NAIST-IS-DD0761031

Doctoral Dissertation

Dependency-based Predicate-Argument Structure Analysis Using Structured Learning and Named Entity Information

Yotaro Watanabe

February 4, 2010

Department of Information Processing
Graduate School of Information Science
Nara Institute of Science and Technology

A Doctoral Dissertation
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Yotaro Watanabe

Dependency-based Predicate-Argument Structure Analysis Using Structured Learning and Named Entity Information*

Yotaro Watanabe

Abstract

A predicate-argument structure is composed of a predicate that represents a state or an event, and its arguments which have logical relations to the predicate. The logical relations are roles of arguments such as Agent, Theme, etc., are called *Semantic Roles*. Predicate-argument structure analysis is an important process for deep semantic analysis of natural language texts, and has been exploited in various natural language processing applications.

The aim of this work is primarily realizing robust predicate-argument structure analysis based on the dependency-based syntactic representation which is currently the mainstream in the field of natural language processing. In this dissertation, we focus on (1) designing models for dependency-based predicate-argument structure analysis with structured learning, (2) acquiring and exploiting named entity information.

There are two types of dependencies in predicate-argument structures. One is inter-dependencies between a predicate and its arguments in which types of arguments depend on their predicate sense, on the other hand, types of arguments of the predicate restrict senses of the predicate. The other is non-local dependencies between arguments. A predicate-argument structure has no duplicate argument roles, and must contain at least one core argument role of the predicate. Although a number of approaches for predicate-argument structure

*Doctoral Dissertation, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD0761031, February 4, 2010.

analysis have been proposed, none of them handles both of the dependencies simultaneously. We propose a structured model based on structured learning that captures both types of dependencies. In the experiments, we achieved performance improvements on both tasks, and competitive results compared to the state-of-the-art systems without any feature engineering. Also, we constructed a system for syntactic and semantic dependency parsing using the structured model. The system achieved the best performance among the existing systems without any feature engineering.

An issue of predicate-argument structure analysis is how to deal with named entities in sentences. Since a large number of named entities exist, unknown expressions appear in texts. Their lexical information do not contribute the analysis, hence it is necessary to generalize them. In order to do this, we assign named entity categories for named entities. We focus on (A) acquiring named entity information from the Web, (B) incorporating named entity information for predicate-argument structure analysis. First, we propose structured models for acquiring named entity information from Wikipedia that captures characteristics of list structures in articles. Second, we propose a method to introduce named entity information for predicate-argument structure analysis. In addition to the features used in the previous work, the global feature that globally captures types in terms of all core arguments is introduced to the model. In the experiments, we achieved performance improvements compared to the previous approaches.

Keywords:

predicate-argument structure analysis, structured learning, joint learning, global features, named entity

依存構造と固有表現情報を用いた構造学習による述語項構造解析[*]

渡邉 陽太郎

内容梗概

述語項構造とは，状態や事態を表す述語と，その述語と関係を持つ項によって構成される構造のことをいう．述語と項の関係には，項が述語に対してどのような意味役割（動作主，対象など）を持っているかが与えられる．述語項構造解析は，高次の意味解析を必要とする自然言語処理の応用において重要な要素技術である．

本研究の目的は，近年自然言語処理分野において主流となっている統語構造の表現形式である依存構造に基づく頑健な述語項構造解析の実現である．本論文では，(1) 構造学習に基づく述語項構造解析ためのモデル設計，(2) 述語項構造解析のための固有表現情報の獲得と利用，の二点について論じる．

述語項構造には，構造の要素間に二種類の依存関係が存在する．一つは，述語の意味 (語義) によって項が制約される，また項によって語義が限定されるという相互の依存関係，もう一つは複数の項が同じ意味役割を持たないこと，述語の必須の項は基本的に存在するといった項の間の依存関係である．既存研究のアプローチでは，これら双方の依存関係を同時に考慮することが困難である．そこで，双方の依存関係を同時に捉える構造学習法に基づくモデルを提案する．評価実験では，語義と意味役割の双方の分類性能が向上し，さらに素性選択法を適用せずに既存研究に匹敵する性能が得られた．さらに，提案モデルを用いて依存構造と述語項構造の双方を解析するシステムを構築した．評価実験の結果，素性選択法を用いていない既存システムの中では最良の結果が得られた．

述語項構造解析解析をおこなう上で問題となるのは，文中に現れる固有表現の存在がある．これらは無数に存在し，未知の固有表現が多く現れるため，これ

[*]奈良先端科学技術大学院大学 情報科学研究科 情報処理学専攻 博士論文, NAIST-IS-DD0761031, 2010 年 2 月 4 日.

らを抽象化した形で扱う必要がある．そこで，文中の固有表現に対して固有表現情報を付与することで，抽象化をおこなう．本論文では，(A) Web 文書からの固有表現情報の自動獲得，(B) 固有表現情報の述語項構造解析モデルへの導入，の二点について論じる．まず，Web 上の百科事典である Wikipedia から，記事に現れるリスト構造の性質を利用した固有表現の獲得手法について提案する．次に，固有表現情報の導入については，既存研究で用いられている方法に加えて，複数の項に関連する大域的な固有表現情報の導入を提案する．評価実験では，既存の導入方法と比較して高い性能が得られた．

キーワード

述語項構造解析，構造学習，結合学習，大域的素性，固有表現

Acknowledgements

主指導教官の松本裕治教授には，終始暖かく御指導していただき，適切な御助言をいただきました．深く感謝いたします．審査委員として本論文を査読してくださいました鹿野清宏教授に深く感謝いたします．乾健太郎准教授には，研究者としてあるべき姿について，多くを学ばせていただきました．今後もお世話になります．どうぞ宜しくお願いいたします．浅原正幸助教には，在籍している間はいなくならないで下さいとお願いし，博士前期課程からの 5 年間，研究指導をしていただきました．一仕事するまでに時間のかかるタイプの私を，終始暖かく見守ってくださり，研究で行き詰まった時には，常に適切な御助言をくださいました．また，浅原助教に背中を押していただき，カーネギーメロン大学への訪問，ニューヨーク大学でのインターンシップへの参加をすることになり，大変貴重な経験をすることができました．深く感謝いたします．新保仁助教には，研究に対する厳しく批判的な姿勢に，多くを学ばせていただきました．また，日頃からよく私に声をかけて下さり，研究に関する議論などをよくさせていただきました．深く感謝いたします．カーネギーメロン大学の三田村照子教授には，訪問研究員として 5 週間受け入れていただき，滞在期間中は終始暖かく御指導いただきました．初めての海外での長期滞在，また多くの優秀な研究者の中で共に研究をするという大変貴重な経験をさせていただきました．深く感謝いたします．ニューヨーク大学の関根聡准教授には，インターンシップで受け入れていただき，滞在期間中は手厚く御指導いただきました．深く感謝いたします．研究室の先輩で現東京工業大学の飯田龍助教 † には，研究面においては議論の中で多くを学ばせていただき，また課題に直面した際には多くの御助言をいただきました．深く感謝いたします．鄭育昌さんには，研究をはじめとする様々な面に置いて支えていただきました．深く感謝いたします．研究室同期の小町守さんには，研究や私生活で支えていただき，進路選択など多くの面でアドバイスをいただきました．博士後期課程への進学は多くの不安がありましたが，小町さんの存在は私にとって大変心強いものでした．深く感謝いたします．木村学さん ‡，吉川勝正さん †，水野淳太さん †，岩立将和さんとは，研究に関してよく議論し，その中から多くのことを学ばせていただきました．また，研究室での生活においても様々な面で支えていただきました．深く感謝いたします．研究室での活動の中で，カレー部の活動は特に有意義なものでした．創立者 (‡)，および活動における主要メンバー (†) に感謝いたします．大川基子さん，北川祐子秘書には，研究活動および大学生活を暖かく支えていただきました．深く感謝いたします．井原瑞希さん，高橋祐さんは博士後期

課程まで進学した同期の友人で，５年間の大学生活における様々な面で支えていただきました．深く感謝いたします．最後に，NAIST への入学および博士後期課程への進学を応援し，暖かく見守り，支えてくれた両親に感謝します．

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1. Background

Over the past decade, the field of natural language processing (NLP) has drastically changed by the availability of large hand-annotated training corpora. The approaches for the tasks of NLP have been shifted from rule-based approaches to statistical approaches, and by efforts of several researchers, the performances have gotten the level of practical use in the fundamental NLP tasks such as POS tagging, base phrase chunking and syntactic parsing. Therefore now is the time to tackle semantic analysis of texts.

Predicate-argument structure analysis is an important process toward understanding meaning of sentences. A predicate-argument structure is composed of a predicate that represents a state or an event, and its arguments which have logical relations to the predicate. The logical relations are roles of arguments including Agent, Theme, etc., are called **semantic roles**. Predicate-argument structure analysis is the process of identifying predicate-argument structures in texts [1].

Predicate-argument structures have potential to absorb different syntactic realizations. For example, in the following sentences:

(1.a) [He]$_{AGENT}$ *opened* [the door]$_{THEME}$.

(1.b) [The door]$_{THEME}$ *opened*.

[1]The task of predicate-argument structure analysis is often called *Semantic Role Labeling*.

(1.a) is a transitive use, and (1.b) is a intransitive use of the verb *opened*. Although the subjects of *opened* in (1.a) and (1.b) are *The door* and *He*, the roles of both are not the same. In terms of (1.a), the subject *The door* has the role *Theme* of *opened*.

(2.a) [John]$_{AGENT}$ *hit* [Mary]$_{PATIENT}$.

(2.b) [Mary]$_{PATIENT}$ was *hit* [by John]$_{AGENT}$.

(2.b) is the passivised sentence of (2.a) where the roles of the subject and the object are alternated.

(3.a) [John]$_{AGENT}$ *gave* [Mary]$_{PATIENT}$ [the book]$_{THEME}$.

(3.b) [John]$_{AGENT}$ *gave* [the book]$_{THEME}$ [to Mary]$_{PATIENT}$.

This is an English dative alternation. Although these two sentences have the same meaning, (3.a) has two objects, on the other hand, (3.b) has only one object.

Since the difference of syntactic realizations can be absorbed by identifying predicate-argument structures, predicate-argument structure analysis is an important process for understanding natural language texts by computers. In recent years, predicate-argument structure analysis has been exploited in various natural language processing applications, such as information extraction [87], statistical machine translation [108], question answering [84] and recognizing textual entailment [106].

So far, corpora that have annotations of predicate-argument structures have been constructed in several languages. For English, predicate-argument structures for verbs are defined and annotated by FrameNet [4] and Proposition Bank (PropBank) [70], and that for nouns are defined and annotated by NomBank. Also, for other languages, predicate-argument structures are annotated in SALSA Corpus [7] for German, Prague Dependency Treebank [39] for Czech, Kyoto Corpus [45] and NAIST Text Corpus [41] for Japanese. The availability of these corpora allows statistical approaches for predicate-argument structure analysis.

Since the fact of necessity of constituent parses in predicate-argument structure analysis was pointed out by Gildea and Palmer [31], a number of work had been widely explored use of the type of syntactic representation [9, 10]. On the other hand, alternatively, another syntactic representation based on the dependency grammar has explored by several researchers [109, 68, 26, 61], and after that more sophisticated higher-order parsing algorithms have been proposed [62, 8, 57]. Because of this background, recently several researchers have paid much attention to dependency-based predicate-argument structure analysis [34, 88, 38].

In addition, by the progress of the field of machine learning, it has become possible to learn models from structured data (**Structured Learning**). For the tasks of natural language processing, several researchers have been explored use of structured learning framework for sequential labeling [17, 49] such as POS tagging and chunking, and for tree-structured data such as dependency parsing [61].

Because of the availability of the corpora and the progress of the field of machine learning, predicate-argument structure analysis could become a reliable fundamental technology of NLP. Toward advanced processing of semantic meaning and inference of natural language texts, methods for robust predicate-argument structure analysis have been demanded.

2. Research Objectives

The research goal of this dissertations is primarily to realize robust dependency-based predicate-argument structure analysis. An example of a dependency-based predicate-argument structure on which we focus in this dissertation is shown in Figure 1.1. The upper part of the figure shows a syntactic dependency structure of the sentence, and the lower part shows predicate-argument structures. This syntactic dependency structure represents, for example, that the main verb *sold* has a subject *maker* and an object *cars*. The corresponding predicate-argument structure has a semantic predicate corresponding to the verb *sold*, and its arguments labeled with semantic role labels. The semantic role labels represent logical relation between the predicate and the arguments.

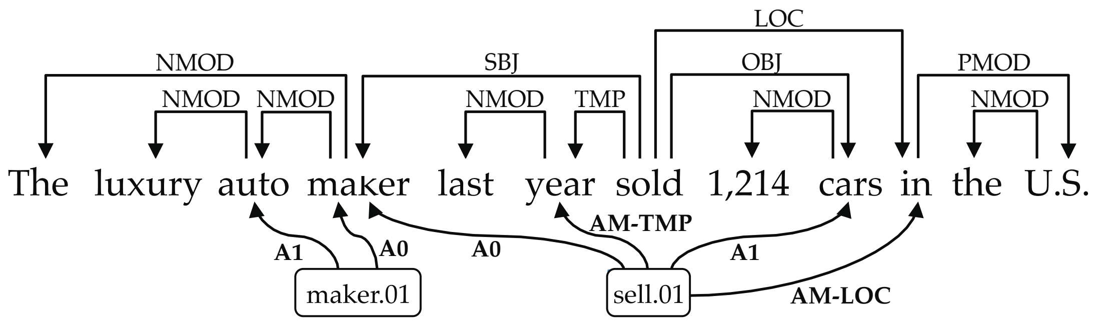


Figure 1.1. Dependency-based representation of a predicate-argument structure.

For example, the argument *maker* of the predicate *sold* labeled with *A0* means that it has a role *Seller* for the predicate *sold*, and the argument *car* labeled with *A1* means that it has a role *Thing Sold* for the predicate *sold*. The label of the predicates *sell.01* represents a word sense which means that the predicate *sold* is an instance of the first sense of *sell* defined in the lexicon.

Predicate-argument structures have strong dependencies among its elements: a predicate and its arguments. One of its dependencies is that more than one arguments in a predicate-argument structure do not have the same argument role. For example, the predicate *sold* in Figure 1.1 do not allow two arguments that have the role *Seller*. The other one of dependencies is that acceptable argument roles differ for each predicate sense. One of the other possible use of *sell* is, for example, "John sold out.". In this case, the allowed argument role is only "entity selling out" which corresponds to "John". By capturing these types of dependencies, it is expected to realize robust predicate-argument structure analysis. So, one of the objectives in this dissertation is to design a model that effectively captures these types of dependencies underlying predicate-argument structures. In order to capture these types of dependencies, a structured learning framework is introduced to construct a structured model where several types of features are introduced. Also, using the structured model, we construct a system that analyzes both syntactic dependencies and predicate-argument structures.

Another objective is to resolve lexical sparseness which stems from existence of named entities. Since a large number of named entities exist, unknown expressions frequently appear in texts. Their lexical information does not contribute the analysis, hence it is necessary to generalize them. In order to resolve this issue, we attempt to generalize lexical information of named entities using its *named entity classes*.

We tackle two problems in terms of named entities. At first we construct a large-scale named entity dictionary from *Wikipedia*, an encyclopedia on the Web. In Wikipedia, there is a useful characteristics that the elements enumerated in list structures tend to have the same named entity class. In order to capture the characteristics, we introduce a structured learning framework where the model structures are determined based on relations between elements in list structures. Next, we investigate how the named entity information contributes resolution of lexical sparseness and performance improvements in predicate-argument structure analysis.

3. Dissertation Outline

The rest of this dissertation is organized as follows.

Chapter 2: Predicate-Argument Structure Analysis This chapter describes the formal problem definitions of predicate-argument structure analysis, three types of syntactic representations used for predicate-argument structure analysis, and approaches of previous work.

Chapter 3: Structured Learning This chapter describes machine learning methods to learn from structured data. Several learning algorithms for linear and log-linear models proposed by previous work are described.

Chapter 4: Joint Learning of Predicate Senses and Argument Roles We propose a structured model to deal with inter-dependent elements of predicate-argument structure – a predicate sense and its argument roles. Four factors that affect to decisions of the model are introduced, and an inference algorithm and a large-margin learning algorithm adopted for the model are described.

Chapter 5: A Partially Joint Approach for Syntactic and Semantic Dependency Parsing Based on the structured model described in Chapter 4, we construct a syntactic and semantic depenency parser which is a partially joint system.

Chapter 6: Acquiring Named Entity Information from Wikipedia This chapter describes acquisition of named entity information from Wikipedia. Capturing structural characteristics of articles in Wikipedia using structured models, we show that named entity information can be extracted accurately.

Chapter 7: Exploiting Named Entity Information for Predicate-Argument Structure Analysis This chapter investigates effects of named entity information for predicate-argument structure analysis. We construct and evaluatte the model in Chapter 4 with the named entity information extracted in Chapter 6.

Chapter 8: Conclusion This chapter summarizes this dissertation and describe further directions.

Chapter 2

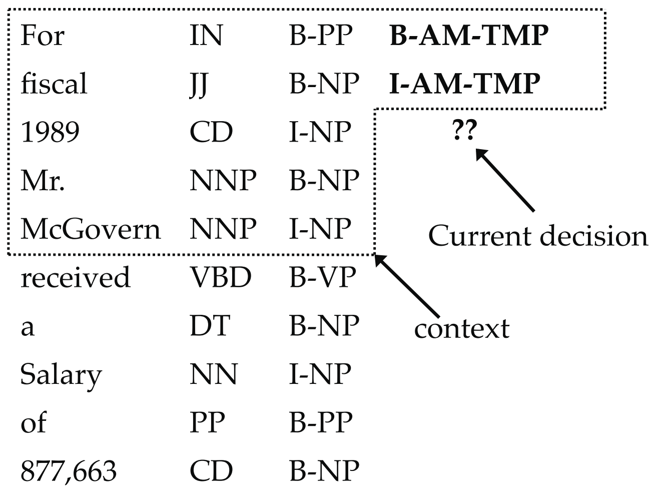
Previous Work on Predicate-Argument Structure Analysis

In this Chapter, we describe various approaches proposed by the previous work.

1. Predicate-Argument Structure Analysis with Shallow Syntactic Information

Hacioglu and Ward [36], Hacioglu [33] and Roth et al. [81] formalized the task of predicate-argument structure analysis as sequential tagging of semantic roles for arguments.

These work use shallow syntactic information instead of syntactic trees for predicate-argument structure analysis. The reason of using it is that obtaining syntactic information such as constituent syntactic trees is computationally expensive, and not all languages have full syntactic parsers. Another reason is that treating the task of predicate-argument structure analysis as sequential labeling is fast. In sequential labeling, if chunks or phrases that consist of two or more words have a particular label, the label is decomposed into role and position labels. The decomposition is sometimes performed by using the IOB



For	IN	B-PP	**B-AM-TMP**
fiscal	JJ	B-NP	**I-AM-TMP**
1989	CD	I-NP	**??**
Mr.	NNP	B-NP	
McGovern	NNP	I-NP	
received	VBD	B-VP	
a	DT	B-NP	
Salary	NN	I-NP	
of	PP	B-PP	
877,663	CD	B-NP	

Figure 2.1. Example of chunking-based predicate-argument structure analysis

tag where "B" denotes the first word of the chunk, "I" means that the word is inside a phrase, and "O" means that the word is not part of a phrase.

Hacioglu and Ward [36] and Hacioglu [33] and Hacioglu et al. [35] used a chunking-based method which is an history-based approach. The procedure of the approach presented in [36] is denoted in Figure 2.1. In this example, the process is determining the argument role label of the word *1989*. For determining argument role label, words, its POSs and its phrase type information that denoted as *context* in the figure are encoded as features for the Support Vector Machine-based chunker.

Roth and Yih [81] used a Linear-chain Conditional Random Fields (Linear-chain CRFs) [49] for argument role labeling. In contrast to Hacioglu's approach, the approach of Roth and Yih is a graph-based. In order to introduce constraints in terms of argument role labels, they used Integer Linear Programming (ILP) for inference on Linear-chain CRFs, not the Viterbi algorithm.

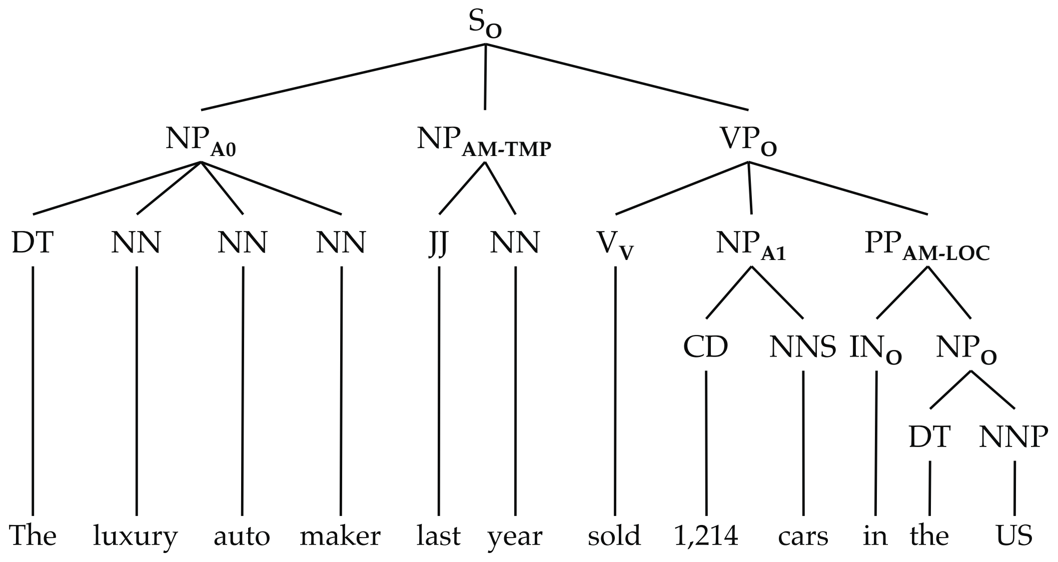


Figure 2.2. Example of constituent-based predicate-argument structure analysis

2. Constituent-based Predicate-Argument Structure Analysis

The task of predicate-argument structure analysis based on constituent-trees is assigning argument role labels for non-terminal nodes of constituent trees.

Figure 2.2 shows an example of constituent-based predicate-argument structure analysis. For the case of the predicate *sold* in Figure 2.2, the argument of the role *A0* is "The luxury auto maker", and the non-terminal node that corresponds to the words is assigned the argument role label.

The first statistical approach for constituent-based predicate-argument structure analysis was probabilistic-based models proposed by [30, 31]. Later, various machine learning approaches such as decision trees [87, 12], Support Vector Machines [76, 75] have been applied and various feature types have been explored by these work.

These researchers have been used unstructured multi-class classifiers, whereafter some work applied structured learning techniques for constituent-based predicate-argument structure analysis. Cohn et al. [16] proposed use of Tree-structured Conditional Random Fields for predicate-argument structure analysis. They applied conditional random fields over constituent trees where argument roles are assigned non-terminal nodes of trees collectively. They reported that allowing parent-child interactions provided performance improvements over a standard maximum entropy classifier, however, their approach does not deal with constraints such as "more than one of arguments do not have a particular core argument role".

On the other hand, some work showed that capturing non-local information using re-ranking is effective for predicate-argument structure analysis. Haghighi et al. [37] and Toutanova et al. [101, 102] proposed use of global features which capture whole predicate-argument structures. One of these global features is *core argument label sequence*. For the example in Figure 2.2, the corresponding feature of the core argument label sequence is [voice:active A0,V,A1]. In order to deal with such global features, they used two classifiers: one for the local features and the other for the global features, and confidences of the classifiers are combined to provide the optimal output of the model. Surdeanu et al. [89] used global features with perceptron-based re-ranker and a combination approach for generating argument candidates using multiple classifiers.

3. Dependency-based Predicate-Argument Structure Analysis

Predicate-argument structure analysis over dependency trees is formalized as the task of assigning argument role labels to nodes of dependency trees.

An example of dependency-based predicate-argument structure analysis (English) is shown in Figure 2.3. The upper part of the figure shows a syntactic dependency structure, and the lower part shows semantic dependencies (a predicate-argument structure). In this example, the arguments that have a particular role for the predicate *wore* are *His daughter* (agent), *a small hat* (theme) and *yesterday* (time). For the phrases that have a particular role, semantic role

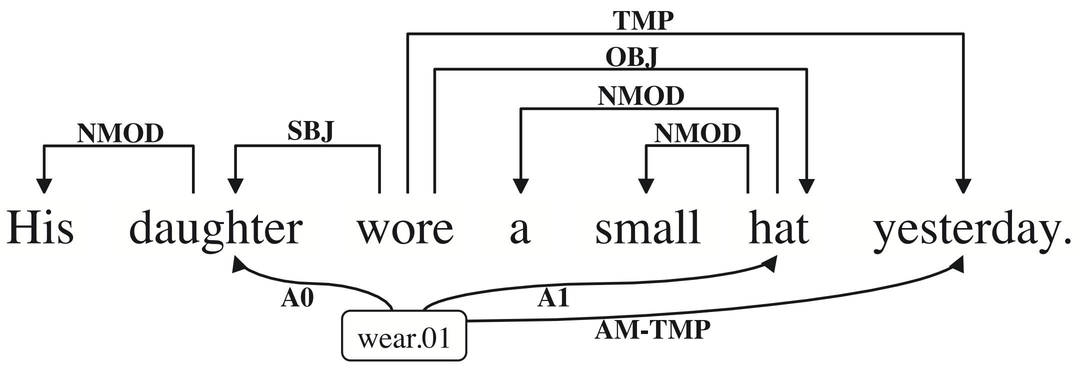


Figure 2.3. Example of dependency-based predicate-argument structure analysis

labels are assigned to head words of each phrase in sentences. For example, in the case of Figure 2.3, as for the phrase *His daughter*, *daughter* is assigned A0 because *His* modifies *daughter*.

Dependency-based predicate-argument structure analysis was first explored by Hacioglu [34]. Later, the challenges of the task was held in CoNLL-2008 Shared Task [88] and CoNLL-2009 Shared Task [38]. In the challenges, predicate-argument structure analysis was treated the task of detecting argument roles of predicates as well as senses of predicates.

Riedel and Meza-Ruiz [79], Meza-Ruiz and Riedel [65, 64] used Markov Logic Networks (MLNs) for predicate-argument structure analysis. Their systems jointly identify predicates, predicate senses, arguments and argument roles by designing formulae that depend on these tasks. The system is learned by MIRA, an online large-margin learning algorithm.

Johansson and Nugues [44] proposed a framework for joint syntactic dependency parsing and predicate-argument structure analysis. Their system at first generates n-best trees using a higher-order dependency parser, and then performs predicate-argument structure analysis for each tree, finally obtains the most plausible structure using a re-ranker. They divided the task of predicate-argument structure analysis into four subtasks: predicate identification, predicate sense disambiguation, argument identification and argument role labeling, and prepared classifiers for each task where the features for the each task is optimized using a greedy forward feature selection algorithm. For the re-ranker, they used global features that are similar to those used in [101]. This approach generates argument candidates of predicate-argument structures in pipelined way, hence it can be seen as a *shallow* joint framework. Their system achieved the top score in the closed challenge of the CoNLL-2008 Shared Task [88].

Henderson et al. [40] and Titov et al. [28] proposed a history-based latent variable model for this task. They extend a Shift-Reduce parsing [67] which generally used for dependency parsing, for predicate-argument structure analysis so as to obtain both of dependency structures. In their framework, either of both tasks is performed deterministically and on the way, tasks are switched while its procedure. Their approach is the most succeeded *pure* joint framework on this task [88].

Lluís and Màrquez [54, 53] extended the Eisner algorithm [26], an span-based parsing algorithm, so as to obtain both joint syntactic dependency trees and predicate-argument structures. In this approach, the optimal structure is obtained based on the joint score function of syntactic trees and predicate-argument structures, however, at first the system need to perform dependency parsing to obtain features used for predicate-argument structure analysis, therefore the approach does not seem to a pure joint approach, and the performances of their system do not have superiority over the other state-of-the-art systems.

Chapter 3

Structured Learning

1. Introduction

In structured learning, the goal is to learn a predictive function $h : \mathcal{X} \rightarrow \mathcal{Y}$ from a structured input $\mathbf{x} \in \mathcal{X}$ to a structured output $\mathbf{y} \in \mathcal{Y}$, where $\mathcal{Y}$ represents a space of possible outputs. For example, in sequential tagging problems such as POS tagging, noun phrase chunking, named entity recognition, etc, each input $\mathbf{x}$ is a word sequence, and $\mathbf{y} \in \mathcal{Y}$ is a sequence of labels. In dependency parsing, $\mathbf{x}$ is a word sequence, its lemmas and POS tags, and $\mathbf{y} \in \mathcal{Y}$ is a possible dependency tree.

The models applied for structured learning can be roughly categorized into two types: **Linear Models** and **Log-linear Models**. In this chapter, we describle learning algorithms applied for each type of models.

2. Linear Models

2.1 Structured Perceptron

The perceptron algorithm proposed by Rosenblatt [80] is the simplest online learning algorithm that estimates parameters of linear models. The procedure of perceptron learning is as follows. At first, the parameter $\mathbf{w}_0$ is initialized to $\mathbf{0}$. For each training sample, if the model with the current parameter $\mathbf{w}_t$ correctly classify the output $\mathbf{y}_t$, then $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t$; otherwise update the parameter

$\mathbf{w}_t$. Rosenblatt's perceptron is specialized to binary-classification problems, and Novikoff's theorem [69] shows that if the training set is separable with nonzero margin, the procedure terminates. Collins [18] generalized the Rosenblatt's perceptron to multi-class and structured cases. In Collins's perceptron, at each iteration t, predict with

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}\in\mathcal{Y}} \mathbf{w}\cdot\Phi(\mathbf{x},\mathbf{y}) \tag{3.1}$$

if $\hat{\mathbf{y}} = \mathbf{y}$, then set $\mathbf{w}_{t+1} = \mathbf{w}_t$; otherwise, i.e. $\hat{\mathbf{y}} \neq \mathbf{y}$, update the parameter $\mathbf{w}_t$ as follows.

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}}) \tag{3.2}$$

Note that since the parameter update is performed only if $\hat{\mathbf{y}} \neq \mathbf{y}$, the parameter $\mathbf{w}$ will not achieve good generalization on unseen test data. For the reason of the shortcoming of the Rosenblatt's perceptron, Krauth and Mézard [48] proposed the perceptron with a margin, and Kazama and Torisawa [46] extended the Krauth and Mézard's perceptron to cases of structured outputs.

Now, we use a margin function $\Delta(\mathbf{y},\mathbf{y}')$ which is defined by the difference between $\mathbf{y}$ and $\mathbf{y}'$. In margin perceptron, the model is stricted so as to ensure the margin $\Delta(\mathbf{y},\mathbf{y}')$ between the gold structure $\mathbf{y}_t$ and the other structure $\mathbf{y}' \neq \mathbf{y}_t$ as follows.

$$\forall \mathbf{y}' \in \{\mathcal{Y}\setminus \mathbf{y}_t\} \;\; \mathbf{w}_t\cdot\Phi(\mathbf{x}_t,\mathbf{y}_t) - \mathbf{w}_t\cdot\Phi(\mathbf{x}_t,\mathbf{y}') \geq \Delta(\mathbf{y}_t,\mathbf{y}') \tag{3.3}$$

In order to impose the above constraints, we have only to replace eq. (3.1) with the following.

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}\in\mathcal{Y}} \mathbf{w}\cdot\Phi(\mathbf{x}_t,\mathbf{y}_t) + \Delta(\mathbf{y}_t,\mathbf{y}) \tag{3.4}$$

By doing this, if no updates occur, then the learned parameter $\mathbf{w}$ satisfies all margin constraints in eq. (3.3). Algorithm 1 shows a perceptron algorithm with a margin for structured prediction [1].

[1]The algorithm we present here has differences with the Kazama's perceptron algorithm [46]. In Kazama's perceptron, (1) The margin value is a constant value C not the margin function $\Delta(\mathbf{y}_t,\mathbf{y})$, (2) The margin constraint is applied for only $\mathbf{y}$ and second-best $\mathbf{y}''$. In terms of (1), we believe that fixing margin is not appropriate for structured prediction problems, because

Algorithm 1 A Perceptron Algorithm with a Margin for Structured Prediction

```
input Training set $\mathcal{T} = \{\mathbf{x}_t, \mathbf{y}_t\}_{t=1}^{T}$, Number of iterations N
$\mathbf{w} \leftarrow \mathbf{0}$
for $i \leftarrow 0$ to $N$ do
  for $(\mathbf{x}_t, \mathbf{y}_t) \in \mathcal{T}$ do
    $\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} \mathbf{w} \cdot \Phi(\mathbf{x}_t, \mathbf{y}) + \Delta(\mathbf{y}_t, \mathbf{y})$
    $\mathbf{w} \leftarrow \mathbf{w} + \Phi(\mathbf{x}_t, \mathbf{y}_t) - \Phi(\mathbf{x}_t, \hat{\mathbf{y}})$
  end for
end for
return w
```

Perceptron algorithms (including margin perceptrons) have shortcomings in that fixed margin becomes relatively small than the parameter **w** because perceptrons add features of positive and negative examples by fixed coefficient (namely 1.0). The state-of-the-art learning algorithms such as Passive-Aggressive Algorithms (PA) [20] and Support Vector Machines (SVMs) [103] attempt to resolve this problem. These algorithms are described later.

2.2 Passive-Aggressive Algorithms (PA)

The Passive-Aggressive Algorithms (PA) [20], are online large-margin learning algorithms which are variants of the (margin) perceptron. These algorithms can be applied for a broad range of problems: binary classification, multi-class classification, structured prediction, regression and one class prediction. Here, we describe the application of PA for structured prediction problems.

The Passive-Aggressive Algorithm finds $\mathbf{w}_{new}$ that satisfies the constraint for only one example by solving the following optimization problem. At first, we explain the case in which soft margins are not considered.

$$\mathbf{w}_{new} = \arg\min_{\mathbf{w} \in \Re^n} \frac{1}{2}||\mathbf{w} - \mathbf{w}_t||^2 \text{ s.t. } \mathbf{w} \cdot \Phi(\mathbf{x}, \mathbf{y}) - \mathbf{w} \cdot \Phi(\mathbf{x}, \hat{\mathbf{y}}) \geq \Delta(\mathbf{y}, \hat{\mathbf{y}}) \quad (3.5)$$

number of elements in the difference feature of $\hat{\mathbf{y}}$ and $\mathbf{y}$ depends on the number of incorrect assignments. In terms of (2), by using the function $\Delta(\mathbf{y}_t, \mathbf{y})$ to decoding, the margin constraint for $\mathbf{y}_t$ and $\mathbf{y}''$ is naturally incorporated, and the resulting algorithm becomes more simple.

In order to solve the optimization problem, at first we define the Lagrangian as follows.

$$\mathcal{L}(\mathbf{w},\tau) = \frac{1}{2}||\mathbf{w}-\mathbf{w}_t||^2 + \tau\{\Delta(\mathbf{y},\hat{\mathbf{y}}) - (\mathbf{w}\cdot\Phi(\mathbf{x},\mathbf{y}) - \mathbf{w}\cdot\Phi(\mathbf{x},\hat{\mathbf{y}}))\} \tag{3.6}$$

where τ is a Lagrange multiplier. Setting the derivative of $\mathcal{L}(\mathbf{w},\tau)$ with respect to $\mathbf{w}$ to zero, we get

$$\frac{\partial\mathcal{L}(\mathbf{w},\tau)}{\partial\mathbf{w}} = \mathbf{w}-\mathbf{w}_t - \tau\Big(\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}})\Big) = 0. \tag{3.7}$$

$$\mathbf{w} = \mathbf{w}_t + \tau(\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}})) \tag{3.8}$$

Substituting the eq. (3.8) into eq. (3.6), we obtain

$$\begin{aligned}
\mathcal{L}(\tau) =& \frac{1}{2}||\mathbf{w}_t + \tau(\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}})) - \mathbf{w}_t||^2 \\
&+ \tau\{\Delta(\mathbf{y},\hat{\mathbf{y}}) - (\mathbf{w}_t + \tau(\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}})))(\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}}))\} \\
=& \frac{1}{2}\tau^2||\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}})||^2 \\
&+ \tau\{\Delta(\mathbf{y},\hat{\mathbf{y}}) - \mathbf{w}_t\cdot(\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}})) - \tau||\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}})||^2\} \\
=& \frac{1}{2}\tau^2||\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}})||^2 \\
&+ \tau\{\Delta(\mathbf{y},\hat{\mathbf{y}}) - \mathbf{w}_t\cdot(\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}}))\} - \tau^2||\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}})||^2 \\
=& -\frac{1}{2}\tau^2||\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}})||^2 + \tau\{\Delta(\mathbf{y},\hat{\mathbf{y}}) - \mathbf{w}_t\cdot(\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}}))\}.
\end{aligned} \tag{3.9}$$

Setting the derivative of $\mathcal{L}(\tau)$ with respect to τ to zero, we get

$$\frac{\partial\mathcal{L}(\tau)}{\partial\tau} = -\tau||\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}})||^2 + \{\Delta(\mathbf{y},\hat{\mathbf{y}}) - (\mathbf{w}_t\cdot(\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}}))\} = 0 \tag{3.10}$$

$$\tau = \frac{\Delta(\mathbf{y},\hat{\mathbf{y}}) - \mathbf{w}_t\cdot(\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}}))}{||\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}})||^2} = \frac{\mathbf{w}_t\cdot\Phi(\mathbf{x},\hat{\mathbf{y}}) - \mathbf{w}_t\cdot\Phi(\mathbf{x},\mathbf{y}) + \Delta(\mathbf{y},\hat{\mathbf{y}})}{||\Phi(\mathbf{x},\mathbf{y}) - \Phi(\mathbf{x},\hat{\mathbf{y}})||^2} \tag{3.11}$$

This update is called **PA**.

Next, we consider the cases with a slack variable ξ. A possible objective function includes ξ is as follows.

$$\mathbf{w}_{new} = \arg \min_{\mathbf{w} \in \Re^n} \frac{1}{2}||\mathbf{w} - \mathbf{w}_t||^2 + C\xi$$
$$\text{s.t. } \mathbf{w} \cdot \Phi(\mathbf{x}, \mathbf{y}) - \mathbf{w} \cdot \Phi(\mathbf{x}, \hat{\mathbf{y}}) \geq \Delta(\mathbf{y}, \hat{\mathbf{y}}) - \xi \text{ and } \xi \geq 0 \quad (3.12)$$

where C is a constant that controls an influence of ξ. This update is called **PA-I**. By solving the optimization problem, we obtain

$$\tau = \min\{C, \frac{\mathbf{w} \cdot \Phi(\mathbf{x}_t, \hat{\mathbf{y}}) - \mathbf{w} \cdot \Phi(\mathbf{x}_t, \mathbf{y}_t) + \Delta(\mathbf{y}_t, \hat{\mathbf{y}})}{||\Phi(\mathbf{x}_t, \mathbf{y}_t) - \Phi(\mathbf{x}_t, \hat{\mathbf{y}})||^2}\}. \quad (3.13)$$

Alternatively, we can consider the objective function that scale quadratically with ξ. In that case, the optimization problem becomes

$$\mathbf{w}_{new} = \arg \min_{\mathbf{w} \in \Re^n} \frac{1}{2}||\mathbf{w} - \mathbf{w}_t||^2 + C\xi^2$$
$$\text{s.t. } \mathbf{w} \cdot \Phi(\mathbf{x}, \mathbf{y}) - \mathbf{w} \cdot \Phi(\mathbf{x}, \hat{\mathbf{y}}) \geq \Delta(\mathbf{y}, \hat{\mathbf{y}}) - \xi \quad (3.14)$$

By solving the optimization problem, we obtain

$$\tau = \frac{\mathbf{w} \cdot \Phi(\mathbf{x}_t, \hat{\mathbf{y}}) - \mathbf{w} \cdot \Phi(\mathbf{x}_t, \mathbf{y}_t) + \Delta(\mathbf{y}_t, \hat{\mathbf{y}})}{||\Phi(\mathbf{x}_t, \mathbf{y}_t) - \Phi(\mathbf{x}_t, \hat{\mathbf{y}})||^2 + \frac{1}{2C}} \quad (3.15)$$

This update is called **PA-II**.

The PA algorithms are shown in Algorithm 2.

The similar large-margin learning algorithm, Margin Infused Relaxed Algorithm (MIRA) proposed by Crammer et al. [21] coincides with the PA-I with the constant value $C = 1$.

2.3 Structured Support Vector Machines

Structured Support Vector Machines (SSVMs) learns the parameter **w** that satisfies all constraints of the dataset. More precisely, SSVMs solves the following optimization problem.

$$\min_{\mathbf{w}, \xi \geq 0} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{t=1}^{n} \xi_t$$
$$\forall t, \forall \mathbf{y} \in \mathcal{Y} \setminus \mathbf{y}_t : \mathbf{w} \cdot \Phi(\mathbf{x}_t, \mathbf{y}_t) - \mathbf{w} \cdot \Phi(\mathbf{x}_t, \mathbf{y}) \geq \Delta(\mathbf{y}_t, \mathbf{y}) - \xi_i \quad (3.16)$$

Algorithm 2 The Passive Aggressive Algorithms

```
input Training set $\mathcal{T} = \{\mathbf{x}_t, \mathbf{y}_t\}_{t=1}^{T}$, Number of iterations N
w ← 0
for $i \leftarrow 0$ to $N$ do
    for $(\mathbf{x}_t, \mathbf{y}_t) \in \mathcal{T}$ do
        $\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} \mathbf{w} \cdot \Phi(\mathbf{x}_t, \mathbf{y}) + \Delta(\mathbf{y}_t, \mathbf{y})$
        $\tau = \begin{cases} \frac{\mathbf{w}\cdot\Phi(\mathbf{x}_t,\hat{\mathbf{y}}) - \mathbf{w}\cdot\Phi(\mathbf{x}_t,\mathbf{y}_t) + \Delta(\mathbf{y}_t,\hat{\mathbf{y}})}{||\Phi(\mathbf{x}_t,\mathbf{y}_t) - \Phi(\mathbf{x}_t,\hat{\mathbf{y}})||^2} & \text{(PA)} \\ \min\{C, \frac{\mathbf{w}\cdot\Phi(\mathbf{x}_t,\hat{\mathbf{y}}) - \mathbf{w}\cdot\Phi(\mathbf{x}_t,\mathbf{y}_t) + \Delta(\mathbf{y}_t,\hat{\mathbf{y}})}{||\Phi(\mathbf{x}_t,\mathbf{y}_t) - \Phi(\mathbf{x}_t,\hat{\mathbf{y}})||^2}\} & \text{(PA - I)} \\ \frac{\mathbf{w}\cdot\Phi(\mathbf{x}_t,\hat{\mathbf{y}}) - \mathbf{w}\cdot\Phi(\mathbf{x}_t,\mathbf{y}_t) + \Delta(\mathbf{y}_t,\hat{\mathbf{y}})}{||\Phi(\mathbf{x}_t,\mathbf{y}_t) - \Phi(\mathbf{x}_t,\hat{\mathbf{y}})||^2 + \frac{1}{2C}} & \text{(PA - II)} \end{cases}$
        $\mathbf{w} \leftarrow \mathbf{w} + \tau(\Phi(\mathbf{x}_t, \mathbf{y}_t) - \Phi(\mathbf{x}_t, \hat{\mathbf{y}}))$
    end for
end for
return w
```

In constrast to the Passive-Aggressive Algorithms that use only the constraint of one example, SSVM finds the parameter **w** under the exponential number of constraints. However, since it is impractical to optimize models with the exponential number of constraints, it is necessary to introduce techniques to make the optimization problem tractable.

Tsochantaridis et al. show that the optimization problem can be solved efficiently by introducing a cutting-plane algorithm [103]. The algorithm tries to find small subset (working set) of the constraints in the problem (3.16). The algorithm iteratively finds the mostly violated constraint under the current model, and then adds the constraint to the working set. In order to construct the current model, an optimization problem is solved using the subset of the constraints. This procedure is repeated until no constraints are selected. A cutting plane algorithm for Structured Support Vector Machines is shown in Algorithm 3.

Algorithm 3 A Cutting Plane Algorithm for Structured Support Vector Machines

```
**input** Training set $\mathcal{T} = \{(\mathbf{x}_1, \mathbf{y}_1, ..., (\mathbf{x}_N, \mathbf{y}_N)\}$ and Parameter C
$S_i \leftarrow \mathbf{0}$ for all $i = 1, ..., N$
**repeat**
  **for** $i = 1, ..., N$ **do**
    $H(\mathbf{y}) = \Delta(\mathbf{y}_i, \mathbf{y}) + \mathbf{w} \cdot \Phi(\mathbf{x}_i, \mathbf{y}) - \mathbf{w} \cdot \Phi(\mathbf{x}_i, \mathbf{y}_i)$
    compute $\hat{\mathbf{y}} = \arg\max_{\mathbf{y} \in \mathcal{Y}} H(\mathbf{y})$
    compute $\xi_i = \max\{0, \max_{\mathbf{y} \in S_i} H(\mathbf{y})\}$
    **if** $H(\hat{\mathbf{y}}) > \xi_i + \epsilon$ **then**
      $S_i \leftarrow S_i \cup \{\hat{\mathbf{y}}\}$
      $\mathbf{w} \leftarrow$ optimize the problem over $S = \cup_i S_i$
    **end if**
  **end for**
**until** no $S_i$ has changed during iteration
```

3. Log-Linear Models

3.1 Conditional Random Fields

Conditional Random Fields (CRFs) [50] are undirected graphical models that give a conditional probability distribution $p(\mathbf{y}|\mathbf{x})$ in a form of exponential model. CRFs are formalized as follows. Let $\mathcal{G} = \{V, E\}$ be an undirected graph over random variables $\mathbf{y}$ and $\mathbf{x}$. when a set of cliques $C = \{\{\mathbf{y}_c, \mathbf{x}_c\}\}$ are given, CRFs define the conditional probability of a state assignment given an observation set.

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{c \in \mathbf{C}} \Phi(\mathbf{x}_c, \mathbf{y}_c) \tag{3.17}$$

where $\Phi(\mathbf{x}_c, \mathbf{y}_c)$ is a potential function defined over cliques, and $Z(\mathbf{x})$ is the partition function defined as follows.

$$Z(\mathbf{x}) = \sum_{\mathbf{y}} \prod_{c \in \mathbf{C}} \Phi(\mathbf{x}_c, \mathbf{y}_c) \tag{3.18}$$

The potentials are factorized according to the set of features $\{f_k\}$.

$$\Phi(\mathbf{x}_c, \mathbf{y}_c) = \exp\left(\sum_k \lambda_k f_k(\mathbf{x}_c, \mathbf{y}_c)\right) \tag{3.19}$$

where $F = \{f_1, ..., f_K\}$ are feature functions on the cliques, $\Lambda = \{\lambda_1, ..., \lambda_K \in \mathcal{R}\}$ are the model parameters, obtained for each feature.

In general, the model parameters Λ are estimated by maximizing conditional log-likelihood of N training samples $\mathcal{D} = \{\mathbf{y}^{(d)}, \mathbf{x}^{(d)}\}_{t=1}^{N}$.

$$\mathcal{L}_\lambda = \sum_{t=1}^{N} \sum_{c \in \mathcal{C}^{(d)}} \sum_{k} \lambda_k f_k(\mathbf{y}_c^{(d)}, \mathbf{x}_c^{(d)}) - \sum_{t=1}^{N} \log Z(\mathbf{x}^{(d)}) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2} \tag{3.20}$$

where third term is prior distribution (Gaussian prior) for parameters [13], and σ^2 is variance for the distribution. By introducing this term, parameters are normalized, and overfitting of the parameters is alleviated. It is called Maximum a posteriori (MAP) estimation.

The derivative of the objective function $\mathcal{L}_\lambda$ is the following.

$$\frac{\partial \mathcal{L}_\lambda}{\partial \lambda_k} = \sum_{t=1}^{N} \sum_{c \in \mathcal{C}^{(d)}} f_k(\mathbf{y}_c^{(d)}, \mathbf{x}_c^{(d)}) - \sum_{t=1}^{N} \sum_{c \in \mathcal{C}^{(d)}} \sum_{\mathbf{y}_c} f_k(\mathbf{y}_c, \mathbf{x}_c^{(d)}) p(\mathbf{y}_c | \mathbf{x}_c^{(d)}) - \frac{\lambda_k}{\sigma^2} \tag{3.21}$$

Since the objective function $\mathcal{L}_\lambda$ is convex, global optimal solution can be obtained. (Stochastic) Gradient Ascent or the Limited Memory BFGS (L-BFGS) [52] are used for this optimization.

In order to get the value of the equation 3.21, it is necessary to calculate marginal probabilities of the form $p(\mathbf{y}_c|\mathbf{x}) = \sum_{\mathbf{y} \backslash \mathbf{y}_c} p(\mathbf{y}|\mathbf{x})$.

The algorithm for marginal probability calculation to be applied depends on structure of the graphical model. The forward-backward algorithm is used for linear chain CRFs that are adopted for sequence labeling tasks such as POS tagging. In the case of tree-structure, marginal probabilities can be calculated exactly by using Belief Propagation (BP) [72] which is the general version of the Forward-Backward algorithm.

In BP, (*) for each clique, cumulative values of the potential functions to a particular label are calculated (which is called *message* $\mathbf{m} = \{m_i(y_j)\}$) and then calculate marginal probabilities with the messages. In the case that the defined cliques consist of at most two nodes, update form of the message $m_i(y_j)$ for node i to node j is the following.

$$m_i(y_j) = \sum_{y_i} \Phi_v(y_i, \mathbf{x}) \Phi_e(y_i, y_j, \mathbf{x}) \prod_{k \in \mathcal{N}(i) \backslash j} m_k(y_i) \tag{3.22}$$

where $\mathcal{N}(i)$ is the adjacent nodes of the i, $\Phi_v(y_i, \mathbf{x})$ is the potential function over the node $i \in V$, and $\Phi_e(y_i, y_j, \mathbf{x})$ is the potential function over the edge $(i, j) \in E$. Note that we denoted the observation $\mathbf{x}_c$ as $\mathbf{x}$ because we can see whole of the sequence. Hereafter, we denote $\mathbf{x}_c$ as simply $\mathbf{x}$.

Using the equation 3.22, the marginal probability $p(y_j|\mathbf{x})$ of the node j can be calculated by the following equation.

$$p(y_j|\mathbf{x}) = \kappa\Phi_v(y_j, \mathbf{x}) \prod_{i\in\mathcal{N}(j)} m_i(y_j) \tag{3.23}$$

where κ is a normalization factor.

In the case that if the model contains no loops, marginal probabilities can be calculated by messages. However, if the model have the structure that contains loops (e.g. Dynamic Conditional Random Fields (DCRFs) [58, 92], Skip-Chain CRFs [91]) approximation is necessary to calculate marginals. Approximation methods for marginal calculation include Loopy Belief Propagation (Loopy BP) [66] and Tree-based Reparameterization (TRP). Loopy BP applies BP no matter whether the model contains loops. TRP generate a set of spanning trees that can be constructed from the model, and then update messages for each tree using BP.

3.2 Max-Margin Markov Networks (M^{3}N)

Max-Margin Markov Networks (M^{3}N) [98] are structured models in which the model parameters $\mathbf{w}$ are estimated by max-margin criterion. M^{3}N have the same form of the distribution as CRFs, however, unlike CRFs, parameters $\mathbf{w}$ are estimated by solving the following optimization problem.

$$\begin{aligned}\mathbf{w}^* = \arg\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{\mathbf{x}} \xi_{\mathbf{x}} \\ \text{s.t. } \forall t\ \forall \mathbf{y}' \neq \mathbf{y}_t;\ \mathbf{w}\cdot(\Phi(\mathbf{x}_t, \mathbf{y}_t) - \Phi(\mathbf{x}_t, \mathbf{y}')) \geq \Delta(\mathbf{y}_t, \mathbf{y}')\end{aligned} \tag{3.24}$$

For the quadratic programming, Taskar et al. used Sequential Minimal Optimization (SMO). On the other hand, the QP of eq. (3.24) can be converted into convex optimization problems in which margin constraints are included into the objective fuction. In this case, the following optimization problem is to be solved.

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2 + C\sum_t \max_{\mathbf{y}'}\{\Delta(\mathbf{y}_t, \mathbf{y}') - \mathbf{w}\cdot(\Phi(\mathbf{x}_t, \mathbf{y}_t) - \Phi(\mathbf{x}_t, \mathbf{y}'))\} \tag{3.25}$$

Since this optimization is not included in any constraints, it can be solved as a convex optimization problem. For solving this optimization problem, Globerson et al., Collins et al. proposed use of Exponentiated Gradient (EG) algorithms [51, 32, 19], Zhu et al. proposed Projected Sub-gradient and EM-style Algorithm [116].

Note that since the M^{3}N optimization problems are the same as the Structured SVMs, some work see M^{3}N as the same model as Structural SVMs (e.g. [116]).

3.3 Markov Logic Networks (MLNs)

Markov Logic Networks (MLNs) [77] is a relational learning framework based on First-order Logic and Markov Networks. It can be seen that MLNs extends first-order logic: formulas can be violated with some penalty, and that formulas of MLNs are templates for instantiating a particular markov network. A markov logic network is defined by a set of weighted formula. Formally, an MLN M is a set of pairs (ϕ, w) where ϕ is a first-order formula and w is a weight of the model. M assigns the probability for the hidden ground atoms $\mathbf{y}$ given the observed ground atoms $\mathbf{x}$.

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})}\exp\Big(\sum_{(\phi,w)\in M} w \sum_{\mathbf{c}\in C^\phi} f_{\mathbf{c}}^{\phi}(\mathbf{x}, \mathbf{y})\Big) \tag{3.26}$$

C^ϕ is the set of all possible bindings of the free variables in ϕ, and $f_{\mathbf{c}}^{\phi}$ is a feature function. This distribution corresponds to a Markov Network where nodes represent ground atoms and factors represent ground formulas.

For learning of MLNs, maximum log-likelihood estimation, which is an ordinary optimization for log-linear models, is used in [85]. On the other hand, Riedel uses MIRA [78] which is an online large-margin learning algorithm.

Chapter 4

Joint Learning of Argument Roles and Predicate Senses

1. Introduction

Several researchers have paid much attention to predicate-argument structure analysis, and the two following important factors have been presented. Johansson and Nugues [44], and Björkelund et al. [5] presented importance of capturing non-local dependencies of core arguments. They used argument sequences tied with a predicate sense (e.g. AGENT-buy.01/Active-PATIENT) as a feature for the re-ranker of the system where predicate sense and argument role candidates are generated by their pipelined architectures. They reported that incorporating the type of features provides substantial gain of the system performance.

The other is inter-dependencies between a predicate sense and argument roles, which relate to selectional preferences, and motivate us to jointly identify a predicate sense and argument roles. We consider the following sentence as an example.

(1) *She drives a car.*

where *drives* is the predicate. In PropBank, two senses (*drive.01* and *drive.02*) are defined for the verb *drive*, and the types of the role *theme* of these senses are *vehicle* and *things in motion* respectively. In the example, corresponding argument of the predicate *drives* is *car*, and it is a *vehicle*. Therefore, the sense of the predicate *drives* can be determined as *drive.01*. This is an example that has a dependency from the argument roles to the predicate sense.

On the other hand, there are opposite cases, namely, dependency from predicate senses to argument roles. Next, we consider the two following sentences.

(2) *Mr. Yeargin comes to work on weekends.*
(3) *Tokyo comes to terms with its new status as the region's economic behemoth.*

In PropBank, the senses of the predicates *comes* in (2) and (3) are defined as *come.01* (move) and *come.14* (come to terms with) respectively. Observe that *Mr. Yeargin* in (2) and *Tokyo* in (3) are both placed at the subject position in each sentence, however, the semantic role of *Mr. Yeargin* is A1 (comer), whereas the role of *Tokyo* is A0 (entity coming to terms). For the two cases, it is necessary to determine the senses formerly to identify the argument roles. This type of dependencies has been explored by Riedel and Meza-Ruiz [79, 65, 64], all of these are Markov Logic Networks (MLN) based systems. These work use the global logic formulae that has atoms in terms of both argument roles and a predicate sense, and the systems identify predicate senses and argument roles jointly.

Ideally, we want to capture both types of dependencies simultaneously. However, the approaches of the previous work can not handle both. The former approaches can not explicitly include features that capture inter-dependencies between a predicate sense and its argument roles. Though these are implicitly incorporated by re-ranking where the most plausible assignment is selected based on a small subset of predicate and argument candidates, these are generated independently. On the other hand, it is difficult to deal with core argument features in MLN. Because, the number of core arguments varies with the role assignments, the type of features is not able to be expressed by a single formula.

In this chapter, we propose a structured model that overcomes limitations of previous approaches. For the model, we introduce several types of features

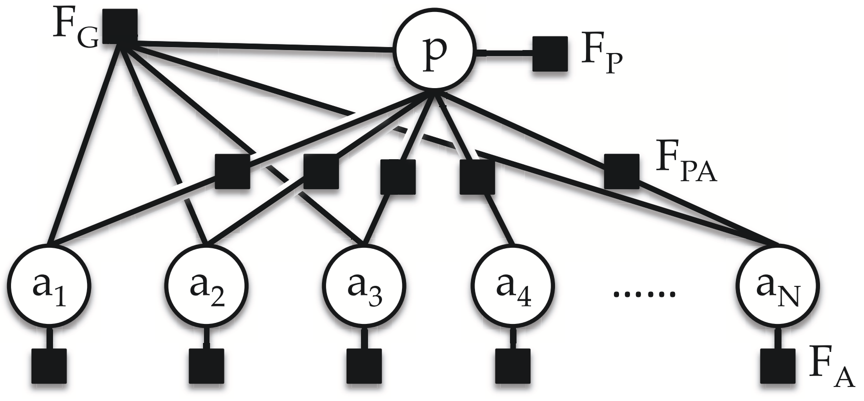


Figure 4.1. Undirected graphical model representation of the joint model

including those that capture both inter-dependencies among arguments roles, and inter-dependencies between a predicate sense and its argument roles. By doing this, while both tasks are mutually influenced, the model determines the most plausible set of assignments of a predicate sense and its argument roles simultaneously. Also, we present an exact inference algorithm for the model, and a new large-margin learning algorithm that handles global features in parallel with local features.

2. A Structured-Prediction Model for Joint Learning of Argument Roles and Predicate Senses

In this section, we describe a structured model that captures both non-local dependencies between arguments, and inter-dependencies between a predicate sense and argument roles.

Figure 4.1 shows a graphical representation of the model. The node p corresponds to a predicate, and the nodes $a_1, ..., a_N$ to arguments of the predicate. Each node is assigned a particular predicate sense or argument role label. The black squares are **factors** which provide scores of label assignments. In the model, the nodes for arguments depend on a predicate sense, and by influencing labels of a predicate sense and argument roles, the most plausible label assignments of the nodes are determined by all factors.

Possible approaches to represent such undirected graphical models are using log-linear models such as Conditional Random Fields [49] or linear models [18, 60]. In this work, we use linear models.

Let $\mathbf{x}$ be words in a sentence, l be a predicate, $\mathcal{P}_l$ be a set of possible senses of the predicate l, $p \in \mathcal{P}_l$ be a particular sense of l, and $\mathcal{A} = \{a_1, a_2, ..., a_N\}$ be a set of possible label assignments for $\mathbf{x}$. A predicate-argument structure is represented by a pair of p and $\mathcal{A}$.

We define the score function for predicate-argument structures as below.

$$s(p, \mathcal{A}) = \sum_{F_k \in \mathcal{F}} F_k(\mathbf{x}, p, \mathcal{A}). \tag{4.1}$$

The pair of the predicate sense p and the set of argument role assignments $\mathcal{A}$ that has the maximum score is returned by the model. $\mathcal{F}$ is a set of all the factors, $F_k(\mathbf{x}, p, \mathcal{A})$, which corresponds to a particular factor in Figure 4.1, is a factor that scores a predicate or argument label assignments, and is defined by the inner product of the model parameter $\mathbf{w}$ and the feature vector Φ_k.

$$F_k(\mathbf{x}, p, \mathcal{A}) = \mathbf{w} \cdot \Phi_k(\mathbf{x}, p, \mathcal{A}) \tag{4.2}$$

The feature vector $\Phi_k(\mathbf{x}, p, \mathcal{A})$ is a set of clues used in the factor F_k. Each elements of $\Phi_k(\mathbf{x}, p, \mathcal{A})$ corresponds to a particular clue which is an observed feature conjoined with one or more labels to be predicted. Let us consider a case of detecting the role of *hat* for *wore* in 2.3. For this case, we might include a feature ϕ_k^j for Φ_k as follows.

$$\phi_k^j(\mathbf{x}, a) = \begin{cases} 1 & \text{if } a = \text{A1} \\ & \text{and predicate lemma} = \text{wear} \\ & \text{and dependency label} = \text{OBJ} \\ 0 & \text{otherwise} \end{cases}$$

j is the j-th dimension of the feature space of F_k. In such a way, we create feature vectors for the model by mapping various features for different dimensions of the feature space.

In the next section, we define four types of factors used for the joint model. Later, the inference algorithm of the model is shown in Section 2.3, and then describe the proposed learning algorithm for the model in 2.3. Finally, in Section 2.4, we present an issue in terms of the predicate-argument pairwise factor.

2.1 Factors for the Joint Model

We define four types of factors for the joint model.

Predicate Factor F_P This factor scores each sense of p, and does not depend on any argument role assignments. The score function is defined by $F_P(\mathbf{x}, p, \mathcal{A}) = \mathbf{w} \cdot \Phi_P(\mathbf{x}, p)$.

Argument Factor F_A This factor scores a label assignment of a particular argument $a \in \mathcal{A}$. The score is determined independently from a predicate sense, and is given by $F_A(\mathbf{x}, p, a) = \mathbf{w} \cdot \Phi_A(\mathbf{x}, a)$.

Predicate-Argument Pairwise Factor F_{PA} This factor captures inter-dependencies between a predicate sense and an argument role. The score function is defined as $F_{PA}(\mathbf{x}, p, a) = \mathbf{w} \cdot \Phi_{PA}(\mathbf{x}, p, a)$. The difference from F_A is that F_{PA} influences both the predicate sense and the argument role. By introducing this factor, the role label can be influenced by the predicate sense, and vise versa. We consider the example sentence 1 in Section 1. For instance, by introducing the following feature, the label of *car* affects scores of the predicate sense label.

$$\phi^{j}_{PA}(\mathbf{x}, p, a) = \begin{cases} 1 & \text{if } p = \text{drive.01 and } a = \text{A1} \\ & \text{and argument lemma} = \textit{car} \\ 0 & \text{otherwise} \end{cases}$$

Global Factor F_G This factor is introduced to be able to capture plausibility of predicate-argument structures. Like the other factors, the score function is defined as an inner product of $\mathbf{w}$ and a global feature vector $\Phi_G(\mathbf{x}, p, \mathcal{A})$: $F_G(\mathbf{x}, p, \mathcal{A}) = \mathbf{w} \cdot \Phi_G(\mathbf{x}, p, \mathcal{A})$. A possible feature that can be considered for this factor is, for instance, if a predicate-argument structure has an agent (A0) followed by the predicate and a patient (A1), we encode the structure as a string *A0-PRED-A1* and use it as a feature.

$$\phi_G^j(\mathbf{x}, p, \mathcal{A}) = \begin{cases} 1 & \text{if } p \text{ and } \mathcal{A} \text{ can be} \\ & \text{expressed by } \textit{A0-PRED-A1} \\ 0 & \text{otherwise} \end{cases}$$

This type of features provide *plausibility* of predicate-argument structures. For instance, if the highest scoring predicate-argument structure with the local factors misses some core arguments , then the global feature demands the model to fill the missing arguments.

The numbers of factors for each factor type in eq. (4.1) are: F_P and F_G are 1, F_A and F_{PA} are $|\mathcal{A}|$. By integrating the all factors, as a result, the score function becomes

$$s(p, \mathcal{A}) = \mathbf{w} \cdot \Phi_P(\mathbf{x}, p) + \mathbf{w} \cdot \Phi_G(\mathbf{x}, p, \mathcal{A}) + \mathbf{w} \cdot \sum_{a \in \mathcal{A}} \{\Phi_A(\mathbf{x}, a) + \Phi_{PA}(\mathbf{x}, p, a)\}. \tag{4.3}$$

Each factor does not share its feature space with the other factors. Let each feature space be $\Phi_P \in \Re^{d_P}, \ \Phi_A \in \Re^{d_A}, \ \Phi_{PA} \in \Re^{d_{PA}}, \ \Phi_G \in \Re^{d_G}$, then the feature space of the parameter vector $\mathbf{w}$ could be written as $\mathbf{w} \in \Re^{d_P + d_A + d_{PA} + d_G}$.

2.2 Inference Algorithm of the Joint Model

In general, dynamic programming is applicable when the features are *factorized*. Among the four factors of the model, the features of three factors are factorized (F_P, F_A and F_{PA}). We call these type of features *Local Features*. The features used in the global factor F_G, we call *Global Features*. Without the global factor F_G, the exact solution can be produced with F_P, F_A and F_{PA}. This is because if the predicate sense label is fixed, then inference with the three factors is reduced to simple multiclass classifications for each argument. A similar idea is used by McDonald et al. [60], in which the prediction model is structured with a rooted tree.

An issue is how to deal with the global factor F_G. In general, efficient inference becomes impractical if we use global features. A naive approach we can easily conceive is that, at first, we enumerate all of possible argument role assignments, and then add scores of global features for each assignment, finally select the argmax. However, enumerating possible assignments is too costly (at least our case), therefore the approach might be impractical. A number of methods have been proposed use of global features for linear models. Daumé III proposed a search-based framework, Learning as Search Optimization (LaSO) [24], in which the search space flexibly varies by the current parameter vector, and parameters are updated if the search gets disabled to achieve the correct answer or the argmax is not equal to the correct answer. The approach is often applied to cases that the exact inferences are impractical due to large search spaces. Kazama proposed an N-best based approach [46]. The approach abandons enumerating all of possible assignments, instead, selects the argmax among N-best candidates that are produced by local features. For the learning of the model with global features, Kazama proposed a variant of the margin perceptron learning algorithm.

In this work, we use the approach proposed by Kazama to deal with global features. Although Kazama's approach is proposed for sequence labeling tasks, it can be easily extended for the proposed model. That is, for each possible predicate sense p of the predicate l, we provide N-best argument role assignments using three local factors F_P, F_A and F_{PA}, and then add scores of the global factor F_G, finally select the argmax from them. In this case, the argmax is selected from $|\mathcal{P}_l|N$ candidates. The inference algorithm for the model is shown in Algorithm 4.

2.3 Learning Algorithm of the Joint Model

In terms of learning algorithm of the model, we borrow a fundamental idea of Kazama's perceptron learning algorithm, however, we use a more sophisticated online-learning algorithm based on the Passive-Aggressive Algorithm (PA) proposed by Crammer et al. [20].

For the sake of simplicity, we introduce some notations. We denote a predicate-argument structure $\mathbf{y} = \langle p, \mathcal{A} \rangle$. a local feature as $\Phi_L(\mathbf{x}, \mathbf{y}) = \Phi_P(\mathbf{x}, p) +$

Algorithm 4 The inference algorithm of the model

```
input sentence $\mathbf{x}$
$\hat{p} = \hat{\mathcal{A}} = null$
for $p_{l_i} \in \mathcal{P}_l$ do
  $\{\mathcal{A}\}^n$ = generate N-best assignments using $F_P$, $F_A$ and $F_{PA}$
  for $\mathcal{A}_j \in \{\mathcal{A}\}^n$ do
    $s(p_{l_i}, \mathcal{A}_j) = \mathbf{w} \cdot \Phi_G(\mathbf{x}, p_{l_i}, \mathcal{A}_j) + \mathbf{w} \cdot \Phi_P(\mathbf{x}, p_{l_i}) + \mathbf{w} \cdot \sum_{a \in \mathcal{A}_j} \{\Phi_A(\mathbf{x}, a) + \Phi_{PA}(\mathbf{x}, p_{l_i}, a)\}$
    if $s(p_{l_i}, \mathcal{A}_j) > s(\hat{p}, \hat{\mathcal{A}})$ then
      $\hat{p} = p_{l_i}, \hat{\mathcal{A}} = \mathcal{A}_j$
    end if
  end for
end for
return $\hat{p}$ and $\hat{\mathcal{A}}$
```

$\sum_{a \in \mathcal{A}} \{\Phi_A(\mathbf{x}, a) + \Phi_{PA}(\mathbf{x}, p, a)\}$, a feature vector coupled both local and global features as $\Phi_{L+G}(\mathbf{x}, \mathbf{y}) = \Phi_L(\mathbf{x}, \mathbf{y}) + \Phi_G(\mathbf{x}, p, \mathcal{A})$, the argmax using Φ_{L+G} as $\hat{\mathbf{y}}^{L+G}$, the argmax using Φ_L as $\hat{\mathbf{y}}^L$. Also, we use a loss function $\rho(\mathbf{y}, \mathbf{y}')$ which is defined by the the number of incorrect assignments in $\mathbf{y}'$.

Kazama's margin perceptron learning algorithm is summarized by the following two points.

(1) Local+Global Update: If $\mathbf{y} \neq \hat{\mathbf{y}}^{L+G}$, then update the parameter $\mathbf{w}$ using both local and global features. Even $\mathbf{y} = \hat{\mathbf{y}}^{L+G}$, if $\mathbf{w} \cdot \{\Phi_{L+G}(\mathbf{x}, \hat{\mathbf{y}}^{L+G}) - \Phi_{L+G}(\mathbf{x}, \mathbf{y}')\} \leq \rho(\hat{\mathbf{y}}^{L+G}, \mathbf{y}')$ $(\exists \mathbf{y}' : \hat{\mathbf{y}}^{L+G} \neq \mathbf{y}')$, then update the parameter $\mathbf{w}$ using both features.

(2) Local Update: Even if $\hat{\mathbf{y}}^{L+G} = \mathbf{y}$, the argmax $\hat{\mathbf{y}}^L$ using only local features is different from $\mathbf{y}$ $(\mathbf{y} \neq \hat{\mathbf{y}}^L)$, then update $\mathbf{w}$ using local features. Also, even $\mathbf{y} = \hat{\mathbf{y}}^L$, if $\mathbf{w} \cdot \{\Phi_L(\mathbf{x}, \hat{\mathbf{y}}^L) - \Phi_L(\mathbf{x}, \mathbf{y}')\} \leq \rho(\hat{\mathbf{y}}^L, \mathbf{y}')$ $(\exists \mathbf{y}' : \hat{\mathbf{y}}^L \neq \mathbf{y}')$, then update $\mathbf{w}$ using local features.

The necessity of the Local Update is that if we perform only Local+Global Update, the algorithm does not guarantee a sufficient margin, and it leads to poor quality N-best assignments.

The margin perceptron learning proposed by Kazama can be seen as an optimization with the following two constrains.

(A) $\mathbf{w} \cdot \Phi_{L+G}(\mathbf{x}, \mathbf{y}) - \mathbf{w} \cdot \Phi_{L+G}(\mathbf{x}, \hat{\mathbf{y}}^{L+G}) \geq \rho(\mathbf{y}, \hat{\mathbf{y}}^{L+G})$

(B) $\mathbf{w} \cdot \Phi_{L}(\mathbf{x}, \mathbf{y}) - \mathbf{w} \cdot \Phi_{L}(\mathbf{x}, \hat{\mathbf{y}}^{L}) \geq \rho(\mathbf{y}, \hat{\mathbf{y}}^{L})$

(A) is the constraint corresponds to Local+Global Update that ensures a sufficient margin $\rho(\mathbf{y}, \hat{\mathbf{y}}^{L+G})$ between $\mathbf{y}$ and $\hat{\mathbf{y}}^{L+G}$. (B) is the constraint corresponds to Local Update. The constraint is applied in cases that (A) is satisfied, and ensures a sufficient margin $\rho(\mathbf{y}, \hat{\mathbf{y}}^{L})$ between $\mathbf{y}$ and $\hat{\mathbf{y}}^{L}$.

The optimization problems of the Passive-Aggressive Algorithm with the above constraints are as follows.

$$\mathbf{w}_{new} = \arg\min_{\mathbf{w}' \in \Re^n} \frac{1}{2}||\mathbf{w}' - \mathbf{w}||^2 + C\xi$$
$$\begin{cases} \text{s.t. } l_{L+G}(\mathbf{w}; (\mathbf{x}, \mathbf{y})) \leq \xi \text{ and } \xi \geq 0 & \text{if } \hat{\mathbf{y}}^{L+G} \neq \mathbf{y} \\ \text{s.t. } l_{L}(\mathbf{w}; (\mathbf{x}, \mathbf{y})) \leq \xi \text{ and } \xi \geq 0 & \text{if } \hat{\mathbf{y}}^{L+G} = \mathbf{y} \neq \hat{\mathbf{y}}^{L} \end{cases} \tag{4.4}$$

$l_{L+G}(\mathbf{w}; (\mathbf{x}, \mathbf{y}))$ is the loss function for the case of using both local and global features, corresponds to applying the constraint (A). $l_{L}(\mathbf{w}; (\mathbf{x}, \mathbf{y}))$ is the loss function for the case of using only local features, corresponds to applying the constraints (B) in cases that (A) is satisfied. ξ is a slack variable for soft margins and C is a constant that controls an influence of ξ.

In this optimization, the parameter $\mathbf{w}$ is selected so as to minimize the l_2-norm of the new parameter $\mathbf{w}$ under a constraints. Either of these two constraints is selected for the optimization according to classification results. The loss function $l_{L+G}(\mathbf{w}; (\mathbf{x}, \mathbf{y}))$ and $l_{L}(\mathbf{w}; (\mathbf{x}, \mathbf{y}))$ are

$$l_{L+G}(\mathbf{w}; (\mathbf{x}, \mathbf{y})) = \mathbf{w} \cdot \Phi_{L+G}(\mathbf{x}, \hat{\mathbf{y}}^{L+G}) - \mathbf{w} \cdot \Phi_{L+G}(\mathbf{x}, \mathbf{y}) + \rho(\mathbf{y}, \hat{\mathbf{y}}^{L+G}) \tag{4.5}$$

$$l_{L}(\mathbf{w}; (\mathbf{x}, \mathbf{y})) = \mathbf{w} \cdot \Phi_{L}(\mathbf{x}, \hat{\mathbf{y}}^{L}) - \mathbf{w} \cdot \Phi_{L}(\mathbf{x}, \mathbf{y}) + \rho(\mathbf{y}, \hat{\mathbf{y}}^{L}) \tag{4.6}$$

The solutions of the optimization problem with the above constraints can be obtained using the method of Lagrange multipliers. The derivations are the same as the usual Passive-Aggressive Algorithm. For the details of the derivations of the Passive-Aggressive Algorithm, see Chapter 3, Section 2.2.

As a result, we get the learning algorithm shown in Algorithm 5. The line denoted by (A) is processed using both local features Φ_L and global features Φ_G if the argmax $\hat{\mathbf{y}}^{L+G}$ is different from the correct answer $\mathbf{y}_t$. It corresponds to the optimization which uses the constraint (4.5). If the argmax $\hat{\mathbf{y}}^{L+G}$ is the same as the correct answer $\mathbf{y}_t$, no updates occur because the two feature vectors are the same ($\Phi_{L+G}(\mathbf{x}_t, \mathbf{y}_t) = \Phi_{L+G}(\mathbf{x}_t, \hat{\mathbf{y}}^{L+G})$). On the other hand, the update denoted by (B) is processed in the case that the argmax $\hat{\mathbf{y}}^{L+G}$ is the same as the correct answer $\mathbf{y}_t$, however, the argmax using only local features $\hat{\mathbf{y}}^L$ is different from the correct answer $\mathbf{y}_t$. It corresponds to the optimization with the constraint (4.6).

The additive type online learning algorithms such as Perceptron and Passive-Aggressive Algorithms can be applied a technique of efficient parameter averaging that described in [23]. Algorithm 5 uses Daumé III's parameter averaging technique. The variables c and $\mathbf{w}_a$ are used for efficient parameter averaging.

The difference between Kazama's margin perceptron learning and the proposed algorithm can be seen in the nature of the two algorithms. In perceptron algorithms, since features of positive and negative examples are added to $\mathbf{w}$ by fixed coefficient (namely 1.0), resulting parameter becomes relatively large. In contrast, for each update, PA finds the smallest weight that satisfies the constraint, hence resulting parameter becomes more small.

2.4 An issue of F_{PA}

As described previously, F_{PA} affects scoring of both a predicate sense and argument roles, and a set of argument roles contains the label "NONE" which means the argument has no role. It is quite unlikely that the information "no role" contributes predicate sense disambiguation, however, if we introduce F_{PA} naively, it remains possible that the score of a particular sense is increased by "no role" arguments. In order to avoid such cases, we use a dummy sense p_{dummy} to "no role" arguments. If the role of argument are "NONE", instead of using the parameter $\mathbf{w}_{\mathrm{NONE} \wedge sense_i}$, we use $\mathbf{w}_{\mathrm{NONE} \wedge dummy}$.

Algorithm 5 The learning algorithm of the model

```
input Training set $\mathcal{T} = \{\mathbf{x}_t, \mathbf{y}_t\}_{t=1}^{T}$, Number of iterations N and Parameter C
$\mathbf{w} \leftarrow \mathbf{0}$, $\mathbf{w}_a \leftarrow \mathbf{0}$, $c \leftarrow 1$
for $i \leftarrow 0$ to $N$ do
  for $(\mathbf{x}_t, \mathbf{y}_t) \in \mathcal{T}$ do
    $\hat{\mathbf{y}}^{L+G}$ = get argmax using Algorithm 4 with penalty $\rho(\mathbf{y}_t, \mathbf{y})$
    $\hat{\mathbf{y}}^{L}$ = get the highest scoring assignment using $F_P$, $F_A$ and $F_{PA}$ with penalty $\rho(\mathbf{y}_t, \mathbf{y})$
    Let $\Delta\Phi_{L+G}(\mathbf{x}_t, \mathbf{y}_t, \hat{\mathbf{y}}^{L+G}) = \Phi_{L+G}(\mathbf{x}_t, \mathbf{y}_t) - \Phi_{L+G}(\mathbf{x}_t, \hat{\mathbf{y}}^{L+G})$
    $\tau_t = \min\left(C, \frac{\mathbf{w}\cdot\Delta\Phi_{L+G}(\mathbf{x}_t,\mathbf{y}_t,\hat{\mathbf{y}}^{L+G}) + \rho(\mathbf{y}_t,\hat{\mathbf{y}}^{L+G})}{||\Delta\Phi_{L+G}(\mathbf{x}_t,\mathbf{y}_t,\hat{\mathbf{y}}^{L+G})||^2}\right)$
    $\mathbf{w} \leftarrow \mathbf{w} + \tau_t(\Delta\Phi_{L+G}(\mathbf{x}_t, \mathbf{y}_t, \hat{\mathbf{y}}^{L+G}))$   (A)
    $\mathbf{w}_a \leftarrow \mathbf{w}_a + c\tau_t(\Delta\Phi_{L+G}(\mathbf{x}_t, \mathbf{y}_t, \hat{\mathbf{y}}^{L+G}))$
    if $\hat{\mathbf{y}}^{L+G} = \mathbf{y}_t$ and $\hat{\mathbf{y}}^{L} \neq \mathbf{y}_t$ then
      Let $\Delta\Phi_{L}(\mathbf{x}_t, \mathbf{y}_t, \hat{\mathbf{y}}^{L}) = \Phi_{L}(\mathbf{x}_t, \mathbf{y}_t) - \Phi_{L}(\mathbf{x}_t, \hat{\mathbf{y}}^{L})$
      $\gamma_t = \min\left(C, \frac{\mathbf{w}\cdot\Delta\Phi_{L}(\mathbf{x}_t,\mathbf{y}_t,\hat{\mathbf{y}}^{L}) + \rho(\mathbf{y}_t,\hat{\mathbf{y}}^{L})}{||\Delta\Phi_{L}(\mathbf{x}_t,\mathbf{y}_t,\hat{\mathbf{y}}^{L})||^2}\right)$
      $\mathbf{w} \leftarrow \mathbf{w} + \gamma_t(\Delta\Phi_{L}(\mathbf{x}_t, \mathbf{y}_t, \hat{\mathbf{y}}^{L}))$   (B)
      $\mathbf{w}_a \leftarrow \mathbf{w}_a + c\gamma_t(\Delta\Phi_{L}(\mathbf{x}_t, \mathbf{y}_t, \hat{\mathbf{y}}^{L}))$
    end if
    $c \leftarrow c + 1$
  end for
end for
return $\mathbf{w} - \mathbf{w}_a / c$
```

3. Experiment

3.1 Experimental Settings

We use the CoNLL-2009 Shared Task dataset for experiments, which is a dataset for multi-lingual syntactic and semantic dependency parsing. The dataset consists of seven languages, Catalan [99], Chinese [71], Czech [39], English [88], German [7], Japanese [45] and Spanish [99], which are annotated syntactic dependencies and predicate-argument structures. The English data is constructed by converting the constituent trees of Penn Treebank using the head rules proposed in [43], and added the predicate-argument structures of Proposition Bank [70] and NomBank [63]. The Japanese data consists of a part of Kyoto Corpus. The data is converted from "bunsetsu-based" dependencies to "word-based" dependencies using the head percolation rules [45]. For the details of the CoNLL-2009 Shared Task dataset, see [38].

In the CoNLL-2009 Shared Task SRL-only challenge, participants are imposed to identify predicate-argument structures of only the specified predicates. Therefore the problems to be solved are predicate sense disambiguation and argument role labeling [1]. Since the Japanese data does not have predicate sense annotation, the task to be solved is only argument role labeling.

We use **Semantic Labeled F1** for evaluation. Semantic Labeled F1 is calculated by the following equations. WSD and SRL mean predicate sense disambiguation and argument role labeling respectively.

$$\textbf{Semantic Labeled Precision} = \frac{\text{\# of correct pred. senses}+\text{\# of correct arg. roles}}{\text{\# predicates}+\text{\# of returned arg.}}$$

$$\textbf{Semantic Labeled Recall} = \frac{\text{\# of correct pred. senses}+\text{\# of correct arg. roles}}{\text{\# predicates}+\text{\# arguments}}$$

$$\textbf{Semantic Labeled F1} = \frac{2\times\text{Sem. Lab. Prec.}\times\text{Sem. Lab. Rec.}}{\text{Sem. Lab. Prec.}+\text{Sem. Lab. Rec.}}$$

In order to investigate the impact of the factors described in Section 2, we experimented the four experimental settings: use (1) only local factors

[1]The reason is that the dataset contains imperfect annotations. For instance, German data (SALSA Corpus) is annotated predicate-argument structures only a part of all verbs. The CoNLL-2008 Shared Task (English only) includes predicate identification task.

(F_P+F_A), (2) local and pairwise factors (F_P+F_A+F_{PA}), (3) local and global factors (F_P+F_A+F_G), (4) all factors (F_P+F_A+F_{PA}+F_G). For generating N-bests, we used the beam-search algorithm, and the number of N-bests was set to $N = 64$.

For learning of the joint model, the loss function $\rho(\mathbf{y}_t, \mathbf{y}')$ of the Passive-Aggressive Algorithm was set to the number of incorrect assignments of a predicate sense and argument roles. Also, the number of iterations of the model used for testing was selected based on the performance on the development data.

3.2 Features for the Joint Model

The features used for our joint model are as follows. In the experiments, we did not perform any features selection procedure.

Features for the Predicate Local Factor F_P

Predicate Token Features Predicted lemma of the predicate and predicate's head, and predicted POS of the predicate, and its conjunctions.

Dependency Label Dependency label between the predicate and predicate's head.

Child Dep Set The concatenation of the dependency labels of the predicate dependents.

Features for the Argument Local Factor F_A

Predicate Token Features Predicted lemma and predicted POS of the predicate.

Argument Token Features Predicted lemma and predicted POS of the argument candidate and the argument's head.

Context Features Predicted lemma and predicted POS of the Leftmost/rightmost dependent and leftmost/rightmost sibling.

Dependency Label The dependency label of predicate, argument candidate and argument candidate's dependent.

Family The position of the argument candidate with respect to the predicate position in the dependency tree (e.g. child, sibling and etc.)

Position The position of the head of the dependency relation with respect to the predicate position in the sentence.

Child Dep Set The left-to-right chain of the predicted dependency labels of the predicate's dependents.

Pred-Arg Dep Path Predicted lemma, predicted POS and dependency label paths between the predicate and the argument candidates.

Distance The number of dependency edges between the predicate and the argument candidate.

Features for the Predicate-Argument Pairwise Factor F_{PA}

Argument Token Features Predicted lemma of the argument candidate, and the conjunction of the predicted lemma and the POS of the argument candidate.

Pred-Arg Dep Path Dependency label path between the predicate and the argument candidates, for instance $\overleftarrow{VC}\overrightarrow{SBJ}$.

Features for the Global Factor F_G

Pred-Arg Label Sequence The sequence of the predicate and the argument labels in the predicate-argument structure (in order of positions), for instance *A0-PRED-A1*.

Presence of Labels Defined in Frame Files Whether the semantic roles defined in the frame of the sense present in the predicate-argument structure, for instance *CONTAINS:A1*. The conjunction of the predicate sense and the frame information, for instance *wear.01&CONTAINS:A1*.

	Ca	Ch	Cz	En	Ge	Jp	Sp
pruning algorithm	3	1	2	1	1	POS	3
coverage (%)	100	98.9	98.5	97.3	98.3	99.9	100
reduction (%)	88.8	69.1	49.1	63.1	64.3	41.0	89.6

Table 4.1. Pruning results. Coverage denotes the recall of true arguments, and reduction denotes the percentage of the number of argument candidates remain after applying the pruning algorithms.

The features for F_P and F_A are used in a number of work for predicate sense disambiguation and semantic role labeling respectively. The features for F_{PA} are lexical, since both types of features contain lemmas. The second feature, lemmas are tied with dependency paths, is not used in other work. The features for F_G include the predicate and argument sequence feature as well as frame-based features.

3.3 Argument Pruning

We observe that most of arguments tend to be near from its predicate on the dependency structure, we can prune argument candidates to reduce search space. Since the characteristics of the languages are slightly different, we apply three types of pruning algorithms.

Let S be the argument candidate set, n be the current node, $h(n)$ be the function that returns the parent node of n, $c(n)$ be the function that returns the children of n, $gc(n)$ be the function that returns grandchildren of n, p be the current node. initialization: $S \leftarrow \phi$ and $n \leftarrow p$.

Pruning Algorithm 1 (1) $S \leftarrow S \cup c(n)$. (2) $n \leftarrow h(n)$.
(3) repeat (1)-(2) until $n =$ ROOT.

Pruning Algorithm 2 (1) $S \leftarrow S \cup c(n) \cup gc(n)$. (2) $n \leftarrow h(n)$. (3) repeat (1)-(2) until $n =$ ROOT.

Pruning Algorithm 3 (1) $S \leftarrow c(n)$

	Avg.	Ca	Ch	Cz	En	Ge	Jp	Sp
F_P+F_A	79.17	78.00	76.02	85.24	83.09	76.76	77.27	77.83
F_P+F_A+F_{PA}	79.58	78.38	76.23	85.14	83.36	78.31	77.72	77.92
F_P+F_A+F_G	80.42	79.50	76.96	85.88	84.49	78.64	78.32	79.21
ALL	80.75	79.55	77.20	**85.94**	84.97	79.62	**78.69**	79.29
Björkelund	**80.80**	80.01	**78.60**	85.41	**85.63**	**79.71**	76.30	79.91
Zhao	80.47	**80.32**	77.72	85.19	85.44	75.99	78.15	**80.46**
Meza-Ruiz	77.46	78.00	77.73	75.75	83.34	73.52	76.00	77.91

Table 4.2. Results on the CoNLL-2009 Shared Task dataset (Semantic Labeled F1).

In addition to the above pruning, we also filtered argument candidates by POS. We filtered argument candidates assigned less frequent POS in the gold arguments. Table 4.1 shows the pruning results. In terms of Japanese, since we could not achieve high precision, we filtered out argument candidates by only POS.

3.4 Results

Table 4.2 shows the results of the experiments, and also shows the results of the top 3 systems on the CoNLL-2009 Shared Task participated as *SRL-only* system [5, 113, 65].

At first, we compare factor effects. By incorporating F_{PA}, we achieved performance improvements for all languages, especially in German (+1.55). This results suggest that it is effective to capture local inter-dependencies between a predicate and an argument. For Japanese, since we do not detect predicate senses, including F_{PA} can be thought to make less sense. However, by using F_{PA}, we achieved the F1 improvement of +0.45. The reason is that the features space of F_{PA} is not completely covered by F_A. For instance, the conjunction of lemma and dependency path is not included in F_A.

Comparing the results with F_P+F_A and F_P+F_A+F_G, incorporating F_G also contributed performance improvements for all languages, especially the significant F1 improvement of +1.88 is obtained in German.

		Avg.	Ca	Ch	Cz	En	Ge	Jp	Sp
SENSE	F_P+F_A	89.65	85.86	94.86	94.09	95.14	83.81	-	84.17
	F_P+F_A+F_{PA}	89.78	85.98	94.94	94.10	95.30	84.00	-	84.36
	F_P+F_A+F_G	89.83	86.66	95.00	94.23	95.24	83.09	-	84.79
	ALL	90.15	86.68	95.09	94.31	95.56	84.18	-	85.10
ARG	F_P+F_A	72.20	74.52	67.43	74.89	77.47	73.08	63.00	75.05
	F_P+F_A+F_{PA}	72.74	75.01	67.69	74.64	77.77	75.35	63.67	75.10
	F_P+F_A+F_G	74.11	76.33	68.75	76.11	79.50	76.34	65.03	76.77
	ALL	74.46	76.39	69.05	76.16	80.05	77.26	65.58	76.74

Table 4.3. Results on the CoNLL-2009 Shared Task data (predicate sense disambiguation and argument role labeling).

Next, we compare our system with CoNLL-2009 Shared Task top 3 systems. By incorporating both F_{PA} and F_G, our joint model achieved competitive results compare to the CoNLL-2009 top 2 systems (Björkelund and Zhao), and achieved the significant difference compare to the Meza-Ruiz's system [2]. Björkelund and Zhao applied feature selection algorithms in order to select the best set of feature templates for each language [3]. On the other hand, since our system uses the same feature templates for all language, there is still room for performance improvements by applying feature selection algorithms.

Meza-Ruiz's system also learns and analyzes predicate senses and argument roles simultaneously, however, the performances are significantly different. The reason is that the Meza-Ruiz's system does not use global features such as the argument label sequence and the frame-based features of F_G. As shown in Table 4.2, these features greatly affect system performances.

Table 4.3 shows the performances of predicate sense disambiguation and argument role labeling separately. The upper part "SENSE" stands for the performance of predicate sense disambiguation (accuracy) and lower part "ARG"

[2]The result of Meza-Ruiz for Czech is significantly worse than the other systems because of inappropriate preprocessing for predicate sense disambiguation. Excepting Czech, the average F1 value of the Meza-Ruiz is 77.75, where as our system is 79.89.

[3]Björkelund and Zhao reported that they took for features selection three to four weeks and up to two months respectively.

stands for the performance of argument role labeling (F1). Since we do not detect predicate senses for Japanese, the corresponding parts are filled by "-" [4].

In terms of sense disambiguation results, incorporating F_{PA} and F_G worked well for all languages. Especially the system achieved relatively high improvements for Spanish and Catalan (+0.93 and +0.86 respectively). Although incorporating either of F_{PA} and F_G provided improvements of +0.13 and +0.18 on average, adding both factors provided improvements of +0.50. This result suggests that combination of these factors is effective for sense disambiguation.

As for argument role labeling results, by incorporating F_{PA} and F_G contributed performances for all languages, especially the substantial gain +4.18 is achieved in German. By incorporating F_{PA} with local factors, the system achieved the F1 improvements of +0.54 on average. This result shows that capturing inter-dependencies between a predicate and an argument contributes argument role labeling. By incorporating F_G with local factors, the system achieved the significant improvement of F1 (+1.91).

Since both tasks are improved by using all factors, we can say that the proposed joint model succeeded in *joint learning* of predicate senses and argument roles.

4. Previous Work

Johansson and Nugues [44], and Björkelund et al. [5] decomposed the task of predicate-argument structure analysis into predicate word sense disambiguation, argument identification and argument classification (incl. predicate identification in the former). They applied a greedy feature selection algorithm for each task in order to select the optimal feature set, and used a re-ranker for selecting the globally plausible predicate-argument structure from candidates. The advantage of our model is that it can deal with inter-dependencies

[4]The reason of differing performance between Semantic Labeled F1 and ARG of Japanese is that the evaluation includes predicate sense disambiguation. In Japanese data, the correct senses are predicate lemmas.

between a predicate and arguments.

Zhao [112] and Zhao [113] applied for predicate-argument structure analysis a "history-based" approach where predicates and arguments are labeled sequentially, and the previous labels are used for the next labeling. They also applied a beam-search algorithm for searching the optimal assignments. Basically history based approaches suffer from error propagation. In contrast our joint model is a "graph-based" where the exact solution can be obtained by the inference algorithm. The other advantage of our model is that it can deal with inter-dependencies between predicates and arguments, while their approaches can not. Also, Zhao's feature selection results can be used for our joint model.

Meza-Ruiz and Riedel used Markov Logic Network (MLN) [77] for predicate-argument structure analysis [79, 64, 65]. Their systems perform the subtasks of predicate-argument structure analysis *collectively*; the predicate senses and the arguments of all predicates are labeled jointly. Hence their approaches have more representational power, however, we described previously, in MLN, it is difficult to deal with the particular type of features such as the core argument sequence feature.

5. Summary

In this paper, we proposed a structured model that captures both non-local dependencies between arguments, and inter-dependencies between argument roles and predicate senses. More precisely, we designed a linear model-based structured model, and defined four types of factors: predicate local factor, argument local factor, predicate-argument pairwise factor and global factor for the model. We also proposed a new online large-margin learning algorithm for linear models with global features.

In the experiments, despite the fact that we did not apply any feature selection algorithms, the proposed model achieved competitive results compare to the state-of-the-art systems.

Chapter 5

A Partially Joint Approach for Syntactic and Semantic Dependency Parsing

1. Introduction

The structured model proposed in Chapter 4 is assumed that predicates are already identified. Hence, in order to obtain predicate-argument structures from open texts, we need to identify predicates in sentences. A possible approach to this is to extend the structured model so as to perform predicate sense disambiguation and argument role labeling as well as predicate identification. However, it is not trivial how to extend the structured model to be able to identify predicates. Also, whether other information in terms of predicate-argument structures (e.g. argument roles) could be helpful for predicate identification is not trivial. The work of Meza-Ruiz [64] jointly performed predicate identification, predicate sense disambiguation, argument identification and argument classification using the MLN-based system. Their result seems that the joint approach did not contribute the performance of predicate identification. Therefore by preparing a predicate identifier and combining it with the structured model, we create a predicate-argument structure analyzer (semantic dependency parser).

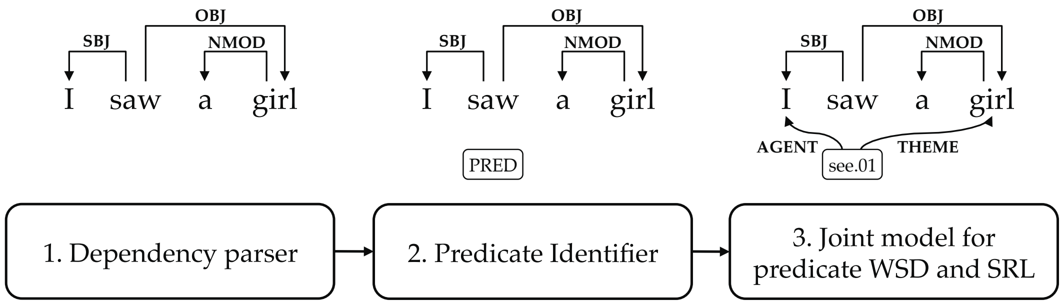


Figure 5.1. A partially joint architecture for syntactic and semantic dependency parsing

Since syntactic dependency parses are also necessary for the structured model, we also need to perform dependency parsing for sentences. In this chapter, we develop a high-performance syntactic and semantic dependency parser using only words and their part-of-speech tags. In the experiments, we use the CoNLL-2008 Shared Task dataset which are provided annotations of both syntactic and semantic dependencies, and compare to state-of-the-art parsers proposed by the previous work.

2. System Architecture

The task of syntactic and semantic dependency parsing can be decomposed into: **syntactic dependency parsing**, **predicate identification**, **predicate sense disambiguation** and **argument role labeling**. Though the argument role labeling task is sometimes decomposed into two tasks: argument identification and role labeling. We deal with them as a single task. Therefore, in order to develop a syntactic and semantic dependency parser, we need to perform the four tasks.

Figure 5.1 shows our system architecture for syntactic and semantic dependency parsing. The system consists of three parts: (1) a state-of-the-art higher-order projective dependency parser, (2) a predicate identifier, (3) the structured model for predicate sense disambiguation and argument role labeling. The procedure of the system is as follows. Given a sentence, the system obtains dependency trees with the dependency parser, and then identifies predicates in sentences using the predicate identifier, finally determines a predicate sense and its argument roles simultaneously. Since the system jointly solve the subtasks of syntactic and semantic dependency parsing, the system is a **partially joint** architecture.

2.1 Higher-order Dependency Parsing

The approaches of dependency parsing proposed by the previous work can be divided into **graph-based** and **transition-based**. The Eisner algorithm [26] for projective dependency parsing and the maximum spanning tree algorithm proposed by McDonald et al. [61] for non-projective dependency parsing are categorized graph-based. On the other hand, the shift-reduce-based parsing algorithms (e.g. Nivre et al. [67]) are categorized transition-based.

In this chapter we focus on graph-based dependency parsing algorithms, and consider labeled dependency parsing. In labeled dependency parsing, each edge in dependency trees are labeled a particular syntactic function such as subject, object, noun modifier, etc. For graph-based algorithms, the basic score function factorizes a dependency tree into pairs of a head and its dependent tied with a dependency label as follows.

$$s(\mathbf{y}) = \sum_{(h,m,l)\in\mathbf{y}} \mathbf{w}\cdot\Phi(h,m,l,\mathbf{x}) \tag{5.1}$$

where (h,m,l) is a particular edge in $\mathbf{y}$, h is a head word, m is a dependent, and l is a dependency label. The highest scoring dependency tree is returned by the model. This type of factorization is called **first-order**.

For the last few years, several researches have investigated higher-order dependency parsing in which scores of a dependency edge are affected by the other dependencies, and showed effectiveness in terms of parsing accuracy [59, 8, 62]. For projective dependency parsing, McDonald et al. extended the Eisner algorithm to second-order case in which scores depend on direct head-dependent relations as well as head-sibling relations [59], and moreover Carreras extended the McDonald's algorithm so as to consider not only sibling relations but also grand children relations [8]. On the other hand, second-order parsing for non-projective dependency parsing is proved to be *NP-hard*. McDonald et al. [62] proposed an approximate second-order projective dependency parsing where at first the parsing algorithm obtains the projective tree using the second-order projective dependency parsing, and then rearranges edges of the tree to expand the search space to non-projective cases. The another approach for higher-order non-projective dependency parsing is proposed by Martins et al. which formalizes the problem of higher-order non-projective dependency parsing with Integer Linear Programming (ILP) [57]. They reported that the significant improvement is obtained with the ILP appraoch compared to the McDonald's approximate algorithm.

We use for our system the higher-order projective dependency parsing algorithm proposed by Carreras [8]. In order to model the dependency parser, as for the structured model in Chapter 4, we use a linear model.

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}\in\mathcal{Y}} \mathbf{w}\cdot\Phi(\mathbf{x},\mathbf{y}) \tag{5.2}$$

where $\mathbf{w}$ is a parameter vector and $\Phi(\mathbf{x},\mathbf{y})$ is a feature vector that represents the tree $\mathbf{y}$. The tree $\mathbf{y}$ is returned if the inner product of the parameter vector and the corresponding feature vector $\Phi(\mathbf{x},\mathbf{y})$ has the highest value among possible trees.

We factorize the eq. (5.2) as sum of factors as follows.

$$s(\mathbf{x},\mathbf{y}) = \sum_{F_k\in\mathcal{F}} F_k(\mathbf{x},\mathbf{y}) = \sum_{F_k\in\mathcal{F}} \mathbf{w}\cdot\Phi_k(\mathbf{x},\mathbf{y}) \tag{5.3}$$

We use the second-order factor used in Carreras [8] defined as follows.

$$\begin{aligned} F_{2o}(h,m,l,\mathbf{x}) &= \mathbf{w}\cdot\Phi(h,m,l,\mathbf{x}) + \mathbf{w}\cdot\Phi(h,m,c_h,l,\mathbf{x}) \\ &\quad + \mathbf{w}\cdot\Phi(h,m,c_{mi},l,\mathbf{x}) + \mathbf{w}\cdot\Phi(h,m,c_{mo},l,\mathbf{x}) \end{aligned} \tag{5.4}$$

where $\mathbf{w}$ is a parameter vector, Φ is a feature vector, c_h is the child of h in the span $[h...m]$ that is closest to m, c_{mi} is the child of m in the span $[h...m]$ that is farthest from m and c_{mo} is the child of m outside the span $[h...m]$ that is farthest from m.

The features used for our dependency parser are based on those listed in [42]. In addition, distance features are used. We use shorthand notations in order to simplify the feature representations: 'h', 'd', 'c', 'l', 'p', '-1' and '$+1$' correspond to head, dependent, sibling or grandchild, lemma , POS, left position and right position respectively.

First-order Features

Token features: hl, hp, hl+hp, dl, dp and dl+dp.

Head-Dependent features: hp+dp, hl+dl, hl+dl, hl+hp+dl, hl+hp+dp, hl+dl+dp, hp+dl+dp and hl+hp+dl+dp.

Context features: hp+hp$_{+1}$+dp$_{-1}$+dp, hp$_{-1}$+hp+dp$_{-1}$+dp, hp+hp$_{+1}$+dp+dp$_{+1}$ and hp$_{-1}$+hp+dp+dp$_{+1}$.

Distance features: The number of tokens between the head and the dependent.

Second-order Features

Head-Dependent-Grandchild (or Sibling): hl+cl, hl+cl+cp, hp+cl, hp+cp, hp+dp+cp, dp+cp, dp+cl+cp, dl+cp, dl+cp+cl

2.2 Predicate Identification

As with the dependency parser, we use a linear model for the predicate identifier as follows.

$$\hat{y} = \arg \max_{y \in \{true, false\}} \mathbf{w} \cdot \Phi(\mathbf{x}, y) \tag{5.5}$$

This model performs binary decisions (*true* or *false*). If the model returns *true*, the example is identified as a predicate. The features used for the predicate identifier are as follows.

Features for the Predicate Identifier

Token Features The lemma of the predicate and predicate's head, and predicted POS of the predicate, and its conjunctions.

Dependency Label The dependency label between the predicate and predicate's head.

Child Dep Set The concatenation of the dependency labels of the predicate dependents.

Lemma in Frame Dictionary Whether the frame dictionary contains the predicate's lemma (e.g. LEMMAINFRAMEDIC or !LEMMAINFRAMEDIC).

2.3 Joint Model for Predicate Sense Disambiguation and Argument Role Labeling

For each predicate identified by the predicate identifier described in Section 2.2, the joint model proposed in Chapter 4 detects both a predicate sense and its argument roles. The score function for a predicate sense p and its argument roles $\mathcal{A}$ is defined as follows.

$$s(p, \mathcal{A}) = \sum_{F_k \in \mathcal{F}} F_k(\mathbf{x}, p, \mathcal{A}) \tag{5.6}$$

$$= \mathbf{w} \cdot \Phi_P(\mathbf{x}, p) + \mathbf{w} \cdot \Phi_G(\mathbf{x}, p, \mathcal{A}) + \mathbf{w} \cdot \sum_{a \in \mathcal{A}} \{\Phi_A(\mathbf{x}, a) + \Phi_{PA}(\mathbf{x}, p, a)\}. \tag{5.7}$$

The set of features used for the joint model is the same as that described in Chapter 4.

3. Experiment

3.1 Experimental Settings

For evaluating our system, we use the CoNLL-2008 Shared Task Dataset which contains annotations of both syntactic dependencies and predicate-argument structures. Although the dataset is almost the same as the English data of the CoNLL-2009 Shared Task Dataset, predicate identification task is included in evaluation of the CoNLL-2008 Shared Task, which is not in CoNLL-2009 Shared Task.

For the dependency parser and the predicate identifier training, we used the averaged Passive-Aggressive Algorithm for learning the dependency parser and the predicate identifier. For the dependency parser, we set the loss function $\rho(\mathbf{y}_t, \hat{\mathbf{y}})$ as the number of incorrect assignments, and C as 1.0. For the predicate identifier, we set $\rho(y_t, y)$ as 1, and C as 1.0. For the joint model, we used the all factors (F_{Lp}, F_{La}, F_{PW} and F_G) and the same features described in Chapter 4.

The following metrics are used for evaluation.

Semantic Labeled Precision $= \frac{\text{\# of correct pred. senses} + \text{\# of correct arg. roles}}{\text{\# predicates} + \text{\# of returned arg.}}$

Semantic Labeled Recall $= \frac{\text{\# of correct pred. senses} + \text{\# of correct arg. roles}}{\text{\# predicates} + \text{\# arguments}}$

Semantic Labeled F1 $= \frac{2 \times \text{Sem. Lab. Precision} \times \text{Sem. Lab. Recall}}{\text{Sem. Lab. Precision} + \text{Sem. Lab. Recall}}$

Labeled Attachment Score (LAS) the proportion of tokens that are assigned both the correct head and the correct dependency label.

Labeled Macro Precision (LMP) $= 0.5 \times Sem.Lab.Precision + 0.5 \times LAS$

Labeled Macro Recall (LMR) $= 0.5 \times Sem.Lab.Recall + 0.5 \times LAS$

Macro F1 harmonic mean of LMP and LMR.

3.2 Results

Table 3.2 shows the experimental results of our system, the top 4 systems [44, 11, 15, 114] in the CoNLL-2008 Shared Task and the subsequent work [112, 100].

Our system outperformed the second best system in the CoNLL-2008 Shared Task (Ciaramita 2008 [15]), and in terms of predicate identification and its sense disambiguation, the performance was competitive to the top system. However, the overall performance (Macro F1) of our system is slightly worse than Johansson's system and Zhao's system because of the relatively low performance of argument role labeling. The top two systems (Johansson 2008 and Zhao 2009) applied greedy feature selection algorithms to obtain the optimal

	Macro F1	LAS	Sem. F1	Pred F1	Arg F1
Johansson 2008 [44]	85.49	89.32	81.65	87.22	79.04
Zhao 2009 [112]	84.93	88.39	80.53	86.80	77.60
Our System	84.51	88.55	80.16	87.03	76.97
Ciaramita 2008 [15]	82.69	87.37	78.00	-	-
Che 2008 [11]	82.66	86.75	78.52	85.31	75.27
Titov 2009 [100]	81.80	87.50	76.10	-	-
Zhao 2008 [114]	81.44	86.66	76.16	78.26	75.18

Table 5.1. Macro F1 scores of our results and the existing systems. Pred F1 denotes F1 value of predicate identification and word sense disambiguation, and Arg F1 denotes F1 value of argument role labeling.

feature set for argument role labeling, while the other systems including our's are constructed without any feature selection algorithms. Also, the top two systems used batch learning algorithms for linear models, while our system uses an online learning algorithm. These facts might affected the system performances. However, the results are promising because our system achieved the best performances compared to the systems that use no feature selection algorithms. By applying features selection algorithms as in Johansson (2008) and Zhao (2009), our system might improve performances.

4. Summary

In this chapter, we proposed a partially joint system for syntactic and semantic dependency parsing. The system consists of three parts: a higher-order projective dependency parser, a predicate identifier and the structured model for joint learning of predicate senses and argument roles proposed in Chapter 4. In the experiment, the system achieved the competitive result in terms of predicate identification and disambiguation compared to the top system, and the best performance compared to the systems which use no feature selection algorithms.

Chapter 6

Acquiring Named Entity Information from Wikipedia

1. Introduction

Named Entities refer to proper nouns (e.g. PERSON, LOCATION and ORGANIZATION), temporal expression and numerical expressions. Since a large number of named entities exist in the world, unknown expressions appear frequently in texts, and they become causes of errors in text analysis. To cope with the problem, it is effective to add a large number of named entities to gazetteers.

Besides, Named Entities play an important role in NLP application such as Relation Extraction(RE), Information Retrieval(IR) and Question Answering(QA). For example, in QA systems, questions are given in natural language, and the systems explore the answer of a question from documents. In the case of factoid type questions in which the answer of a question is a named entity, systems first try to identify the NE class of the answer, and explore the NEs that belong to the identified class, and select the correct answer among them. If named entity classes of terms are unidentifiable, systems cannot find the correct answer.

In recent years, NE extraction has been performed with machine learning based methods. However, such methods cannot cover all of entities in texts. Therefore, it is necessary to extract entities from existing resources and use them to identify more entities. There are many useful resources on the Web. We focus on Wikipedia[1] as the resource for acquiring NEs. Wikipedia is a free multilingual online encyclopedia and a quickly growing resource. In Wikipedia, a huge number of named entities are described in titles of articles with useful information such as HTML tree structure and categories. Each article has anchor texts which refer to other related articles. According to these characteristics, they could be an appropriate resource for extracting named entities.

Since a specific entity or concept is glossed in a Wikipedia article, we can regard the named entity extraction problem as a document classification problem of the Wikipedia article. In traditional approaches for document classification, in many cases, documents are classified independently. However, the Wikipedia articles are hypertexts and they have rich structure that is useful information for categorization. For example, mentions included in hyperlinks (we call them **anchor text**) enumerated in a list tend to refer to the articles that describe other named entities belonging to the same class. It is expected that more accurate named entity categorization is accomplished by capturing such dependencies.

In order to incorporate such dependencies into the graph structure, we define three types of cliques that correspond to pairs of anchor texts over the tree structure, and then provide a probabilistic model induced by HTML document structure.

So far, several statistical models that can capture dependencies between examples have been proposed. There are two types of classification methods that can capture dependencies: iterative classification methods [56, 55] or collective classification methods [29, 97]. In this paper, we use Conditional Random Fields [50] for named entity categorization in Wikipedia.

[1]http://wikipedia.org/

2. Graph-based CRFs for Named Entity Categorization in Wikipedia

In this section we describe how to apply CRFs for named entity categorization in Wikipedia.

In Wikipedia, each article describes a specific entity or concept. Each article has a heading word, definition, and one or more categories. One possible approach is to classify each of the NE described in an article into an appropriate category by exploiting the definition of the article. This process can be done one by one without considering the relationship with other articles.

On the other hand, articles in Wikipedia are semi-structured texts, and they have some characteristics which do not exist in unstructured texts. Especially lists `(<UL> or <OL>)` and tables `(<TABLE>)` have important characteristics, that is, occurrence of elements in them have some sort of dependencies. Figure 6.1 shows an example of a part of HTML document and corresponding tree structure. The first anchor texts in each list tag (`<LI>`) tend to be in a same NE category. This characteristics is useful feature for the categorization task. In this paper we focus on lists which appear frequently in Wikipedia.

Furthermore, there are anchor texts in articles. Anchor texts are glossed entity or concept described in the linked page. With this in mind, NE categorization problem can be regarded as NE category labeling problem for anchor texts in articles. Exploiting dependencies of anchor texts that are induced by HTML structure, it can be expected to improve categorization accuracy.

We use CRFs for categorization in which anchor texts correspond to random variables V in a graph $\mathcal{G}$ and a dependency between anchor texts are treated as edges E in $\mathcal{G}$. In the next section, we describe the concrete way to construct graphs.

2.1 Constructing graphs based on DOM structure

An HTML document can be regarded as an ordered tree. We define a graph $\mathcal{G}$ on the ordered tree $\mathcal{T}^{ordered} = (V^{\mathcal{T}}, E^{\mathcal{T}})$: the nodes $V^{\mathcal{G}}$ are anchor texts in the HTML text; three types of cliques are introduced as the edges $E^{\mathcal{G}}$: **Sibling,**

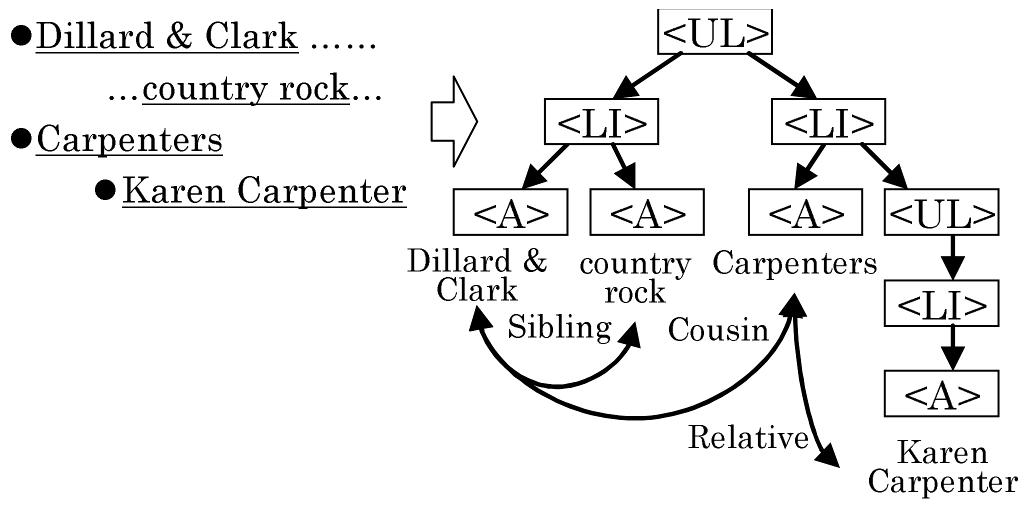


Figure 6.1. Correspondence between the HTML tree structure and the defined cliques

Cousin, and **Relative**. These cliques are introduced to encode NE label dependency on which the two NEs tend to be in a same class, or one NE affects the other NE label determination.

Then we consider dependent anchor text pairs in Figure 6.1. First, "Dillard & Clark" and "country rock" have a sibling relation over the tree structure, and appearing the same element of the list. The elements that have this relation tend to have a relation in which the following element is an attribute or a concept of the preceding element. Second, "Dillard & Clark" and "Carpenters" have a cousin relation over the tree structure, and they can be described that they tend to have a common attribute such as "Artist". The elements that have this relation tend to belong to the same class. Third, "Carpenters" and "Karen Carpenter" have a relation in which "Karen Carpenter" is a sibling's grandchild in relation to "Carpenters" over the tree structure. The elements that have this relation tend to have a relation in which the following element is a constituent part of the preceding element. It can be thought that the model can capture dependencies by dealing with anchor texts that dependent each other as cliques. Then, based on the observations above, we treat anchor text pairs as cliques satisfying the following three definitions over ordered tree.

Sibling $E_S = \{(v_i^{\mathcal{T}}, v_j^{\mathcal{T}}) | v_i^{\mathcal{T}}, v_j^{\mathcal{T}} \in V^{\mathcal{T}},\ d(v_i^{\mathcal{T}},\ cpa(v_i^{\mathcal{T}},\ v_j^{\mathcal{T}})) = d(v_j^{\mathcal{T}},\ cpa(v_i^{\mathcal{T}},\ v_j^{\mathcal{T}})) = 1,\ v_j^{\mathcal{T}} = ch(pa(v_j^{\mathcal{T}}, 1), k), v_i^{\mathcal{T}} = ch(pa(v_i^{\mathcal{T}}, 1),\ \max\{l | l < k\})\}$

Cousin $E_C = \{(v_i^{\mathcal{T}}, v_j^{\mathcal{T}}) | v_i^{\mathcal{T}},\ v_j^{\mathcal{T}} \in V^{\mathcal{T}}, d(v_i^{\mathcal{T}},\ cpa(v_i^{\mathcal{T}}, v_j^{\mathcal{T}})) = d(v_j^{\mathcal{T}},\ cpa(v_i^{\mathcal{T}}, v_j^{\mathcal{T}})) \geq 2,$
$v_i^{\mathcal{T}} = ch(pa(v_i^{\mathcal{T}}), k), v_j^{\mathcal{T}} = ch(pa(v_j^{\mathcal{T}}), k),\ pa(v_j^{\mathcal{T}}, d(v_j^{\mathcal{T}}, cpa(v_i^{\mathcal{T}}, v_j^{\mathcal{T}})) - 1)$
$= ch(pa(v_j^{\mathcal{T}}, d(v_j^{\mathcal{T}}, cpa(v_i^{\mathcal{T}}, v_j^{\mathcal{T}}))), k), pa(v_i^{\mathcal{T}}, d(v_i^{\mathcal{T}}, cpa(v_i^{\mathcal{T}}, v_j^{\mathcal{T}})) - 1)$
$= ch(pa(v_i^{\mathcal{T}}, cpa(v_i^{\mathcal{T}}, v_j^{\mathcal{T}})), \max\{l | l < k\})\}$

Relative $E_R = \{(v_i^{\mathcal{T}}, v_j^{\mathcal{T}}) | v_i^{\mathcal{T}}, v_j^{\mathcal{T}} \in V^{\mathcal{T}}, d(v_i^{\mathcal{T}}, cpa(v_i^{\mathcal{T}}, v_j^{\mathcal{T}})) = 1, d(v_j^{\mathcal{T}}, cpa(v_i^{\mathcal{T}}, v_j^{\mathcal{T}})) = 3,$
$pa(v_j^{\mathcal{T}}, 2) = ch(pa(v_j^{\mathcal{T}}, 3), k), v_i^{\mathcal{T}} = ch(pa(v_i^{\mathcal{T}}, 1), \max\{l | l < k\})\}$

Figure 6.2. The definitions of sibling, cousin and relative cliques, where E_S, E_C, E_R correspond to sets which consist of anchor text pairs that have sibling, cousin and relative relations respectively.

Consider ordered tree $\mathcal{T}^{ordered} = (V^{\mathcal{T}}, E^{\mathcal{T}})$, where $V^{\mathcal{T}}$ and $E^{\mathcal{T}}$ are nodes and edges over the tree. Let distance $d(v_i^{\mathcal{T}}, v_j^{\mathcal{T}})$ be the number of edges between $v_i^{\mathcal{T}}, v_j^{\mathcal{T}} \in V^{\mathcal{T}}$, $pa(v_i^{\mathcal{T}}, k)$ be k-th generation ancestor of $v_i^{\mathcal{T}}$, $ch(v_i^{\mathcal{T}}, k)$ be $v_i^{\mathcal{T}}$'s kth child, $cpa(v_i^{\mathcal{T}}, v_j^{\mathcal{T}})$ be common ancestor of $v_i^{\mathcal{T}}, v_j^{\mathcal{T}} \in V^{\mathcal{T}}$. Then we define three relations as cliques. The definitions of cliques are shown in Figure 6.2.

Note that the defined cliques are restricted to pairs of nearest vertices because of computational cost. Consider a case in which a graph consist of 8 anchor texts and each pair has Cousin relation, then the number of cliques increases exponentially.

The pairs of vertices satisfying the definitions are treated as cliques . That is, $C = E_S \cup E_C \cup E_R \cup V$.

2.2 Model

We introduce potential functions for cliques to define conditional probability distribution over CRFs. Conditional distribution over label set $\mathbf{y}$ given observation set $\mathbf{x}$ are given as:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \left(\prod_{(v_i, v_j) \in E_S, E_C, E_R} \Phi_{SCR}(y_i, y_j, \mathbf{x}) \right) \left(\prod_{v_i \in V} \Phi_V(y_i, \mathbf{x}) \right) \tag{6.1}$$

where $\Phi_{SCR}(y_i, y_j)$ are the potentials over sibling, cousin, and relative edges, $\Phi_V(y_i, \mathbf{x})$ are the potentials over the nodes, and $Z(\mathbf{x})$ is the partition function. The potentials $\Phi_{SCR}(y_i, y_j)$ and $\Phi_V(y_i, \mathbf{x})$ factorize according to the features f_k and weights λ_k as:

$$\Phi_{SCR}\left(y_i, y_j\right) = \exp\left(\sum_k \lambda_k f_k(y_i, y_j)\right) \tag{6.2}$$

$$\Phi_V\left(y_i, \mathbf{x}\right) = \exp\left(\sum_{k'} \lambda'_k f'_k(y_i, \mathbf{x})\right) \tag{6.3}$$

$f_k(y_i, y_j)$ captures co-occurrences between labels, where $k \in \{(y_i, y_j)|\mathcal{Y} \times \mathcal{Y}\}$ corresponds to the particular element of the Cartesian product of the label sets $\mathcal{Y}$. $f'_k(y_i, \mathbf{x})$ captures co-occurrences between label $y_i \in \mathcal{Y}$ and observation features, where k' corresponds to the particular element of the label set and observed features.

The weights of a CRF, $\Lambda = \{\lambda_k, \ldots, \lambda'_k, \ldots\}$ are estimated to maximize the conditional log-likelihood of the graph in a training dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^N$. The log-likelihood function can be defined as follows:

$$\begin{aligned}\mathcal{L}_\lambda = \sum_{d=1}^{N} [\sum_{(v_i,v_j)\in E_S^{(d)},E_C^{(d)},E_R^{(d)}} & \sum_k \lambda_k f_k(y_i^{(d)}, y_j^{(d)}) \\ & + \sum_{v_i \in V^{(d)}} \sum_{k'} \lambda_{k'} f_{k'}(y_i^{(d)}, \mathbf{x}^{(d)}) - logZ(\mathbf{x}^{(d)})] \\ & - \sum_k \frac{\lambda_k^2}{2\sigma^2} - \sum_{k'} \frac{\lambda_{k'}^2}{2\sigma^2}\end{aligned} \tag{6.4}$$

where the last two terms are due to the Gaussian prior [13] used to reduce overfitting. Quasi-Newton methods, such as L-BFGS [52] can be used for maximizing the function.

2.3 Tree-based Reparameterization

Since the proposed model may include loops, it is necessary to introduce an approximation to calculate marginal probabilities. We use Tree-based Repa-

Named Entity Classes		SIZE
NAME	EVENT	121
	PERSON	3315
	UNIT	15
	LOCATION	1480
	FACILITY	2449
	TITLE	42
	ORG	991
	VOCATION	303
	NATURAL_OBJ	1132
	PRODUCT	1664
	NAME_OTHER	24
TIMEX/NUMEX	TIMEX/NUMEX	2749
OTHER		1851
ALL		16136

Table 6.1. The number of anchor texts per NE-class in evaluation datasets.

rameterization (TRP) [105] for approximate inference. TRP enumerates a set of spanning trees $Y = \{\mathcal{T}\}$ from the graph. Then, inference is performed by applying an exact inference algorithm such as Belief Propagation to each of the spanning trees, and updates of marginal probabilities is continued until they converge.

3. Experiment

In this section, we report the experimental results of named entity extraction from Wikipedia using the proposed model.

3.1 Dataset

Our dataset is a random selection of 2300 articles from the Japanese version of Wikipedia as of October 2005. All anchor texts appearing under HTML `<LI>` tags are hand-annotated with NE class label. We use the Extended Named

	SCR	SC	SR	CR
# loopy examples	318 (36%)	324 (32%)	101 (1%)	42 (2%)
# linear chain or tree examples	555 (64%)	631 (62%)	2883 (27%)	1464 (54%)
# one node examples	0 (0%)	60 (6%)	7800 (72%)	1176 (44%)
# total examples	873	1015	10784	2682
# nodes per example on avg.	18.5	15.8	1.5	6.0
	S	C	R	I
# loopy examples	0 (0%)	0 (0%)	0 (0%)	0 (0%)
# linear chain or tree examples	2913 (26%)	1631 (54%)	237 (2%)	0 (0%)
# one node examples	8298 (74%)	1380 (46%)	15153 (98%)	16136 (100%)
# total examples	11211	3011	15390	16136
# nodes per example on avg.	1.4	5.4	1.05	1

Table 6.2. The dataset details constructed from each model.

Entity Hierarchy (ENE) [83] as the NE class labeling guideline, but reduce the number of classes to 13 from the original 200+ by ignoring fine-grained categories and nearby categories in order to avoid data sparseness. The correspondence of the ENE classes and the number of anchors in the dataset is shown in Table 6.1. Also, we eliminate examples that consist of less than two nodes in the SCR model. There are 16136 anchor texts with 14285 NEs. The number of Sibling, Cousin and Relative edges in the dataset are $|E_S| = 4925$, $|E_C| = 13134$ and $|E_R| = 746$ respectively.

3.2 Experimental Settings

The aims of experiments are the two-fold. Firstly, we investigate the effect of each type of cliques. The several graphs are composed with the three sorts of edges. We also compare the graph-based models with a node-wise method – just MaxEnt method not using any edge dependency. Secondly, we compare the proposed method by CRFs with a baseline method by Support Vector Machines (SVMs) [104].

The experimental settings of CRFs and SVMs are as follows.

CRFs

In order to investigate which type of clique boosts classification performance, we perform experiments on several CRFs models that are constructed from combinations of defined cliques. Resulting models of CRFs evaluated on this experiments are SCR, SC, SR, CR, S, C, R and I (independent). Figure 6.3 shows representative graphs of the eight models. When the graph are disconnected by reducing the edges, the classification is performed on each connected subgraph. We call it an *example*. We name the *examples* according the graph structure: ″loopy examples″ are subgraphs including at least one cycle; ″linear chain or tree examples″ are subgraphs including not a cycle but at least an edge; ″one node examples″ are subgraphs without edges. Table 1 shows the distribution of the examples of each model. Since SCR, SC, SR and CR model have loopy examples, TRP approximate inference is necessary. To perform training and testing with CRFs, we use GRMM [90] with TRP. We set the Gaussian Prior variances for weights as $\sigma^2 = 10$ in all models.

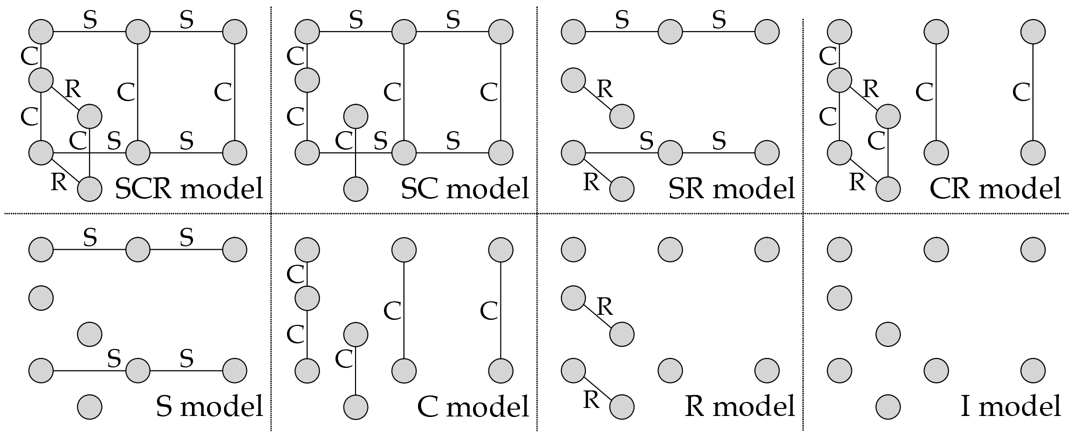


Figure 6.3. An example of graphs constructed by combination of defined cliques

types	feature	SVMs	CRFs
observed	definition (bag-of-words)	$\surd$	$\surd\,(V)$
features	heading of articles	$\surd$	$\surd\,(V)$
	heading of articles (morphemes)	$\surd$	$\surd\,(V)$
	categories articles	$\surd$	$\surd\,(V)$
	categories articles (morphemes)	$\surd$	$\surd\,(V)$
	anchor texts	$\surd$	$\surd\,(V)$
	anchor texts (morphemes)	$\surd$	$\surd\,(V)$
	parent tags of anchor texts	$\surd$	$\surd\,(V)$
	last header of anchor texts	$\surd$	$\surd\,(V)$
	last header of anchor texts (morphemes)	$\surd$	$\surd\,(V)$
label	between-label feature		$\surd\,(S,C,R)$
features	previous label	$\surd$	

Table 6.3. Features used in experiments. "$\surd$" means that the corresponding features are used in classification. The V, S, C and R in CRFs column corresponds to the node, sibling edges, cousin edges and relative edges respectively.

SVMs

We introduce two models by SVMs (model I and model P). In model I, each anchor text is classified independently. In model P, we ordered the anchor texts in a linear-chain sequence. Then, we perform a history-based classification along the sequence, in which $j-1$-th classification result is used in j-th classification. To perform training and testing with SVMs, we use TinySVM [2] with a linear-kernel, and one-versus-rest is used for multi-class classification. We used the cost of constraint violation $C = 1.0$.

Features for CRFs and SVMs

The features used in the classification with CRFs and SVMs are shown in Table 6.3. Japanese morphological analyzer MeCab [3] is used to obtain morphemes.

[2]http://www.chasen.org/~taku/software/TinySVM/
[3]http://mecab.sourceforge.net/

		CRFs								SVMs	
NE CLASS	N	C	CR	I	R	S	SC	SCR	SR	I	P
PERSON	3315	.7419	.7429	.7453	.7458	.7507	.7533	**.7981**	.7515	.7383	.7386
TIMEX/NUMEX	2749	.9936	**.9944**	.9940	.9936	.9938	.9931	.9933	.9940	.9933	.9935
FACILITY	2449	.8546	.8541	.8540	.8516	.8500	.8530	.8495	.8495	.8504	**.8560**
PRODUCT	1664	.7414	**.7540**	.7164	.7208	.7130	.7371	.7418	.7187	.7154	.7135
LOCATION	1480	**.7265**	.7239	.6989	.7048	.6974	.7210	.7232	.7033	.7022	.7132
NATURAL_OBJ	1132	.3333	.3422	.3476	.3513	.3547	.3294	.3304	.3316	**.3670**	.3326
ORG	991	.7122	.7160	.7100	.7073	.7122	.6961	.5580	.7109	.7141	**.7180**
VOCATION	303	.9088	.9050	.9075	.9059	.9150	.9122	.9100	**.9186**	.9091	.9069
EVENT	121	.2740	.2345	.2533	.2667	.2800	.2740	.2759	.2667	.3418	**.3500**
TITLE	42	.1702	.0889	.2800	.2800	**.3462**	.2083	.1277	**.3462**	.2593	.2642
NAME_OTHER	24	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	**.0690**	.0000
UNIT	15	.2353	.1250	.2353	.2353	.2353	.1250	.1250	.2353	**.3333**	.3158
ALL	14285	.7846	**.7862**	.7806	.7814	.7817	.7856	.7854	.7823	.7790	.7798
ALL (no articles)	3898	.5476	**.5495**	.5249	.5274	.5272	.5484	.5465	.5224	.5278	.5386

Table 6.4. Comparison of F1-values of CRFs and SVMs.

3.3 Evaluation

We evaluate the models by 5 fold cross-validation. Since the number of examples are different in each model, the datasets are divided taking the examples – namely, connected subgraphs – in SCR model. The size of divided five subdata are roughly equal. We evaluate per-class and total extraction performance by F1-value.

3.4 Results

Table 3 shows the classification accuracy of each model. The second column "N" stands for the number of nodes in the gold data. The second last row "ALL" stands for the F1-value of all NE classes. The last row "ALL (no article)" stands for the F1-value of all NE classes which have no gloss texts in Wikipedia.

In terms of results with SVMs, only the results with $C = 1.0$ are shown since we achieved the best results with the setting.

Relational vs. Independent

Among the models constructed by combination of defined cliques, the best F1-value is achieved by CR model, followed by SC, SCR, C, SR, S, R and I. We performed McNemar paired test on labeling disagreements between CR model of CRFs and I model of CRFs. More precisely, at first, we count the number of examples (a) correctly classified by only m_1, (b) correctly classified by only m_2. Next we test the null hypothesis H_0, namely (a)=(b). If the result is contained in rejection region $p < 0.01$, then it is determined that the two models have significant difference. We compared the CR model and I model, and the difference was significant ($p < 0.01$). These results show that considering dependencies work positively in obtaining better accuracy than classifying independently. The Cousin cliques provide the highest accuracy improvement among the three defined cliques. The reason may be that the Cousin cliques appear frequently in comparison with the other cliques, and also possess strong dependencies among anchor texts. As for PERSON, better accuracy is achieved in SC and SCR models. In fact, the PERSON-PERSON pairs frequently appear in Sibling cliques (435 out of 4925) and in Cousin cliques (2557 out of 13125) in the dataset. Also, as for PRODUCT and LOCATION, better accuracy is achieved in the models that contain Cousin cliques (C, CR, SC and SCR model). 1072 PRODUCT-PRODUCT pairs and 738 LOCATION-LOCATION pairs appear in Cousin cliques. "All (no article)" row in Table 3 shows the F1-value of nodes which have no gloss texts. The F1-value difference between CR and I model of CRF in "ALL (no article)" row is larger than the difference in "All" row. The fact means that the dependency information helps to extract NEs without gloss texts in Wikipedia.

CRFs vs. SVMs

The best model of CRFs (CR model) outperforms the best model of SVMs (P model). We performed McNemar paired test on labeling disagreements between CR model of CRFs and P model of SVMs. The difference was significant ($p < 0.01$). In the classes having larger number of examples, models of CRFs achieve better F1-values than models of SVMs. However, in several classes having smaller number of examples such as EVENT and UNIT, models of SVMs achieve significantly better F1-values than models of CRFs.

What type of edge features contribute?

Although the potential function of the proposed model for edges capture only combinations of pairs of labels, the model can deal with more sparse features in which a particular observation features is tied with a pair of labels. That is, the edge potential function Φ_{SCR} of the proposed model is defined by the following.

$$\Phi_{SCR} = \exp\left(\sum_k \lambda_k f_k(y_i, y_j, \mathbf{x})\right)$$

It is unclear that what kinds of observation features $x_l \in \mathbf{x}$ should be selected for the potential function Φ_{SCR} in order to determine two labels y_i, y_j. The observation features that capture relations between both anchors might appropriate for the edge features. However, among the features in Table 6.3, words and categories in articles should be included in the corresponding node potentials. Because, if the number of nodes is 1, then such manner is identical to that of document classification. The possible edge features are "relation of anchors over the structure" and "relation of anchors in terms of contents". Among these, including "relation of anchors over the structure" features for Φ_{SCR} might not contribute classification performance, because relations such as enumeration and back and forth are considered in Cousin and Sibling cliques, and activated as edge features. A possible features of "relation of anchors in terms of contents" is that, for instance, shared categories between two articles. If the two articles share categories, these might tend to have the same named entity category.

In order to investigate whether shared category features contribute categorization accuracy, we experimented the two settings: we included shared category features for one, and the other is not. Both are experimented with CRFs (CR model). As a result, we can not achieved improvement of classification accuracy by including shared category features (0.7797 (incl.) and 0.7862 (not incl.) of F1). A possible reason is that since Wikipedia categories are also

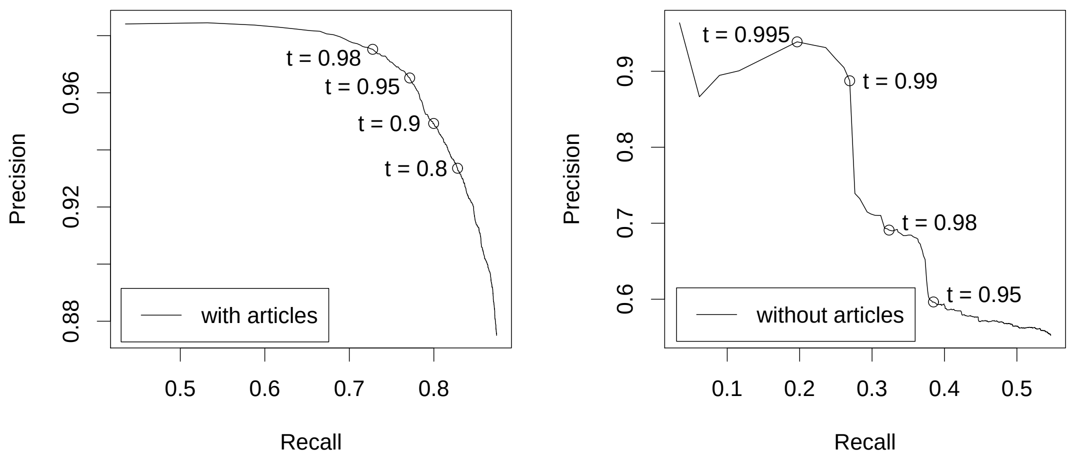


Figure 6.4. Precision-recall curve of the CR model. The left curve is drawn using anchors with articles, and the right figure is drawn using only anchors without articles.

used for node features and these are duplicated, the features did not contribute the performances.

Although it might be possible to improve categorization accuracy by finding informative features of a pair of anchors, in order to capture dependencies over list structures, it might enough to use naive label pair features.

Filtering NE Candidates using Marginal Probability

If we construct a named entity dictionary with the proposed model, it is desirable that the construction is performed with less human cost. For this reason, we investigate whether marginal probabilities $p(y_i|\mathbf{x})$ of CRFs can be exploited for filtering named entity candidates. Figure 6.4 shows the precision-recall curve obtained by thresholding the marginal probability of the MAP estimation in the CR models. For cases of anchors with articles, we achieved 97% of precision and 75% of recall with the threshold $p = 0.968$. In contrast, if we restrict the anchors without articles, since we have less clues for categorization, we obtained low recall compare to the case of anchors with articles.

4. Related Work

Wikipedia has become a popular resource for NLP. Bunescu and Pasca used Wikipedia for detecting and disambiguating NEs in open domain texts [6]. Strube and Ponzetto explored the use of Wikipedia for measuring Semantic Relatedness between two concepts [86], and for Coreference Resolution [74].

Several CRFs have been explored for information extraction from the web. Tang et al. proposed Tree-structured Conditional Random Fields (TCRFs) [96] that capture hierarchical structure of web documents. Zhu et al. proposed Hierarchical Conditional Random Fields (HCRFs) [115] for product information extraction from Web documents. TCRFs and HCRFs are similar to our approach described in section 4 in that the model structure is induced by page structure. However, the model structures of these models are different from our model.

There are statistical models that capture dependencies between examples. There are two types of classification approaches: iterative [56, 55] or collective [29, 97]. Lu et al. [55, 56] proposed link-based classification method based on logistic regression. This model iterates local classification until label assignments converge. The results vary from the ordering strategy of local classification. In contrast to iterative classification methods, collective classification methods directly estimate most likely assignments. Getoor et al. proposed Probabilistic Relational Models (PRMs) [29] which are built upon Bayesian Networks. Since Bayesian Networks are directed graphical models, PRMs cannot model directly the cases where instantiated graph contains cycles. Taskar et al. proposed Relational Markov Networks (RMNs) [97]. RMNs are the special case of Conditional Markov Networks (or Conditional Random Fields) in which graph structure and parameter tying are determined by SQL-like form.

As for the marginal probability to use as a confidence measure shown in Figure 6.4, Peng et al. [73] has applied linear-chain CRFs to Chinese word segmentation. It is calculated by constrained forward-backward algorithm [22], and confident segments are added to the dictionary in order to improve segmentation accuracy.

5. Summary

In this chapter, we proposed a method for categorizing NEs in Wikipedia. We defined three types of cliques that are constitute dependent anchor texts in construct CRFs graph structure, and introduced potential functions for them to reflect classification. The experimental results show that the effectiveness of capturing dependencies, and proposed CRFs model can achieve significant improvements compare to baseline methods with SVMs. The results also show that the dependency information from the HTML tree helps to categorize entities without gloss texts in Wikipedia. The marginal probability of MAP assignments can be used as confidence measure of the entity categorization. We can control the precision by filtering the confidence measure as PR curve in Figure 6.4.

Chapter 7

Exploiting Named Entity Information for Predicate-Argument Structure Analysis

1. Introduction

In this chapter we discuss use of named entity information for predicate-argument structure analysis. As we described previously, named entities refer to proper nouns (e.g. PERSON, LOCATION and ORGANIZATION), time expressions or numeric expressions. Since a large number of named entities exist, unknown expressions appear in texts. Their lexical information do not contribute the analysis, hence it is necessary to generalize them.

Also, use of named entity information for predicate-argument structure analysis can be seen as considering selectional preferences. Selectional Preferences (SPs) capture the fact that predicates prefer arguments in a certain semantic class. For example, a verb *drive* prefers human as *Agent*, and vehicle as *Theme*. This preferences can be incorporated into models by assigning named entity classes to nominal and named entities in sentences. Therefore, introducing classes of entities can be seen as capturing selectional preferences.

Although use of named entity information for constituent-based predicate-argument structure analysis has been explored [87, 76], there is no research of dependency-based work that focused on named entity information. One of our aim is to investigate what types of features in terms of named entities are effective for dependency-based predicate-argument structure analysis. In addition, we also investigate *global features* in terms of named entity tags of arguments, since this type of features has not been explored by previous work.

2. Named Entity Tags

In this chapter, we use the tagsets used in the CoNLL-2003 Shared Task (named entity recognition) [82] and the BBN Pronoun Coreference and Named Entity Type Corpus [107] to generalize named entities. We refer to them as CoNLL-2003 tagset and BBN tagset respectively.

In the CoNLL-2003 tagset, four types of tags are defined: PERSON (PER), LOCATION (LOC), ORGANIZATION (ORG), MISCELLANEOUS (MISC). The MISCELLANEOUS type include entities that do not belong to the previous three groups.

The BBN tagset is more fine-grained and widely covers entities. The tagset includes not only proper nouns but also nominal nouns. For instance, the word "researcher" is not covered by the CoNLL-2003 tagset, however, it is treated as an named entity of the type "PERSON_DESCRIPTOR". The types of named entities defined in the BBN corpus are Person, Facility, Organization, GPE, Location, Nationality, Product, Event, Work of Art, Law, Language and Contact-Info, and the types of nominal entities are Person, Facility, Organization, GPE, Product, Plant, Animal, Substance, Disease and Game. Also the BBN tagset defines seven numeric types: Date, Time, Percent, Money, Quantity, Ordinal and Cardinal.

If the named entity consists of two or more words, it is necessary to represent such chunk by the tags. We represent such chunks using IOB tag encoding. "B" of IOB denote the first word of the chunk. "I" means that the word is inside a chunk. A word with tag "O" is not part of a chunk. Table 7.1 shows an example annotation of both the CoNLL-2003 tagset and the BBN tagset.

Table 7.1. Examples of the CoNLL-2003 and the BBN Named Entity Tags

	POS	**CoNLL-2003**	**BBN**
Honda	NNP	B-ORG	B-E:ORGANIZATION:CORPORATION
Motor	NNP	I-ORG	I-E:ORGANIZATION:CORPORATION
Co.	NNP	I-ORG	I-E:ORGANIZATION:CORPORATION
′s	POS	O	O
sales	NNS	O	O
of	IN	O	O
domestically	RB	O	O
built	VBN	O	O
vehicles	NNS	O	B-E:PRODUCT_DESC:VEHICLE
plunged	VBD	O	O
21.7	CD	O	B-N:PERCENT
%	NN	O	I-N:PERCENT
from	IN	O	O
a	DT	O	B-T:DATE:DATE
year	NN	O	I-T:DATE:DATE
earlier	RBR	O	I-T:DATE:DATE
.	.	O	O

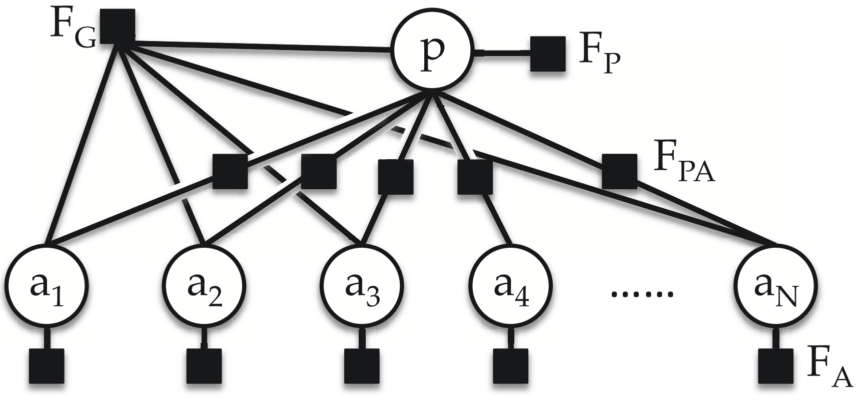


Figure 7.1. Undirected graphical model representation of the joint model.

3. Model

In order to investigate effects of generalizing named entities, we use the structured model described in Chapter 4 which identifies a predicate sense and its argument roles simultaneously. Figure 7.1 shows the structured models.

The black squares in Figure 7.1 are factors which provide scores of label assignments, and the score function for predicate-argument structures is defined by the sum of the factor scores.

$$s(p,\mathcal{A}) = \sum_{F_k \in \mathcal{F}} F_k(\mathbf{x}, p, \mathcal{A}) \tag{7.1}$$

$$= \mathbf{w} \cdot \Phi_P(\mathbf{x}, p) + \mathbf{w} \cdot \Phi_G(\mathbf{x}, p, \mathcal{A}) + \mathbf{w} \cdot \sum_{a \in \mathcal{A}} \{\Phi_A(\mathbf{x}, a) + \Phi_{PA}(\mathbf{x}, p, a)\}. \tag{7.2}$$

4. Additional Named Entity Features for the Model

In addition to the features described in Chapter 4, the following features are added to the factors F_{PA} and F_G of the structured model.

Named Entity Features for the Predicate-Argument Pairwise Factor F_{PA}

Named Entity Tags The named entity tag of the argument (e.g. *B-ORG*). This feature generalize the lexical information of the named entity.

Named Entity Tag-Dependency Path Conjunctions The conjunctions of named entity tag and dependency path (e.g. *B-ORG & $\overrightarrow{OBJ}$*). This feature captures the type of an argument and its syntactic position.

Named Entity Features for the Global Factor F_G

Named Entity Tag Sequences of Core Arguments The sequence of core argument labels and corresponding named entity tags (e.g. *A0:I-PER&PRED&A1:B-ORG*). This feature non-locally captures types in terms of all core arguments, and can be seen capturing non-local selectional preferences. The feature is not included the predicate sense information to avoid it being sparse.

5. Experiment

5.1 Experimental Settings

We use the CoNLL-2008 Shared Task data for experiments. The named entity tagsets used in this experiments are the CoNLL-2003 Shared Task tagset and the BBN Pronoun Coreference and Entity Type Corpus tagset. These tags are given in the CoNLL-2008 Shared Task Open Challenge data, and are annotated by a state-of-the-art sequential tagger proposed by Ciaramita and Altun [14]. The dependency parses used in this experiments are outputs of a history-based parser, Malt Parser, given in the open challenge data of the CoNLL-2008 Shared Task, and outputs of a higher-order projective dependency parser proposed by Carreras [8], and the gold parse trees.

One of the test data used for experiments are Wall Street Journal which a part of the Penn TreeBank, and it is the same domain with the training data (in-domain). The other data is taken from the Brown Corpus (out-of-domain).

For training and testing of the model, we used the all factors (F_P, F_A, F_{PA} and F_G). For learning of the joint model, the loss function $\rho(\mathbf{y}_t, \mathbf{y}')$ of the Passive-Aggressive Algorithm was set to the number of incorrect assignments of a predicate sense and argument roles. Also, the number of iterations of the model used for testing was selected based on the performance on the development data.

5.2 Distributions of Named Entity Tags in the Data

It is important to investigate how many arguments are tagged a particular named entity tag. Table 7.2 shows the coverages of the named entity tags of both CoNLL-2003 tagset and BBN tagset in the training data. The overall ratios of arguments tagged with a particular named entity tags are 8.33% on the CoNLL-2003 tagset and 22.71% on the BBN tagset. The difference is because the BBN tagset covers named entities as well as nominal entities (some tags of nominal entities have relatively high coverage: PER_DESC for 6.88% and ORG_DESC for 3.83%). Though the same types of named entity types such as PERSON and ORGANIZATION are included in the both tagsets, the ratios of the both are slightly different. A possible reason is that the named entity tags are automatically annotated by taggers, not the gold tags, the taggers might annotated NE labels inconsistently.

5.3 Results

Table 7.3 shows the performances of core argument role labeling. The results of the column $+NE_{PA}$ denote the F1 values of the results with NE features for F_{PA}, and the results of the column $+NE_{PA\&G}$ denote the F1 values of the results with NE features for both F_{PA} and F_G.

By introducing NE features for F_{PA}, we achieved slightly better performances compared to the results without NE features. Furthermore, adding the global feature for F_G provided performance improvements on both WSJ (+0.98) and on Brown (+1.74), especially the significant improvement is achieved for A0 in Brown (+3.30). The results suggest that the proposed global feature

Table 7.2. CoNLL-2003 and BBN named entity tag distribution of arguments in the training data

CoNLL NE Tag	Ratio	#
ORG	3.31%	13440
PER	3.31%	13438
LOC	1.00%	4066
MISC	0.71%	2877
Coverage	8.33%	33821/406169

BBN NE Tag	Ratio	#
PER_DESC	6.88%	27954
ORG	4.63%	18809
ORG_DESC	3.83%	15541
PERSON	2.67%	10845
MONEY	0.77%	3132
GPE	0.66%	2695
PERCENT	0.65%	2634
CARDINAL	0.42%	1734
DATE	0.39%	1594
NORP	0.39%	1572
...	...	...
Coverage	22.71%	92258/406169

is effective for predicate-argument structure analysis especially for core arguments of verbs.

6. Related Work

Use of named entity information for predicate-argument structure analysis has been explored by some of previous work, all of which are constituent-tree based approaches. Surdeanu et al. [87] reported that named entity information contributed performance improvements of the constituent-based predicate-argument structure analyser. In this work, seven types of named entities (PERSON, ORGANIZATION, LOCATION, MONEY, PERCENT, TIME and DATE) are used. These named entity tags are encoded by the two ways: use named entity tag of the content word (argument candidate), use 7 types of binary features which are activated when the phrase contains the corresponding named entities.

Pradhan et al. [76] followed the work of Surdeanu et al. [87]. They re-

	WSJ				Brown			
	N	w/o NE	$+NE_{PA}$	$+NE_{PA\&G}$	N	w/o NE	$+NE_{PA}$	$+NE_{PA\&G}$
VB*-A0	3509	92.15	92.44	93.04	551	87.23	87.77	90.53
VB*-A1	4844	92.83	93.14	93.86	669	84.30	84.37	85.22
VB*-A2	1085	85.61	86.71	86.83	145	69.18	69.62	68.34
VB*-A3	169	79.74	81.03	81.01	12	36.36	33.33	31.58
VB*-A4	99	88.78	88.33	88.33	15	47.62	75.00	60.87
VB*-A5	5	88.89	88.89	88.89	-	-	-	-
Total	9711	91.53	91.93	**92.51**	1392	83.09	84.15	**84.83**

Table 7.3. Impact of Named Entity features for argument role labeling of verb predicates (F1 of core argument labels)

ported that named entity information contributed a part of modifier argument roles such as location (AM-LOC) and temporal (AM-TMP), the performance improvements of them are 59 $\rightarrow$ 68 and 78.8 $\rightarrow$ 83.4 respectively. However, the overall performance was not significant compared to the result without named entity information.

Some of work have been used selectional preferences of nouns, not named entities. Zapirain et al. (2007) [110] exploited WordNet as a resource of selectional preferences for predicate-argument structure analysis. They extracted word categories from WordNet (e.g. food $\rightarrow$ nutrient) and use it as a feature for the Maximum Entropy Markov Model (MEMM) based classifier. In the experiments, the performances of some argument role label assignment (A3 of core arguments, and two modifier arguments of location (LOC) and temporal (TMP)) are improved, however, it has no significant difference compared to the results without selectional preferences features.

7. Summary

In this chapter, we proposed a new approach of introducing named entity information for predicate-argument structure analysis. More precisely, in addition to the named entity features used in the previous work, we incorporated the global feature that non-locally captures types in terms of all core arguments of the predicate, and can be seen capturing non-local selectional preferences. For training and testing, we used a structured model that can deal with global features. In the experiments, the proposed global feature contributed performance improvements especially for core arguments of verbs.

Chapter 8

Conclusion

This chapter summarizes this dissertation and gives future directions we intend to explore.

1. Summary

This dissertation has explored dependency-based predicate-argument structure analysis.

In Chapter 4, we proposed a structured model that captures dependencies underlying in predicate-argument structures – non-local dependencies between arguments and inter-dependencies between a predicate and its arguments. In order to capture both types of dependencies simultaneously, we introduced four types of factors including pairwise factor that captures local dependencies between a predicate and one of its arguments, and global factor that captures non-local dependencies between arguments. We also described an inference algorithm and a large-margin learning algorithm which handle both local features and global features. We conducted experiments on analyzing predicate-argument structures in multiple languages, and the model achieved performance improvements on both predicate sense disambiguation and argument role labeling, and competitive results compared to the state-of-the art systems which use time-consuming feature selection algorithms.

Using the structured model described in Chapter 4, we constructed a syntactic and semantic dependency parser in Chapter 5. The system consists of three parts: a higher-order projective dependency parser, a predicate identifier and the structured model. The system achieved the competitive result in terms of predicate identification and sense disambiguation, and the best performance compared to the systems which use no feature selection algorithms.

From Chapter 6 to 7, this dissertation had focused on named entities – acquiring named entity information from the Web and exploiting it for predicate argument structure analysis. We selected Wikipedia which is an encyclopedia on the Web for acquiring named entities, and proposed CRF-based structured models that captures characteristics of list structures in articles. The experimental results showed that capturing characteristics of list structures improves classification accuracy, and marginal probabilities provided by CRFs can be used as a confidence measure for filtering named entity candidates.

Finally in Chapter 7, we discussed use of named entity information for predicate argument structure analysis. In addition to the features used in the previous work, we explored use of the global feature that non-locally captures types in terms of all core arguments is introduced to the model. In the experiments, global named entity information of predicate-argument structures contributed performance improvements especially for core arguments of verbs.

2. Future Directions

2.1 Incorporating Lexical Knowledge

We used named entity information as a knowledge for predicate-argument structure analysis in Chapter 7. However, since arguments are not only named entities but also nouns, it is necessary to incorporate various other information to deal with lexical sparseness of other words except for named entities. In order to treat lexical sparseness of nouns, the work of Zapirain et al. [110] used WordNet categories as additional features for predicate-argument structure analysis. However, they reported that WordNet features did not provide significant improvements compared to the results without the features. Also, one of their successive work [111] used selectional preferences for predicate-argument structure analysis. The work explored various selectional preference measures, and these preferences contributed improvements over the model with only lexical information. However, they used these preference models directly for labeling argument roles, hence how to incorporate these preferences to machine learning-based predicate-argument structure models is still an open problem.

2.2 Incorporating Unlabeled Data

Predicate-argument structure analysis suffers from data sparseness. Although the structured model proposed in Chapter 4 is a supervised model which only uses labeled data, it is expensive to prepare sufficient labeled data for each predicate. In recent years, a number of work have been proposed methods for incorporating unlabeled data (semi-supervised learning), and these methods have been applied for word sense disambiguation [1], sequential labeling tasks such as syntactic chunking, POS tagging, and named entity recognition [3, 2, 93, 94], and dependency parsing [47, 95].

Lately, some work have been explored semi-supervised methods for predicate-argument structure analysis. Fürstenau et al. [27] proposed a method for semi-supervised predicate-argument structure analysis where training samples are expanded from seed training samples using similarity measures of predicate-argument structures. They reported performance improvements by expanding training data with the method, however, they used a very small samples for training and the resulting model has poor performances, hence the resulting model can not be used for real applications. Deschacht and Moens [25] proposed latent variable models for predicate-argument structure analysis. They used word distributions that can be obtained from the models as new features for supervised discriminative models. The approach achieved significant improvements compared to an approach without information of unlabeled data, however, this type of incorporation of information in terms of unlabeled data is the most naive approach, and as Suzuki et al. [94, 95], it might be possible to combine models trained on labeled and unlabeled data for accurate predicate-argument structure analysis. Therefore, there is still room for further improvements of predicate-argument structure analysis using unlabeled data.

References

[1] Rie Kubota Ando. Applying alternating strcture optimization to word sense disambiguation. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, 2006.

[2] Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, 2005.

[3] Rie Kubota Ando and Tong Zhang. A high-performance semi-supervised learning method for text chunking. In *Proceedings of the 43rd Annual Meeting of the ACL (ACL-2005)*, 2005.

[4] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, 1998.

[5] Anders Björkelund, Love Hafdell, and Pierre Nugues. Multilingual semantic role labeling. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, 2009.

[6] Razvan Bunescu and Marius Pasca. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of EACL*, 2006.

[7] Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC-2006)*, Genoa, Italy, 2006.

[8] Xavier Carreras. Experiments with a higher-order projective dependency parser. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007.

[9] Xavier Carreras and Lluís Màrquez. Introduction to the conll-2004 shared task: Semantic role labeling. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004)*, 2004.

[10] Xavier Carreras and Lluís Màrquez. Introduction to the conll-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, 2005.

[11] Wanxiang Che, Zhenghua Li, Yuxuan Hu, Yongqiang Li, Bing Qin, Ting Liu, and Sheng Li. A cascaded syntactic and semantic dependency parsing system. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL-2008)*, 2008.

[12] John Chen and Owen Rambow. Use of deep linguistic features for the recognition and labeling of semantic arguments. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP-2003)*, 2003.

[13] Stanley F. Chen and Ronald Rosenfeld. A gaussian prior for smoothing maximum entropy models. Technical report, Carnegie Mellon University, 1999.

[14] Massimiliano Ciaramita and Yasemin Altun. Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP-2006)*, 2006.

[15] Massimiliano Ciaramita, Giuseppe Attardi, Felice Dell'Orletta, and Mihai Surdeanu. Desrl: A linear-time semantic role labeling system. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL-2008)*, 2008.

[16] Trevor Cohn and Philip Blunsom. Semantic role labelling with tree conditional random fields. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, 2005.

[17] Michael Collins. Discriminative training methods for hidden markov models. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, 2002.

[18] Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-2002)*, 2002.

[19] Michael Collins, Amir Globerson, Terry Koo, Xavier Carreras, and Peter L. Barlett. Exponentiated gradient algorithms for conditional random fields and max-margin markov networks. *The Journal of Machine Learning Research*, 9:1775.

[20] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, 2006.

[21] Koby Crammer and Yoram Singer. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3:951.

[22] Aron Culotta and Andrew McCallum. Confidence estimation for information extraction. In *Proceedings of HLT-NAACL*, 2004.

[23] Hal Daumé III. *Practical Structured Learning Techniques for Natural Language Processing*. PhD thesis, University of Southern California, Los Angeles, CA, August 2006.

[24] Hal Daumé III and Daniel Marcu. Learning as search optimization: Approximate large margin methods for structured prediction. In *Proceedings of the Twenty-Second International Conference on Machine Learning (ICML-2005)*, 2005.

[25] Koen Deschacht and Marie-Francine Moens. Semi-supervised semantic role labeling using the latent words language model. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP-2009)*, 2009.

[26] Jason Eisner. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, 1996.

[27] Hagen Fürstenau and Mirella Lapata. Graph alignment for semi-supervised semantic role labeling. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP-2009)*, 2009.

[28] Andrea Gesmundo, James Henderson, Paola Merlo, and Ivan Titov. A latent variable model of synchronous syntactic-semantic parsing for multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, 2009.

[29] Lise Getoor, Eran Segal, Ben Taskar, and Daphne Koller. Probabilistic models of text and link structure for hypertext classification. In *IJCAI Workshop on Text Learning: Beyond Supervision*, 2001.

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

[31] Daniel Gildea and Martha Palmer. The necessity of syntactic parsing for predicate argument recognition. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, 2002.

[32] Amir Globerson, Terry Koo, Xavier Carreras, and Michael Collins. Exponentiated gradient algorithms for log-linear structured prediction. In *Proceedings of ICML*, 2007.

[33] Kadri Hacioglu. A lightweight semantic chunking model based on tagging. In *Proceedings of the 2004 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2004)*, 2004.

[34] Kadri Hacioglu. Semantic role labeling using dependency trees. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, 2004.

[35] Kadri Hacioglu, Sammer Pradhan, Wayne Ward, James H. Martin, and Daniel Jurafsky. Semantic role labeling by tagging syntactic chunks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004)*, 2004.

[36] Kadri Hacioglu and Wayne Ward. Target word detection and semantic role chunking using support vector machines. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2003)*, 2003.

[37] Aria Haghighi, Kristina Toutanova, and Christopher D. Manning. A joint model for semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, 2005.

[38] Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, Boulder, Colorado, USA, 2009.

[39] Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeněk Žabokrtský. Prague Dependency Treebank 2.0, 2006.

[40] James Henderson, Paola Merlo, Gabriele Musillo, and Ivan Titov. A latent variable model of synchronous parsing for syntactic and semantic dependencies. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL-2008)*, 2008.

[41] Ryu Iida, Mamoru Komachi, Kentaro Inui, and Yuji Matsumoto. Annotating a japanese text corpus with predicate-argument and coreference relations. In *ACL Workshop 'Linguistic Annotation Workshop'*, 2007.

[42] Richard Johansson. *Dependency-based Semantic Analysis of Natural-language Text*. PhD thesis, Lund University, 2008.

[43] Richard Johansson and Pierre Nugues. Extended constituent-to-dependency conversion for english. In *Proceedings of the 16th Nordic Conference of Computational Linguistics NODALIDA-2007*, 2007.

[44] Richard Johansson and Pierre Nugues. Dependency-based syntactic-semantic analysis with propbank and nombank. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL-2008)*, 2008.

[45] Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. Construction of a Japanese relevance-tagged corpus. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC-2002)*, pages 2008–2013, Las Palmas, Canary Islands, 2002.

[46] Jun'Ichi Kazama and Kentaro Torisawa. A new perceptron algorithm for sequence labeling with non-local features. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007.

[47] Terry Koo, Xavier Carreras, and Michael Collins. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, 2008.

[48] Werner Krauth and Marc Mèzard. Learning algorithms with optimal stability in neural networks. *Journal of Physics*, 20:745.

[49] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML-2001)*, 2001.

[50] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, 2001.

[51] Peter L.Berlett, Michael Collins, Ben Taskar, and David McAllester. Exponentiated gradient algorithms for large-margin structured classification. In *Proceedings of the 2004 Conference on Advances in Neural Information Processing Systems (NIPS)*, 2004.

[52] Dong C. Liu and Jorge Nocedal. The limited memory bfgs methods for large scale optimization. In *Mathematical Programming 45*, 1989.

[53] Xavier Lluís, Stefan Bott, and Lluís Màrquez. A second-order joint eisner model for syntactic and semantic dependency parsing. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, 2009.

[54] Xavier Lluís and Lluís Màrquez. A joint model for parsing syntactic and semantic dependencies. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL-2008)*, 2008.

[55] Qing Lu and Lise Getoor. Link-based classification using labeled and unlabeled data. In *Proceedings of ICML*, 2003.

[56] Qing Lu and Lise Getoor. Link-based text classification. In *Proceedings of IJCAI*, 2003.

[57] Andre Martins, Noah Smith, and Eric Xing. Concise integer linear programming formulations for dependency parsing. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing (ACL-IJCNLP-2009)*, 2009.

[58] Andrew McCallum, Khashayar Rohanimanesh, and Charles Sutton. Dynamic conditional random fields for jointly labeling multiple sequences. In *NIPS Workshop on Syntax, Semantics, and Statistics*, December 2003.

[59] Ryan McDonald. *Discriminative Learning and Spanning Tree Algorithms for Dependency Parsing*. PhD thesis, University of Pennsylvania, 2006.

[60] Ryan McDonald, Kerry Hannan, Tyler Neylon, Mike Wells, and Jeff Reynar. Structured models for fine-to-coarse sentiment analysis. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL-2007)*, 2007.

[61] Ryan McDonald, Fernand Pereira, Kiril Ribarov, and Jan Hajič. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT-EMNLP 2005)*, 2005.

[62] Ryan McDonald and Fernando Pereira. Online learning of approximate dependency parsing algorithms. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2006)*, 2006.

[63] Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. The nombank project: An interim report. In *HLT-NAACL 2004 Workshop on Frontiers in Corpus Annotation*, 2004.

[64] Ivan Meza-Ruiz and Sebastian Riedel. 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 (HLT/NAACL-2009)*, 2009.

[65] Ivan Meza-Ruiz and Sebastian Riedel. Multilingual semantic role labelling with markov logic. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, 2009.

[66] Kevin Murphy, Yair Weiss, and Michael Jordan. Loopy belief propagation for approximate inference. In *Proceedings of UAI*, 1999.

[67] Joakim Nivre, Johan Hall, Jens Nilsson, Gulsen Eryigit, and Svetoslav Marinov. Pseudo-projective dependency parsing with support vector machines. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL-X)*, 2006.

[68] Joakim Nivre and Jens Nilsson. Pseudo-projective dependency parsing. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005)*, 2005.

[69] ALBERT B. J. Novikoff. On convergence proofs on perceptrons. In *Proceedings of the Symposium on the Mathematical Theory of Automata*, volume 12, page 615.

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

[71] Martha Palmer and Nianwen Xue. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172, 2009.

[72] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.

[73] Fuchun Peng, Fangfang Feng, and Andrew McCallum. Chinese segmentation and new word detection using conditional random fields. In *Proceedings of COLING*, 2004.

[74] Simone Paolo Ponzetto and Michael Strube. Exploiting semantic role labeling, wordnet and wikipedia for coreference resolution. In *Proceedings of HLT-NAACL*, 2006.

[75] Sameer Pradhan, Kadri Hacioglu, Wayne Ward, James H. Martin, and Daniel Jurafsky. Semantic role chunking combining complementary syntactic views. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, 2005.

[76] Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James H. Martin, and Dan Jurafsky. Shallow semantic parsing using support vector machines. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2004)*, 2004.

[77] Matt Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62:107–136, 2006.

[78] Sebastian Riedel. Improving the accuracy and efficiency of map inference for markov logic. In *Proceedings of the Annual Conference on Uncertainty in AI (UAI-2008)*, 2008.

[79] Sebastian Riedel and Ivan Meza-Ruiz. Collective semantic role labelling with markov logic. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL-2008)*, 2008.

[80] Frank Rosenblatt. *Principles of Neurodynamics: Perceptron and Theory of Brain Mechanisms*. Spartan-Books, 1962.

[81] Dan Roth and Wen tau Yih. Integer linear programming inference for conditional random fields. In *Proceedings of the 22nd International Conference on Machine Learning*, 2005.

[82] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning (CoNLL-2003)*, 2003.

[83] Satoshi Sekine, Kiyoshi Sudo, and Chikashi Nobata. Extended named entity hierarchy. In *Proceedings of LREC*, 2002.

[84] Dan Shen and Mirella Lapata. Using semantic roles to improve question answering. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007.

[85] Parag Singla and Pedro Domingos. Discriminative training of markov logic networks. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*, 2005.

[86] Michael Strube and Simone Paolo Ponzetto. Wikirelate! computing semantic relatedness using wikipedia. In *Proceedings of AAAI*, 2006.

[87] Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. Using predicate-argument structures for information extraction. In *Pro-*

ceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-2003), 2003.

[88] Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL-2008)*, 2008.

[89] Mihai Surdeanu, Lluis Marques, Xavier Carreras, and Pere R. Comas. Combination strategies for semantic role labeling. *Journal of Artificial Intelligence Research*, 29, 2007.

[90] Charles Sutton. GRMM: A graphical models toolkit. http://mallet.cs.umass.edu, 2006.

[91] Charles Sutton and Andrew McCallum. An introduction to conditional random fields for relational learning. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2007.

[92] Charles Sutton, Khashayar Rohanimanesh, and Andrew McCallum. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. In *Proceedings of ICML*, 2004.

[93] Jun Suzuki, Akinori Fujino, and Hideki Isozaki. Semi-supervised structured output learning based on a hybrid generative and discriminative approach. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007.

[94] Jun Suzuki and Hideki Isozaki. Smi-supervised sequential labeling and segmentation using giga-word scale unlabeled data. In *Proceedings of ACL-08: HLT*, 2008.

[95] Jun Suzuki, Hideki Isozaki, Xavier Carreras, and Michael Collins. An empirical study of semi-supervised structured conditional models for dependency parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP-2009)*, 2009.

[96] Jie Tang, Mingcai Hong, Juanzi Li, and Bangyong Liang. Tree-structured conditional random fields for semantic annotation. In *Proceedings of ISWC*, 2006.

[97] Ben Taskar, Pieter Abbeel, and Daphne Koller. Discriminative probabilistic models for relational data. In *Proceedings of UAI*, 2002.

[98] Ben Taskar, Carlos Guestrin, and Daphne Koller. Max-margin markov networks. In *Proceedings of the 2003 Conference on Advances in Neural Information Processing Systems (NIPS)*, 2003.

[99] Mariona Taulé, Maria Antònia Martí, and Marta Recasens. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC-2008)*, Marrakesh, Morroco, 2008.

[100] Ivan Titov, James Henderson, Paola Merlo, and Gabriele Musillo. Online graph planarisation for synchronous parsing of semantic and syntactic dependencies. In *Proceedings of the Twenty-first International Joint Conference on Artificial Intelligence (IJCAI-09)*, 2009.

[101] Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. Joint learning improves semantic role labeling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005)*, 2005.

[102] Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. A global joint model for semantic role labeling. *Computational Linguistics*, 34(2), 2008.

[103] Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the 21th International Conference on Machine Learning (ICML-2004)*, 2004.

[104] Vladimir Vapnik. *Statistical Learning Theory*. Wiley Interscience, 1998.

[105] Martin Wainwright, Tommi Jaakkola, and Alan Willsky. Tree-based reparameterization framework for analysis of sum-product and related algorithms. *IEEE Transactions on Information Theory*, 45(9):1120–1146, 2003.

[106] Rui Wang and Yi Zhang. Recognizing textual relatedness with predicate-argument structures. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP-2009)*, 2009.

[107] Ralph Weischedel and Ada Brunstein. BBN pronoun coreference and entity type corpus, 2005.

[108] Dekai Wu and Pascale Fung. Can semantic role labeling improve SMT? In *Proceedings of the 13th Annual Conference of the European Association for Machine Translation (EAMT-2009)*, 2009.

[109] Hiroyasu Yamada and Yuji Matsumoto. Statistical dependency analysis with support vector machines. In *Proceedings of the Eighth International Workshop on Parsing Technology (IWPT-2003)*, 2003.

[110] Benat Zapirain, Eneko Agirre, and Lluís Márquez. UBC-UPC: Sequential srl using selectional preferences. an approach with maximum entropy markov models. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007)*, 2007.

[111] Benat Zapirain, Eneko Agirre, and Lluís Márquez. Generalizing over lexical features: Selectional preferences for semantic role classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing (ACL-IJCNLP-2009)*, 2009.

[112] Hai Zhao, Wenliang Chen, and Chunyu Kit. Semantic dependency parsing of nombank and propbank: An efficient integrated approach via a large-scale feature selection. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP-2009)*, 2009.

[113] Hai Zhao, Wenliang Chen, Chunyu Kit, and Guodong Zhou. Multilingual dependency learning: A huge feature engineering method to se-

mantic dependency parsing. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, 2009.

[114] Hai Zhao and Chunyu Kit. Parsing syntactic and semantic dependencies with two single-stage maximum entropy models. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL-2008)*, 2008.

[115] Jun Zhu, Zaiqing Nie, Ji-Rong Wen, Bo Zhang, and Wei-Ying Ma. Simultaneous record detection and attribute labeling in web data extraction. In *Proceedings of ACM SIGKDD*, 2006.

[116] Jun Zhu, Eric P. Xing, and Bo Zhang. Priman sparse max-margin markov networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009.

List of Publications

Journal Papers

1. Yotaro Watanabe, Masayuki Asahara and Yuji Matsumoto. "A Structured Prediction Model for Joint Learning of Predicate Senses and Argument Roles". Transactions of the Japanese Society for Artificial Intelligence, Vol. 25, No.2, pp.252-261, January 2010. (in Japanese)

2. Yotaro Watanabe, Masayuki Asahara and Yuji Matsumoto. "Graph-structured Conditional Random Fields for Named Entity Categorization in Wikipedia". Transactions of the Japanese Society for Artificial Intelligence, Vol. 23, No. 4, pp.245-254, April 2008. (in Japanese)

International Conference/Workshop Papers

1. Yotaro Watanabe, Masayuki Asahara and Yuji Matsumoto. "Multilingual Syntactic-Semantic Dependency Parsing with Three-Stage Approximate Max-Margin Linear Models". In Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009), pp. 114-119, Boulder, Colorado, June 2009.

2. Yotaro Watanabe, Masakazu Iwatate, Masayuki Asahara and Yuji Matsumoto. "A Pipeline Approach for Syntactic and Semantic Dependency Parsing". In Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL-2008), pp. 228-232, Manchester, August 2008.

3. Yotaro Watanabe, Masayuki Asahara and Yuji Matsumoto. "A Graph-based Approach to Named Entity Categorization in Wikipedia Using Conditional Random Fields". In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007), pp. 649-657, Prague, June 2007.

Other Publications

1. Yotaro Watanabe, Masayuki Asahara and Yuji Matsumoto. “Multilingual Syntactic-Semantic Dependency Parsing Using Online Large-Margin Learning Algorithms”. In Information Processing Society of Japanese SIG Notes, NL-192, No.2, pp.1-8, 2009 (in Japanese).

2. Yotaro Watanabe, Masayuki Asahara and Yuji Matsumoto. “Named Entity Categorization in Wikipedia Using Conditional Random Fields on DOM Structure”. In the 21st Annual Conference of the Japanese Society for Artificial Intelligence, 2G4-5, 2007 (in Japanese).

3. Yotaro Watanabe, Masayuki Asahara and Yuji Matsumoto. “Named Entity Categorization Using Conditional Random Fields on HTML Tree Structure: Semi-Automatic Thesaurus Construction from Wikipedia”. In Information Processing Society of Japanese SIG Notes, NL-179, pp.73-78, 2007 (in Japanese).

4. Masayuki Asahara, Kenta Fukuoka, Ai Azuma, Chooi-Ling Goh, Yotaro Watanabe, Yuji Matsumoto and Takashi Tsuzuki. “Combination of Machine Learning Methods for Optimum Chinese Word Segmentation”. 4th SIGHAN Workshop Bakeoff (IJCNLP 2005).