	.	.	.	.	.	.	.	2	.	.	.	.
ProfessionalAspect	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	.	.	.	.	.	.	.	.	.
Purpose	.	.	.	3	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	.	.	.	.	.	.	.	.
Recipient	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	5	.	.	.	.	.	.	.
Separation	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	5	.	.	.	.	.	.
Source	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	5	.	5	.	.	.
Species	.	.	3	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	2	.	.	.	.	.	.	.	.	.	.	.	.	.
StartState	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	.	.	.
Temporal	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	4	.	.
Topic	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	11	.
Via	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1

Table C.34: Confusion matrix for annotated preposition relations. This represents the data from one of our annotators. The rows denote the labels from the SemEval-based relation data and the columns are the annotated labels. Dots denote that pairs of labels that were never confused.

	Activity	Agent	Attribute	Beneficiary	Cause	Co-Particiants	Destination	Direction	EndState	Experiencer	Instrument	Location	Manner	MediumOfCommunication	Numeric	ObjectOfVerb	Opponent/Contrast	Other	PartWhole	Participant/Accompanier	PhysicalSupport	Possessor	ProfessionalAspect	Purpose	Recipient	Separation	Source	Species	StartState	Temporal	Topic	Via
Activity	26	.	.	1	3	.	.	.	.	.	.	6	.	.	.	1	.	.	.	1	.	.	.	.	.	.	.	.	.	.	1	.
Agent	.	144	.	.	.	.	.	.	.	.	5	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	9	.	.	.	.	.
Attribute	.	4	213	.	1	.	2	.	.	.	3	20	3	2	.	4	.	.	6	1	.	2	.	.	.	.	1	4	.	.	.	.
Beneficiary	.	.	.	83	4	.	1	.	.	.	.	.	.	.	2	4	.	.	.	.	.	.	.	11	.	.	.	.	.	.	.	.
Cause	.	.	.	1	246	.	.	.	.	.	2	2	.	.	.	16	.	.	.	.	.	.	.	7	.	1	4	3	.	7	.	.
Co-Particiants	.	.	.	.	.	56	.	.	.	.	.	2	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Destination	.	.	.	.	.	.	479	.	6	.	.	28	2	.	.	.	.	.	.	2	.	.	.	1	6	.	.	.	.	2	.	.
Direction	.	.	.	.	1	.	.	430	.	.	.	6	1	.	.	1	.	.	.	.	.	.	.	.	.	2	.	.	.	.	.	.
EndState	.	.	.	1	.	.	20	.	79	.	.	2	.	.	.	.	.	.	.	1	.	.	.	.	1	.	.	.	.	.	.	.
Experiencer	.	.	.	.	.	.	.	1	.	43	.	1	.	1	.	.	.	.	.	.	5	.	.	.	.	.	.	.	.	.	3	.
Instrument	.	17	2	.	3	.	.	.	.	.	243	7	1	.	1	9	2	.	.	1	4	.	.	.	.	.	.	.	.	.	.	.
Location	3	1	9	.	1	4	17	8	.	3	2	1427	6	1	3	2	1	.	.	.	10	.	1	.	1	3	11	.	.	4	12	1
Manner	.	.	2	.	4	.	2	.	.	.	10	22	200	1	.	2	1	.	.	.	.	.	.	.	.	.	.	.	.	1	.	.
MediumOfCommunication	.	.	.	.	.	.	.	.	.	.	.	5	.	18	.	1	.	.	.	.	6	.	.	.	.	.	.	.	.	.	.	.
Numeric	.	1	.	1	2	.	.	.	.	.	.	2	.	.	37	3	.	.	.	.	.	.	.	.	.	.	.	.	.	1	1	.
ObjectOfVerb	.	.	.	2	12	.	.	.	.	2	2	8	.	.	.	813	.	.	7	2	1	13	1	2	3	.	1	8	.	1	4	.
Opponent/Contrast	.	.	.	.	.	2	.	.	.	.	2	1	.	.	.	1	118	.	.	4	.	.	.	.	.	.	3	.	.	.	.	.
Other	.	.	.	2	2	.	.	.	.	1	.	5	.	.	.	3	.	26	.	.	.	.	.	2	.	.	.	.	.	1	.	.
PartWhole	.	.	1	.	.	4	.	.	.	.	.	4	.	.	1	3	.	.	450	.	.	6	.	.	.	.	.	2	.	.	.	.
Participant/Accompanier	.	.	.	2	.	.	10	.	1	.	4	4	.	.	.	3	.	.	.	116	.	.	.	.	2	.	.	.	.	.	.	.
PhysicalSupport	.	.	.	.	.	.	.	.	.	.	.	9	.	1	.	.	1	.	.	.	188	.	.	.	.	.	.	.	.	.	1	2
Possessor	.	.	4	.	.	.	.	.	.	.	.	.	.	.	.	18	.	.	6	.	1	218	.	.	.	.	1	18	.	.	3	.
ProfessionalAspect	.	.	.	1	.	.	.	.	.	.	.	.	.	.	.	2	.	.	.	.	.	.	16	2	.	.	.	.	.	1	.	.
Purpose	.	.	.	1	7	.	.	.	.	.	.	.	.	.	.	2	.	.	.	.	.	.	.	102	.	.	.	.	.	1	.	.
Recipient	.	.	.	1	.	.	6	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	183	.	.	.	.	.	.	.
Separation	.	1	.	.	.	.	.	5	.	.	.	2	.	.	.	.	.	.	.	.	.	.	.	.	.	143	21	.	.	.	.	.
Source	.	7	.	.	3	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	8	335	3	1	.	.	.
Species	.	.	2	.	.	.	.	.	.	.	.	.	.	.	1	18	.	.	3	.	.	13	.	.	.	.	1	160	.	.	.	.
StartState	.	.	1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	4	.	31	.	.	.
Temporal	.	.	.	.	2	.	.	.	.	.	.	10	1	.	2	2	.	.	.	.	2	.	.	.	.	.	2	.	1	133	2	.
Topic	.	.	.	.	.	.	4	.	1	.	.	2	.	1	.	1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	453	.
Via	.	2	.	.	.	.	.	.	.	.	2	5	.	.	.	1	.	.	.	.	1	.	.	.	.	.	.	.	.	.	1	14

Table C.35: Confusion matrix for predicted preposition relations. This represents the performance of one of the runs of Model 2 from Chapter 6. The rows denote the labels from the SemEval-based relation data and the columns are the predicted labels. Dots denote that pairs of labels that were never confused.

References

Agirre, E., Baldwin, T., and Martinez, D. (2008). Improving Parsing and PP Attachment Performance with Sense Information. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 317–325, Columbus, USA.

Andrew, G., Grenager, T., and Manning, C. D. (2004). Verb sense and subcategorization: Using joint inference to improve performance on complementary tasks. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*, pages 150–157, Barcelona, Spain. Association for Computational Linguistics.

Baker, C., Ellsworth, M., and Erk, K. (2007). SemEval'07 Task 19: Frame Semantic Structure Extraction. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 99–104.

Baldwin, T., Kordoni, V., and Villavicencio, A. (2009). Prepositions in applications: A survey and introduction to the special issue. *Computational Linguistics*, 35(2):119–149.

Banko, M., Cafarella, M., Soderland, M., Broadhead, M., and Etzioni, O. (2007). Open information extraction from the web. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2670–2676.

Bar-Haim, R., Dagan, I., Greental, I., and Shnarch, E. (2007a). Semantic Inference at the Lexical-Syntactic Level. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, Vancouver. AAAI.

Bar-Haim, R., Dagan, I., Greental, I., Szpektor, I., and Friedman, M. (2007b). Semantic inference at the lexical-syntactic level for textual entailment recognition. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 131–136, Prague. Association for Computational Linguistics.

Barker, C. and Dowty, D. (1993). Non-verbal Thematic Proto-Roles. In Schafer, A., editor, *Proceedings of NELS 23, Vol. 1*, pages 49–62, Amherst. GSLA.

Bayraktar, M., Say, B., and Akman, V. (1998). An Analysis of English Punctuation: The Special Case of Comma. *International Journal of Corpus Linguistics*, 3(1):33–57.

Bengtson, E. and Roth, D. (2008). Understanding the Value of Features for Coreference Resolution. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*, pages 294–303.

Braz, R., Girju, R., Punyakanok, V., Roth, D., and Sammons, M. (2005). Knowledge representation for semantic entailment and question-answering. In *IJCAI-05 Workshop on Knowledge and Reasoning for Question Answering*.

Burton-Roberts, N. (1997). *Analysing sentences.* Longman.

Carreras, X. and Màrquez, L. (2004). Introduction to the CoNLL-2004 shared tasks: Semantic role labeling. In *Proceedings of the Annual Conference on Computational Natural Language Learning (CoNLL)*, pages 89–97. Boston, MA, USA.

Carreras, X. and Màrquez, L. (2005). Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the Annual Conference on Computational Natural Language Learning (CoNLL)*, pages 152–164, Ann Arbor, Michigan.

Chang, M., Goldwasser, D., Roth, D., and Srikumar, V. (2010a). Discriminative learning over constrained latent representations. In *Proceedings of the Annual Meeting of the North American Association of Computational Linguistics (NAACL).*

Chang, M., Ratinov, L., and Roth, D. (2012). Structured learning with constrained conditional models. *Machine Learning*, 88(3):399–431.

Chang, M., Srikumar, V., Goldwasser, D., and Roth, D. (2010b). Structured output learning with indirect supervision. In *Proceedings of the International Conference on Machine Learning (ICML).*

Chang, Y. and Collins, M. (2011). Exact decoding of phrase-based translation models through lagrangian relaxation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 26–37, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Charniak, E. (1997). Statistical parsing with a context-free grammar and word statistics. In *Proceedings of the National Conference on Artificial Intelligence (AAAI).*

Charniak, E. and Johnson, M. (2005). Coarse-to-Fine n-Best Parsing and MaxEnt Discriminative Reranking. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 173–180, Ann Arbor, Michigan. ACL.

Chen, D. and Mooney, R. (2008). Learning to sportscast: a test of grounded language acquisition. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 128–135.

Chierchia, G. and McConnell-Ginet, S. (2000). *Meaning and grammar: An introduction to semantics.* MIT press.

Clarke, J., Srikumar, V., Sammons, M., and Roth, D. (2012). An NLP Curator (or: How I Learned to Stop Worrying and Love NLP Pipelines). In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:37–46.

Collins, M. (1997). Three generative, lexicalised models for statistical parsing. In *Proceedings of the annual meeting of the ACL*, pages 16–23, Morristown, NJ, USA. Association for Computational Linguistics.

Collins, M. (1999). *Head-driven statistical models for natural language parsing.* PhD thesis, University of Pennsylvania.

Collins, M. (2000). Discriminative reranking for natural language parsing. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 175–182. Morgan Kaufmann, San Francisco, CA.

Collins, M. (2002). Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*.

Collins, M. and Duffy, N. (2002). New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20:273–297.

Culotta, A. and Sorenson, J. (2004). Dependency tree kernels for relation extraction. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 423–429.

Dagan, I., Glickman, O., and Magnini, B., editors (2006). *The PASCAL Recognising Textual Entailment Challenge.*, volume 3944. Springer-Verlag, Berlin.

Dahlmeier, D., Ng, H. T., and Schultz, T. (2009). Joint Learning of Preposition Senses and Semantic Roles of Prepositional Phrases. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*, pages 450–458.

Dang, H. T. and Palmer, M. (2005). The Role of Semantic Roles in Disambiguating Verb Senses. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 42–49, Ann Arbor, Michigan. Association for Computational Linguistics.

Das, D., Schneider, N., Chen, D., and Smith, N. (2010). Probabilistic Frame-Semantic Parsing. In *Proceedings of the Annual Meeting of the North American Association of Computational Linguistics (NAACL)*, pages 948–956, Stroudsburg, PA, USA.

Davidov, D. and Rappoport, A. (2008). Unsupervised discovery of generic relationships using pattern clusters and its evaluation by automatically generated sat analogy questions. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Davidov, D., Rappoport, A., and Koppel, M. (2007). Fully Unsupervised Discovery of Concept-Specific Relationships by Web Mining. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 232–239, Prague, Czech Republic.

Doddington, G., Mitchell, A., Przybocki, M., Ramshaw, L., Strassel, S., and Weischedel, R. (2004). The automatic content extraction (ACE) program–tasks, data, and evaluation. In *Proceedings of LREC*, volume 4, pages 837–840.

Doran, C. D. (1998). *Incorporating punctuation into the sentence grammar: A lexicalized tree adjoining grammar perspective.* PhD thesis, University of Pennsylvania.

Dowty, D. (1991). Thematic proto-roles and argument selection. *Language*, 67(3):547–619.

Duda, R. O., Hart, P. E., and Stork, D. G. (2001). *Pattern Classification.* Wiley-Interscience, New York, 2nd edition.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database.* MIT Press.

Fillmore, C. (1976). Frame Semantics and the Nature of Language. *Annals of the New York Academy of Sciences*, 280(1):20–32.

Fillmore, C., Johnson, C., and Petruck, M. (2003). Background to FrameNet. *International Journal of Lexicography*, 16(3):235–250.

Fillmore, C. J. (1968). The case for case. In Bach and Harms, editors, *Universals in Linguistic Theory*, pages 1–88. Holt, Rinehart, and Winston, New York.

Finkel, J. R. and Manning, C. D. (2009). Joint parsing and named entity recognition. In *Proceedings of the Annual Meeting of the North American Association of Computational Linguistics (NAACL)*, pages 326–334, Boulder, Colorado, USA.

Freund, Y. and Schapire, R. E. (1999). Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296.

Gildea, D. and Jurafsky, D. (2002). Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Goldberg, Y. and Elhadad, M. (2010). An Efficient Algorithm for Easy-First Non-Directional Dependency Parsing. In *Proceedings of the Annual Meeting of the North American Association of Computational Linguistics (NAACL)*, pages 742–750.

Goldwasser, D. and Roth, D. (2011). Learning from natural instructions. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Grishman, R. and Sundheim, B. (1996). Message Understanding Conference-6: A Brief History. In *Proceedings the International Conference on Computational Linguistics (COLING)*, volume 96, pages 466–471.

Gruber, J. S. (1965). *Studies in Lexical Relations*. PhD thesis, MIT, Cambridge, MA.

Gurobi Optimization, Inc. (2012). Gurobi optimizer reference manual.

Haghighi, A. and Klein, D. (2010). Coreference resolution in a modular, entity-centered model. In *Proceedings of the Annual Meeting of the North American Association of Computational Linguistics (NAACL)*, pages 385–393, Los Angeles, California.

Haghighi, A., Ng, A., and Manning, C. (2005). Robust textual inference via graph matching. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 387–394.

Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M., Màrquez, L., Meyers, A., Nivre, J., Padó, S., Štěpánek, J., et al. (2009). The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–18.

Hirano, T., Matsuo, Y., and Kikui, G. (2007). Detecting semantic relations between named entities in text using contextual features. In *Proc. of the Annual Meeting of the ACL*, pages 157–160.

Hockenmaier, J. and Steedman, M. (2007). CCGbank: a corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396.

Hovy, D., Tratz, S., and Hovy, E. (2010). What's in a Preposition? Dimensions of Sense Disambiguation for an Interesting Word Class. In *Coling 2010: Posters*, pages 454–462, Beijing, China. Coling 2010 Organizing Committee.

Jackendoff, R. (1976). Toward an explanatory semantic representation. *Linguistic Inquiry*, 7(1):89–150.

Jiang, Z. P. and Ng, H. T. (2006). Semantic Role Labeling of NomBank: a maximum entropy approach. In *EMNLP '06: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 138–145, Morristown, NJ, USA. Association for Computational Linguistics.

Joachims, T., Finley, T., and Yu, C. (2009). Cutting-plane training of structural svms. *Machine Learning*, 77(1):27–59.

Joshi, A. K. and Schabes, Y. (1997). Tree-adjoining grammars. In *Handbook of formal languages*, pages 69–123. Springer.

Kingsbury, P. and Palmer, M. (2002). From Treebank to PropBank. In *Proceedings of LREC-2002*, Spain.

Kingsbury, P. and Palmer, M. (2003). PropBank: The next level of TreeBank. In *Proceedings of Treebanks and Lexical Theories*, Växjö, Sweden.

Kipper, K. (2005). *VerbNet: A broad-coverage, comprehensive verb lexicon.* PhD thesis, University of Pennsylvania.

Kipper, K., Snyder, B., and Palmer, M. (2004). Using prepositions to extend a verb lexicon. In *HLT-NAACL 2004: Workshop on Computational Lexical Semantics*, pages 23–29, Boston, Massachusetts, USA.

Koo, T., Rush, A., Collins, M., Jaakkola, T., and Sontag, D. (2010). Dual decomposition for parsing with non-projective head automata. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1288–1298. Association for Computational Linguistics.

Koomen, P., Punyakanok, V., Roth, D., and Yih, W. (2005). Generalized inference with multiple semantic role labeling systems (shared task paper). In Dagan, I. and Gildea, D., editors, *Proceedings of the Annual Conference on Computational Natural Language Learning (CoNLL)*, pages 181–184.

Kundu, G. and Roth, D. (2011). Adapting text instead of the model : An open domain approach. In *Proceedings of the Annual Conference on Computational Natural Language Learning (CoNLL).*

Levin, B. (1993). *English Verb Classes and Alternations: A Preliminary Investigation.* University Of Chicago Press.

Lin, D. (1998). Automatic retrieval and clustering of similar words. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 768–774.

Lin, D. and Pantel, P. (2001). DIRT: discovery of inference rules from text. In *Proc. of ACM SIGKDD Conference on Knowledge Discovery and Data Mining 2001*, pages 323–328.

Litkowski, K. (2012). Proposed Next Steps for The Preposition Project. Technical Report 12-01. Damascus, MD: CL Research.

Litkowski, K. and Hargraves, O. (2005). The preposition project. In *Proceedings of the Second ACL-SIGSEM Workshop on the Linguistic Dimensions of Prepositions and their Use in Computational Linguistics Formalisms and Applications*.

Litkowski, K. and Hargraves, O. (2007). SemEval-2007 Task 06: Word-Sense Disambiguation of Prepositions. In *SemEval-2007: 4th International Workshop on Semantic Evaluations*.

Magerman, D. M. (1995). Statistical decision-tree models for parsing. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 276–283. Association for Computational Linguistics.

Marcus, M., Kim, G., Marcinkiewicz, M. A., MacIntyre, R., Bies, A., Ferguson, M., Katz, K., and Schasberger, B. (1994). The Penn Treebank: annotating predicate argument structure. In *HLT '94: Proceedings of the workshop on Human Language Technology*, pages 114–119, Morristown, NJ, USA.

Meyers, A. (2007). Those Other NomBank Dictionaries. Technical report, Technical report, New York University.

Meyers, A., Reeves, R., Macleod, C., Szekely, R., Zielinska, V., Young, B., and Grishman, R. (2004). The NomBank project: An interim report. In *HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation*, pages 24–31.

Meza-Ruiz, I. and Riedel, S. (2009). Jointly identifying predicates, arguments and senses using markov logic. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 155–163, Boulder, Colorado. Association for Computational Linguistics.

Miltsakaki, E., Prasad, R., Joshi, A., and Webber, B. (2004). The Penn Discourse Treebank. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*.

Minsky, M. (1975). A framework for representing knowledge. In Winston, P., editor, *The Psychology of Computer Vision*. McGraw-Hill.

Nunberg, G. (1990). *CSLI Lecture Notes 18: The Linguistics of Punctuation*. CSLI Publications, Stanford, CA.

Old, L. (2003). An Analysis of Semantic Overlap among English Prepositions in Roget's Thesaurus. In *Proceedings of the Association for Computational Linguistics SIG Semantics Conference*, pages 13–19.

Palmer, M., Gildea, D., and Kingsbury, P. (2005). The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1):71–106.

Palmer, M., Gildea, D., and Xue, N. (2010). *Semantic Role Labeling*, volume 3. Morgan & Claypool Publishers.

Pantel, P. and Pennacchiotti, M. (2006). Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 113–120.

Pasca, M., Lin, D., Bigham, J., Lifchits, A., and Jain, A. (2006). Names and similarities on the web: Fact extraction in the fast lane. In *Proc. of the Annual Meeting of the ACL*, pages 809–816.

Petrov, S., Das, D., and McDonald, R. (2011). A universal part-of-speech tagset. *Arxiv preprint ArXiv:1104.2086*.

Pradhan, S., Ward, W., Hacioglu, K., Martin, J. H., and Jurafsky, D. (2004). Shallow semantic parsing using support vector machines. In *Proceedings of NAACL-HLT 2004*, pages 233–240.

Punyakanok, V., Roth, D., and Yih, W. (2008). The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2).

Punyakanok, V., Roth, D., Yih, W., and Zimak, D. (2004). Semantic role labeling via integer linear programming inference. In *Proceedings the International Conference on Computational Linguistics (COLING)*, pages 1346–1352, Geneva, Switzerland.

Ratinov, L., Downey, D., Anderson, M., and Roth, D. (2011). Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Ratinov, L. and Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proc. of the Annual Conference on Computational Natural Language Learning (CoNLL)*.

Riedel, S. (2008). Improving the Accuracy and Efficiency of MAP Inference for Markov Logic. *Proceedings of the 24th Annual Conference on Uncertainty in AI (UAI '08)*, pages 468–475.

Rizzolo, N. and Roth, D. (2007). Modeling Discriminative Global Inference. In *Proceedings of the First International Conference on Semantic Computing (ICSC)*, pages 597–604, Irvine, California. IEEE.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psych. Rev.*, 65:386–407. (Reprinted in Neurocomputing (MIT Press, 1988).).

Roth, D. (1998). Learning to resolve natural language ambiguities: A unified approach. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 806–813.

Roth, D. and Yih, W. (2004). A linear programming formulation for global inference in natural language tasks. In Ng, H. T. and Riloff, E., editors, *Proceedings of the Annual Conference on Computational Natural Language Learning (CoNLL)*, pages 1–8. Association for Computational Linguistics.

Roth, D. and Yih, W. (2007). Global inference for entity and relation identification via a linear programming formulation. In Getoor, L. and Taskar, B., editors, *Introduction to Statistical Relational Learning*. MIT Press.

Ruppenhofer, J., Ellsworth, M., Petruck, M., Johnson, C., and Scheffczyk, J. (2006). FrameNet II: Extended theory and practice. *Berkeley FrameNet Release*, 1.

Rush, A. and Collins, M. (2011). Exact decoding of syntactic translation models through lagrangian relaxation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 72–82, Portland, Oregon, USA. Association for Computational Linguistics.

Rush, A. M., Sontag, D., Collins, M., and Jaakkola, T. (2010). On dual decomposition and linear programming relaxations for natural language processing. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*.

Russell, S. J. and Norvig, P. (2003). *Artificial Intelligence: A Modern Approach*. Pearson Education.

Sammons, M., Vydiswaran, V., and Roth, D. (2010). "Ask not what Textual Entailment can do for you...". In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Uppsala, Sweden. Association for Computational Linguistics.

Sammons, M., Vydiswaran, V., Vieira, T., Johri, N., Chang, M., Goldwasser, D., Srikumar, V., Kundu, G., Tu, Y., Small, K., Rule, J., Do, Q., and Roth, D. (2009). Relation alignment for textual entailment recognition. In *Text Analysis Conference (TAC)*.

Say, B. and Akman, V. (1997). Current approaches to punctuation in computational linguistics. *Computers and the Humanities*, 30(6):457–469.

Sekine, S. (2006). On-demand information extraction. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 731–738.

Shalev-Shwartz, S., Singer, Y., Srebro, N., and Cotter, A. (2011). Pegasos: primal estimated sub-gradient solver for SVM. *Mathematical Programming*, 127:3–30.

Sontag, D. and Jaakkola, T. (2007). New outer bounds on the marginal polytope. *Advances in Neural Information Processing Systems*, 20:1393–1400.

Srikumar, V., Kundu, G., and Roth, D. (2012). On amortizing inference cost for structured prediction. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*.

Srikumar, V., Reichart, R., Sammons, M., Rappoport, A., and Roth, D. (2008). Extraction of entailed semantic relations through syntax-based comma resolution. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, Columbus, OH, USA. Association for Computational Linguistics.

Srikumar, V. and Roth, D. (2011). A joint model for extended semantic role labeling. In *Proceedings of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*, Edinburgh, Scotland.

Srikumar, V. and Roth, D. (2013). Modeling Semantic Relations Expressed by Prepositions. *Transactions of the Association for Computational Linguistics*.

Surdeanu, M., Johansson, R., Meyers, A., Màrquez, L., and Nivre, J. (2008). The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 159–177. Association for Computational Linguistics.

Surdeanu, M., Màrquez, L., Carreras, X., and Comas, P. R. (2007). Combination strategies for semantic role labeling. *Journal of Artificial Intelligence Research*, 29:105–151.

Szpektor, I., Tanev, H., Dagan, I., and Coppola, B. (2004). Scaling web-based of entailment relations. In *Proc. of EMNLP*, pages 49–56.

Toutanova, K., Haghighi, A., and Manning, C. D. (2005). Joint learning improves semantic role labeling. In *Proceedings of ACL 2005*.

Toutanova, K., Haghighi, A., and Manning, C. D. (2008). A global joint model for semantic role labeling. *Computational Linguistics*, 34(2):161–191.

Tratz, S. and Hovy, D. (2009). Disambiguation of preposition sense using linguistically motivated features. In *HLT-NAACL*, pages 96–100.

Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. (2005). Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484.

van Delden, S. and Gomez, F. (2002). Combining finite state automata and a greedy learning algorithm to determine the syntactic roles of commas. In *Proc. of ICTAI*, pages 293–300.

Vanderwende, L. and Dolan, W. (2006). What syntax can contribute in the entailment task. *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 205–216.

White, M. and Rajkumar, R. (2008). A more precise analysis of punctuation for broad-coverage surface realization with CCG. In *Coling 2008: Proceedings of the workshop on Grammar Engineering Across Frameworks*, pages 17–24, Manchester, England. Coling 2008 Organizing Committee.

Xue, N. and Palmer, M. (2004). Calibrating features for semantic role labeling. In *Proceedings of EMNLP*, volume 4.

Ye, P. and Baldwin, T. (2006). Semantic role labeling of prepositional phrases. *ACM Transactions on Asian Language Information Processing (TALIP)*, 5(3):228–244.

Ye, P. and Baldwin, T. (2007). MELB-YB: Preposition Sense Disambiguation Using Rich Semantic Features. In *SemEval-2007*, pages 241–244. Association for Computational Linguistics.

Yu, C. and Joachims, T. (2009). Learning structural svms with latent variables. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Zapirain, B., Agirre, E., Màrquez, L., and Surdeanu, M. (2012). Selectional preferences for semantic role classification. *Computational Linguistics*, pages 1–33.

Zelenko, D., Aone, C., and Richardella, A. (2003). Kernel methods for relation extraction. *Journal of Machine Learning Research*.

Zettlemoyer, L. and Collins, M. (2009). Learning context-dependent mappings from sentences to logical form. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, pages 976–984, Stroudsburg, PA, USA.