NAIST-IS-DD1061033

Doctoral Dissertation

Syntactic Dependency Structure-based Approaches for Chinese Semantic Role Labeling

Yanyan Luo

September 4, 2013

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 SCIENCE

Yanyan Luo

Thesis Committee:

Professor Yuji Matsumoto	(Supervisor)
Professor Satoshi Nakamura	(Co-supervisor)
Associate Professor Masashi Shimbo	(Co-supervisor)
Assistant Professor Kevin Duh	(Co-supervisor)

Syntactic Dependency Structure-based Approaches for Chinese Semantic Role Labeling[*]

Yanyan Luo

Abstract

Semantic roles, logical relations such as *Agent*, *Theme*, etc. that hold between predicates and their participants and circumstances, need to be determined automatically in a wide range of natural language processing applications. This process is referred to as *Semantic Role Labeling*. This dissertation describes how to construct statistical models for Chinese semantic role labeling, and explores what is helpful for it.

Recently, dependency parsing based methods have achieved much success in semantic role labeling. However, due to errors in dependency parsing, there remains a large performance gap between semantic role labeling based on oracle parses and semantic role labeling based on automatic parses in practice. In light of this, this work devotes considerable effort to investigating the statistical models for semantic role labeling and what additional features are necessary to close this gap.

Semantic role labeling is often divided into three subtasks: predicate disambiguation, argument identification and argument classification. In this thesis, we propose a dual decomposition algorithm to alleviate the error propagation between argument identification subtask and argument classification subtask. In the experiments, we achieved competitive results compared to the state-of-the-art systems.

Apart from the statistical models for semantic role labeling, we investigate two kinds of additional features: additional dependency information in the form of N-best parse features and orthogonal non-dependency information. We compare the above features in a semantic role labeling system that achieves state-of-the-art results on the corpus. In the experiments, we achieved top result to our best knowledge.

[*]Doctoral Dissertation, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD1061033, September 4, 2013.

Keywords:

predicate argument structure, semantic role labeling, dual decomposition, deep/shallow syntactic information

統語的依存構造を用いた中国語述語項構造解析[*]

羅彦彦

内容梗概

動作主や対象のような述語と項の間の意味役割を同定する "述語項構造解析 "は様々な自然言語処理応用で利用されている。本論文ではどのように中国語述語項構造解析のための統計的機械学習モデルをどのように構築するか、またモデルにどのような情報が有効かを明らかにする。

近年、述語項構造解析器を構成する際に統語的依存構造解析結果を用いる手法が成功を収めている。しかしながら、この手法において述語項構造解析器の性能は依存構造解析器の誤りの影響を大きく受ける。例えば、述語項構造解析器に正しい依存構造解析結果を与えたときと依存構造解析器の出力を与えたときとでは述語項構造解析結果の性能に大きなギャップがある。このことから、本研究は述語項構造の統計的機械学習モデルを調査するとともに、このギャップを埋めるために必要なさらなる素性を調査する。

述語項構造解析は三つの部分問題に分割して解くことが多い。一つ目は述語の語義の曖昧性解消で、語義により異なる取りうる項の数と種類を決定する。二つ目は項同定で、述語の項になりうる名詞句の同定を行う。三つ目は項の意味役割分類で、同定された項に対し動作主や対象といった意味役割を識別する。この重畳的な問題設定に対して、本論文では双対分解法を用いた述語項構造解析手法を提案する。提案手法では項同定と項の意味役割分類間の誤り伝搬を緩和することが確認され、評価実験において先行研究の最新の手法と遜色ない結果を達成した。

解析手法と独立して、述語項構造解析器の性能向上に寄与する二つの素性について調査した。一つは依存構造解析器の N-best 解析結果を用いる手法で、もう一つは高性能の浅い統語解析器（チャンカー）の出力を用いる手法である。評価実験においてこれら二つの素性が有効であることを示した。さらに上記解析手法と組み合わせて用いることで現在のところ中国語述語項構造解析において最高の性能を達成した。

[*]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-DD1061033, 2013 年 9 月 4 日.

キーワード

述語項構造、意味役割付与、双対分解、統語構造素性

Acknowledgments

First and foremost I would like to thank my advisor and mentor Professor Yuji Matsumoto. He has always been very kind to me since I came to Japan. As a supervisor, he always gives me insightful advices that lead me to the right way of my research and life. It would be impossible for me to start my life and research in Naist smoothly without his endless support and help. I feel highly honored and proud that I can be one of his students.

I am also grateful to Associate Professor Masayuki Asahara for his kind help and supervision on my research. Every time when I come across with problems in my research, he always helped me with my problems patiently. And I am also indebted to him for pointing out important details to improve my research papers and final dissertation report. It is with his sincerely help that I have the chance to finish my dissertation and write this acknowledgements.

I also acknowledge members of my Doctoral advisory committee for the great patience on reading this thesis and give invaluable feedback.

I would also like to thank Assistant Professor Kevin Duh for his insightful advices and deep discussions about improving my work. Thanks also to the staff in International Student Affairs. Thanks to their kind help and support, I learned a lot about the culture of Japan and my study life in Japan become very colorful and meaningful. I would also like to thank Ms. Yuko Kitagawa for her help in my private life. She taught me a lot about Japan and helped me to adjust to life in Japan.

I also want to thank China Scholarship Council (CSC) for providing me financial support to complete my Ph.D course.

Finally I would like to thank my parents for always trusting me and supporting me to study aboard. With their enormous amount of support and inspiration, I have the strength to overcome all the difficulties I had met. I dedicate this thesis to them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Semantic Role Labeling (SRL), the task of identifying arguments for a predicate and assigning semantically meaningful labels to them, has been widely used by many applications, like machine translation [26], question answering [34, 42], information retrieval [41, 47] and text categorization [36]. A semantic role represents semantic relations that hold between a predicate and its one of the arguments. While the task of full natural language understanding is poorly understood in general, semantic role structures constitute a tractable fragment of the full spectrum of human semantic processing.

It is generally agreed that the problem of SRL is closely tied to syntactic analysis. Most previous implementations of semantic role labelers have used constituents of a phrase structure parsed tree as the syntactic input. However, constituent parsing acts as a central bottleneck in some languages, like Chinese, that limits the improvement of SRL. Recently, dependency parsing-based SRL has received much attention. Dependency parsing views the syntactic structure of a sentence as a graph of labeled word-to-word relations. It became very common especially after the challenges of the shared tasks held in the CoNLL-2008 and 2009. Our work focuses on syntactic dependency structure-based SRL since dependency representations make the syntactic-semantic interface conceptually simpler and more intuitive, and dependency parsing does not need complicated linguistic knowledge as constituent parsing.

Although dependency parsing has better results than constituent parsing, the performance is still not ideal and the performance of SRL fluctuates according to performance of dependency parsing. For example, in our experiments, the SRL performance achieves an F-measure of 87.47%, when we use the gold parse trees. This performance drops to 77.34% when a real parser (MALT) is used instead. It is worthwhile to investigate what statistical models for SRL and what additional information is necessary to improve the SRL performance based on automatic parses in practice and close the gap between it and SRL based on oracle parses.

1.2 Contributions of This Research

Our principal contributions through this research work are summarized in the following.

Based on a SRL model that can jointly label the predicate senses and their corresponding arguments in Chapter 3, we propose a dual decomposition algorithm to alleviate the error propagation between argument identification subtask and argument classification subtask by benefitting the argument identification subtask greatly.

The goal is to make use of the possible dependencies between different argument identification models without greatly increasing the complexity of inference. The dual decomposition inference approach allows us to explore sub-models for solving optimization problems. In particular, we can iteratively apply exact inferences to the sub-models, adjusting their potentials to the reflect the constraints of the full problem. The technique of dual decomposition has recently been shown to yield state-of-the-art performance in dependency parsing [25], text alignment [11], biomedical event extraction [41] and so on. However, to our best knowledge, it has not been applied to SRL.

Furthermore, two additional features: base chunk constituents related and N-best parse related features, are explored to our syntactic dependency parsing based SRL respectively and increasingly in Chapter 6 and Chapter 7. To date, almost all the researches for features that will be helpful for SRL focused on given syntactic dependency results. Our work shows that we are not limited to increasing SRL performance via depending too much on syntactic parsing, but that we can explore other information to improve SRL.

1.3 Dissertation Outlines

The rest of dissertation is organized as follows:

Chapter 2: SRL Literature Review This chapter formally defines the concepts of semantic roles and predicate argument structures, gives an overview of semantic-role linguistic resources and reviews approaches of previous work.

Chapter 3: Syntactic Dependency Based Predicate Argument Structure Analysis In this chapter, we construct Chinese SRL to describe the formal definitions for predicate argument structure analysis using dependency syntactic structures as input and compare several different methods of predicate argument structure analysis for SRL. Based on this model, we propose other approaches and explore more new features to improve Chinese predicate argument structure analysis results.

Chapter 4: Robust Integrated Models for SRL First, we describe the dual decomposition inference approach which forms the basis of our approach in this chapter. Based on this theory, we propose a dual decomposition algorithm for integrating two argument identification models for SRL. Both argument identification models are nearly the same, except for the label tag sets and the feature sets used. And each argument identification model together implements the predicate sense disambiguation simultaneously. The integrated argument identification result is highly improved, therefore, the error propagation from argument identification to argument classification is greatly decreased.

Chapter 5: Exploring Shallow Parsing Information for SRL In this chapter, we compare the syntactic dependency parsing based SRL and shallow parsing based SRL, and investigate the effect of shallow parsing information for syntactic dependency parsing based SRL.

Chapter 6: Exploring Additional Dependency Information for SRL This chapter investigates effects of additional dependency features from N-best parsing results for SRL. We also combine the features in Chapter 5 to evaluate the performance of SRL and investigate what information is helpful for syntactic dependency based semantic role labeling.

Chapter 7: Conclusions This chapter summarizes the dissertation by summarizing our main contributions and describes our possible future directions.

Chapter 2

SRL Literature Review

Semantic roles are the equivalence classes which slot connectors associated with different predicates can be meaningfully grouped into. And they are one of the oldest classes of constructs in linguistic theory. They are used to indicate the role played by each entity in a sentence and are ranging from very specific to very general. The entities that are labelled should have participated in an event. Some of the domain-specific roles like from airport, to airport, and depart time. Some of the verb-specific roles like eater and eaten for the verb eat. Although there is no consensus on a definitive list of semantic roles, some basic semantic roles such as agent, instrument, etc are followed by all.

Since the work of Gildea and Jurafsky [16], statistical and machine learning approaches have been the predominant research paradigm in SRL, like most of the subfields in natural language processing and computational linguistics. A prerequisite for statistical and machine learning approaches to semantic role labeling is the availability of a significant amount of semantically interpreted corpora from which automatic systems can learn.

2.1 Semantic Role Corpora

PropBank and FrameNet are the two most widely used corpora in developing automatic SRL systems. Although both corpora provide semantic roles annotations, they use very different semantic role labels. The following example taken from Xue [55] shows the same sentences with different labels in both corpora respectively.

FrameNet

[**Buyer** We] always [**LU** bought] [**Goods** a few dark-red carnations] [**Seller** from her.]
During the later part of the nineteenth century, [**Seller** the landowners] [**LU** sold] [**Goods** the land] [**Buyer** to developers] in very small lots.

PropBank

[**Arg0** We] always [**Rel** bought] [**Arg1** a few dark-red carnations] [**Arg2** from her.]
During the later part of the nineteenth century, [**Arg0** the landowners] [**Rel** sold] [**Arg1** the land] [**Arg2** to developers] in very small lots.

Both corpora add a layer of semantic role annotation on top of the Penn Treebank of constituent syntactic annotation [55]. However, In FrameNet, the semantic roles of a predicate (called a Lexical Unit (LU)) are organized by semantic frames, which are conceptual structures that describe a particular situation or event along with their participants. For example, the *sell* and *buy* both belong to the semantic frame, which involves *Buyer* and *Seller* exchanging *Money* and *Goods*. For the examples from FrameNet, the seller always labeled *Seller* and the buyer always labeled *Buyer*. Unlike in FrameNet, there is no reference ontology like the semantic frame that provides a general set of semantic roles. In PropBank, the allowed core arguments for every predicate are listed. For the core arguments, they are represented by an integer prefixed by *Arg* and the integer is not more than 5. Although these labels are not meaningful across different predicates. A general rough convention is that *Arg0* corresponds to the argument having most properties of a ”proto-agent” and *Arg1* to a ”proto-patient” [13]. For the adjunct-like arguments, each is represented as *ArgM*, followed by a secondary tag indicating the type of the modifier, such as ArgM-DIR(direction), ArgM-TMP(temporal). The predicate-specific nature of the PropBank semantic roles is clear when compared with the roles in FrameNet. The buyers shown in examples from PropBank are labeled as *Arg0* and *Arg2* respectively.

It generally seems that SRL is easier with PropBank than with FrameNet because of the much larger annotation coverage in PropBank. Statistical systems that carry out a PropBank-style semantic analysis typically treat role label assignment as a well-defined classification problem and PropBank based semantic role labelers have been implemented for other languages than English, for instance Chinese [55].

2.1.1 Chinese PropBank

Chinese PropBank adopts the descriptive framework of English PropBank. The semantic arguments of a predicate are labeled with a contiguous sequence of integer, and the integer is from 0 to 5. These labels can only be interpreted in the context of a specific predicate and they can not be repeated for a specific predicate. That' s to say, they are predicate-specific and can occur at most once for a specific predicate. For the semantic adjuncts, they are still annotated similarly as those in English PropBank. In other words, they are annotated as *ArgM* followed by a secondary tag that represents the semantic classification of the adjunct. Unlike the semantic arguments of a predicate, they are not predicate-specific and can repeated more than one time for a specific predicate. In Chinese PropBank, there is a limited set of such secondary tags and are listed in Table 2.1.

Table 2.1: The complete list of functional tags defined in the Chinese PropBank

ADV	adverbial	FRQ	frequency
EXT	extent	TMP	temporal
LOC	locative	MNR	manner
BNF	beneficiary	CND	conditional
DGR	degree	DIR	direction
DIS	discourse marker	PRP	purpose
QTY	quantity	TPC	topic
VOC	vocative	PRD	predicate

Figure 2.1 shows an example from Chinese PropBank, there are two semantic arguments: *Arg0*[1] 西藏金融工作 (”Finance work in Tibet”) and *Arg1* 顯著成績 (”remarkable achievement”). And they represent *agent/cause* and *thing achieved/acquired* respectively; the noun 去年 (”Last year”) is labeled *ArgM-TMP* where the secondary tag *TMP* indicates a temporal modifier.

[1]In order to save space, core argument Arg0 is short for A0; adjunct-like argument ArgM-TMP is represented by TMP. Without special explanation, the label representations that appear later will follow the same rule.

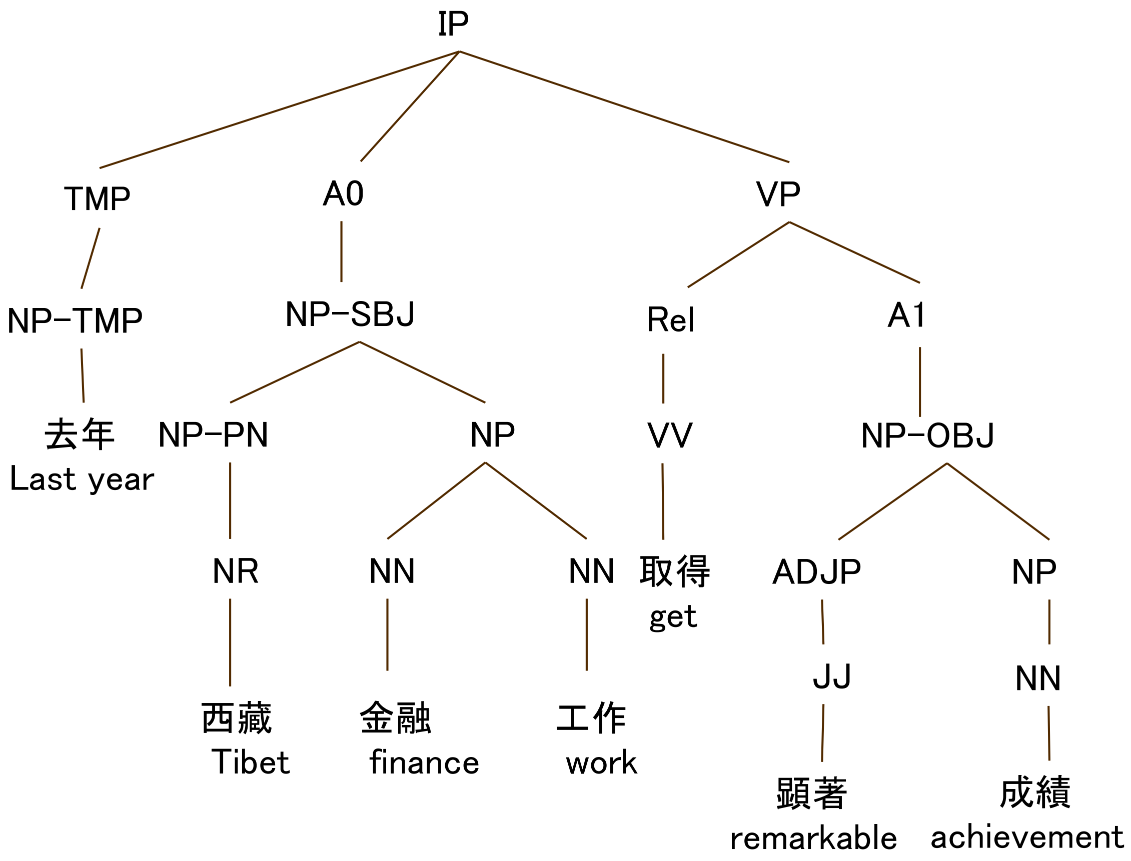


Figure 2.1: An example of SRL from CPB.

2.2 Previous Work

Before Hacioglu's work [18], most previous implementations of semantic role labelers [16, 17] have used constituents as the syntactic input. For the predicate 取得 **(get)** in the example of constituent based SRL analysis from Chinese PropBank in Figure 2.1, the argument of the role *A*0 is " 西藏金融工作 (finance work in Tibet)" and the non-terminal node **NP-SBJ** that corresponds to the words is assigned the argument role label. A wide range of statistical and machine learning techniques have been applied to SRL. Pradhan et.al [37, 38] used Support Vector Machine; Zhao and Kit [58] used Maximum Entropy; Cohn and Blunsom [7] used Conditional Random Fields to classify the semantic roles. As for Chinese, at present, most Chinese semantic role labeling systems convert the methods that work for English semantic role labeling systems, especially, after the occurrence of the English/Chinese PropBank.

In comparison, syntactic dependency based SRL has received relatively little attention for the SRL until the challenges of the SRL tasks held in CoNLL-2008 [48] and CoNLL-2009 [19]. Hacioglu [18] first explored Support Vector Machines to implement the semantic role classifiers using the dependency syntactic input. Generally, SRL and predicate sense disambiguation are regarded as two independent tasks. In the CoNLL-2008 and 2009 shared tasks, identifying the correct senses of predicates is also considered together with SRL. A few researchers have used semantic roles to help the predicate sense disambiguation [58]. In [58], they first implemented semantic classification task, then used the semantic role results as features for predicate sense disambiguation. More people used the estimated predicate senses in semantic role labelers [21, 3]. Still other work paid more attention to joint implementation of predicate sense disambiguation and SRL tasks recently [53, 30].

It is easy to represent a predicate and its semantic roles using the predicate argument structure. Moreover, the predicate argument structure can work for almost all the languages. Before the CoNLL-2008 shared task, a predicate argument structure is composed of a word that is specified as a predicate and a number of word groups that are considered as arguments accompanying the predicate. Figure 2.2 illustrates the predicate argument structure for the sentence in Figure 2.1. The predicate is the word 取得, and its arguments with their associated word groups are also illustrated. After

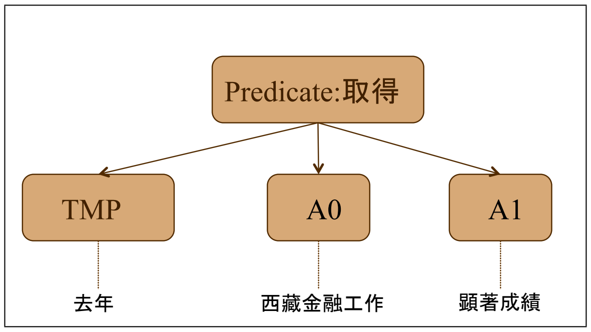


Figure 2.2: Predicate-argument structure of sample sentence in Figure 2.1.

CoNLL-2008 shared task, the predicate argument structure is also extended and shows more and more powerful vitality. In this new structure, the predicate word is changed to predicate sense, and a number of words are considered as arguments accompanying the predicate sense. The predicate argument structure for sentence in Figure 2.1 under the dependency-based SRL is shown in the lower part of Figure 2.3. Comparing with Figure 2.1, it is obvious that syntactic dependency structures offer a more transparent encoding of predicate argument relations. In Figure 2.3, the arguments that have semantic roles for the predicate 取得 (get) with the meaning representation 取得.01 are 工作 (work), 去年 (last year) and 成績 (achievement). Johansson and Nugues [21, 20]

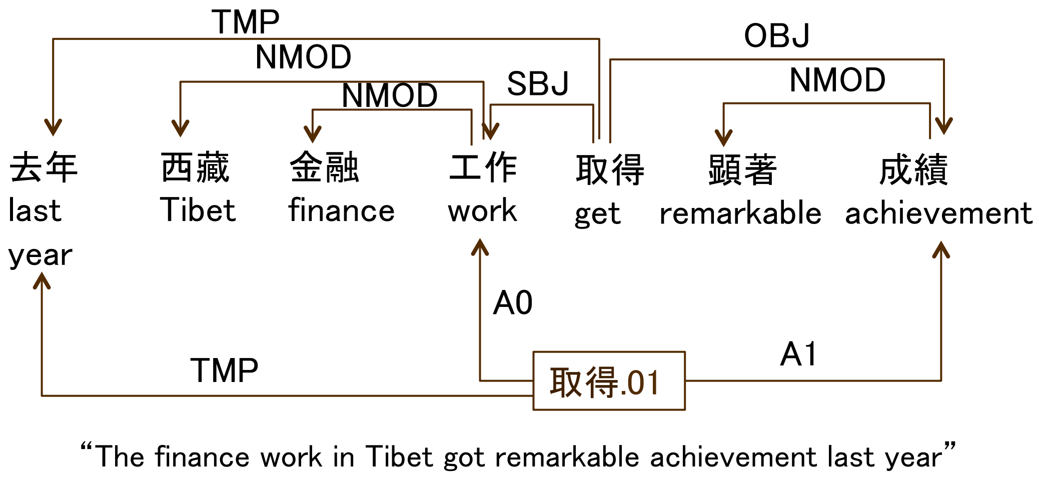


Figure 2.3: Example under dependency-based predicate argument structure analysis.

proposed a framework for joint syntactic dependency parsing and predicate argument structure analysis. They first generated N-best dependency results to input semantic pipeline which include: predicate sense disambiguation, argument identification and argument labeling. After getting the predicate argument structure for each dependency result from the semantic pipeline, the predicate argument structures together with N-best dependency results are re-ranked by a linear model. In their framework, they also used global linguistic constraints to re-rank the predicate argument structures. Their system achieved the top score in the closed challenge of the CoNLL-2008 shared task [48].

Meza-Ruiz and Riedel [30] used Markov Logic Networks (MLN), a statistical relational learning framework, to implement the syntactic dependency based predicate argument analysis, their systems jointly identified predicates, predicate senses, arguments and argument roles by designing formulae that depend on these tasks. Their experiments indicated integrating predicate disambiguation into the SRL helps to increase the robustness of SRL.

Watanabe et al. [53] proposed a structured model and introduced several types of features to capture strong dependencies among the elements in predicate argument structure. Their model successfully handled both non-local dependencies and semantic dependencies between predicate and arguments and achieved comparable results with the other state-of-the-art systems. Furthermore, it provided ways that more effective structure learning methods can be applied to learn from predicate argument structure data.

2.3 Task Description of Semantic Role Labeling

Usually, the overall semantic role labeling (SRL) process can be analyzed as three different tasks as introduced by Pradhan [39].

1. Argument identification—The words in a sentence can be classified whether it represents a semantic argument or not.
2. Argument classification—The appropriate semantic role label can be assigned on the word to identified as a semantic argument.
3. Argument identification and classification—A combination of the two tasks. In this process, the automatically identified arguments are fed through the argument classification system.

2.4 Predicate Argument Structure Analysis

Predicate argument structure analysis is the process of identifying predicate argument structure in texts. Obviously, if we can get the predicate argument structure of a sentence, then the SRL representation of the sentence can also be obtained. Under the extended representation of the predicate argument structure, like that in Figure 2.3, the task of predicate argument structure analysis contains SRL task and predicate sense disambiguation task.

2.5 Summary

In this chapter, we briefly show some background about syntactic dependency based semantic role labeling. Comparing with constituent based semantic, syntactic dependency based SRL offer a more transparent encoding of predicate argument relations.

Predicate argument structure can well reflect the semantic dependency relations between a special predicate and its arguments and can be convenient to add any features between elements in the structure. Furthermore, with this structure, it easy to construct semantic role labeling models for almost all the languages. Therefore, we still would like to use the predicate argument structure to construct models and explore what features are helpful for Chinese semantic role labeling in our following work.

Chapter 3

Syntactic Dependency Based Predicate Argument Structure Analysis

Predicate argument structure analysis has been proved to be effective for SRL in many previous work [52, 21]. Most previous work pays little attention to the interaction between the predicate senses disambiguation task and the argument classification task. Meza-Ruiz and Riedel [30] and Watanabe et al. [53] indicated that predicate senses are very helpful for judging the semantic relations of its arguments. However, without arguments roles, sometimes it is difficult to determine the senses of a predicate.

The following examples having the same verb predicate ”drive” with similar syntactic environments show semantic roles are helpful for predicate sense disambiguation. The types of the role *proto-patient* of these senses are *vehicle* and *thing in motion* respectively. Therefore, if one ”proto-patient” of the predicate ”drive” is a *vehicle*, it is easy to determine the predicate sense for ”drive” as ”driver.01”.

1. Paul drove his car fast.
   drive.01: drive a vehicle
   A0: Paul (driver) **A1**: car (vehicle)

2. Traders drove price sharply higher.
   drive.02: cause to move
   A0: trader (driver) **A1**: price (thing in motion)
   A2: higher (secondary predication on A1)

Watanabe et al. [53] proposed a structured model that overcame this limitation and can identify predicate sense disambiguation and argument roles jointly, which made both tasks be mutually influenced and determined the most plausible set of assignments of a predicate sense and its argument roles simultaneously. In their work, a linear model is applied for predicate argument structure learning. Although linear models are much easier to add more variables, they can not provide more control over the interaction of the variables than log linear models. Inspired by this work, we would like to construct a SRL model that inherits the advantage in [53] and can provide more control over the interaction of the variables.

3.1 Model Definition for Joint Learning of Argument Roles and Predicate Senses

The joint model is constructed on the predicate argument structure graph which is similar as that in Figure 3.1. First, we define an instance as a predicate word and its corresponding argument words. If there are m predicates in a sentence, then there will be m instances. Given an instance $X = \{x_1, \ldots, x_p, \ldots, x_n\}$ with the predicate position p, we want to find the corresponding sequence of argument labels and predicate sense $S = a_1, \ldots, a_{p-1}, P, a_{p+1}, \ldots, a_n = \langle P, A \rangle$. Each a_i for the i-th word in the instance X is drawn from a set of tags $T(A)$ which contains all the semantic role labels in the corpus and which follows the definition criteria in Chinese PropBank. In addition, a special label $NONE$ is added to $T(A)$. If a word is labeled as $NONE$, the word will not be an argument of the current predicate. As for P, this is a member of a sense set $T(x_p)$ which contains all possible senses of predicate word x_p. We propose two sorts of label assignment models Pr_{local} and Pr_{global}. The former incorporates local features only; the latter also incorporates global features.

Figure 3.1 shows the predicate argument structure graph for sentence in Figure 2.1. The nodes in the frame are all the arguments of the current predicate: 取得 (get) under the sense of 取得.01. The solid squares are factors which provide scores of label assignments. By influencing labels of predicate sense and argument roles, the most plausible label assignments of the nodes are determined by all factors.

3.2 Feature Categories

Factors refer to feature sets and one factor stands for one feature category. In this section, we will explain four factors which are categorized according to their contribu-

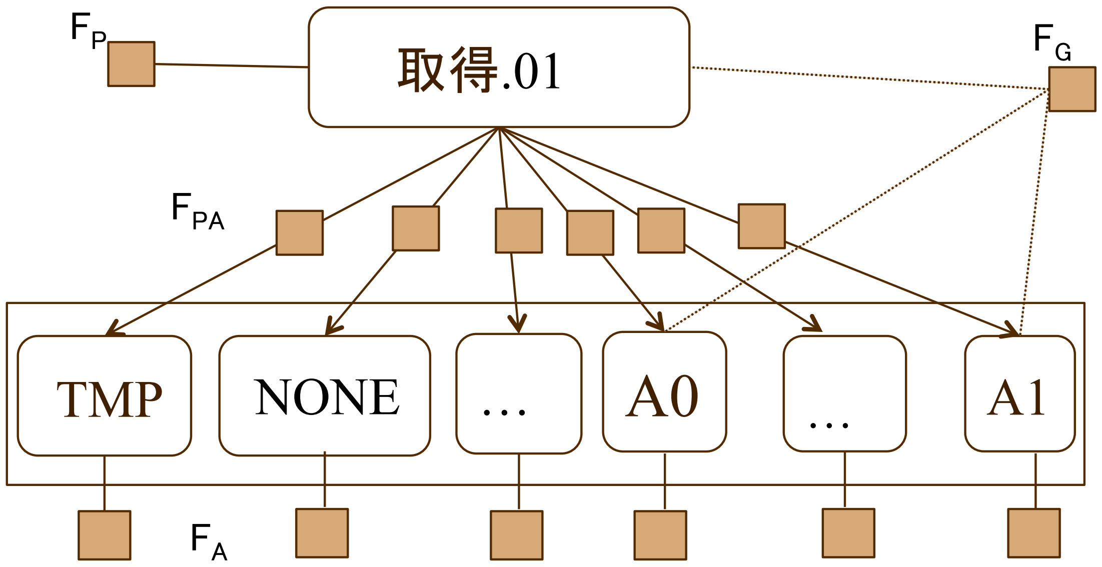


Figure 3.1: Direct graphical representation of the predicate arugment structure.

tions to role labels and predicate sense assignments.

To illustrate the feature representational examples that we will describe in this section, we will refer to the situation denoted by the Figure 2.3.

3.2.1 Predicate Factor

Features in this category represented by $F_P = \{f_P\}$ are used for scoring the sense of x_p and are independent of the arguments of the predicate x_p. For example,

$$f_P(X,p,P) = \begin{cases} 1 & \text{if predicate lemma = 取得 (get)} \\ & \text{and P = 取得.01} \\ 0 & \text{otherwise} \end{cases}$$

3.2.2 Argument Factor

This category, defined by $F_A = \{f_A\}$, scores the semantic label assignment for an argument and is independent of a predicate sense. For example,

$$f_A(X, a_i) = \begin{cases} 1 & \text{if current argument lemma} \\ & = \text{工作 (work) and } a_i = A_0 \\ 0 & \text{otherwise} \end{cases}$$

3.2.3 Predicate-Argument Pairwise Factor

These features associate semantic label and sense simultaneously which permit not only to use all predicate senses disambiguated, but also to enable semantic role labeling to simultaneously help in the predicate sense disambiguation. These features are defined by $F_{PA} = \{f_{PA}\}$ and an example is as follows:

$$f_{PA}(X, p, P, a_i) = \begin{cases} 1 & \text{if current argument lemma} \\ & = \text{工作 (work) and } a_i = A_0 \\ & \text{ and P = 取得.01} \\ 0 & \text{otherwise} \end{cases}$$

3.2.4 Global Factor

This feature set is defined $F_G = \{f_G\}$ for two purposes: (1) to ensure that the sequence of the arguments is assigned according to the predicate frame; (2) to reflect the joint relation between predicate sense disambiguation and argument assignments. The sequence of the predicate sense and core argument labels (e.g.A0- 取得.01-A1) is used to filter out undesirable results and has been successfully exploited in several previous systems [50, 20].

3.3 Predicate Sense Disambiguation and SRL with a Local Model

Since the predicate cannot be an argument of itself for Chinese, we define the following local probabilistic model for argument classification and predicate sense disambiguation.

$$Pr_{local}(S|X) = \prod_{i=1(i \neq p)}^{n} Pr(a_i|P, X, i, p) \cdot Pr(P|X, p) \qquad (3.1)$$

where $Pr(a_i|P,X,i,p)$ and $Pr(P|X,p)$ are estimated according to the following equation:

$$Pr(a_i|P,X,i,p) = \frac{1}{Z^A(X)} \exp\{\sum_{f_{A_j} \in F_A} \lambda_{f_{A_j}} f_{A_j}(X,a_i) + \sum_{f_{PA_k} \in F_{PA}} \lambda_{f_{PA_k}} f_{PA_k}(X,p,P,a_i)\},$$

$$Pr(P|X,p) = \frac{1}{Z^P(X)} \exp\{\sum_{f_{P_l} \in F_P} \lambda_{f_{P_l}} f_{P_l}(X,p,P)\},$$

where Z^A and Z^P are normalization functions, i.e.,

$$Z^A = \sum_{a_i \in T(A)} \exp\{\sum_{f_{A_j} \in F_A} \lambda_{f_{A_j}} f_{A_j}(X,a_i) + \sum_{f_{PA_k} \in F_{PA}} \lambda_{f_{PA_k}} f_{PA_k}(X,p,P,a_i)\};$$

$$Z^P = \sum_{P \in T(x_p)} \exp\{\sum_{f_{P_l} \in F_P} \lambda_{f_{P_l}} f_{P_l}(X,p,P)\};$$

f are the features with associated weight λ learned via training. The features are, typically binary valued functions, defined as four types as mentioned in Section 3.2.

3.4 Predicate Sense Disambiguation and SRL with the Global Model

Global information is known to be useful in NLP tasks. Nakagawa [33] proposed a whole-sentence exponential language model for dependency parsing and this model can incorporate arbitrary features in a sentence. We consider a similar global probabilistic model Pr_{global} here for SRL as follows:

$$Pr_{global}(S|X) = \frac{1}{Z} Pr_{local}(S|X) \cdot exp\left\{\sum_{f_{G_m} \in F_G} \lambda_{f_{G_m}} f_{G_m}(S,X)\right\} \quad (3.2)$$

where Z is a normalizing factor over all candidate sequences and is defined as:

$$Z = \sum_{S \in S(X,p)} Pr_{local}(S|X) \cdot exp\left\{\sum_{f_{G_m} \in F_G} \lambda_{f_{G_m}} f_{G_m}(S,X)\right\},$$

and $S(X,p)$ is the set of possible configurations of semantic tags and predicate senses given X and predicate location p. To get the whole sequence of S, we need to perform computationally expensive search. As is done in previous work [53], we use a simple approach, N-best relaxation. Unlike the $Pr_{local}(S|X)$, the product of probability distributions of each word, the probability distribution $Pr_{global}(S|X)$ is calculated by feature functions f_G defined on an instance X with assignment S. Thereby, we can use any information in an instance without the independence assumption for assignments of words in it.

3.5 Online Learning

The perceptron-based online algorithm by Collins [8] is the simplest online algorithm and can provide the state-of-the-art results on various domains including syntactic parsing, text chunking, etc. The main drawback of this perceptron style algorithm is that it does not have a mechanism for attaining large margin in the training phase. It may be difficult to obtain good generalization on unseen data. We therefore use a modified online-learning based on the Passive-Aggressive algorithm (PA) [9].

For the sake of convenience, we introduce the following notations respectively: $\Lambda = \{\lambda_{f_A}, \lambda_{f_{PA}}, \lambda_{f_P}, \lambda_{f_G}\}$ and $F = \{F_A, F_{PA}, F_P, F_G\}$. Using these global vectors, $\mathrm{Pr}_{global}(S|X)$ can also be represented as $\mathrm{Pr}_{global}(S|X) = \frac{1}{Z} exp(\Lambda \cdot F)$. The best predicate-argument sequence $\hat{S}$ for the instance X is then given by

$$\hat{S} = \underset{S}{argmax}\ Pr_{global}(S|X) = \underset{S}{argmax}\ \frac{1}{Z} exp(\ \Lambda \cdot F).$$

We define the *margin* attained by the algorithm on round j for the example (X_j, S_j) as,

$$\gamma(\Lambda_j, (X_j, S_j)) = \Lambda_j \cdot F(X_j, S_j) - \Lambda_j \cdot F(X_j, \hat{S}_j)$$

The margin is positive only if all the relevant labels are ranked higher than all of the irrelevant labels. However, in the spirit of the binary classification [9], we are not satisfied by a mere positive margin, so we define the margin to be at least $\Delta(S, \hat{S})$ which is defined by the number of words which have incorrect semantic role and incorrect predicate sense predictions. Obviously, the largest $\Delta(S, \hat{S})$ that can have is the length of the sentence. The instantaneous loss is defined by the following hinge-loss function.

$$\iota(\Lambda, (X, S)) = \begin{cases} 0 & \gamma(\Lambda, (X, S)) \geq \Delta(S, \hat{S}) \\ \Delta(S, \hat{S}) - \gamma(\Lambda, (X, S)) & \text{otherwise} \end{cases}$$

Therefore, The PA algorithm can be seen as the following optimization problem that satisfies the corresponding constraint.

$$\Lambda = \underset{\Lambda \in \Re^n}{argmin} \frac{1}{2} \|\Lambda - \Lambda_t\|^2 \quad \text{s.t.} \quad \Lambda \cdot F(X, S) - \Lambda \cdot F(X, \hat{S}) \geq \Delta(S, \hat{S})$$

ALGORITHM 1: The Passive-Agressive Algorithm for SRL

```
Input: Training set $\tau = \left\{X_j, S_j\right\}_{j=1}^{\mathbb{N}}$, maximum iteration value $\mathbb{IT}$ and
       regularization parameter $\mathbb{C}$.
Output: The parameter $\Lambda$ that minimizes the hinge-loss function and the margin
        of learning data $\Delta(S, \hat{S})$ .
$\Lambda \leftarrow 0$, $\upsilon \leftarrow 0$, $c \leftarrow 0$;
for $itr$ to $\mathbb{IT}$ do
    for each $(X_j, S_j)$ in $\tau$ do
        $\hat{S}_j = \underset{S}{argmax}\ \frac{1}{Z} exp(\Lambda \cdot F(X_j, S))$;
        $\sigma_i = min\left(\mathbb{C}, \frac{\Lambda \cdot F(X_j, \hat{S}_j) - \Lambda \cdot F(X_j, S_j) + \Delta(S_j, \hat{S}_j)}{\left\|F(X_j, S_j) - F(X_j, \hat{S}_j)\right\|^2}\right)$;
        $\Lambda \leftarrow \Lambda + \sigma_j(F(X_j, S_j) - F(X_j, \hat{S}_j))$;
        $\upsilon \leftarrow \upsilon + c\sigma_j(F(X_j, S_j) - F(X_j, \hat{S}_j))$;
        $c \leftarrow c + 1$;
    end
end
return $\Lambda = \Lambda - \upsilon / c$;
```

Using the Lagrangrian and its derivative, it is easy to solve this optimization problem. The detail of PA algorithm for SRL is presented in Algorithm 1, where N is the number of instances in the training data. Also, in order to reduce the overfitting problem, we apply an efficient parameter averaging technique [10]. The vector υ and the variable c in Algorithm 1 play this role.

3.6 Features

Earlier research [16, 55] recognized the necessity and importance of syntactic parsing for SRL. Therefore, it is critical to effectively utilize the syntactic structure features in argument labeling. Unfortunately, to find a useful feature set is usually a nontrivial task. In our experiments we ignored this task by adapting more of the features that have been described in recent work on English SRL to Chinese. The features used in our experiments are listed in Table 3.1 according to the four feature categories defined above.

Table 3.1: Features in dependency-based Chinese predicate argument structure analysis

Predicate Factor	
POS	part of speech of the predicate and its parent
Dependency	dependency relation between the predicate and its head
Dependency Set	set of dependency labels for predicate's dependents
Argument Factor	
Lemma	lemmas of the argument and its head lemma of the leftmost/rightmost dependents of the argument lemma of the leftmost/rightmost siblings of the argument
POS	part of speech of the argument and its head part of speech of of the leftmost/rightmost dependents of the argument part of speech of of the leftmost/rightmost siblings of the argument
Location	location relation between the argument and predicate in the instance position of the argument with respect to the predicate in the dependency tree
Path	dependency label/lemma/part of speech paths between the argument and predicate with the direction of the edge
Dependency	dependency relation between the argument and its head dependency relations between the argument's dependents and their heads
Predicate-Argument Pairwise Factor	
Lemma	lemma of the current candidate argument
POS	part of speech of the argument
Lemma and POS	combination of the lemma and part of speech of the argument
Path	dependency label path between the argument and predicate with the direction of the edge
Global Factor	

Continued on next page

Table 3.1 – *Continued from previous page*

Semantic Sequence	the core semantic role sequence with the sense of the predicate in consistent with their location in the sentence.

3.7 Experiments and Discussion

To assess our predicate argument structure analysis model, we evaluated the results by comparing the output from its pipeline-based model.

3.7.1 Pipeline-based Models

Figure 3.2 shows the architecture of the pipeline-based models. In this method,

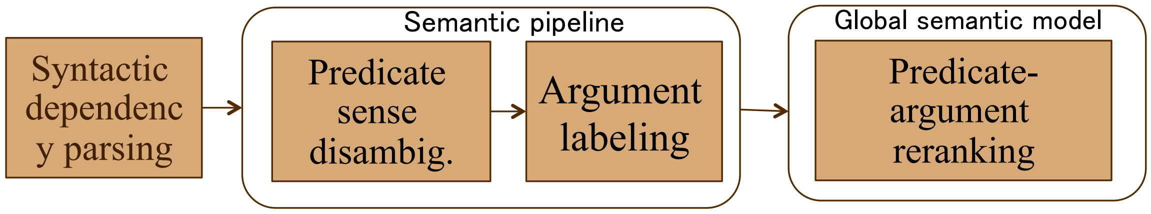

Syntactic dependenc y parsing
Semantic pipeline
Predicate sense disambig.
Argument labeling
Global semantic model
Predicate-argument reranking

Figure 3.2: The architecture of the pipeline-based predicate argument structure analyzer.

first, we determined the best sense for each predicate using Equation 3.3, then use the predicted sense as features for argument label assignment according to Equation 3.4.

$$\begin{aligned}\hat{P} &= \underset{P}{argmax} \Pr(P|X,p) \\ &= \underset{P}{argmax} \frac{1}{Z^P(X)} \exp\{\sum_{f_{P_l} \in F_P} \lambda_{f_{P_l}} f_{P_l}(X,p,P)\}\end{aligned} \tag{3.3}$$

$$Pr_{local}(S|X) = \prod_{i=1(i \neq p)}^{n} Pr(a_i|\hat{P},X,i,p) \tag{3.4}$$

Correspondingly, the $Pr(a_i|\hat{P},X,i,p)$ is estimated according to the following equation:

$$Pr(a_i|\hat{P},X,i,p) = \frac{1}{Z^A(X)}\exp\{\sum_{f_{A_j}\in F_A}\lambda_{f_{A_j}} f_{A_j}(X,a_i) + \sum_{f_{PA_k}\in F_{PA}}\lambda_{f_{PA_k}} f_{PA_k}(X,p,\hat{P},a_i)\},$$

and the normalization factor Z^A becomes as follows:

$$Z^A = \sum_{a_i\in T(A)}\exp\{\sum_{f_{A_j}\in F_A}\lambda_{f_{A_j}} f_{A_j}(X,a_i) + \sum_{f_{PA_k}\in F_{PA}}\lambda_{f_{PA_k}} f_{PA_k}(X,p,\hat{P},a_i)\};$$

finally, based on the predicate argument structures from semantic pipeline, we detected the highest scored argument assignment using the global model in Section 3.4.

3.7.2 Dataset and Setup

We used the Chinese dataset provided by CoNLL-2009 shared task for experiments. For the SRL-only task in the shared task, the dataset came with MALTParser [35] dependencies. In the training corpus, there are 102,813 predicates including verb predicates and noun predicates; 8,104 predicates are contained in the development corpus and 12,282 predicates are included in the test corpus. For the details of this dataset, see [19].

The overall performance of semantic role labeling is calculated using the semantic evaluation metric of the CoNLL-2009 shared task scorer[1]. It measures the precision, recall and $F_{\beta=1}$ as below:

$$\text{semantic labeled precision } Sem.P = \frac{\text{\# of correct pred.senses} + \text{\# of correct arg.roles}}{\text{\#predicates} + \text{\# of returned arg}}$$

$$\text{semantic labeled recall } Sem.R = \frac{\text{\# of correct pred.senses} + \text{\# of correct arg.roles}}{\text{\#predicates} + \text{\# arguments}}$$

$$\text{semantic labeled } F_1 \quad Sem.F = \frac{2Sem.P \cdot Sem.R}{Sem.P + Sem.R}.$$

3.7.3 Results and Discussion

Table 3.2 shows the results of the experiments, and also shows the results of the top 3 systems [3, 57, 30] on the CoNLL-2009 Shared Task participated as *SRL-only* system. Furthermore, in order to show clearly our approach can provide more control

[1]`http://ufal.mff.cuni.cz/conll2009-st/eval09.pl`

over the interaction of the variables than that in Watanabe et al. [52, 53], Table 3.2 also shows the result achieved by applying their approach and using our feature sets to get the SRL result for comparison.

Table 3.2: Predicate argument structure analysis results on Chinese corpus

Systems	Sem.P	Sem.R	Sem.F
Joint Syntactic Dependency-based SRL	82.64	72.68	77.34
Pipeline Dependency-based SRL	82.28	72.42	77.04
[Watanabe, 2009]	82.62	71.11	76.43
[Björkelund, 2009]	82.42	75.12	78.60
[Meza-Ruiz, 2009]	82.66	73.36	77.73
[Zhao, 2009]	80.42	75.20	77.72

Comparing the two results from joint method and pipeline method respectively, we see that improvement can be made if we jointly perform inference for predicate sense disambiguation and argument labeling. For a predicate word with multi-senses, the pipeline model may not have enough confidence to assign the correct meaning to it. In that case, this error may easily be propagated to the argument assignment step. However, the joint probability that they can be correctly disambiguated simultaneously is larger than incorrect cases.

On the other hand, it can be seen that the improvement is not so big. The main reason is that most predicates (13.9%) in test corpus only have one sense, while in train corpus there are more than 30% predicates with multi-senses.

Comparing our result with the result achieved by using Watanabe's method, it can be seen that our model truly provides more control over the interaction of the variables.

Comparing our results with the top 3 systems in CoNLL-2009 Shared Task, our results is not ideal. The main reason may be that Björkelund [3] and Zhao [57] applied feature selection algorithms in order to select the best set of feature templates. On the other hand, we only used the common feature templates appeared in most previous research, thus there is still room for performance improvement by applying feature selection algorithms.

3.8 Summary

In this chapter, we proposed a structured model that can jointly implement predicate sense disambiguation and argument role assignment. This model inherits the advantage of the model proposed by Watanabe et al. [53] that can capture both non-local dependencies between arguments, and inter-dependencies between argument and predicate senses. More precisely, we designed a log linear model-based structure model. And this model plays an important part in the latter chapters where other methods and features are explored to improve the Chinese predicate argument structure analysis results.

In our work, the argument identification subtask was ignored by adding a special label *NONE* to the predefined semantic role set in argument classification subtask. Label *NONE* indicates a word is not an argument. In this case, the argument classification system plays the role of identifying the arguments and classifying them simultaneously. This method alleviates the error propagation problem to some extents. However, the benefit from the argument identification subtask is also ignored. As experiment results show, our results have lower recall value. In our following work, we would like to apply different approaches and explore helpful features to improve this problem.

Chapter 4

Robust Integrated Models for SRL

Predicate argument (PA) structure analysis is most often divided into three subtasks: predicate sense disambiguation, argument identification and argument classification. To date, they have mostly been modeled in isolation. However, this approach neglects logical constraints between them. We therefore exploited integrating predicate sense disambiguation with the latter two subtasks respectively, which verifies the automatic predicate sense disambiguation could help the semantic role labeling task. In addition, a dual decomposition algorithm was used to alleviate the error propagation between argument identification subtask and argument classification subtask by benefitting the argument identification subtask greatly. Experiment results show that our approach leads to a higher performance for predicate-argument analysis than other pipeline approaches.

4.1 Introduction

Predicate argument structure analysis is a fundamental task in natural language processing to find a clause-level semantic representation. An example is shown in Figure 4.1.

Given the predicate token " wore " with multiple senses, we are to distinguish the sense of " wore " to be " to put something on one 's body " and the sense label is " wear.01 "; to identify the argument headed by the token " she " as the " wearer " and the argument headed by the token " coat " as the " clothing ". It has obvious applications for tasks which involve determining **who** does **what** to **whom**, such as question answering [22, 34, 42], information extraction [47] and text categorization

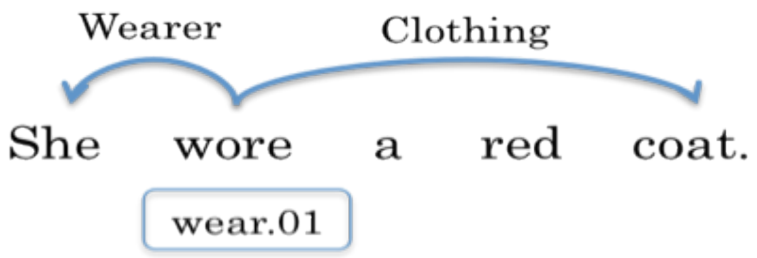


Figure 4.1: An example for predicate-argument structure.

[36].

Normally, PA structure analysis is regarded as three subtasks: predicate sense disambiguation, argument identification and argument classification. A few researchers [58] performed the latter two subtasks first and used the semantic roles to help the predicate sense disambiguation. Most researchers [3, 21, 20] performed predicate sense disambiguation first and used the predicted predicate senses to help the argument identification and argument classification subtasks. However, both the two pipeline methods ignore possible dependencies between the three subtasks and can result in error propagation problem. Still other researchers ignored the argument identification subtask and jointly implemented the predicate sense disambiguation subtask and the argument classification subtask [53]. This method alleviates the error propagation problem to some extents, while the benefit from the argument identification subtask is also ignored.

To address these issues, we implemented the three subtasks for PA structure analysis in the following three steps and Figure 4.2 illustrates the architecture of our system.

Firstly, we incorporated non-local dependencies features between predicate senses and their corresponding argument candidates to help argument identification. Secondly, the dual decomposition [32, 25, 11] was introduced to integrate two argument identification models, which were integrated with predicate sense disambiguation. This method enforces the outputs of both argument identification models to agree with each other as much as possible. In this approach, we iteratively applied the similar dynamic programming algorithms for the whole joint model and thereby optimized a dual bound on the model object. In cases where our algorithm converged, we had a certificate of optimality under the full model.

Finally, we run a joint argument classification model for the identified arguments from dual decomposition. This joint model implemented predicate sense disambigua-

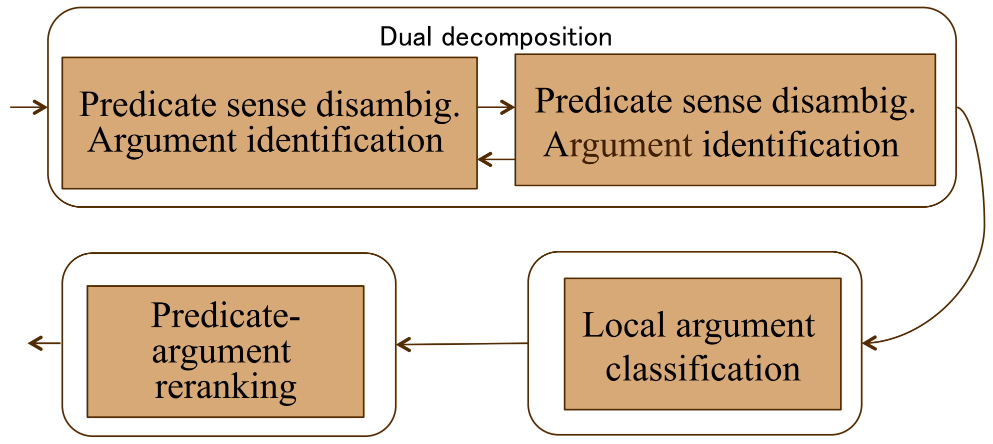

Dual decomposition
Predicate sense disambig. Argument identification
Predicate sense disambig. Argument identification
Local argument classification
Predicate-argument reranking

Figure 4.2: The overall architecture of our system.

tion and argument classification simultaneously, which made both subtasks can help each other. For the sentence in Figure 4.1, if we know " wear " has a sense of " to put something on one 's body ", then we will guess there should be two arguments: one is something related(clothing) and another is somebody related(wearer). In the same way, if we know two arguments: one is wearer and the other is clothing, we can guess the predicate sense is " to put clothing on wearer 's body ". The joint probability that they are disambiguated correctly simultaneously is larger than incorrect cases.

Experiments on the Chinese corpus provided by CoNLL-2009 shared task show that (1) the automatic predicate sense disambiguation is helpful for semantic role labeling; (2) argument identification subtask greatly benefits from the dual decomposition method, which leads to higher final performance for PA structure analysis.

The rest of this chapter is organized as following:

Before presenting our algorithm, the theory for our approach about dual Decomposition will be described in Section 4.2; Section 4.3 outlines two joint models:(1) argument identification; (2) joint argument classification and predicate sense disambiguation, followed by our dual decomposition method to achieve the final argument identification problem formulation in Section 4.4. Section 4.5 gives the experiment results and analysis in detail. Finally, Section 4.6 concludes the chapter.

4.2 Dual Decomposition

4.2.1 Introduction

Dual decomposition, also called Lagrangian relaxation, is a classical method for combinatorial optimization; it has recently been applied to several inference problems in natural language processing. Dual decomposition leverages the observation that many decoding problems can be decomposed into two or more sub-problems that can be solved efficiently using some exact algorithms. And these subproblems must satisfy linear constraints to enforce some notion of agreement between solutions to them. The agreement constraints are incorporated using Lagrange multipliers, and an iterative algorithm is used to minimize the resulting dual. Dual decomposition algorithm have the following properties [31].

- They are typically simple and efficient.
- They have well-understood formal properties, in particular through connections to linear programming (LP) relaxations.
- They produce an exact solution to the original decoding problem with a certificate of optimality in cases where the underlying LP relaxation is tight.

4.2.2 Lagrangian Relaxation

Consider we have some finite set Y, which is a subset of $\mathbb{R}^d$. The score associated with any vector $y \in Y$ is

$$h(y) = y \cdot \theta$$

where θ is also a vector in $\mathbb{R}^d$. The decoding problem is to find

$$y^* = \underset{y \in Y}{argmax}\, h(y) = \underset{y \in Y}{argmax}\, y \cdot \theta \tag{4.1}$$

In some cases the problem in Equation 4.1 is intractable and we thus need to resort to approximations. The first key step in Lagrangian relaxation will be to choose a finite set $Y' \subset \mathbb{R}^d$ and it must has the following properties [31].

- $Y \subset Y'$. Thus all the vectors in Y can be found in Y'.

- For any value $\theta \in \mathbb{R}^d$, it can be easily found

$$\underset{y \in Y'}{argmax}\, y \cdot \theta$$

- Assuming there exits linear constraints that make

$$Y = \left\{ y : y \in Y' \text{ and } Ay = b \right\}.$$

The original problem can be converted to be an constrained optimization problem. A solution to the relaxed problem is an approximate solution to the original problem, and provides useful information. By using a Lagrangian multiplier, the constraint $Ay = b$ can be introduced into the objective:

$$L(u, y) = y \cdot \theta + u \cdot (Ay - b)$$

This function combines the original objective function $y \cdot \theta$, with a second term that incorporates the linear constraints and the Lagrange multipliers. The dual objective is

$$L(u) = \max_{y \in Y'} L(u, y) \tag{4.2}$$

Since Y' also contains some vectors that are not in Y, $L(u) \geqslant h(y)$. In other words, $L(u)$ will be an upper bound on the original problem. The dual problem is to find

$$\min_{u \in \mathbb{R}^p} L(u)$$

where p is the number of linear constraints on y.

To minimize the approximate objective $L(u)$, local updates can be used. Each iteration the value returned by $L(u)$ is a candidate upper bound to the original problem, the smallest of which is kept as the best upper bound. We can iterate until the best upper bound and the cost of the best feasible solution to a desired tolerance.

Subgradient method, which has been widely applied to solving Lagrangian relaxation problems and is often surprisingly effective, in spite of it being a simple method. The subgradient method is similar to gradient descent, but is applicable to non-differentiable objective. We set the initial Lagrange multiplier values to be $u^0 = 0$. For $k = 0, 1, 2, \cdots$, this method iterates

$$y^{k+1} = \underset{y \in Y'}{argmax}\, L(u^k, y)$$

$$u^{k+1} = u^k - \alpha_k(Ay^{k+1} - b)$$

where α_k is the step size at the k'th iteration. At each iteration, a structure y^{k+1} is found, and is used to update the Lagrange multipliers. y^{k+1} can be found efficiently for that:

$$\begin{aligned} \underset{y\in Y'}{argmax}\, L(u^{(k)}, y) &= \underset{y\in Y'}{argmax}\, (y\cdot\theta + u^{(k)}\cdot(Ay - b)) \\ &= \underset{y\in Y'}{y} \cdot \theta' \end{aligned}$$

where $\theta' = \theta + A^T u^{(k)}$. Therefore the Lagrange multiplier terms are easily incorporated into the objective function.

A well-known theoretical result is that the subgradient method is guaranteed to solve the optimality whenever the step-sizes are chosen such that $\lim_{k\to\infty}\alpha_k = 0$ and $\sum_{k=0}^{\infty}\alpha_k = \infty$ [43, 1]. One example of such a step-size is $\alpha_k = \frac{1}{k}$. Komodakis et al. [23] introduced further possibilities of how to choose the step size.

4.2.3 Dual Decomposition

Dual decomposition is a special case of Lagrangian relaxation. Assuming there are several finite sets $Y \subset \mathbb{R}^d$ and some finite sets $Z \subset \mathbb{R}^{d'}$. Each set $y \in Y$, $z \in Z$ have the following respective associated scores

$$f(y) = y\cdot\theta^{(1)}$$

$$g(z) = z\cdot\theta^{(2)}$$

$\theta^{(1)}$, $\theta^{(2)}$ are vectors in $\mathbb{R}^d$ and $\mathbb{R}^{d'}$ respectively. The decoding problem is then to find

$$\begin{aligned} &\underset{y\in Y, z\in Z}{max}\; y\cdot\theta^{(1)} + z\cdot\theta^{(2)} \\ &such\ that\ Ay + Cz = b \end{aligned} \tag{4.3}$$

where $A \in \mathbb{R}^{p\times d}$, $C \in \mathbb{R}^{p\times d'}$ and $b \in \mathbb{R}^p$.

Under the linear constraints specified by $Ay + Cz = b$, the decoding problem is to find the optimal pair of structures. In practice, the linear constraints often specify agreement constraints between y and z. These agreement constraints specify that the two vectors are in some sense coherent [31]. The decoding is to find

$$\underset{y\in Y, z\in Z, Ay+Cz=b}{max}\; y\cdot\theta^{(1)} + z\cdot\theta^{(2)} \tag{4.4}$$

Unfortunately, Equation 4.4 is not easily to solve most time, but if for any value of $\theta^1 \in \mathbb{R}^d$, it is easy to calculate $\underset{y\in Y}{argmax}\, y \cdot \theta^{(1)}$. Similarly, for any value of $\theta^2 \in \mathbb{R}^{d'}$, we can easy calculate $\underset{z\in Z}{argmax}\, z \cdot \theta^{(2)}$. Then it is easy to get Equation 4.5 by setting

$$y^* = \underset{y\in Y}{argmax}\, y \cdot \theta^{(1)}, z^* = \underset{z\in Z}{argmax}\, z \cdot \theta^{(2)}$$

and dropping the linear constraints $Ay + Cz = b$.

$$(y^*, z^*) = \underset{y\in Y, z\in Z}{argmax}\, y \cdot \theta^{(1)} + z \cdot \theta^{(2)} \tag{4.5}$$

The dual decomposition algorithm is then derived in a similar way to before. After introducing a vector of Lagrange multipliers, $u \in \mathbb{R}^p$, the Lagrangian is now

$$L(u, y, z) = y \cdot \theta^{(1)} + z \cdot \theta^{(2)} + u \cdot (Ay + Cz - b)$$

And the dual objective is

$$L(u) = \underset{y\in Y, z\in Z}{max}\, L(u, y, z) \tag{4.6}$$

To find $\underset{u\in\mathbb{R}^p}{min}\, L(u)$, a common method is the subgradient algorithm. Initialize the Lagrange multipliers to $u^{(0)} = 0$. For $k = 0, 1, 2, \cdots$, the following steps are performed:

$$(y^{k+1}, z^{k+1}) = \underset{y\in Y, z\in Z}{argmax}\, L(u^k, y, z)$$

$$u^{k+1} = u^k - \alpha_k (Ay^{k+1} + Cz^{k+1} - b)$$

where α_k stepsize.

From the assumptions for Equation 4.5, it can be seen that the solution y^{k+1}, z^{k+1} are calculated easily, because it is easily verified that

$$\underset{y\in Y, z\in Z}{argmax}\, L(u^k, y, z) = (\underset{y\in Y}{argmax}\, y \cdot \theta'^{(1)}, \underset{z\in Z}{argmax}\, z \cdot \theta'^{(2)})$$

where $\theta'^{(1)} = \theta^{(1)} + A^T u^k$ and $\theta'^{(2)} = \theta^{(2)} + C^T u^k$. Thus the dual decomposes into two easily solved maximization problems [31].

4.3 Model Definition

All our models were constructed based on the direct graphical representation of the predicate argument structure which is shown in Figure 3.1.

There are two kinds of vertices in the structure: a predicate vertex and argument vertices. The label of predicate vertex corresponds to the sense of the predicate. For argument identification subtask and argument classification subtask, the argument vertices represent in different ways. In argument identification subtask, the argument vertices correspond to the identification labels, which indicate whether their corresponding words in the sentence are arguments; in argument classification subtask, the argument vertices correspond to the semantic roles under this predicate sense. The solid squares are factors, which are vectors of associated potential functions. Each potential function is assigned a real-valued potential value given by features. There are four kinds of factors and we have presented them in Chapter 3, Section 3.2 in detail. In order to distinguish their different functions in argument classification subtask and argument identification, we use B to note factors that work for argument identification and A for argument classification.

4.3.1 Argument Identification

Given a sentence $X = \{x_1, \ldots, x_p, \ldots, x_n\}$ with the predicate position p, we defined the local probabilistic model $Pr^{id}_{local}(B|X)$ in Equation 4.7. Since for Chinese, the predicate cannot be an argument of itself, we define a label sequence $B = \left\langle b_1, \cdots, b_{p-1}, b_{p+1}, \cdots, b_n \right\rangle$ for X with each b_i indicating whether the index word is an argument or not.

$$Pr^{id}_{local}(B|X) = \prod_{i=1(i \neq p)}^{n} Prob^{id}_{local}(b_i|P,X,i,p) \cdot Prob^{predlabel}_{local}(P|X,p) \tag{4.7}$$

where P indicates the sense of the predicate word x_p. For this joint task, three kinds of features are applied. $F_B = \left\langle f_{B_j} \right\rangle$ is an argument factor for identification assignment of one argument candidate. $F_{PB} = \langle f_{PB_k} \rangle$ is a predicate argument pair-wise factor for identification assignment of one argument candidate and predicate sense determination. $F_P = \langle f_{P_l} \rangle$ is a predicate factor for disambiguating predicate senses. We also define their corresponding weight vectors $\Lambda_{F_B} = \left\langle \lambda_{f_{B_j}} \right\rangle$, $\Lambda_{F_{PB}} = \left\langle \lambda_{f_{PB_k}} \right\rangle$ and $\Lambda_{F_P} = \left\langle \lambda_{f_{P_l}} \right\rangle$ respectively.

Based on these definitions, the $prob^{id}_{local}$ and $prob^{predlabel}_{local}$ could be represented as

$$\begin{aligned} prob^{id}_{local}(b_i|P,X,i,p) = \frac{1}{Z^{id}_{local}(X)} \exp\{ & \sum_{f_{B_j} \in F_B} \lambda_{f_{B_j}} f_{B_j}(X, b_i) \\ & + \sum_{f_{PB_k} \in F_{PB}} \lambda_{f_{PB_k}} f_{PB_k}(X, p, P, b_i)\}, \end{aligned}$$

and

$$prob^{predlabel}_{local}(P|X,p) = \frac{1}{Z^{predlabel}_{local}(X)} \exp\{ \sum_{f_{P_l} \in F_P} \lambda_{f_{P_l}} f_{P_l}(X, p, P)\},$$

where Z^{id}_{local} and $Z^{predlabel}_{local}$ are normalized factors and are defined below:

$$\begin{aligned} Z^{id}_{local} = \sum_{b_i \in T(B)} \exp\{ & \sum_{f_{B_j} \in F_B} \lambda_{f_{B_j}} f_{B_j}(X, b_i) \\ & + \sum_{f_{PB_k} \in F_{PB}} \lambda_{f_{PB_k}} f_{PB_k}(X, p, P, b_i)\}; \end{aligned}$$

$$Z^{predlabel}_{local} = \sum_{P \in T(x_p)} \exp\{ \sum_{f_{P_l} \in F_P} \lambda_{f_{P_l}} f_{P_l}(X, p, P)\};$$

Normally, a binary label set is enough for argument identification. In our model, we defined two label sets. In our model, we defined two label sets. One is a binary label set $T(B) = \{ARG, NONE\}$ and the other is $T(B') = T(A) \cup \{NONE\}$. $T(A)$ contains all the semantic role labels in the training corpus. It is obvious that the $T(B)$ can be viewed as the subdivision of $T(B')$. In addition, for argument identification classifier with $T(B)$, the features in F_{PB} are not used. Whereas for the classifier with the $T(B')$, another global model is proposed:

$$Pr^{id}_{global}(B'|X) = \frac{1}{Z^{id}_{global}} Pr^{id}_{local}(B'|X) \cdot exp \left\{ \sum_{f_{G_m} \in F_G} \lambda_{f_{G_m}} f_{G_m}(B', X) \right\} \quad (4.8)$$

where f_{G_m} is one of the feature functions defined on global features and $\lambda_{f_{G_m}}$ is its corresponding feature value; Z^{id}_{global} is for normalizing and is defined as:

$$Z^{id}_{global} = \sum_{B' \in B'(X,p)} Pr^{id}_{local}(B'|X) \cdot exp \left\{ \sum_{f_{G_m} \in F_G} \lambda_{f_{G_m}} f_{G_m}(B', X) \right\},$$

and $B'(X, p)$ stands for all the possible label sequences. For the computation convenience, the beam search algorithm was used.

4.3.2 Joint Predicate Sense Disambiguation and Argument Classification

Given the identified argument sequence $B = b_1, \cdots, b_m$, $m \leqslant |x|$, this joint task is to determine such a sequence $S = a_1, \cdots, P, \cdots, a_m = \langle A, P \rangle$. Each a_i is a semantic role for the *ith* word in the sequence and this word is indicated as an argument. i.e. $b_i = ARG$ is determined in argument identification step. P still indicates the sense of the predicate word x_p.

In this model, we propose two sorts of label assignment models: $Pr^{arglabel}_{local}$ and $Pr^{arglabel}_{global}$. And these two models are similar with those mentioned in Chapter 3.

First, we define the label assignment model $Pr^{arglabel}_{local}$ in Equation 4.9 only using local factors. This model is also similar with the argument identification model. In argument identification model ,the final results for predicate sense are ignored.

$$prob^{arglabel}_{local}(S|X,B) = \prod_{i=1, i \neq p}^{m} prob^{arglabel}_{local}(a_i|X,b,P) \cdot prob^{predlabel}_{local}(P|X,p) \tag{4.9}$$

Also, we introduce several new feature vectors. $F_A = \langle f_{A_j} \rangle$ is an argument factor for semantic role assignment of one argument. $F_{PA} = \langle f_{PA_k} \rangle$ is a predicate argument pair-wise factor for semantic role assignment of one argument and predicate sense assignment. We also define their corresponding weight vectors: $\Lambda_{F_A} = \left\langle \lambda_{f_{A_j}} \right\rangle$ and $\Lambda_{F_{PA}} = \left\langle \lambda_{f_{PA_k}} \right\rangle$ respectively. In addition, a global factor $F_G = \langle f_{G_q} \rangle$ is introduced, which is a feature vector for label assignments of all arguments and predicate sense and $\Lambda_{F_G} = \left\langle \lambda_{f_{G_q}} \right\rangle$ is a learned weight or parameter associated with F_G. Using these expressions, $prob^{arglabel}_{local}$ can be represented as :

$$\begin{aligned} prob^{arglabel}_{local}(a_i|X,b_i,P) = \frac{1}{Z^{arglabel}_{local}} exp\{ &\sum_{f_{A_j} \in F_A} \lambda_{f_{A_j}} f_{A_j}(X,b_i,a_i) \\ + &\sum_{f_{PA_k} \in F_P A} \lambda_{f_{PA_k}} f_{PA_k}(X,p,P,b_i,a_i)\} \end{aligned} \tag{4.10}$$

where $Z^{arglabel}_{local}$ is a normalization factor over the identified argument b_i, i.e.,

$$Z^{arglabel}_{local} = \sum_{a_i \in T(A)} exp\{ \sum_{f_{A_j} \in F_A} \lambda_{f_{A_j}} f_{A_j}(X,b_i,a_i) + \sum_{f_{PA_k} \in F_P A} \lambda_{f_{PA_k}} f_{PA_k}(X,p,P,b_i,a_i)\}$$

as for the $prob^{predlabel}_{local}$, it has the same formulation mentioned in Section 4.3.1.

Second, we define a label assignment model $pr^{arguable}_{global}$ using not only feature vectors (F_P, F_{PA}, F_A) but also global feature vector F_G as follows:

$$Pr^{arglabel}_{global}(S|X) = \frac{1}{Z^{arglabel}_{global}} pr^{arglabel}_{local} \cdot exp\left\{ \sum_{f_{G_q} \in F_G} \lambda_{f_{G_q}} f_{G_q}(S,X) \right\} \tag{4.11}$$

Where the normalization factor over the label sequence from local model is defined as:

$$Z^{arglabel}_{global} = \sum_{S \in S(X,p)} pr^{arglabel}_{local} \cdot exp\left\{ \sum_{f_{G_q} \in F_G} \lambda_{f_{G_q}} f_{G_q}(S,X) \right\}$$

4.3.3 Features

The features used in our models are listed in Table 4.1. We use **ID** and **CL** to distinguish features used in argument identification and argument classification steps respectively; we use superscript **B** and **B'** to indicate classifiers with tag sets $T(B)$ and $T(B')$ in the argument identification step.

Table 4.1: Features in our system

Factor	Features	Step
F_P	POS, lemma of the predicate and its parent	IDB ',IDB,CL
	Dependency label of the predicate	IDB ',IDB,CL
	Frameset of the predicate	IDB ',IDB
	Set of the dependency labels of predicate 's dependents	IDB ',IDB,CL
F_A/F_B	POS, lemma of the argument (candidate) and its parent	IDB ', IDB,CL
	Lemma, POS of the leftmost/rightmost dependent/sibling of the argument (candidate)	IDB ',IDB,CL
	Dependency labels of the argument (candidate) and its dependents	IDB ', IDB,CL
	Position of the argument (candidate) with respect to predicate in the dependency tree. Such as parent, child, grandparent et al.	IDB ',IDB,CL
	Location relation between the argument (candidate) and predicate in the sentence with values in before, after, equal	IDB ',IDB,CL

Continued on next page

Table 4.1 – *Continued from previous page*

	Lemma, POS and dependency paths between the argument (candidate) and predicate with the direction of the edge. i.e. N $\uparrow$ VV	IDB	’,IDB,CL
F_{PA}/F_{PB}	Lemma of the current argument (candidate)	IDB	’,CL
	Combination of Lemma and POS of the current argument (candidate)	IDB	’,CL
	POS of the current argument (candidate)	IDB	’
	Dependency label path between the argument (candidate) and the predicate in the dependency tree.	IDB	’,CL
F_G	Sequence of the predicate sense and core argument labels in the structure.	IDB	’, CL

The features in our model can be categorized into four types by the factors (solid squares) in Figure 3.1. Three factors are local factors and each factor is a vector of local features. The last factor is a global factor consisting of global features. Since factors F_A, F_{PA}, F_G have been introduced in Chapter 3 Section 3.2, we just introduce factors F_B and F_{PB} in the following.

F_B

features belong to this type action as scoring a identification assignment for a specified argument and these features do not depend on predicate sense.

F_{PB}

features in this type are used for capturing the local dependencies between a predicate sense and one of its arguments. These features influence both the predicate sense and the argument identification subtasks and make both subtasks can learn from each other.

4.3.4 Parameter Estimation

Unlike the Part-of-Speech(PoS) tagging and word segmentation tasks [27], the feature space in predicate argument structure analysis is much larger. We chose predication-based Passive-Aggressive online learning [9] with parameter averaging technique used in [53] to estimate the weights. Passive-Aggressive(PA) learning is an error-driven learner that shifts weights towards features of the gold solution and away from features of the current guess, whenever the current model makes a mistake.

The PA learning algorithm we used here is the same as that in Algorithm 1. And the $\mathbb{C}$ in this PA algorithm is set to 1.0.

4.4 Our Dual Decomposition Algorithm

Dual decomposition is a method of solving global optimization problems that can be decomposed into efficiently solvable local sub-problems [32]. Under two labels sets $T(B)$ and $T(B')$, there are two argument identification models and their outputs are represented by B and B' respectively. In order to implement joint inference, we iteratively applied exact inferences to the both models, adjusting their potentials to reflect the constraints of the full problem.

4.4.1 Dual Problem Formulation

Before describing the dual decomposition inference for two argument identification model integrations, we restate the inference problem under our joint model.

Firstly, we introduce a boolean matrix C^B. Using this matrix, it is easy to encode the output B: $C^B \in \{0,1\}^{|B|\times|\{ARG,NONE\}|}$. Similarly, we could define a matrix $C^{B'} \in \{0,1\}^{|B'|\times|\{ARG,NONE\}|}$ by mapping $T(B')$ to $\{ARG,NONE\}$ with a simple rule, for any $b_i \in B'$ and $b_i = tag \in T(B')$, if $tag \neq NONE$, then tag is mapped to ARG, this matrix encodes the output from the joint model:

$$B_C = \left\{(i,tag) : C_{i,tag} = 1 \wedge C_{i,tag} \in C^{B'}\right\}.$$

If the word x_i with the label "tag" is included in the output of the B', then $C_{i,tag} = 1$. Otherwise, it equals 0; finally, let I be the index set of all (i,tag) for matrix C, then the maximum likelihood assignment to our original model can be found by optimizing:

$$\begin{aligned} &\underset{B,B'}{argmax}\left(Pr^{id}_{local}\left(B,C^B,X\right) + Pr^{id}_{global}\left(B',C^{B'},X\right)\right) \\ &\qquad \text{such that } \; C^B = C^{B'} \end{aligned} \tag{4.12}$$

The Lagrangian relaxation of this optimization problem is :

$$
\begin{aligned}
L(B', B, u) = Pr^{id}_{global}\left(B', C^{B'}, X\right) + Pr^{id}_{local}\left(B, C^{B}, X\right) \\
+ \sum_{(i,tag)\in I} u(i,tag)\left(C^{B'}(i,tag) - C^{B}(i,tag)\right)
\end{aligned}
\tag{4.13}
$$

If L^* is the optimal value of Equation 4.12, then for any value of u, $L^* = \text{argmax}\, L(B, B', u)$ under the constraints in Equation 4.12. Since $L(u)$ is maximized over a larger space, the dual problem obtains the upper bound $L(u)$.

In this case, the original problem is decomposed into two terms:

$$
\begin{aligned}
\underset{u}{argmin}\Bigg(\underset{B', C^{B'}}{argmax}\left[Pr^{id}_{global}\left(B', C^{B'}, X\right) + \sum_{(i,tag)\in I} u(i,tag) C^{B'}(i,tag) \right] \\
+ \underset{B, C^{B}}{argmax}\left[Pr^{id}_{local}\left(B, C^{B}, X\right) - \sum_{(i,tag)\in I} u(i,tag) C^{B}(i,tag) \right] \Bigg)
\end{aligned}
$$

As in previous work [32], we solve the dual variable u by subgradient optimization methods which are iterative algorithm with updates:

$$
u^{r+1} = u^{r} + \alpha^{r} \cdot \left(C^{\hat{B}^{r}}(i,tag) - C^{\hat{B'}^{r}}(i,tag)\right)
$$

where α^r is a step size. It is obvious that u can be solved by repeatedly performing inference in the argument identification task explained in Section 4.3.1.

4.4.2 Convergence

According to Korte [2], if $\underset{r\to\infty}{lim}\, \alpha^r = 0$ and $\sum_{r=1}^{\infty} \alpha^r = \infty$, the subgradient method can be shown to solve the dual problem:

$$
\lim_{r\to\infty} L(u^r) = \underset{u}{L(u)}.
$$

As mentioned before, the dual provides an upper bound on the primal problem:

$$
\underset{B, B', C^{B} = C^{B'}}{argmax}\left(Pr^{id}_{local}\left(B, C^{B}, X\right) + Pr^{id}_{global}\left(B', C^{B'}, X\right)\right) \le \min_{u\in\mathbb{R}^{\|\mathbb{I}\|}} L(u)
\tag{4.14}
$$

Our dual decomposition algorithm provides an inference method that is exact upon convergence. In this case, the equality in Equation 4.14 is satisfied. When the equality is not satisfied, i.e. when the Algorithm 2 does not converge, the B' can still be used. While B and B' may differ, B' will likely be more similar to B than the assignment of the independent model.

ALGORITHM 2: The parsing algorithm for inter two argument identification classifiers

```
for $r = 1\ to\ \mathbb{IT}$ do
    $\alpha \leftarrow 0.1 \times \frac{1}{r}$;
    $C^{\hat{B'^r}} \leftarrow \underset{B',C^{B'}}{argmax} \left[ Pr^{id}_{global}\left(B', C^{B'}, X\right) + \sum_{(i,tag)\in I} u^r(i,tag) C^{B'}(i,tag) \right]$;
    $C^{\hat{B^r}} \leftarrow \underset{B,C^{B}}{argmax} \left[ Pr^{id}_{global}\left(B, C^{B}, X\right) - \sum_{(i,tag)\in I} u^r(i,tag) C^{B}(i,tag) \right]$ ;
    if $C^{\hat{B'^r}} = C^{\hat{B^r}}$ then;
    return $C^{\hat{B'^r}}$;
    $u^{r+1} \leftarrow u^r + \alpha \cdot \left( C^{\hat{B^r}}(i,tag) - C^{\hat{B'^r}}(i,tag) \right)$
end
return $B_{C^{\hat{B'}^{\mathbb{IT}}}}$;
```

4.4.3 Joint Parsing Algorithm

The joint algorithm for a u that optimizes the argmin $L(u)$ which can be iteratively obtained via sub-gradient descent. Algorithm 2 shows our joint parsing procedure. The learning rate $1/r$ decays with the number of iterations r.

If the algorithm converges, then we have found a u, which optimizes the value of $C^{B'}$ and C^B by B' and B. Hence, it is still a solution to our original optimization problem. However, when the algorithm does not converge, the result sequence B' is alternative for the optimized result of argument identification, because the independent classifier with $T(B')$ has better performance than classifier with $T(B)$. Even in that case, the two results may include some inconsistency, but they will likely be more consistent than that of independent models.

4.5 Experiments and Discussions

4.5.1 Experimental Settings

The dataset setting is the same as that mentioned in Chapter 3 Section 3.7.2. In order to get the syntactic parsing features for Predicate argument structure analysis.

The second order MST parser [4] is trained using the Passive-Aggressive algorithm which is similar to Algorithm 1 in Chapter 3.

The number of iteration for Passive-Aggressive algorithm was set to 5. Since it is difficult to calculate all the possible assignment sequences for global model, we apply beam search to generate N-bests by local model and apply global features for them. A larger N makes the training and calculation expensive and more N local outputs would lead to more local results having the same global features. In our experiments, the value of N is 3.

As for the evaluation, we still use the criteria listed in Chapter 3 Section 3.7.2 which are provided by CoNLL-2009 shared task.

4.5.2 Predicate Argument Analysis and Discussion

Table 4.2 lists the performance of predicate argument analysis in our model and the newly top 3 systems in the latter three rows. The first two rows show when the predicate senses are used to label semantic roles, the performance of the predicate argument analysis can be improved from 77.47 to 78.97. And as expected, the lower recall problem also is improved. Comparing with the SRL result without argument identification step in first row, the recall is improved by 0.2 percent.

Table 4.2: Results on Chinese dataset of CoNLL-2009 shared task

Systems	Sem.P	Sem.R	Sem.F
Without argument identification and *PA* factor	81.5	73.82	77.47
Without argument identification	82.94	75.36	78.97
With dual decomposition	83.31	75.92	79.44
[Björkelund, 2009]	82.42	75.12	78.60
[Meza-Ruiz, 2009]	82.66	73.36	77.73
[Zhao, 2009]	80.42	75.20	77.72

Since the dual decomposition is applied in argument identification step, we can also see that the predicate argument structure analysis can benefit from argument identification task. Also, with dual decomposition method, the result is better than the current best results.

4.5.3 Argument Identification Performance with Dual Decomposition

In order to see how many improvement can be obtained from dual decomposition method, the performance of predicate argument structure analysis is calculated using the unlabeled semantic evaluation metric of the CoNLL-2009 shared task score. It measures the performance of the argument identification task by the unlabeled precision (UP), unlabeled recall (UR) and unlabeled F_1 (UF) score.

$$\text{semantic unlabeled precision } UP = \frac{\text{\# predicates} + \text{\# of correct arguments}}{\text{\#predicates} + \text{\# of returned arguments}}$$
$$\text{semantic unlabeled recall } UR = \frac{\text{\# predicates} + \text{\# of correct argument}}{\text{\#predicates} + \text{\# arguments}}$$
$$\text{semantic unlabeled } UF = \frac{2UP \cdot UR}{UP + UR}.$$

Table 4.3 shows the changes of the performance with different maximum iteration $\mathbb{IT}$ in Algorithm 2.

Table 4.3: Performances on the argument identification task assuming a fixed number of iteration

Iterations	UP	UR	UF
0	90.3	82.03	85.97
1	90.41	82.1	86.06
3	90.48	82.21	86.15
5	**90.55**	82.25	86.2
10	90.44	**82.41**	**86.24**

From these results, it can be seen that the larger $\mathbb{IT}$ is, the better the performance is when $\mathbb{IT}$ is not big enough, which can be explained by the reason that the number of iterations are required for convergence (we donot give the detail about the convergence. This convergence can be reflected by the results with tiny fluctuations); also the dual decomposition method gives a significant gain in recall and precision over the method without argument identification task, which boosts the performance of predicate argument structure analysis on the Chinese test set.

4.6 Summary

In this chapter, we briefly describe some knowledge related dual decomposition inference approach. By applying this inference approach, it provides a possible way that we can use the important information from argument identification subtask, while alleviate the error propagation between argument identification and argument classification subtasks.

As summarizing in Chapter 3, by adding a special label *NONE* to argument classification subtask, the subtask can play the role of identifying the arguments. In this case, the argument classification subtask can also be viewed as argument identification subtask. By introducing linear constraints, we present a dual decomposition algorithm to combine this argument classification subtask and another simpler argument identification subtask. This approach is greatly helpful for improving the performance of argument identification, which is indirectly related to the final selection preference. As seen from the experiments in Section 4.5.2, this approach improves our lower recall problem and we achieved the best result reported so far on the same dataset.

It is noteworthy that the runtime penalty is kept minimal by using dual decomposition, the total time consuming is still considerable since the expensive computation time is required for predicate argument structure analysis. Table 4.4 lists the asymptotic of our models with respect to one prediction with n candidate arguments; each predicate has $|T(P)|$ senses and $\mathbb{IT}$ iterations for dual decomposition. Other symbols have in Table 4.4 have the same definitions as in Section 4.3.1. In the future, we want to apply some pruning techniques to reduce the number of argument candidates.

Table 4.4: Time complexity analysis

Systems	Complexity
Without argument identification	$O(n\|T(P)\|\|T(B')\|)$
With dual decomposition	$O(n\mathbb{IT}\|T(P)\|\|T(B')\|)$

In our method, we just focused on applying the dual decomposition method in argument identification. Ideally, we look forward to discovering the best way to take advantage of this method in the whole predicate argument structure analysis, which can lead to a true joint system. In addition, we also need to conduct feature engineering experiments to figure out which features are useful for predicate argument structure analysis.

Chapter 5

Exploring Shallow Parsing Information for SRL

The bulk of previous work on automatic SRL has primarily focused on using full constituent parse of sentences to define argument boundaries and to extract relevant information for training classifiers. However, there have been some attempts at relaxing the necessity of using syntactic information derived from full parse trees. Sun et.al [46, 45] addressed the Chinese SRL problem on the basis of shallow syntactic information at the level of phrase chunks. In their approach, Chinese SRL is formulated as a sequence labeling problem, performing IOB2 decisions on the syntactic chunks of a sentence. However, this method ignores the full syntactic parsing information entirely, and we believe that even the accuracy of full syntactic parsing is not ideal, it is still helpful for SRL. Moreover, their method is inapplicable to syntactic dependency based SRL since a chunk usually consists of successive words. Little is known how the SRL over word units performs with shallow syntactic information.

In this chapter, we first implemented a SRL system, which only used shallow syntactic related information, to see how the SRL over word units performances without any full dependency syntactic information. Then we added these shallow syntactic related features to our syntactic dependency based SRL and observed how shallow syntactic related information influence the SRL results.

5.1 Shallow Parsing-based SRL

Shallow parsing-based SRL in our method is formalized as the task of assigning argument role labels and predicate sense to word units without using dependency information but use shallow syntactic related information.

5.1.1 Chinese Shallow Parsing

Chinese shallow parsing has been researched for years and a variety of chunk definitions have been proposed. In our system, we used the chunk definition presented in [6] which also provided a chunk extraction tool[1]. This tool extracted chunks from Chinese Tree Bank (CTB) and was developed by modifying the English tool[2] used in the CoNLL-2000 shared task. We also use the sentence in Figure 2.1 to illustrate the definition of chunks in line **CH** in Figure 5.1. In this example, " 金融工作" (finance work) is a noun phrase and is composed by two nouns.

WORD	去年	西藏	金融	工作	取得	显著	成绩
POS	NN	NR	NN	NN	VV	JJ	NN
CH	[NP]	[NP]	[NP	]	[VP]	[ADJP]	[NP]
TAG	B-NP	B-NP	B-NP	I-NP	B-VP	B-ADJP	B-NP
SRL	TMP	NONE	NONE	A0	取得.01	NONE	A1

Figure 5.1: Example under shallow parsing-based SRL.

The problem of Chinese chunking has been regarded as a sequential labeling task in previous research. As the Inside/Outside representation for proper chunks that was first introduced in [40], we use the following set of three tags for presenting proper chunks which can be reflected by **TAG** line in Figure 5.1 .

B	Current word is the beginning of a chunk.
I	Current word is inside of a chunk.

[1] `http://www.nlplab.cn/chenwl/chunking.html`
[2] `http://ilk.uvt.nl/team/sabine/chunklink/chunklink\_2-2-2000\_for\_conll.pl`

O Current word is outside of a chunk.

We carried out Chinese chunking task with software package CRF++[3]. Unlike hidden Markov models(HMMs), it can capture many correlated features of the inputs. As this package has a single exponential model for the joint probability of the entire paths given the input sentence, it overcomes the label bias problem in maximum entropy Markov models (MEMMs). To illustrate conveniently, we denote a word in focus with a fixed window $x_{-2}x_{-1}x_0x_{+1}x_{+2}$, where x_0 is the current word. The feature template used is defined as:

- Uni-gram word/POS tag features: x_{-2}, x_{-1}, x_0, x_{+1} and x_{+2}.
- Bi-gram word/POS tag features: $x_{-2}x_{-1}$, $x_{-1}x_0$, x_0x_{+1} and $x_{+1}x_{+2}$.

5.1.2 Features

In Figure 5.1, words, labeled as *NONE* in the line *SRL*, are not arguments for the predicate: 取得 (get) and the predicate sense is represented as 取得.01. Obviously, words in chunks do not have equal importance for SRL. Headwords represent the main meaning of the chunks. The base phrase chunking related features are only applied to these headwords. The rules described in Sun and Jurafsky [44] are used to extract headwords as listed in Table 5.1. For non-headwords, only the lemma and POS information is used.

Table 5.2 shows the chunking related features for headwords. These features also meet the classification definition in Chapter 3 Section 3.2. Verb Class in Table 5.2 is represented similarly as *Verb.C1C2*, which means this *verb* has two senses. For its first sense, it has one core argument and for its second sense, it has two core arguments. These verb classes are extracted from Chinese PropBank [55].

Table 5.2: Features in shallow parsing-based Chinese SRL

Predicate Factor	
POS	part of speech of the predicate part of speech of the words that immediately precede and follow the predicate

Continued on next page

[3]http://crfpp.sourceforge.net/

Table 5.2 – *Continued from previous page*

Verb Class	the verb classification according to its framesets annotated in CPB
Lemma	lemma of the predicate lemma of the words that immediately precede and follow the predicate
	Argument Factor
Chunk	chunk tag of headword with IB representation (e.g. $B-NP$) chunk tag of the chunk where the headword belongs to the number of words in a chunk the POS sequence of words in a chunk, for example, for "金融工作" (finance work) is "NN_NN" the position of the chunk with respect to the predicate(Position) the conjunctions of Position and headword, predicate and verb class the conjunctions of Position and POS of headword, predicate and verb class lemma/POS of one word immediately before/after of the chunk
Path	a chain of chunk types between the headword and the predicate the length of the chunk chain between the headword and the predicate
Predicate	POS tag/lefomma of the predicate verb class of the predicate verb formation of the predicate[4]
	Predicate-Argument Pairwise Factor
Lemma	lemma of the current argument
POS	part of speech of the argument
Lemma and POS	combination of the lemma and part of speech of the argument

Continued on next page

[4]Sun [46] defined 9 kinds of word formation for Chinese compound words and put forward an algorithm to abstract them. This information is shown very useful for Chinese SRL.

Table 5.2 – *Continued from previous page*

Global Factor	
Semantic Sequence	the core semantic label sequence with the sense of the predicate in consistent with their location in the sentence.

5.2 Shallow Parsing Information for Syntactic Dependency based SRL

As the syntactic dependency based SRL models in Chapter 3 can achieve competitive results compared to the state-of-the-art systems, we still use the same models in this chapter.

5.3 Results and Discussions

5.3.1 Experimental Setting

We use the same dataset as that in Chapter 3. In order to be convenient for comparison, two kinds of dependency parsing results are provided, the first is from MALT parser, the second is from second-order MST parser [5] [4]. For this parser, McDonald et al [28] modified the MIRA learning algorithm for structured domains in which the optimization problem can be solved by using the Hidreth's algorithm [5]. It is obvious that using such an optimization technique is more expensive than not using an optimization technique. Thus, instead of the modified MIRA, we adopted the Passive-aggressive algorithm that similar to Algorithm 1. Table 5.3 compares its performance with given MALT dependencies.

Most of features templates are "standard" which have been widely used in previous dependency-based SRL research and has been listed in Chapter 3, we do not explain "standard" features here. As for the base phrase chunking related features, they are the same as shown in Table 5.2.

[5]`http://www.seas.upenn.edu/~strctlrn/MSTParser/MSTParser.tar.gz`

Table 5.1: Head rules for Chinese chunks

Parent	Direction	Priority List
ADJP	Right	ADJP JJ AD
ADVP	Right	ADVP AD CS JJ NP PP P VA VV
CLP	Right	CLP M NN NP
CP	Right	CP IP VP
DNP	Right	DEG DNP DEC QP
DVP	Right	DEV AD VP
DP	Left	M(r) DP DT OD
IP	Right	VP IP NP
LST	Right	CD NP QP
LCP	Right	LCP LC
NP	Right	NP NN IP NR NT
PP	Left	P PP
PRN	Left	PU
UCP	Left	IP NP VP
VCD	Left	VV VA VE
VP	Left	VE VC VV VNV VPT VRD VSB VCD VP
QP	Right	QP CLP CD
VPT	Left	VA VV
VRD	Left	VVI VA
VSB	Right	VV VE

Parsers	labeled attachment score(LAS)	unlabeled attachment score(UAS)	Label accuracy score(LA)
MALT parser	78.46	80.70	85.45
MST parser	82.00	84.90	88.57

Table 5.3: Dependency performance comparison among MALT parser and second-order MST parser on the test corpus.

5.3.2 SRL Performance

Still using the semantic evaluation metric of the CoNLL-2009 shared task scorer, Table 5.4 gives the comparison of SRL performance before and after adding the proposed base phrase chunking related features on the test data. Lines with $-/+$ show the SRL performance without/with base phrase chunking related features. As seen on this table, without gold dependency parse, the best SRL is up to 80.52 in F_1 score. To the best of author's knowledge, there are few Chinese SRL results more than 80%.

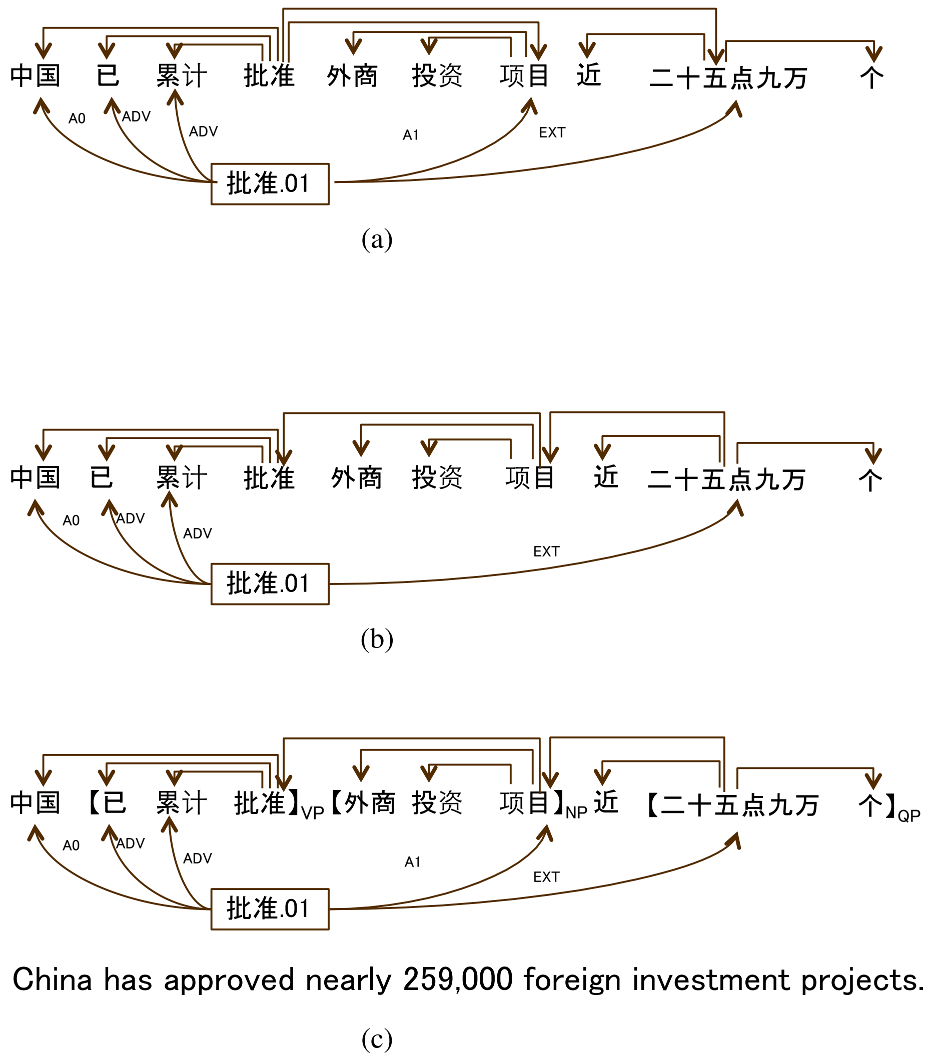
	P(%)	**R(%)**	**$\mathbf{F_1}$(%)**
Gold parsing $-$	88.68	86.30	87.47
Gold parsing $+$	**90.03**	**87.71**	**88.86**
MALT $-$	82.64	72.68	77.34
MALT $+$	**84.17**	**74.67**	**79.13**
MST-2 $-$	83.01	75.39	79.02
MST-2 $+$	**84.49**	**76.92**	**80.52**
$+$	80.77	66.62	73.02

Table 5.4: SRL results without/with base phrase chunking information.

The last row in Table 5.4 also shows the SRL performance without any dependency syntactic information, but only with base phrase chunking related features. Comparing the results only with dependency syntactic features, it confirmes our belief that even the accuracy of full syntactic parsing is not ideal, it is still helpful for SRL. And to our best knowledge, it is also the first to apply shallow parsing information to word units for SRL.

On the other hand, comparing the lines with $-$, it shows dependency parsing play the central role in Chinese SRL as most research does. Comparing their corresponding lines with $+$, Chinese SRL can still benefit a lot from shallow parsing information. An example from the corpus is shown in Figure 5.2.

Figure 5.2a shows the gold dependency parsing result and the gold predicate argument structure; Figure 5.2b shows the dependency parsing result from MALT parser and the predicate argument structure as a result of the predicted parse; Figure 5.2c shows the predicate argument structure which is predicted after adding base phrase chunking related features. In Figure 5.2c, the subscripts stand for chunk types. From Figure 5.2b, it can be seen that the argument $A1$ is not identified by the dependency based SRL because of dependency errors. Comparing Figure 5.2b and 5.2c, we can



中国 已 累计 批准 外商 投资 项目 近 二十五点九万 个
A0
ADV
ADV
A1
EXT
批准.01
(a)
中国 已 累计 批准 外商 投资 项目 近 二十五点九万 个
A0
ADV
ADV
EXT
批准.01
(b)
中国 【已 累计 批准】VP【外商 投资 项目】NP 近 【二十五点九万 个】QP
A0
ADV
ADV
A1
EXT
批准.01
China has approved nearly 259,000 foreign investment projects.
(c)

Figure 5.2: An example that the argument prediction error brought by MALT parse errors is corrected by introducing base phrase chunking related features.

see that after adding the base phrase chunking related features, this SRL error brought by dependency parsing errors is corrected.

Line MALT+ and line MST-2− show that even the dependency parsing result from MALT is not better than that from second order MST, with the aid of chunking related features, Chinese SRL can still get comparable results.

5.4 Summary

In this chapter, we first explored the SRL performance over word units by the shallow parsing based SRL that did not use any dependency syntactic information. Then added these base phrase chunking related features to a dependency based SRL system and investigated the benefit that our Chinese SRL model can get from them. Evaluations on the corpus show that chunking information well complements dependency based SRL. With these features, the F-measure of our dependency based SRL reached more than 80. However, without any dependency syntactic information, the performance of shallow parsing based SRL can not compete with the performance of dependency based SRL.

Chapter 6

Exploring Additional Dependency Information for SRL

6.1 Introduction

In recent years, SRL has become an important component in many kinds of deep natural language processing applications, such as question answering [34], event extraction [41], document categorization [36]. SRL aims at identifying the semantic relations between predicates in a sentence and their associated arguments, with these relations drawn from a pre-specific list of possible semantic roles for corresponding predicates. Syntax information is essential in SRL systems. To date, both constituent parsing and dependency parsing based SRL have been investigated [55, 21], with dependency based systems giving superior results in CoNLL 2008 [48] and CoNLL 2009 shared tasks [19].

However, the performance gap is still quite large between SRL systems using oracle ”perfect” dependency parses and SRL systems using automatic dependency parses. We observe as much as 10% F-score difference in our experiments. Clearly, errors in the 1-best dependency parse affects SRL prediction. This leaves an open question: in order to improve dependency based SRL, is it more worthwhile to incorporate more dependency information (in the form of N-best parse), or to incorporate an entirely separate source of information, such as base phrase chunks? We perform such an analysis in this paper, using the state-of-the-art Chinese SRL system in Chapter 3.

In the following, we first describe related work in Section 6.2. Then present how to get N-best dependency parsing results and how to apply the N-bests to SRL in Section 6.3. The experimental results and discussions are shown in Section 6.4. Furthermore, combining the chunking related features in Table 5.2, we also explored which information is important for SRL and answered the open question by the experimental results. Finally, our work is concluded in Section 6.5.

6.2 Related Work

A substantial amount of research has focused on dependency-based SRL [30, 27] since the CoNLL-2009 shared task and rich linguistic features [57] are applied. For dependency related features, most studies focused on extracting them from the best dependency result. Johansson and Nugues [21] tried to use N-best dependency parsing results. In their work, they applied 16-best dependency trees to generate predicate-argument structures and applied both syntactic trees and predicate-argument structures to a linear model. This model reranked the predicate-argument structures and the top 16 dependency trees at the same time. Though their work suggests that N-best dependency parsing can enhance the SRL, little is known about how the N-best dependency parsing related features perform on SRL.

6.3 Dependency Parsing

Dependency parsing has been researched for years and a variety of methods have been proposed [56, 14]. In this chapter, we used the MSTParser [28, 29] which translates the problem of dependency parsing into finding maximum spanning trees for directed graphs. Since in our experiments, we used the second-order spanning tree parsing and we will focus on introducing it.

6.3.1 First-order Spanning Tree Parsing

Given a generic input sentence $x = x_1, \cdots, x_n$, if y represents a generic dependency tree for sentence x and contains a set of edges like $(i, j) \in y$ if there is a dependency in y from word x_i to x_j.

If the score of an edge can be calculated as the dot product between a high dimensional feature representation of the edge and a weight vector

$$s(i, j) = \omega \cdot f(i, j)$$

then the score of a dependency tree y for sentence x can be calculated as the sum of the scores of all edges in the tree.

$$s(x,y) = \sum_{(i,j)\in y} s(i,j) = \sum_{(i,j)\in y} \omega \cdot f(i,j)$$

And dependency parsing is the task of finding the dependency tree y with highest score for a given sentence x.

6.3.2 Second-order Spanning Tree Parsing

Different from first-order spanning tree parsing, the score of a tree is the sum of adjacent edge pairs. To quantify this, consider the example from Figure 2.3, with words indexed: root(0) 去年 (1) $\cdots$ 成績 (7). Under the first-order spanning tree formulation, the score of this tree would be,

$$s(0,5)+s(5,1)+s(4,2)+s(4,3)+s(5,4)+s(5,7)+s(7,6).$$

However, in the second-order spanning tree model, the score of this tree would be,

$$\begin{aligned} s(0,-,5)+s(5,-,1)+s(4,-,2)+s(4,2,3) \\ +s(5,-,4)+s(5,-,7)+s(7,-,6). \end{aligned}$$

Here the score function is $s(i,k,j)$, which is the score of creating a pair of adjacent edges, from word x_i to word x_k and x_j. For instance, $s(4,2,3)$ is the score of creating dependency edges from 工作 to 西藏 and from 工作 to 金融. The score functions are relative to the left or right of the head, but can not on different sides of the head. This left/right dependence makes it is possible to define polynomial second-order projective parsing algorithms. For example, if the word x_i has the modifiers as shown, the score can be defined:

$$\begin{aligned} s(x_i,-,x_{i_1}) + \sum_{k=1}^{i_{j-1}} s(x_i,x_{i_k},x_{i_{k+1}}) \\ +s(x_i,-,x_{i_{j+1}}) + \sum_{k=j+1}^{i_{m-1}} s(x_i,x_{i_k},x_{i_{k+1}}) \end{aligned}$$

Eisner [14] proposed a first-order algorithm in $O(n^3)$. This algorithm parses the left and right dependents of a word independently and combines them at a later stage. McDonald and Pereira [29] extended this algorithm to second-order case. Figure 6.1 illustrates the extended algorithm. For this algorithm, the key insight is to delay completion of items until all the dependents of the head have been gathered.

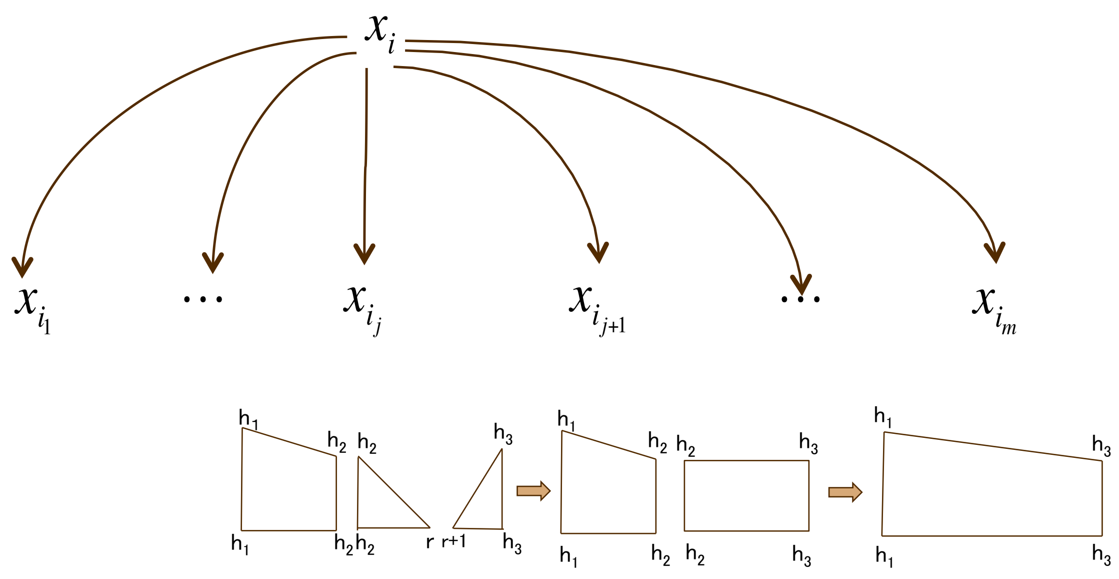


Figure 6.1: An extension of the Eisner algorithm to second-order dependency parsing.

6.3.3 Online Learning

Online learning algorithm have been shown to be robust in structure learning. McDonald et al [28] modified the MIRA learning algorithm for dependency parsing. Instead of this optimization technique that is more expensive than not using an optimization technique, the Passive-aggressive algorithm is adopted. Algorithm 3 illustrates the details.

6.3.4 Error Analysis for Dependency-based SRL

Using the gold parse of dependency relations between a predicate and its arguments and according to these relations, we classified SRL errors into following three types, and used C, G, O to represent them separately.

- C: children of a predicate should be arguments but they are tagged incorrectly.
- G: grand children of a predicate should be arguments but they are tagged incorrectly.
- O: arguments are incorrectly tagged and these arguments are not children or grandchildren of a predicate

ALGORITHM 3: The Passive-Agressive Algorithm for Dependency Parsing

```
Input: Training set $\tau = \{x_t, y_t\}_{t=1}^{\mathbb{T}}$, maximum iteration value $\mathbb{IT}$ and
       regularization parameter $\mathbb{C}$.
Output: The parameter $\omega$ that minimizes the hinge-loss function and the margin
        of learning data $\Delta(y, \hat{y})$ .
$\omega \leftarrow 0$, $\upsilon \leftarrow 0$, $c \leftarrow 0$;
for $itr$ to $\mathbb{IT}$ do
    for each $(x_t, y_t)$ in $\tau$ do
        $\hat{y}_t = \underset{y}{argmax}\ \omega \cdot F(x_t, y)$;
        $\sigma_i = min\left(\mathbb{C}, \frac{\Lambda \cdot F(x_t, \hat{y}_t) - \Lambda \cdot F(x_t, y_t) + \Delta(y_t, \hat{y}_t)}{\|F(x_t, y_t) - F(y_t, \hat{y}_t)\|^2}\right)$;
        $\omega \leftarrow \omega + \sigma_j(F(x_t, y_t) - F(x_t, \hat{y}_t))$;
        $\upsilon \leftarrow \upsilon + c\sigma_j(F(x_t, y_t) - F(x_t, \hat{y}_t))$;
        $c \leftarrow c + 1$;
    end
end
return $\omega = \omega - \upsilon / c$;
```

C	9.59%	2,072/21,604
G	90.85%	139/153
O	43.93%	2,626/5,955

Table 6.1: The distribution of SRL errors on Test corpus by the joint model.

Table 6.1 shows the distribution of three errors observed in the test corpus(see Section 5.3.1) after tagging by our joint model. For example, there are a total of 21,604 arguments that are children of predicates and among them, and 2,072(9.59%) are errors.

6.3.5 Features from N-best Dependency Parsing

According to the statistics of corpus, it is found that about 77.96% arguments are children of predicates. If we can reduce the head errors for dependents, the C errors caused by dependency parsing errors should be decreased, and SRL tagging results

N-best	P(%)	R (%)	F_1(%)
1	**83.01**	75.39	79.02
3	82.52	**77.16**	79.75
5	82.74	77.10	**79.82**
10	82.44	76.98	79.62

Table 6.2: SRL results with N-best dependency parsing related features.

would be improved. Under this hypothesis, we extracted the following features from N-best dependency parsing results. These features are also included in the ”standard ” feature set when N = 1. For the ”standard” features which has been listed in Chapter 3, we do not explain once again.

Arguments′ heads: lemma/pos; lemma and pos; dependency label; whether are predicates.

Position: position of the argument candidates with respect to the predicate position in the tree; position of the heads of the argument candidates with respect to the predicate position in the sentence.

Chain: the left-to-right chain of the dependency labels of the predicate's dependents.

6.4 Experiments and Discussions

The experimental settings are the same as the settings in the former chapters.

6.4.1 Performance

Table 6.2 shows the Chinese SRL results after adding the N-best dependency parsing related features. It is not surprising that SRL can get better performance when $N > 1$, because the larger N, a more accurate dependency parsing results can be likely obtained. When $N = 5$, SRL gets the best performance 79.82 in F_1 with 0.8 point improvement.

Figure 6.2a shows the gold dependency parsing result and the gold predicate argument structure; the dependency parsing result in Figure 6.2b is from second-order MST parser, and the predicate argument structure in this figure is predicted based on it; Figure 6.2c shows SRL result after adding 5-best dependency related features. From

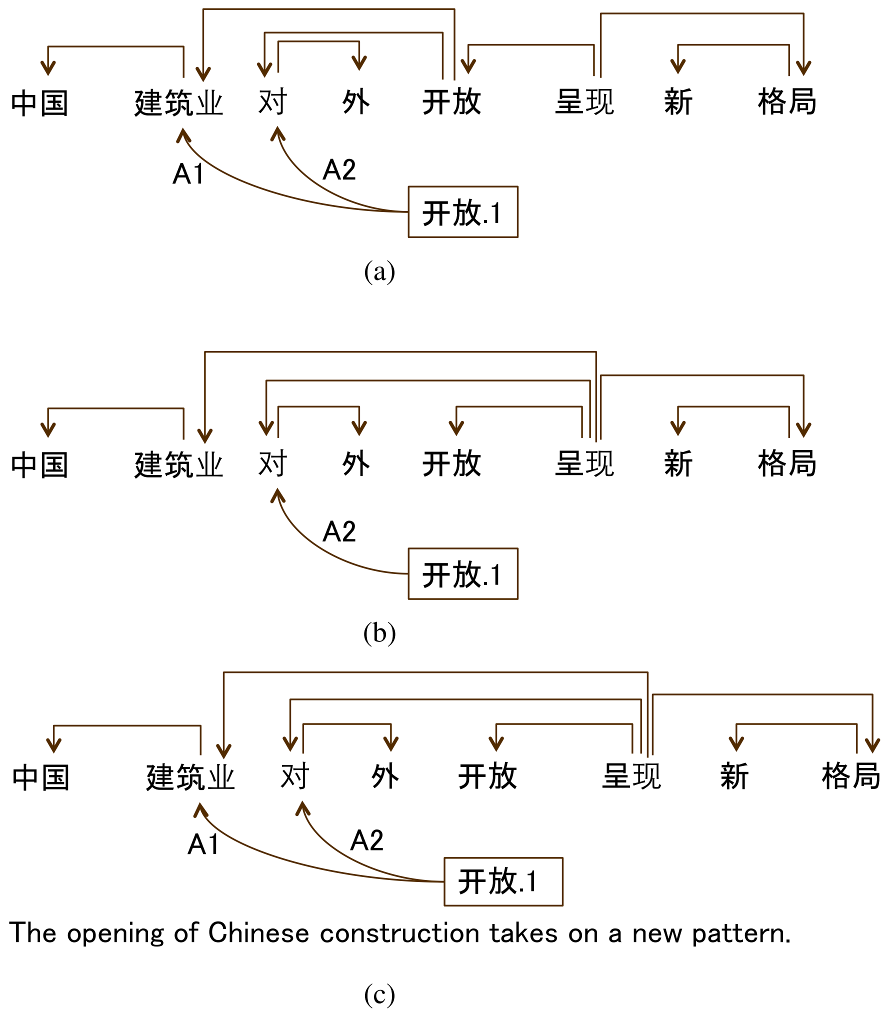

中国 建筑业 对 外 开放 呈现 新 格局
A1
A2
开放.1
(a)
中国 建筑业 对 外 开放 呈现 新 格局
A2
开放.1
(b)
中国 建筑业 对 外 开放 呈现 新 格局
A1
A2
开放.1
The opening of Chinese construction takes on a new pattern.
(c)

Figure 6.2: An example that the argument prediction error brought by second order MST parse errors is corrected by introducing N-best dependency related features.

N-best	Correct (#)	Error(#)	Noise(#)
1	18,428	3,176	-
3	19,071	2,533	4,636
5	19,392	2,212	5,667
10	19,738	1,866	7,699

Table 6.3: Dependency accuracy and the noise changes with different N.

this figure, we can see that the N-best dependency related features helped in correcting the wrong argument prediction.

However, the improvement declines when $N = 10$. A larger N may result in adding more accurate dependency parsing, however, it can also result in including more noises. For the MST parser using second order algorithm, Table 6.3 shows how the choice of the value of N affects the dependency parsing. The Correct(#) column represents the number of cases where the correct parent of an argument is predicted within the N-best. For example, in 3-best, it counts the number of arguments where their parents are correctly predicted in at least one of the 3 predictions. In the case where the parent is not predicted in any tree, they are counted as an error, as listed in the second column. The third column (Noise), is defined under a hypothesis: correct dependency relations generate correct SRL results, wrong dependency relations generate incorrect SRL results. It represents the number of wrong dependency relations in Correct case which can cause bad influence for SRL results. For example, if 3 best heads for an argument are top-1, top-2, top-3 respectively, and top-1 is the correct one, then this case is a Correct case and the number of noise are 2; if none of the three results are correct, then this case is an Error case, and no noise. From this table, it obviously indicates that the benefit for dependency parsing brought by a larger N is less than the noise brought by the N.

Now, it is time to answer the open question: in order to improve dependency based SRL, is it more worthwhile to incorporate more dependency information in the form of N-best parse, or to incorporate base phrase chunks information ? Table 5.4 and 6.2, it can be seen that SRL benefit more from chunking related features than from N-best parse related features.

Table 6.4 shows the the results of Chinese SRL after adding base phrase chunking information and N-best parsing related features and gives the comparison with the previous work. From Table 6.2 and 6.4 we can see that after adding the chunking

	N-best	P(%)	R (%)	F_1(%)
[Björkelund, 2009]	-	82.42	75.12	78.60
[Meza-Ruiz, 2009]	-	82.66	73.36	77.73
[Zhao, 2009]	-	80.42	75.20	77.72
MST-2 +	1	**84.49**	76.92	80.52
MST-2 +	3	83.81	**78.51**	**81.07**
MST-2 +	5	83.71	78.40	80.97

Table 6.4: SRL results with base phrase chunking information and N-best parsing related features.

	N-best	C(%)	G(%)	O(%)
MST-2−	1	25.37	86.93	59.06
MST-2−	3	22.83	78.43	56.84
MST-2−	5	22.83	78.43	57.36
MST-2+	1	23.93	86.93	54.70
MST-2+	3	21.5	76.47	53.05
MST-2+	5	21.66	76.47	53.38

Table 6.5: SRL error changes with different features

related features, the impact of N-best parsing related features is a little reduced.

6.4.2 Discussion

In Section 6.4.1, we see that both chunking and N-best parsing related features are helpful for Chinese SRL to some extent. In order to understand how they affect SRL, we analyze the results from three types of errors introduced in Section 6.3.4. Table 6.5 shows the error changes when different features are added.

Since accurate dependency information is not always available, the three types of errors should become larger when automatic dependency parsers are used. From Table 6.1 and 6.5, the *C* and *O* errors increased as expected, while *G* decreased, the main reason is that arguments, that are grandchildren of predicates, are relocated in the dependency trees because dependency errors, and these locations make them easier to be tagged. From first and fourth rows, it suggests that shallow parsing information are helpful to reduce the *C* and *O* errors. Comparing the fourth line with second and third rows, it explains why SRL achieving more improvements from chunking than from N-best dependency. When N changed from 1 to 3, the errors decreased obviously, however, when the $N = 5$, there are no obviously different changes.

6.5 Summary

In this paper, we first introduced how to get the N-best dependency parsing results and explored how to use this information to SRL. Then combined these N-best dependency parsing related features with base phrase chunking related features to investigate the benefit that our Chinese SRL model can get from them. Evaluations on the CoNLL 2009 Chinese corpus show that chunking information well complements dependency based SRL, achieving more improvements compared to N-best dependency information. With those additional features, our dependency based SRL achieves the best result on the same Chinese corpus to our knowledge.

Chapter 7

Conclusions

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

7.1 Conclusions

This dissertation explored which models and what information are helpful for dependency based semantic role labeling. Our contributions by this thesis have the following characters.

- We proposed a log linear model-based structure model that joint implement predicate sense disambiguation and semantic role labeling simultaneously: This model can capture dependencies underlying in predicate argument structure: including dependencies between arguments, dependencies between predicate and its corresponding arguments under its some predicate sense. We conducted experiments for our and a pipeline dependency based SRL models using the same feature sets. Comparing the performance of the pipeline model, our model achieved performance improvement and achieved competitive results compared to the state-of-the-art systems which use time-consuming feature selection algorithms.

- We applied dual decomposition inference to construct a pipeline syntactic dependency based SRL: In this system, we proposed a dual decomposition algorithm which can greatly improve the performance of argument identification and indirectly improve the final selection performance. In the evaluation experiments, the system achieved the best performance reported so far on the same dataset. And by using the proposed dual decomposition algorithm, the runtime penalty is kept minimal.

- We focused on investigating the benefit that SRL can get from shallow parsing: Firstly, we got chunking information by CRF model; then explored the SRL performance by the shallow parsing based SRL which applied the chunking related features only. Finally, we applied these base phrase chunking related features together with the dependency related features to our Chinese SRL model and observed the SRL performance. From experiments, it can be seen that chunking related features can well complement dependency based SRL. However, shallow parsing based SRL can not compete with syntactic dependency based SRL over the word units.

- We discussed the use of N-best dependency parsing related features for syntactic dependency based SRL: We first explored the influence for SRL brought by adding several N-best parsing related features to our dependency based SRL system. In addition to these features, we also added the chunking related features used in previous chapter. With these features, the F_1 value of our dependency based SRL is more than 80%. From experiments, it can also seen that chunking information nicely complements syntactic dependency based SRL, achieving more improvement compared to N-best dependent information.

While all our experiments are for Chinese, it is possible to design experiments for other languages with our approaches.

7.2 Future Directions

There are several directions we would like to try.

7.2.1 Exploring More Features for SRL

Our experiment results show that we are not limited to increasing SRL performance via more accurate syntactic parsing, but that we can explore other information, which is easier to get and is helpful for SRL. This also determines our future work. In our future work, we would like to explore more features and their influence for SRL.

7.2.2 Incorporating Unlabeled Data

All the models for predicate argument structure analysis in our work are supervised methods which only use labeled data. As it is known, it is very expensive to prepare labeled data for each predicate. Recently, semi-supervised learning methods have been explored in a number of work, like word segmentation [54], pos tagging [51], dependency parsing [49, 24] and so on.

Recently, semi-supervised learning methods for semantic role labeling [15, 12] have attracted much researchers' attention. But there is still room for further improvements of predicate argument structure analysis using unlabeled data.

7.2.3 Apply SRL to Other NLP Applications

One possible way may be exploring opinion mining of product reviews based on predicate argument structure information. Online product reviews are becoming increasing available. In order to make an informed decision, potential customers usually prefer to referring to a lot of online reviews. It is worth to explore whether SRL can help to understand consumer reviews.

Appendix A

List of Publications

Journal Papers

- Luo Yanyan, Asahara Masayuki, Matsumoto Yuji. ”Robust Integrated Models for Chinese Predicate-Argument Structure Analysis”. China Communications, Vol.9, No.3, 10-18, 2012. (Chapter 5).

International Conference Papers

- Yanyan Luo, Masayuki Asahara and Yuji Matsumoto. ”Dual decomposition method for Chinese predicate-argument structure analysis”. In Proceedings of the 7th International Conference on Natural Language Processing and Knowledge Engineering (NLPKE’11). 409-414. 2011. (Chapter 5).
- Yanyan Luo, Kevin Duh and Yuji Matsumoto. ”What information is helpful for dependency based semantic role labeling”. International Joint Conference on Natural Language Processing (IJCNLP), 2013. Accepted. (Chapters 6 and 7).

Awards

- Best paper award of the 7th International Conference on Natural Language Processing and Knowledge Engineering, 2011. ”Dual decomposition method for Chinese predicate-argument structure analysis”

Bibliography

[1] Kurt M. Anstreicher and Laurence A. Wolsey. Two ”well-known” properties of subgradient optimization. *Mathematical Programming*, 120(1):213–220, August 2009.

[2] Jens Vygen Bernhard Korte. *Combinatorial Optimization:Theory and Algorithm*. Springer, 3rd edition, 2008.

[3] Anders Björkelund, Love Hafdell, and Pierre Nugues. Multilingual semantic role labeling. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning(CoNLL)*, pages 43–48, Boulder,Colorado, June 2009.

[4] 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*, pages 957–961, Prague, June 2007. ACL.

[5] Yair Censor and Stavros A. Zenios. *Parallel optimization:theory, algorithms and applications*. Oxford University Press, 1997.

[6] Wenliang Chen, Yujie Zhang, and Hitoshi Isahara. An empirical study of Chinese chunking. In *Proceeding Proceedings of the COLING/ACL 2006 on Main conference poster sessions*, pages 97–104, Sydney, Australia, July 2006.

[7] Trevor Cohn and Philip Blunsom. Semantic role labeling with tree conditional random fields. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 169–172, 2005.

[8] Michael Collins. Discriminative training methods for hidden markov models:theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, pages 1–8, July 2002.

[9] 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.

[10] Hal Daumé. *Practical structured learning techniques for natural language processing*. PhD thesis, University of Southern California, Los Angeles, Canada, August 2006.

[11] John DeNero and Klaus Macherey. Model-based aligner combination using dual decomposition. In *Proceeding of the 49th Annual Meeting on Association for Computational Linguistics*, pages 420–429, Portland, Oregon, June 2011.

[12] 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: Volume 1-Volume 1*, pages 21–29. Association for Computational Linguistics, 2009.

[13] David Dowty. Thematic proto-roles and argument selections. *Language*, 67(3):574–619, September 1991.

[14] Jason Eisner. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistic*, pages 340–345, 1996.

[15] Hagen Fürstenau and Mirella Lapata. Semi-supervised semantic role labeling. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 220–228. Association for Computational Linguistics, 2009.

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

[17] Daniel Gildea and Martha Palmer. The necessity of syntactic parsing for predicate argument recognition. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 239–246. Association for Computational Linguistics, 2002.

[18] Hadri Hacioglu. Semantic role labeling using dependency trees. In *Proceeding of the 20th International Conference on Computer Linguistics*, 2004.

[19] Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antòia 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*, pages 1–18, Boulder,Colorado, June 2009.

[20] Richard Johansson and Pierre Nugues. Dependency-based semantic role labeling of propbank. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 69–78, October 2008.

[21] Richard Johansson and Pierre Nugues. Dependency-based syntactic-semantic analysis with propbank and nombank. In *Proceeding of the Twelfth Conference on Computational Natural Language Learning*, pages 183–187, August 2008.

[22] Daisuke Kawahara, Nobuhiro Kaji, and Sadao Kurohashi. Question and answering system based on predicate-argument matching. In *Proceedings of the Third NTCIR Workshop. 2002*, 2002.

[23] Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. Mrf energy minimization & beyond via dual decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):531–552, March 2011.

[24] Terry Koo, Xavier Carreras, and Michael Collins. Simple semi-supervised dependency parsing. In *Proc. of ACL-08: HLT*, pages 595–603, Columbus, USA, June 2008.

[25] Terry Koo, Alexander M.Rush, Michael Collins, Tommi Jaakkola, and David Songtag. Dual decomposition for parsing with non-projective head automata. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1288–1298, Cambridge, Massachusetts, October 2010.

[26] Ding Liu and Daniel Gildea. Semantic role features for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics. Association for Computational Linguistics*, pages 716–724, August 2010.

[27] Yanyan Luo and Gegen Huang. Chinese word segmentation based on the marginal probabilities generated by crfs. *Journal of Chinese Information Processing*, 23(5):3–8, 2009.

[28] Ryan McDonald, Koby Crammer, and Fernando Pereira. Online large-margin training of dependency parsers. In *Proceeding of the 43th Annual Meeting on Association for Computational Linguistics*, pages 91–98, 2005.

[29] Ryan McDonald and Fernando Pereira. Online learning of approximate dependency parsing algorithms. In *Proceedings of the 11th Conference of European Chapter of the Association for Computational Linguistics*, pages 81–88, Trento,Italy, April 2006. ACL.

[30] Ivan Meza-Ruiz and Sebastian Riedel. Jointly identifying predicates, arguments and senses using markov logic. In *Proceedings of Human Language Technology Conference/North American Chapter of the Association for Computational Linguistics(HLT/NAACL-2009)*, pages 155–163, Boulder,Colorado, 2009.

[31] Alexander M.Rush and Michael Collins. A tutorial on dual decomposition and lagrangian relaxation for inference in natural language processing. *Journal of Artificial Intelligence Research*, 45:305–362, 2012.

[32] Alexander M.Rush, David Songtag, Michael Collins, and Tommi Jaakkola. On dual decomposition and linear programming relaxations for natural language processing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1–11, Cambridge, Massachusetts, October 2010.

[33] Tetsuji Nakagawa. Multilingual dependency parsing using global features. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 952–956, Prague, June 2007.

[34] Srini Narayanan and Sanda Harabagiu. Question answering based on semantic structures. In *Proceeding of the 20th International Conference on Computer Linguistics*, pages 693–701, Geneva,Switzerland, 2004.

[35] Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, GÜLŞEN Eryiğit, Sandra KÜBLER, Svetoslav Marinov, and Erwin Marsi. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135, 2007.

[36] Jacob Persson, Richard Johansson, and Pierre Nugues. Text categorization using predicate-argument structure. In *NODALIDA 2009 Conference Proceedings*, pages 142–149, Odense, Denmark, May 2009.

[37] Sameer Pradhan, Kadri Hacioglu, Valerie Krugler, Wayne Ward, James H Martin, and Daniel Jurafsky. Support vector learning for semantic argument classification. *Machine Learning*, 60(1-3):11–39, 2005.

[38] Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James Martin, and Dan Jurafsky. Shallow semantic parsing using support vector machines. In *Proceedings of HLT/NAACL*, page 233, 2004.

[39] Sameer S. Pradhan, Wayne Ward, and James H. Martin. Towards robust semantic role labeling. *Computational Linguistics*, 34(2):289–310, 2008.

[40] Lance A. Ramshaw and Mitchell P.Marcus. Text chunking using transformation-based learning. In *Proceedings of the 3rd Workshop on Very Large Corpora*, pages 88–94, Cambridge, Massachusetts, 1995.

[41] Sebastian Riedel and Andrew McCallum. Fast and robust joint models for biomedical event extraction. In *Proceeding of the 2011 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Languages Learning*, pages 1–12, Edinburgh,Scotland, July 2011.

[42] Dan Shen and Mirella Lapata. Using semantic roles to improve question answering. In *Proceeding of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Languages Learning*, pages 12–21, 2007.

[43] David Songtag, Amir Globerson, and Tommi Jaakkola. Introduction to dual decomposition for inference. *Optimization for Machine Learning*, 1, 2011.

[44] Honglin Sun and Daniel Jurafsky. Shallow semantic parsing of Chinese. In *Proceedings of Human Language Technology Conference/North American Chapter of the Association for Computational Linguistics(HLT/NAACL-2004)*, pages 249–256, Boston, USA, May 2004.

[45] Weiwei Sun. Semantic-driven shallow parsing for Chinese semantic role labeling. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 103–108, Uppsala, Sweden, July 2010.

[46] Weiwei Sun, Zhifang Sui, Meng Wang, and Xin Wang. Chinese semantic role labeling with shallow parsing. In *Proceedings of the 2009 Conference on Em-*

pirical Methods in Natural Language Processing, pages 1475–1483, Singapore, August 2009.

[47] Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. Using predicate-argument structures for information extraction. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 8–15, Sapporo Japan, 2003.

[48] 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 12th Conference on Computational Natural Language Learning*, pages 157–177, Manchester, August 2008.

[49] 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: Volume 2-Volume 2*, pages 551–560. Association for Computational Linguistics, August 2009.

[50] Kristina Toutanova, Aria Haghighi, and Christoper D. Manning. A global joint model for semantic role labeling. *Computational Linguistics*, 34(2):161–191, 2008.

[51] Kristina Toutanova and Mark Johnson. A bayesian lda-based model for semi-supervised part-of-speech tagging. In *Proceedings of NIPS*, volume 20, 2007.

[52] 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*, pages 114–119, Boulder,Colorado, June 2009.

[53] Yotaro Watanabe, Masayuki Asahara, and Yuji Matsumoto. A structured model for joint learning of argument roles and predicate senses. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 98–102, Uppsala, Sweden, July 2010.

[54] Jia Xu, Jianfeng Gao, Kristina Toutanova, and Kermann Ney. Bayesian semi-supervised Chinese word segmentation for statistical machine translation. In

Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1, pages 1017–1024. Association for Computational Linguistics, August 2008.

[55] Nianwen Xue. Labeling Chinese predicates with semantic roles. *Computational Linguistics*, 34(2):225–255, 2008.

[56] Hiroyasu Yamada and Yuji Matsumoto. Statistical dependency analysis with support vector machines. *Machine Learning*, 34(1-3):151–175, 1999.

[57] Hai Zhao, Wenliang Chen, Chunyu Kit, and Guodong Zhou. A huge feature engineering method to semantic dependency parsing. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning Conference of European Chapter of the Association for Computational Linguistics*, pages 55–60, Boulder,Colorado, June 2009.

[58] Hai Zhao and Chunyu Kit. Parsing syntactic and semantic dependencies with two single-stage maximum entropy models. In *Proceedings of the 12th Conference on Computational Natural Language Learning*, pages 203–207, Manchester, August 2008.