MASTER THESIS

IMPROVING HPSG PARSE DISAMBIGUATION BY ENRICHING SYNTACTICAL CONTEXT INFORMAION

Tzu-Yi KUO
September, 2011

European Masters in Language and Communication Technologies

Supervisors:
Prof. Dr. Hans Uszkoreit and Dr. Yi Zhang
Universitat des Saarlandes

Co-supervisor:
Prof. Guy Perrier
Universite Nancy 2 UFR Mathematiques et Informatique

Declaration of the author

Eidesstattliche Erklarung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbststandig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Declaration

I hereby confirm that the thesis presented here is my own work, with all assistance acknowledged.

_ _

Signature: Tzu-Yi KUO

_ _ _ _ _ _ _ _ _ _

Date

Abstract

With the advance of science and technology, Nature Language Processing plays a more important role nowadays. Natural languages are abundant of ambiguities. While ambiguity facilitates efficient communication between human speakers, it causes processing difficulties in language technology. Many applications that may benefit from deep linguistic analysis (such as Question Answering, Machine Translation, etc.) require a single parse tree as their input. The task of parse disambiguation is becoming increasingly important.

Broad coverage grammar such as HPSG or LFG has been introduced and welcomed in recent year. This kind of grammar is developed to offer a more detailed analysis for different languages. However, multiple analyses will inevitably rise in such detailed analysis, both due to the genuine ambiguity of the language and the spurious ambiguity produced by the processing. The number of correct structures that are licensed by the broad coverage grammar is often vast. In such kind of environment, how to identify the intended analysis from a handful or usually a ton of reading becomes a critical issue.

In this work, we discuss some existent methods and consider new approaches which intent to improve the parse disambiguation accuracy on HPSG. We employed a statistical model which fine-tunes the information from the output of HPSG framework in order to find valuable features on the disambiguation task. In particular, we focus on the combination of syntactic categories and the rule schemata of HPSG which were used to license the signs. The features in the combination result are more robust and active than they were when presented separately.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, many natural language processing tasks have reached levels of wider applicability and demand. Along with the tendency, the NLP gets higher attention and becomes a more and more prominent research area. Noteworthy applications are Machine Translation, Question Answering, Information Extraction, Grammar Checking, among many others. While some simple NLP applications treat language as a strings of words, most applications do need deeper understanding of the inner structure of the human languages. Parsing is therefore an important step for such NLP tasks. Parsing is an intermediate process that uncovers the syntactic structure of natural language sentences, and facilitates the interfacing between the surface structure of the language and the embodied meaning of it. In practice, there are different parsing techniques according to the depth of analysis, from the very basic shallow parsing (such as chunking) to the deep parsing which use linguistically informed grammars. For example, given a sentence like “the dog barks”, shallow paring may only recognize the noun/verb phrase groups – [the dog] [barks]. But the deep parsing will assign a more complex structure, such as a dependency or phrase structure tree, or even a feature structure.

However, ambiguity remains as one of the critical problem of parsing task (we focus on syntactic ambiguity in this thesis). A sentence may have more than one syntactic analysis depending on the different type of grammar and its richness. Thus, we need a useful tool to help the higher level NLP applications for selecting a correct syntactic analysis given an input sentence to be passed on to the downstream processing. This task is called *parsing disambiguation*, or *parse selection*. Conceptually, we may view a parsing re-ranking as a two-stage process [Charniak and Johnson, 2005]. The first part is a device that has an underlying grammar and creates an initial list of possible parses for a given utterance. Usually, this list consists of an *N*-best parse tree in order to guarantee a reasonable runtime. This may results another problem in losing the

correct analysis simply because of it is not in the set of candidates. The second stage, a disambiguator selects a single most likely tree from the list of parse candidates.

There is a widely adopted approach of parse selection. This approach treats the disambiguation problem as a classification task. Collecting labeled data from treebank to learn classifier and build training models to classify unseen data. Previous studies have developed two approaches that use treebanks for disambiguation models – *generative* and *discriminative* models. The generative model only uses the gold parse result and needs independent assumption between features in one parse result. The other approach uses discriminative model, which is created by training Maximum Entropy Model on treebanks (that two approaches will be discussed in detail in Chapter 3). There are many features can be extracted from a given treebank and used to build the latter model, such as local configurations, ancestor information, n-gram information, etc. We will talk about the state-of-the-art feature templates and the feature sets we used in this work in Chapter 4.

1.1 Motivation

In this thesis, we try to improve the HPSG parse disambiguation accuracy by enriching the context information in HPSG rules. The primary question is whether there is additional useful information to the state-of-the-art methods. And if yes, how can one to extract and exploit that information for solving this problem? In another way of thinking, it might be effective to combine the different types of information which are proved to be helpful, and use them for the prediction of preferred parse result.

Ambiguity is a central problem in natural language parsing. Combinatorial effects mean that even relatively short sentences can receive a considerable number of parses under a wide-coverage grammar [Collins, 1997]. The higher the grammar coverage is, the higher of chance that it may introduce a greater number of analyses for structurally ambiguous sentences. State-of-the-art feature templates such as local derivational configurations, annotation of several ancestors, semantic information or other internal and external categories have increased performance dramatically (will be discussed in Chapter 3 and Chapter 4). However, some models might be limited in one particular approach and lose the chance to get advantage from others. If a new model can be build by combining various profits from different approach, the features

would help to re-rank the parse candidates and find the best parse result. Another potential prospect is, if we can successfully improve the performance of parse disambiguation system, it will also make advantage for various other natural language processing tasks.

1.2 Outline of the thesis

The remainder of this thesis is organized as following:

First we provide a background overview in Chapter 2, where we explain the basic concepts and methods which are used in this work. Chapter 3 discusses the previous related works. It starts with the development on statistical parsing. It then continues with a review of the related literature while comparing the differences description of the two approaches (discriminative and generative). Chapter 4 starts with the experiment environment of our work. It presents the setting of our experiments, such as the corpora, grammar and the parser. Then we briefly talk about the state-of-the-art feature type in parse disambiguation models. Finally, we describe the feature sets that we use in this work. We also discuss how disambiguation model is trained and used. Chapter 5 offers a detailed analysis of the performance of our work, comparing with the state-of-the-art feature type. Lastly, in Chapter 6, we summarize the contribution of this work and what we have learnt from this thesis. We also conclude by giving suggestions to future research.

Chapter 2

Background

In computer science and linguistic, parsing, or more formally, syntactic analysis, is the process of analyzing sequences of tokens, to determine its grammatical structure with the help of a set of elementary structures in the form of rules, that are defined in a given grammar and a set of operations to combine these structures. It is a central research topic in the automatic processing of human language, and has seen its wide usage in NLP applications such as Information Extraction and Retrieval, Machine Translation, Question Answering, etc. However, due to the inherent ambiguity in the natural languages, this has proved itself to be challenging. In order to resolve structural ambiguities, several statistical parser selection models have been developed.

Depending on the richness of the analysis, various parsing systems can be roughly categorized as either *shallow* or *deep*. Shallow parsing (also chunking or light parsing) is frequently used in order to design robust parsing system. It identifies the constituents (noun groups, verb groups, etc.), but does not specify their internal structure, nor their role in the main sentence. Primary advantage of shallow parsers lies in their high processing efficiency as opposed to full parsers and their ability to support a variety of NLP applications, such as Text Summarization, Named Entity Identification, etc.

Deep parsing computes deep linguistic structures from input text. It is directly based on grammars. A subset is positively characterized if it satisfies the constraints of a grammar. But the main problem of this approach lies in choosing the correct syntactic analysis of a sentence in-between all the possible analyses that the system is able to generate. The more complex the grammar is, the more difficult it is for the parser to make decisions in assigning analyses. Nevertheless, deep parsing is very helpful to

many nature language processing applications.

In the following of this chapter, we briefly talk about the grammar formalisms which are used to constrain the languages in Section 2.1. We present in more detail in Head-driven Phrase Structure Grammar (HPSG). Section 2.2 explains the ambiguity problem. We also introduce the mathematical preliminaries of this work, defining entropy and maximum entropy models in Section 2.3.

2.1 Grammar formalisms

A language is a possibly infinite set of strings. Grammar formalisms are artificial languages whose purpose is to characterize precisely other artificial or natural languages. Grammar formalisms can characterize languages procedurally, by providing a method or algorithm for generating all of the string of the language, or declaratively, by providing a direct description of the language elements. Further, a procedural description needs a clear and organized instruction to specify how the string of the language can be constructed or analytic. The particular instruction is the essential part that makes the procedural description be synthesized [Bright, 1992].

Lexicalized grammar formalism is a type of modern frameworks for explaining syntactic structures of natural language sentences. The main concept is that structures and rules to derive syntactic structures are mainly introduced by *lexical entries*, which are structures assigned to words. Intuitively, terminal nodes of parse trees incorporate most structures, and parse trees are constructed by checking the consistency of the lexical structures. Most modern syntactic theories are lexicalized grammars, examples include Head-Driven Phrase Structure Grammar (HPSG) [Pollard and Sag, 1994 (2)], [Pollard and Sag, 1994 (1)]; [Sag and Wasow, 1999], Lexicalized Tree Adjoining Grammar (LTAG) [Schabes, 1992], Lexical Functional Grammar (LFG) [Bresnan, 1982], and Combinatory Categorical Grammar (CCG) [Steedman, 2000]. In the field of linguistics, lexicalized grammars have provided a mathematical framework for describing linguistic structures, and were extensively studied in order to explicate the structure of syntax in a mathematically sound manner. They also attracted researchers working in the field of computer science, because they were defined with a strict mathematical formulation and could be implemented as computer programs. In fact, several systems are now being implemented with the aim of being used in real

applications.

In the rest of this section, we describe details of the theory of Head-driven Phrase Structure Grammar (HPSG), which is a syntactic theory extensively studied from both linguistic and computational points of view.

HPSG grammars use Typed Feature Structures (TFS) as their background formalism. The TFS uses a set of attribute-value pairs (features) to represent linguistic objects. It also includes a type inheritance system to check the compatibility by using a set of highly generalized rules during the parse process. Therefore, each passive edge on the parsing chart corresponds to a TFS.

An HPSG grammar includes two essential components, a set of grammatical constrains and highly structured representation of grammatical categories. Each of the components is encoded as Typed Feature Structures. From a linguistic perspective, the set of descriptive constraints expressing the theory of an HPSG grammar consist of a) a lexicon licensing basic words, b) lexical rules licensing derived words, c) immediate dominance schemata licensing constituent structure, d) linear precedence statements constraining constituent order, and e) a set of grammatical principles expressing generalizations about linguistic objects [Levine and Meurers, 2006]. The immediate dominance schemata are meant to be the sole basis to combine words and phrases (e.g., head-complement, head-subject, head-specifier, and head-modifier).

The basic unit of linguistic representation in HPSG is the SIGN, which is a complex feature structure representing information of different linguistic levels of a lexical item. All sings carry two features, PHON and SYNSEM. Each type of information within a sign is called an attribute, and each attribute must have a value. This notion can be specified using a specific description language, Attribute Value Matrix (AVM).

The example of SIGN structure is shown in Figure 2.1, consider the verb ‘walks’, for which a partial lexical entry. HPSG is not just a syntactic theory, but a comprehensive theory of language willing to acknowledge the place of semantics, pragmatics and other grammatical levels in grammatical theory.

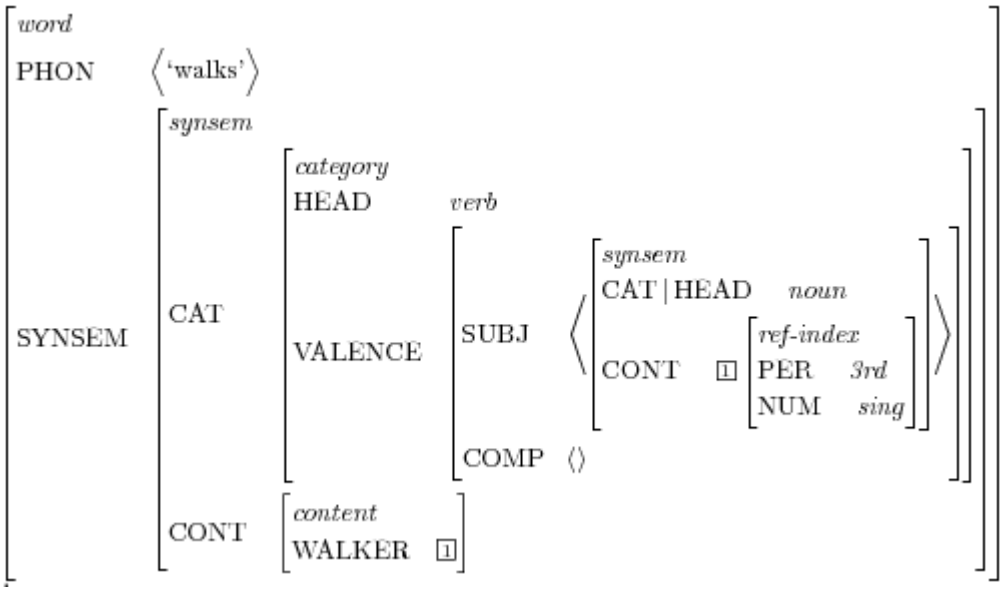

Figure 2.1: A basic lexical entry (an example from Wikipedia)

Due to its rigid mathematical foundation, HPSG has been widely adopted in the development of linguistically motivated large-scale precision grammars for different languages.

2.2 What is Ambiguity?

The general meaning of ambiguity is where information can be understood or interpreted in more than one way in a statement without the lack of precision contained or available in the information. In natural language processing, ambiguity may occur in every step. [Hutchins, 1992] subdivides the ambiguity into different types, the *lexical ambiguity* and *syntactic ambiguity*.

The most straightforward type of lexical ambiguity is that of *category ambiguity*: a given word may be assigned to more than one grammatical or syntactic category (e.g. noun, verb or adjective) according to the context. There are many examples of this in English: *light* can be a noun, verb or adjective; *control* can be a noun or verb. Category ambiguities can often be resolved by morphological inflection. However, the problems increase when several categorically ambiguous words occur in the same sentence, each requiring being resolved syntactically.

The second type of lexical ambiguity occurs when a word can have two or more different meanings. Linguists distinguish between homographs and polysemes. *Homographs* are two (or more) 'words' with quite different meanings which have the same spelling: examples are *bank* (riverside and financial institution), *light* (not dark and not heavy). *Polysemes* are words which exhibit a range of meanings related in some way to each other, e.g. by metaphorical extension or transference (*mouth* of a river, *branch* of a bank, etc.).

Those are the question of identifying the sense in context of a particular written 'word'. Despite the individual ambiguities, there is in fact only one correct analysis of this sentence, although it requires more than simple knowledge of possible category combinations to arrive at it.

In this work, we are interested on the *syntactic ambiguity*. Ambiguity arises when there is more than one way of analyzing the underlying structure of a *sentence* according to the *grammar* used in the system [Hutchins, 1992]. Whereas lexical ambiguity involves possible different interpretations of a word, structural ambiguity involves possible different interpretations of the meaning of the whole sentence. Structural ambiguity, however, can be divided into two types: *genuine ambiguities* (true ambiguity of the language) and *spurious ambiguities* (false ambiguity due to the over-generation of the grammar). As the classic example, "The boy saw the man with the telescope". For this sentence, it is possible for human readers to find more than a single interpretation. The different interpretations of this sentence are shown in Figure 2.2. In real life, language is abundant of true ambiguities. A good parsing model needs the capacity to solve both situations and finds a best parse result.

With broad-coverage grammars, even moderately complex sentences typically have multiple analyses. Depending on the richness of the grammar, a sentence can have multiple numbers (sometimes thousands) of syntactic analyses. The task of parse disambiguation is becoming increasingly important. Roughly speaking, parse disambiguation approaches fall into two categories: *symbolic* and *statistical*.

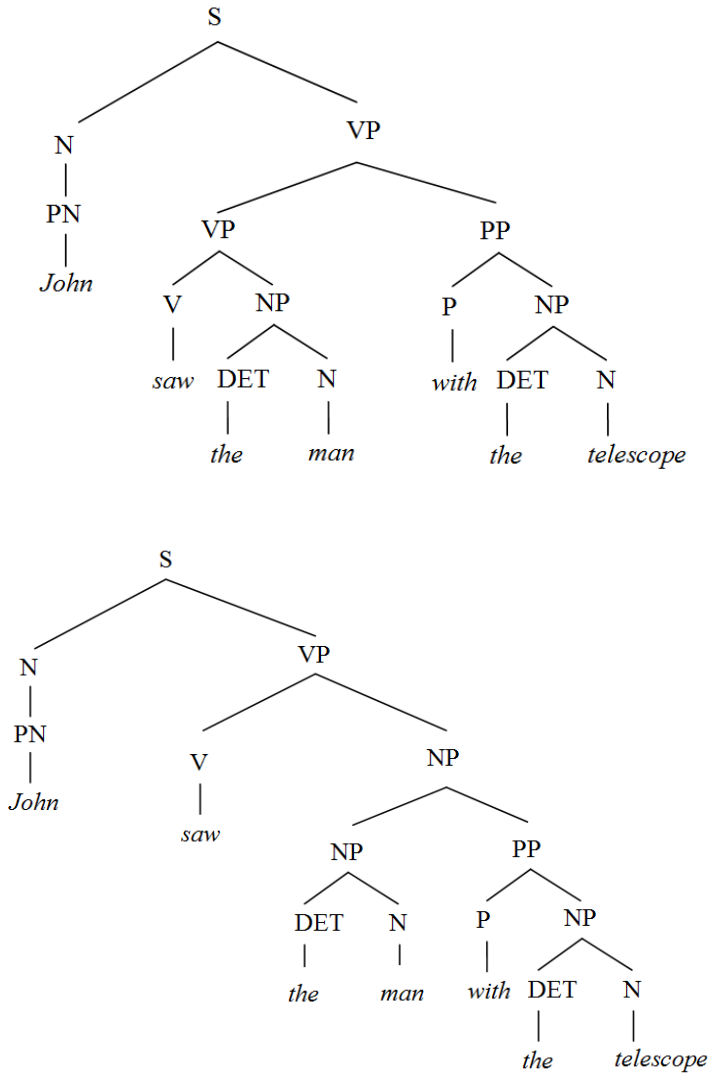

Figure 2.2: Two interpretations of the sentence "*John saw the man with the telescope.*"

Symbolic approach using constraints/grammar rules to restrict the possible ambiguous structures generated by the grammar. It does not require massive amounts of data but often intensive human effort required to create appropriately annotated data. The primary source of evidence in symbolic systems comes from human-developed rules and lexicons, usually the hand-written deep grammars are more restrictive than shallow treebank-induced grammars. In contrast, symbolic approaches are limited by slow manual analysis and rule-building, which is generally costly, difficult, and often incomplete.

Statistical parsing approaches tackle the ambiguity problem by assigning a probability to each parse tree, ranking competing trees in order of plausibility and selecting a single preferred analysis. Therefore such approach is also called parser ranking. There have been plenty of works on statistical parsing in last two decades, and historically, a major portion of them are focused on generative models (probabilistic context-free grammar) or discriminative models (e.g. maximum entropy model). The generative model uses only the gold or preferred parse trees but the discriminative models are trained by maximizing the probability of all the preferred analyses relative to all the alternative and non-preferred analyses. We will discuss about these two models in more detail in next chapter. In contrast to symbolic approaches, statistical approaches use observable data as the primary source of evidence.

2.3 Maximum Entropy Models

The maximum entropy model is the mathematical foundations of the statistical model that we used for parse disambiguation in this work. This model has been applied to variety of problems in Natural Language Processing.

2.3.1 Entropy

The key concept, entropy, is a measure of information, and is invaluable throughout natural language processing. It can be used as a metric for how much information there is in a particular grammar, for how well a given grammar matches a given language, for how predictive a given N-gram grammar is about what the next word could be [Jurafsky and Martin, 2006]. In order to compute entropy, we need to establish a random variable *X* that ranges over what we are predicting. The set of might be words, letters which has a particular probability function – *p(x)*. The entropy of this random variable *X* is then:

$$H(X) = -\sum_{x \in X} p(x) \log p(x)$$

The Maximum entropy approach is one a family of log-linear approaches to classification in which many features are computed for the sentence to be analyzed, and all the features are combined in a model base on multinomial logistic regression. It offers a clean way to combine diverse pieces of contextual evidence in order to estimate the probability of a certain linguistic class occurring with a certain linguistic context.

2.3.2 Maximum Entropy Model

After the brief introduction of maximum entropy model, we try to use this approach to create a model to classify our data. In this work, we want to estimate a probability model in order to use the observed data (the training set) to make prediction from the candidate parses of unseen data (the test set). Maximum entropy models, or discriminative log-linear models, have previously been used for stochastic unification-based grammars by [Johnson et al., 1999]. [Toutanova et al., 2005] has shown that log-linear models for parse disambiguation considerably outperform PCFG models in Redwoods treebank. This model is now becoming a widely accepted for the probabilistic modeling of various tasks because of its high accuracy.

A maximum entropy model is given by a set of feature functions and corresponding weights. The specified feature sets $F = f_1, \ldots, f_m$ describe properties of the events, and the associated set of learned weights $\lambda = \lambda_1, \ldots, \lambda_m$ determines the contribution or importance of each feature. After the model has been trained, in our case, we need to construct a probability distribution p over a set of parses $T = t_1, \ldots, t_m$ which are characterized by features F. The conditional probability of parse t for sentence w is given by

$$P(t \mid w) = \frac{\exp \sum_i \lambda_i f_i(t, w)}{\sum_{t \in T} e \sum_i \lambda_i f_i(t', w')}$$

The value of $f_i(t, x)$ reflects the frequency of the *i−th* feature in a given parse *t*. The parameters λ_i, which is the weight of the corresponding *i−th* feature. The weights of a corresponding set of features produced from annotated training material. The process of training is used to estimate the parameters of the model in particular weights vector

λ. Then we can predict the preferred parse for each sentence by finding the highest figure of its feature weight score.

$$t = \arg\max_{t \in T} \sum_{i} \lambda_i f_i(t, w)$$

An attractive property of the maximum entropy models is that, it is possible to integrate distinct but potentially overlapping sources of information, and features can be defined to take into account whatever aspects of the parse trees are considered important. A potential drawback of is this formula requires access to all parses of a given corpus sentence, which is inefficient because a sentence can have an exponential number of parses.

Chapter 3

Previous Related Work

As we discussed in previous chapters, language is abundant of ambiguities in real life. With the development of Nature Language Processing, parsing get more and more attention from both the linguistic and computer science field. However, it is still rather difficult to parse a sentence fully correct because of the ambiguity inherent in the natural language. Recall to the Section 2.1, the task of parse disambiguation can be roughly fall into two different methods, *symbolic approach* and *statistical approach*. The symbolic approach using constraints/grammar rules to restrict the possible ambiguous structures generated by the grammar. Compare with statistical approach, it is limited by slow manual analysis and rule-building, which is generally costly, difficult, and often incomplete. In contrast, the statistical approach has been wildly applied on various different researches and performs well. [Velldal, 2008] noted that stochastic elements are effective in parsing because of its two properties, *robustness* and *the ability of dealing with ambiguity*. In many real-world natural language applications, the input can be expected to be ungrammatical or with noises. A strict grammar will lead the parser fail in face of such input, even often ungrammatical strings are apprehensible. The second issue is dealing with the ambiguity. A statistically guided parser often ranks the competing analyses and selects one single preferred reading to complete the disambiguation task. During the last decade, the statistical approach becomes the major trend in natural language parsing, or syntactic disambiguation.

The disambiguation task is closely connected to statistical parsing and parse selection. As mentioned before, the goal of this thesis to improve the performance of parse disambiguation model by using context information with statistical methods. In this chapter we take a look at some recent works that were previously done in particular areas which incorporate a statistical component that related to the approaches in this thesis. We begin in Section 3.1 with the development of statistical parsing which use

log-linear models. And talk about the two main approaches – generative and discriminative in Section 3.2. We also talk about the parse disambiguation work on dynamic treebanks, e.g. the Redwoods Treebank in Section 3.3. Finally end up with a brief discussion in Section 3.4.

3.1 Statistical Parse Selection

The task of statistical parse selection is to solve the problem of selecting a preferred analysis given by an input string. There have been numerous of works on statistical parsing in the past twenty years, and historically, a major portion of them are focused on probabilistic context-free grammar (PCFG). The PCFG is a context-free grammar in which it captures syntactic regularities in languages probabilistically. [Abney, 1997] notes the distribution of derivations of a unification-based grammar (such as HPSG and LFG which can encode richer syntactic and semantic constraints) may not perform as well as the class of PCFG grammars defined by its context-free base. He brings up the use of log-linear model for parse ranking and motivates [Johnson et al., 1999] for further development. The main idea of [Abney, 1997] and [Johnson et al., 1999] is to find the optimal parameters of the weights that maximize the log-likelihood of the training corpus according to a log-linear model. In recent years, various approaches based on log-linear models for Stochastic Unification-based grammars (SUBGs) have been developed. These approaches are mostly leaded by [Abney, 1997] and [Johnson et al., 1999]. Regarding to this thesis, the most relevant one among them is the work of [Toutanova et al., 2005] on training and comparing different conditional log-linear models for parse selection. We will discuss this work in detail in Section 3.3. In the next section, we will explain the two different probabilistic models which estimate joint and conditional distribution respectively.

3.2 Generative and Discriminative Approaches

A probability model is a statistical analysis tool that estimates the probability of an unseen event (the test set) occurs based on the observed historical data (the training set). In this work we try to learn such a model in order to predict the correct analysis in a large amount of candidates. The approaches can be divided into two categories: generative models and discriminative models. We will go into detail of these approaches further below.

3.2.1 Generative method

The maximum likelihood estimate for a PCFG gives us a *generative model* that predicts a full joint distribution of strings and trees – *P*(String, Tree). That is to say the generative model use the likelihood $P(x_1,\ldots, x_n|Y)$ and the prior $P(Y)$ to generate observable (training) data-sets. This model needs to compute the probability of a certain observation which belongs to a certain class $P(Y|x_1,\ldots, x_n)$ in order to be used as a classifier. Using Bayes rule we can adapt the probability as

$$P(Y \mid x_1,...x_n) = \frac{P(x_1,...x_n \mid Y)P(Y)}{P(x_1,...x_n)}$$

It specifies a joint probability distribution over observation and label sequences. The joint probability needs vast number of estimation to calculate. We can use an independence assumption to reduce complexity of such model. If we assume that all the features $x_1,\ldots, x_n$ are conditionally independent from the given Y. We can greatly simplify the model by estimate the summation of each feature which occurs in a give class and they are independently from other features.

$$P(Y \mid x_1,...x_n) = \frac{\sum_i P(x_i \mid Y)P(Y)}{P(x_1,...x_n)}$$

In fact, this conditional independent assumption is not true in most of situation, especially in Natural Language Processing. The different components of natural language are very much related. The independence assumption becomes the main weakness of this approach with the result that it fails to capture some important linguistic notions. Even though, there still some researches use generative model before the discriminative approach getting popular and try to get over the drawback. Such as Collins parser [Collins, 1997] [Collins, 2003] and Dan and Manning's work [Klein and Manning, 2003].

In [Klein and Manning, 2003], they demonstrate that an unlexicalized PCFG can parse much more accurate than previously shown, by making use of simple, linguistically motivated state splits, which break down false independence assumptions latent in a unmodified treebank grammar. Indeed, its final label

bracketing result is better than that of early lexicalized PCFG models, and surprisingly close to the current state-of-the-art.

However, the development on the utility of PCFG for parse disambiguation and language modeling were at a standstill in the early 1990s. A new opinion arise that lexicalized PCFGs are the key tool for high performance PCFG parsing. [Manning and Schutze, 1999] declare that "the most remarkable is their lack of lexicalization". In history-based models, a parse tree is represented as a sequence of decisions. In the case of Contet-Free grammars, the decisions are conditioned only on the previous node. This implies that *words* are forgotten during selection of higher nodes in the derivation of the tree. The most straightforward way to incorporate lexical preferences is to associate every non-terminal in the tree with its head-word [Collins, 2003]. A great success in terms of parse disambiguation and even language modeling was achieved by various lexicalized PCFG models, such as [Collins, 1997], [Collins, 2003].

3.2.2 Discriminative method

Discriminative models estimate the probability $P(Y|x_1,\ldots, x_n)$ directly use log-linear models. Most of the works in discriminative models associate with the concept of Maximum Entropy Model, which we discussed in Section 2.3. [Johnson et al., 1999] note that the conditional distribution intuitively seems important for the disambiguation task while the marginal distribution does not. Better results might therefore be achieved by optimizing the conditional likelihood directly. This means to learn a conditional distribution but not a full joint distribution. The state-of-the-art works such as Toutanova's work [Toutanova et al., 2005] and Charniak & Jhonson's reranking parser [Charniak and Johnson, 2005] all adopt this model.

[Toutanova et al., 2005] elaborate how they train a stochastic model by using the fundamental data stored in the Redwoods treebank – derivation tree and the deep semantic information, Minimal Recursion Semantics (MRS), provided by the HPSG signs. In this approach, they demonstrated the usefulness of building models over derivation trees of HPSG analyses and showed how they can be supplemented with semantic and lexical item sequence information for significant accuracy improvement. The learned probabilistic models were used to rank possible parses of unseen test sentences according to the probabilities assigned by those models. In the next section,

we will talk about the work of [Toutanova et al., 2005] in depth. The experimental result in [Toutanova et al., 2005] on the Redwoods Treebank shows the discriminative models indeed performs better than the generative models in the parse disambiguation task.

3.3 Parse Disambiguation on the Redwoods Treebanks

In this section, we talk about a very extensive and impressive work on parse disambiguation made by [Toutanova et al., 2005]. This work employed the same framework as ours; the DELPH-IN collaboration, an HPSG grammar and treebank (see Section 4.2). [Toutanova et al., 2005] experiment the generative and discriminative methods by using different feature set and made systematic comparison of them. They reported almost 14% error reduction when using discriminative models over generative models with the same feature set.

As we mentioned before, several variant approaches of [Abney, 1997] and [Johnson et al., 1999] which use log-linear models on parse disambiguation have been published. Particularly relevant for the experiments of this thesis is the work of [Toutanova et al., 2005]. They executed this work on training conditional log-linear models for parse disambiguation task on the LinGO Redwoods HPSG Treebank [Oepen et al., 2002]. Each sentence in the Treebank is annotated with an HPSG analysis licensed by the LinGO English Resource Grammar (ERG; [Copestake and Flickinger, 2000]), more detail description in Section 4.2. The annotations contain fine-grained syntactic information represented by typed feature structures – the HPSG signs. It also includes logical-form meaning representation based on Minimal Recursion Semantics (MRS; [Copestake et al., 1995]). MRS is a representation language for describing semantic structures. The labeled formulas, Elementary Predicates (EPs), are the basic elements of MRS. They are used to represent the semantic relations. HPSG use a new type – *mrs-struct* to contain the translated information from MRS.

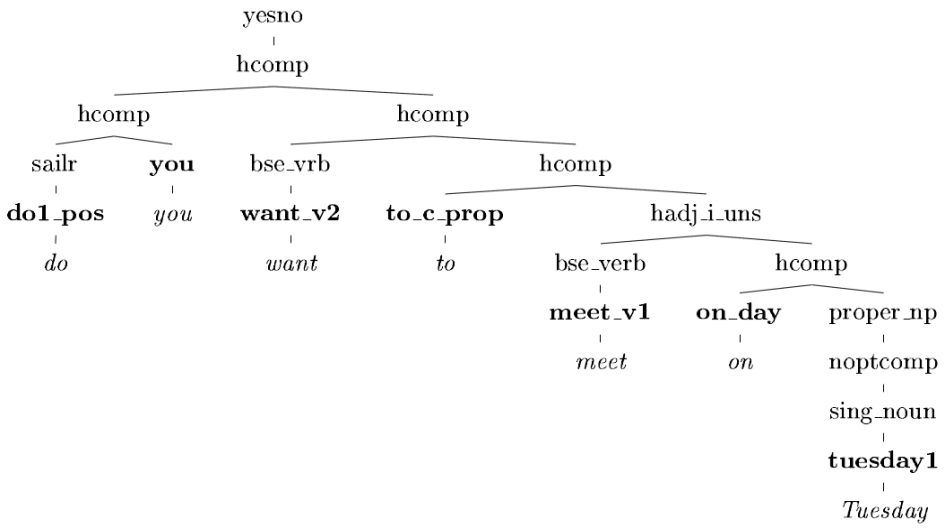

Figure 3.1: Redwoods derivation tree using unique rule and lexical item identifiers for the sentence "*Do you want to meet on Tuesday?*" [Toutanova et al., 2005]

The work of [Toutanova et al., 2005] use different features sets extracted from Redwoods Treebank to train and test several disambiguation models. The Redwoods Treebanks have been created by a *grammar-based* approach, which means the treebanks are annotated under an agreement of an existing hand-crafted grammar. It makes the resource more dynamic, the annotations can be updated whenever the grammar develops and improves over time. Figure 3.1 shows the fundamental data structure of the Redwoods resources – the *derivation tree*. The derivation tree is significantly different from the phrase structure tree (shown as in Figure 3.2) in many traditional resources. The internal nodes of derivation trees (e.g. the head-complement, head-specifier schemas) correspond to rule schemas of the underlying grammar. The pre-terminals (the bold characters in Figure 3.1) of the trees are fine-grained lexical identifiers (lexical items). This kind of representations offers more explicit description than the phrase structure tree. For example, rather than 45 Part-Of-Speech tags and 27 phrase node labels compares with about 8,000 lexical item identifiers and 70 derivational schemas [Toutanova et al., 2005]. Another important feature is the deep semantic information of sentences. Figure 3.3 shows an example of elementary dependency graph extracted from the MRS meaning representation.

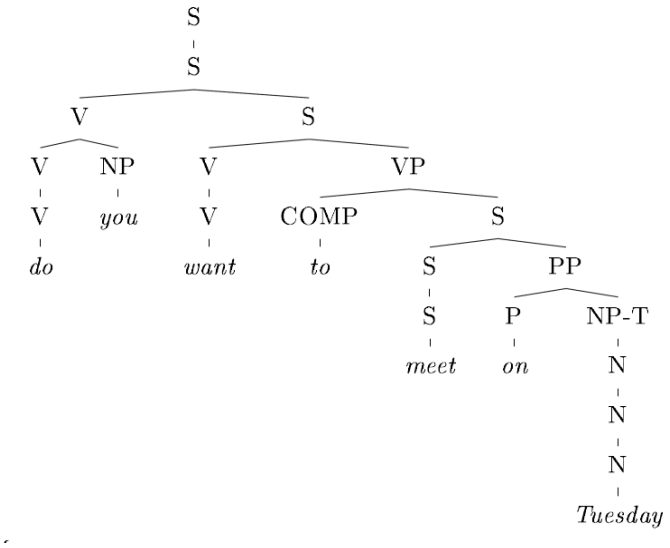

Figure 3.2: Phrase structure tree labeled with user-defined, parameterizable category abbreviations for the sentence "*Do you want to meet on Tuesday?*" [Toutanova et al., 2005]

```
_4:{
    _4:int_rel[SOA e2:_want2_rel]
    e2:_want2_rel[ARG1 x4:pron_rel, ARG4 _2:hypo_rel]
    _1:def_rel[BV x4:pron_rel]
    _2:hypo_rel[SOA e18:_meet_v_rel]
    e18:_meet_v_rel[ARG1 x4:pron_rel]
    e19:_on_temp_rel[ARG e18:_meet_v_rel, ARG3 x21:dofw_rel]
    x21:dofw_rel[NAMED :tue]
    _3:def_np_rel[BV x21:dofw_rel]                              }
```

Figure 3.3: Elementary dependency graph extracted from the MRS for the sentence "*Do you want to meet on Tuesday?*" [Toutanova et al., 2005]

In [Toutanova et al., 2005], they experiment several different feature sets on generative and discriminative model (more detail description in Section 3.2). The first generative model is a PCFG model with the rule schemas in the derivation trees of the HPSG analyses for sentences in the treebank as features. The second model is based on the previous one but extended to include ancestor information for up to a maximum of four grand-parenting levels. The parameters in these two models are specified to maximize the likelihood of the set of preferred analyses of sentences in

the training treebank. They also train similar PCFG models using information extracted from phrase structure trees as features for comparison in this work. However, the performance is not as good as the original models which trained on the derivation tree.

A PCFG-style model is trained over the semantic dependency trees extracted from the HPSG signs. They also implement a standard trigram HMM tagger which define a joint probability distribution over the pre-terminal tag sequences and yields of the derivation trees. The linear interpolation of lower order models are used as smoothing technique. Finally, several of the generative models are combined into a single model. The integral parts are the PCFG model using ancestor information trained on derivation trees, the PCFG model trained on semantic dependency trees and the HMM tagger which only uses the trigram tag probabilities. The score of a given parse tree is a summation of the log-probabilities of individual models with experimental weights.

Furthermore, [Toutanova et al., 2005] also experiment corresponding discriminative log-linear models for all the generative models introduced in the previous paragraphs. Each of the models uses exactly the same feature sets but trained to maximize the conditional log-likelihood of the treebank data. The result of these two different approaches is that the discriminative model outperforms the generative model greatly. The combined model which trained by the discriminative approach achieves 13% of error reduction.

The study of [Toutanova et al., 2005] results several valuable points. The features extracted from derivation trees improve the results of parse disambiguation model indeed and performance even better than the standard phrase structure trees. Extending the feature set with ancestor information also gives contribution on the accuracy. Moreover, it seems that the information in semantic dependency trees does not assist the disambiguation task according to the result. From a higher level of view, the discriminative model been proved to have better results than the generative model no matter the individual models or the final combined models. Besides, the final combined models always have better results than the individual models. It shows that the disambiguation model can be benefit by adding various useful features.

In the following of this thesis, we will introduce the model we developed which based on the discriminative log-linear model in [Toutanova et al., 2005]. The feature sets which have been further extended are defined in more detail in Section 4.3. In spite of

the failure of feature set extracted from phrase structure tree reported in [Toutanova et al., 2005]. We combine the information from derivation tree and phrase structure tree and make some adaption to test the usefulness of syntactical information in parse disambiguation task.

3.4 Discussion

We started this chapter with an overall description of researches in parse selection field, and further talk about the different approaches – generative and discriminative methods. The early researches who uses generative methods, mainly extend PCFG to assign probabilities to different operation of the grammar which results in a score to each analysis. The main drawback of this approach is the independence assumption. As we show in Section 3.2.1, the generative method needs conditional independent assumption between features and the given condition in order to reduce the complexity. But this assumption is often not true, especially in nature language processing. This assumption needs a very careful design of distribution model and the feature sets. Secondly, the difficulty of adding new features and the lack of an underlying deep grammar make this approach unpractical.

In the later development of parse disambiguation field, we can find out that the discriminative methods (in particular Maximum Entropy models) become the main stream. The use of extensive feature number with the support of an underlying human designed deep grammar such as HPSG or LFG allows systems access more information and more linguistically motivated properties of the parse. The discriminative models make the integration of extra features (even the non-local features) more flexible without the scruple about the dependency between features. This approach also gets support from an underlying deep grammar. However, the deeper information about the sentence often leads a bigger data sparsity problem. At the same time, the higher the grammar covers, the higher of chance that it licenses a greater number of analyses for genuine ambiguous sentences. The mass number of candidates will lead the disambiguation task more difficult.

So far we present mainly two approaches and some previous work that bear similarities to the task of statistical parse disambiguation which is the object of this thesis. We see that discriminative methods have much success for their better performance and flexibility. In the next chapter, we turn to present the experiment environment and setting of our work.

Chapter 4

Experiments

While in the previous chapters, we explained the general concepts of the maximum entropy model and the statistical parse disambiguation approaches. In this chapter, we describe the key details related to the setup of our experiments. Roughly speaking, the first few sections of this chapter concern with aspects of data, grammar and parser we employed in this experiment. The later sections illustrate our disambiguation model.

In the first part of the chapter (section 4.1), we describe the corpus we used and the detailed figures of our development and test data. Section 4.2, we talk about the grammar and the parser which used to create the treebank corpora in section 4.1. Section 4.3 illustrates the feature sets, we describe the various features that are extracted from the dataset. We first discuss the baseline model and then illustrate the discriminant feature sets. Finally, we talk about the tools we used for training and evaluation.

4.1 Data

The whole experiment in this work used two different corpora, LOGON[1] and WeScience[2], both analyzed with the same underlying grammar, which is the LinGo English Resource Grammar (ERG). Since the ERG is an on-going developing project, it is necessary to decide and conduct all experiments with the same version of the grammar, we here use ERG grammar version LinGo (July-09). The next section will give more details on this grammar.

[1] the LOGON project web-pages can be found at http://www.emmtee.net/

[2] see http://wiki.delph-in.net/moin/LogonTop for details

The data we used for our experiment is a collection of English sentences from the LOGON parallel tourist corpus [Oepen et al., 2002] provided by Text Laboratory at the University of Oslo. This dataset is also known as JHPSTG, an acronym of its three smaller treebank components called Jotunheimen (JH), Prekestolen (PS) and Turglede (TG). The JH, PS and TG sub-corpora contain approximately 27,000 and 3,700 words in Norwegian respectively, and have at least one commissioned translation into English [Velldal, 2008]. In the following, we will refer to this collection of treebanks simply as the *LOGON* data set. LOGON contains 9,410 sentences in total. We use 8,927 sentences for which the parser is able to generate at least one preferred parse tree in the first 500 syntactic analyses in our experiment.

WeScience corpus [Ytrestøl et al., 2009] consists of a domain specific selection of Wikipedia articles in Natural Language Processing. The aim of this corpus is to help the accessibility of scholarly literature and digital libraries, with a special emphasis on community or open access resources. For this purposes, they chose the category Computational Linguistic and all its sub-categories as seed. They applied a simple link analysis and counted the number of cross-references to other Wikipedia articles from their initial seed set. They also did some pre-processed to strip irrelevant markup and segmented into sentence-like units. Therefore this corpus is more useful than data randomly collected from the Internet that lacks structure consistency and may be ungrammatical[3]. This dataset has 10,143 sentences in total.

Dataset	Total sentence	Average number of reading	Average length of sentence	More than 100 possible reading
Training set	17,225	251	14.46	51.9%
Development set	955	245	14.25	54.86%
Test set	890	269	15.94	58.42%

Table 4.1: Distribution and detailed information of dataset

[3] see http://wiki.delph-in.net/moin/WeScience for details

In a brief summary, our training dataset contains 6,098 sentences in JH, 927 sentences in PS, 947 sentences in subset TG and 9,253 sentences in WeScience corpus. The total number of sentences in the whole training dataset is 17,225 and the average number of reading per sentences is 251[4]. Base on the average number of reading in our training dataset, we choose the subset tg2 in TG as our development dataset (955 sentences). We take the subset ws13 in WeScience (890 sentences) as our test dataset. The development set is used for fine-turn our methods, and the final results presented in the next chapter are with respect to the test set. Table 4.1 shows the detailed figure of our data.

4.2 Grammar and Parser

4.2.1 An HPSG grammar for English

The LinGO English Resource Grammar (ERG) [Copestake and Flickinger, 2000] is a general-purpose and wide-coverage HPSG grammar for English and has developed at Stanford University and other groups for many person years. Following the theory of HPSG, the developers have been concentrating on the description of precision linguistic constraints, as well as the expanding of a grammar to be used in practical applications. After making some additions and adjustments, mostly to accommodate domain dependent vocabulary, the grammar provides accurate analyses for most of the reference translations in the LOGON development corpora. The grammar is developed and applied using an open-source grammar engineering system - Linguistic Knowledge Builder (LKB; [Copestake, 1992]) and efficient run-time parsing – PET.

Although the LinGO ERG is an English grammar, they are also interested in explicating language independent structures of natural language grammars. The Grammar Matrix [Bender et al., 2002] is a framework for the development of

[4] The average reading figures are influenced by the maximum readings that are allowed by the parser. In our setting the upper limit of readings has been set to 500 after which the parser does not continue to process any more derivations

wide-coverage HPSG grammars for various languages. They provide a starter kit to create an initial language-dependent grammar given some specifications of a language, such as a word order and complement categories. Their purpose is to assist the rapid development of grammars in other languages, as well as to share experiences in the development of various grammars.

4.2.2 The PET parser

The parser we employ, the PET parser is an agenda-driven, robust open-source runtime system for processing with unification based grammars. It is a core component of the deep linguistic processing with HPSG initiative (DELPH-IN) originally developed by DFKI GmbH and Saarland University [Callmeier, 2000].

DELPH-IN is a community effort on deep linguistic process with HPSG. It has delivered the most promising multi-lingual parallel grammar development withHPSG to date. The aim of this project is to investigate the interaction and compare the difference between new techniques in HPSG processing. It provides an extendible set of building blocks for the implementation of efficient processors. This enables straightforward comparison of different approaches, rapid development of new techniques and easy synthesis of known techniques.

The parsing module in PET is a bottom-up chart parser. The parsing process is guided by the parsing tasks on an *agenda*. A parsing task represents the combination of a passive chart edge and an active chart edge or a rule. Tasks are popped from the agenda. The parser terminates either when the task agenda is empty. The outputs of the PET parser are tree-like structures in brackets notation. The parse tree includes the phrase structure rules information which was used at each step of the parsing process. The Figure 4.1 below shows the notation of a correctly parsed derivation tree of the sentence "*welcome to the jotunheiment!*" [5].

[5] we omit some notation in this derivation tree because of the lengthy.

```
[3000011:0] (active)

(ROOT_FRAG
 (195 FRAG_ADJ -1.85114 0 4
  (194 HADJ_I_UNS -2.03375 0 4
   (30 WELCOME_A1 0.747602 0 1
    ("welcome" 28
     "token [ +TRAIT native_trait ...])
   (193 HCOMP -4.04071 1 4
    (55 TO -0.910553 1 2
     ("to" 23
      "token [ +ID *diff-list* +FROM ...])
    (192 HSPEC -3.70619 2 4
     (81 THE_1 -0.427019 2 3
      ("the" 24
       "token [ +ID *diff-list* +FROM ...])
     (191 PUNCT_BANG_ORULE -2.59518 3 4
      (190 SING_NOUN_IRULE -2.23084 3 4
       (88 JOTUNHEIMEN 0 3 4
        ("jotunheimen!" 29
         "token [ +TRAIT native_trait +CLASS ...)))))))))
```

Figure 4.1: The brackets notation of a derivation tree

The first line includes the unique ID number 3000011, the reading number and the notation of whether this parsed result is correct (active and inactive). The character enclosed by the quotation marks is the terminal node. The capitalized words are the combining rule schemata during the parsing process.

4.3 The Feature sets

As we discussed in chapter 3, discriminative statistical parse selection model uses valuable features which extracted from the training treebank to distinguish the correct parse result from a bunch of candidates. Some feature templates such as local derivational configurations and active edges were introduced before and has proved to be effective.

There are three state-of-the-art feature templates defined over HPSG derivation trees – local configuration, active edge and n-gram. Their usefulness has been investigated and published [Toutanova et al., 2005], [Zhang et al., 2007]. They all require exhaustive search in every node of the parse trees to generate their corresponding features. The *local configuration* is the local sub-tree. This kind of features shows the full information (the daughter(s) and itself) in a local sub-tree. *Active edge* records only one of the daughters in turn which allows reducing the effects of data sparseness.

N-gram records n-gram of lexical types, extracted from the pre-terminal of the derivation trees. Features extracted using this feature template can be extended with lexicalized information. Table 4.2 shows the examples of these three stat-of-the-art features extracted from the derivation tree in Figure 4.2. The n-gram examples here have addition of surface tokens. The integer that starts each feature indicates the degree of grandparenting level in the case of local configuration and active edge or the n-gram size. If the grandparenting level is greater than 0 then the feature is non-local. This is the special property of discriminative model.

Feature type	Sample Features
local configuration	`0 HADJ_I_UNS aj_i_le HCOMP`
local configuration	`0 HCOMP p_np_i-num_le HSPEC`
local configuration	`1 HADJ_I_UNS HCOMP p_np_i-num_le`
active edge	`0 HCOMP p_np_i-num_le`
active edge	`1 HCOMP HSPEC d_the_le`
n-gram	`1 n_pn_le jotunheime`
n-gram	`2 d_the_le n_pn_le jotunheime`

Table 4.2: Examples of structural features extracted from the derivation tree in Figure 4.2.

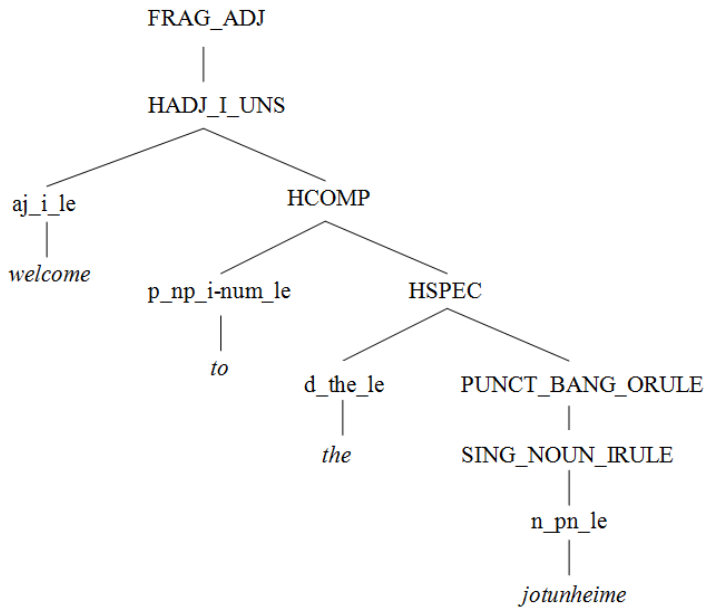

Figure 4.2: Sample HPSG derivation tree for the sentence "*welcome to the jotunheime*". Phrasal nodes are labeled with identifiers of grammar rules, and pre-terminal lexical nodes with the lexical types of ERG.

In order to capture the most distinguished properties of parses, we examine different set of features.

4.3.1 Baseline

Refer to the previous work we discussed in chapter 3, the study of [Toutanova et al., 2005] demonstrate the usefulness of building models over derivation trees of HPSG analyses. The result of using ancestor information improved parse ranking accuracy significantly. Also, [Zhang et al., 2007] suggests that properties of larger contexts and especially grandparenting can greatly improve parse selection accuracy. Here we adopt the conditional log-linear model LPCFG-A in [Toutanova et al., 2005] with three level expansion of its ancestors. Base on previous works and our experiences, three level expansion of grandparenting in English grammar performs higher efficiency and effectiveness.

The conditional log-linear model LPCFG-A has one feature for each expansion of each non-terminal in the derivation trees $A_i \rightarrow \alpha_j$. For a derivation tree t, it has as value the number of times this expansion occurs in the tree. As we discussed in chapter 2, this model has a correspond parameter λ_{ij} for every expansion $A_i \rightarrow \alpha_j$. Figure 4.3 shows the sample features of Figure 4.2 in our baseline model. The integer that starts each feature indicates the degree of grandparenting level. The ancestor of each feature expanded up to level three in this model.

```
0  HCOMP p_np_i-num_le  HSPEC
1  HADJ_I_UNS  HCOMP p_np_i-num_le  HSPEC
2  FRAG_ADJ  HADJ_I_UNS  HCOMP p_np_i-num_le  HSPEC
0  HSPEC  d_the_le  PUNCT_BANG_ORULE
...
3  FRAG_ADJ  HADJ_I_UNS  HCOMP  HSPEC  d_the_le  PUNCT_BANG_ORULE
```

Figure 4.3: Sample features of Figure 4.2 in baseline model

4.3.2 Discriminant sets

After establish the baseline, we would like to extract different features from the resource we've got from the PET parser to extend our disambiguation model. The main idea of this work is investigating the contribution of syntactic information in the parse selection model. As we talked in the previous section, [Toutanova et al., 2005] mainly focus on building models over derivation trees of HPSG analyses. They also conclude the standard phrase structure trees perform poorer result than the features defined over derivation trees. We combine the derivation tree and phrase structure tree in our first model, and we explore other features within the full HPSG sign which are useful for disambiguation. Three different discriminant sets are proposed as follow.

- Model 1 (baseline + syntactic categories)

This model is our primary idea; attaching phrase category information to the HPSG derivation result. Since both of the phrase structure tree and derivation tree are useful in the parsing disambiguation model. We assume that the more extensive features with the potential to provide greater information to aid parse disambiguation. The derivation tree records the combining rule schemas of the HPSG grammar which were used in the phase of construct possible reading for each sentence. Figure 3.1 shows the example of derivation tree of sentence "*Do you want to meet on Tuesday?*". The internal nodes represent, for example, head-adjunct, head-complement and head-specifier schemas. Although in [Toutanova et al., 2005], the derivation trees have proven to be quite effective for parse disambiguation. It is still not enough with only use the information in derivation tree in statistical parse disambiguation model. There are two reasons, 1) HPSG is a broad coverage grammar, which the restrictive potential of grammar constraints is fairly low because of a large number of alternatives has to be accommodated. This loss of constraining information needs to be compensated for. 2) Our system is integrated with a Maximum Entropy estimation toolkit (more detailed introduction in Section 4.4). It accepts the event files which contain the relationship between each parse candidate and their features in a special format as input. After training, it returns the discriminative model with real-valued weight of every inputted feature. The insufficient information might cancel out or mix up the effect of positive features. As we mentioned in Section 3.1, the main idea of statistical parse selection model is to find the useful features to help the re-ranker find the correct parse result from all the candidates. The features which combine

information from both derivation tree and phrase structure trees are more precise and make good effects.

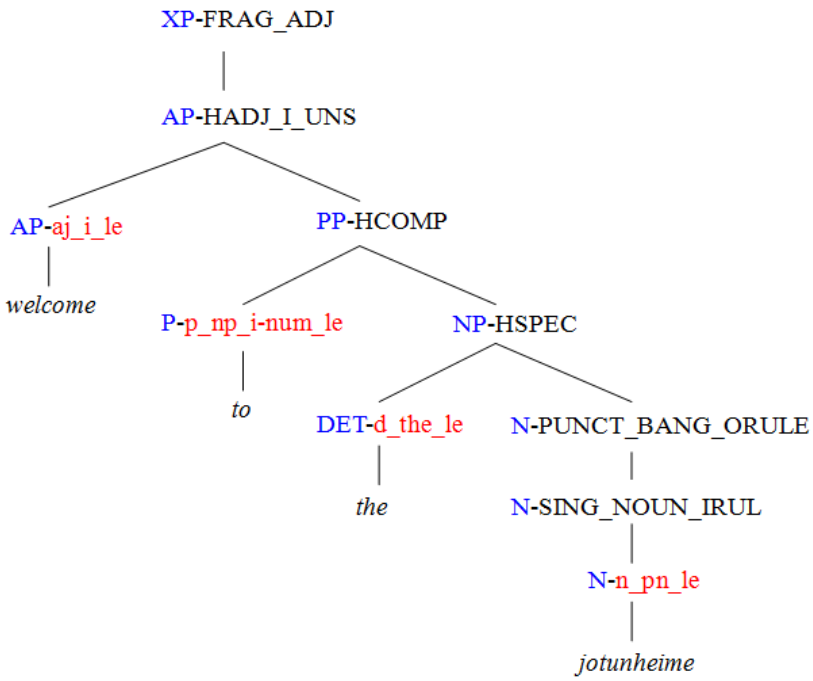

Figure 4.4: Composite tree of derivation tree and phrase structure tree of sample sentence "*welcome to the jotunheime*".

The graph above shows the composite tree of sentence "*welcome to the jotunheime*". The color blue indicates the syntactic category information, which is the mainly difference from the features in our baseline model. We simply combine those information in the derivation tree and phrase structure tree of the identical reading in the same sentence. Integrating the internal nodes and extract the features by using the feature extractor we are going to talk about in the next section. The format is same as the feature in our baseline model (see Section 4.3.1). Figure 4.5 shows the sample features of model 1 (baseline + syntactic categories) from Figure 4.4. The format is the same as baseline.

```
0  PP-HCOMP  P-p_np_i-num_le  NP-HSPEC
1  AP-HADJ_I_UNS  PP-HCOMP  P-p_np_i-num_le  NP-HSPEC
2  XP-FRAG_ADJ  AP-HADJ_I_UNS  PP-HCOMP  P-p_np_i-num_le  NP-HSPEC
0  NP-HSPEC  DET-d_the_le  N-PUNCT_BANG_ORULE
...
3  XP-FRAG_ADJ  AP-HADJ_I_UNS  PP-HCOMP  NP-HSPEC  DET-d_the_le
   N-PUNCT_BANG_ORULE
```

Figure 4.5: Sample features of Figure 4.4 in model 1

- Model 2 (model1 + sibling information)

In parsing, every node has a vertical history, including the node itself, parent, grandparent, and so on. A reasonable assumption is that only the past *v* vertical ancestors matter to the current expansion. Similarly, only the previous *h* horizontal ancestor matter. [Klein and Manning, 2003] has proved that the horizontal information also improved the parsing accuracy, they got the best improvement when the parent annotation corresponds to $v = 3$ and $h <= 2$. The horizontal information is the sibling information in our model. We first test the horizontal information in level 1, which means the sibling information in the local tree. The LinGo ERG is maximally binary grammar, with extensive use of unary schemas for implementing morphology and type-changing operations. The head-initial or head-final property will help us to define the new features. The sibling information will be added on the head-daughter branch. Figure 4.6 shows the sample features of model 2 (model 1 + sibling information) extracted from Figure 4.4.

```
0  AP-HADJ_I_UNS  AP-aj_i_le:r-PP  PP-HCOMP
1  XP-FRAG_ADJ  AP-HADJ_I_UNS  AP-aj_i_le:r-PP  PP-HCOMP
0  PP-HCOMP  P-p_np_i-num_le:r-NP  NP-HSPEC
1  AP-HADJ_I_UNS  PP-HCOMP  P-p_np_i-num_le:r-NP  NP-HSPEC
...
0  NP-HSPEC  DET-d_the_le  N-PUNCT_BANG_ORULE:l-DET
...
0  N-PUNCT_BANG_ORULE:l-DET  N-SING_NOUN_IRUL
1  NP-HSPEC  N-PUNCT_BANG_ORULE:l-DET  N-SING_NOUN_IRUL
```

Figure 4.6: Sample features of Figure 4.4 in model 2

We can see from the ERG grammar and Figure 4.4, for example, the HADJ_I_UNS rule is a head-initial rule. We then add the sibling information to its head-daughter branch (AP-aj_i_le) automatically. The characters after the colon indicate the position and the syntactic category of the sibling information. We use *r* and *l* to express whether the head-daughter has a right or left sibling. The blue and capitalized characters indicate the syntactic category of the sibling branch. We also retain the new label when we extend the grandparenting level.

Such features provide a reasonable approximation of trees that implicitly encodes many of the interesting relationships that are typically gathered from them, such as grandparent and sibling relations. They also capture further relationships that cross the brackets of the actual tree, providing some more long-distance relationships than the configurational features.

- Model 3 (Model1 + co-ordinate information)

After illustrate the two models above, we go a step further to look in-depth of the performance of individual HPSG rule and discovered that the coordination rules generally get an ill score. The long sentences usually include complex construction, such as long-distance dependencies, co-ordinations and insertions. Coordinate structures are a potential source of syntactic ambiguity in natural language, since their interpretation directly affects the meaning of the text. Some studies have been focusing on coordination disambiguation and suggested that conjuncts in coordinate structures have syntactic or semantic similarities.

Coordination disambiguation consists of the following two tasks, the detection of coordinate conjunctions, and finding the scope of coordinate structures. We start at adding more features to improve our disambiguation model by simply finding the scope of coordinate structures. The new feature we are going to introduce here is the measurement of the divergence in the size of coordination constituents. The coordination rule was used to combine two similar phrases. If we have a constituent (XP1 COORD XP2) in English, we assume that in a normal condition, each of the two or more conjuncts belongs to the same category (at some relevant level of abstraction) and similar length as the others. We calculate the remainder of the scope between different conjuncts and divide those new features into 5 different levels

according to the difference. Figure 4.7 shows a simple phrase which uses co-ordinate word.

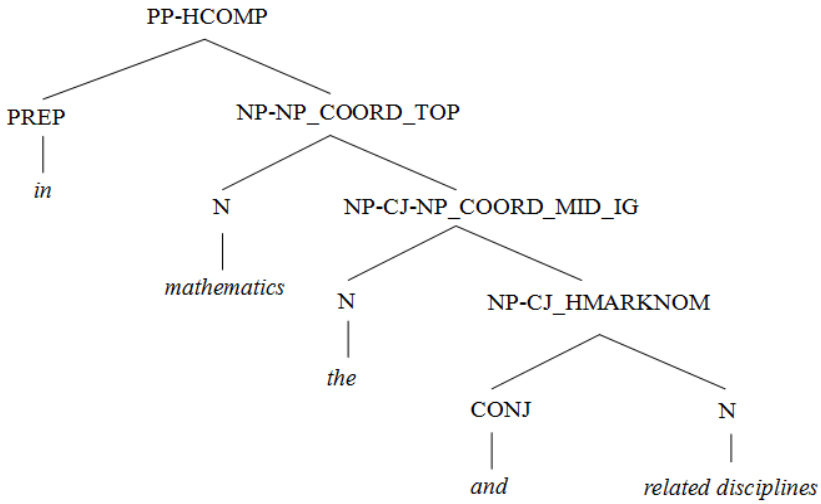

Figure 4.7: A simple co-ordinate phrase.

In this example, we first separate the phrase into three parts; *mathematics*, *linguistics* and *related disciplines*. The next step is to calculate the difference of length between every two sub-phrase from the button of the parse tree and add it as our new feature in model 3. Figure 4.8 shows the sample features of model 3 (model 1 + co-ordinate information) extracted from Figure 4.7. We first calculate the difference of *related disciplines* and *linguistics* and add a new feature *NP-CJ-NP_COORD_MID_IG 1*. Then we go one level upper and find the new feature *NP-NP_COORD_TOP 0* by comparing the difference of length of *mathematics* and *linguistics*.

```
NP-CJ-NP_COORD_MID_IG   1
NP-NP_COORD_TOP  0
```

Figure 4.8: sample features of Figure 4.7 in model 3

4.3.3 Feature Extractor

The process of feature extraction is described by Figure 4.9. There system extracts three different feature sets according to the schema we introduced above for each reading of every sentence in training dataset. Since we have to extract features such as parenting or sibling levels from the parse tree, we need the dynamic programming

methods of storing all the relevant information in charts or hash tables and further process them to retrieve the desired property. This phrase is the most time-consuming part in the process.

```
1:  featureSet = null
2:  for each sentence in training set do
3:    for each reading in sentence do
4:      GET pure derivation tree
5:        Replace the lexical entry by the unique lexical type in ERG
6:        GetFeature (derivation tree)
7:      GET phrase structure tree
8:        new_tree = Merge (derivation tree, phrase structure tree)
9:        GetFeature (new_tree)
10:     Add feature to featureSet if it doesn't exist
11:   end for
12: end for
```

Figure 4.9: The pseudo code of our feature extractor

First, we replaced the pre-terminal lexical entry by the unique HPSG lexical types, the native word classes in the grammars. Lexical types are used to categorize lexical entries into groups that share certain characteristics. These lexical types can be viewed as a very extensive set of Part-of-Speech tags. Lexical rules might be applied to those types in order to extend the grammar to understand inflections and derivations of different words without having to add all forms of each word to the lexicon. The name of lexical type represents its part-of speech, complements and optional annotations which determine most of its properties. As a few examples from the English grammar, v_np_le represents a basic transitive verb, n_pp_c-of _le represents a count noun that optionally takes a prepositional phrase complement headed by *of*. Lexical types play a more specific role than pre-terminal lexical entry.

4.4 Tools

The methods proposed in this dissertation were implemented in the following software packages. They are all available on-line.

4.4.1 A Maximum Entropy Ranker

Maximum entropy model has been applied to a wide variety of problems in natural processing. However, the feature set for typical natural language tasks are usually large. Estimation of such large models is not only expensive, but also can easily lead to very unexpected and erroneous results. Hence, our system uses the *Toolkit for Advanced Discriminative Modeling* (TADM) [6] to complete the parameter estimation work. It is based on the open-source estimate package for Maximum Entropy estimation by Rob Malouf [Levine and Meurers, 2006]. TADM accepts event files as input and returns the discriminative model in the form of real-valued feature weights (parameters) after training. Example of an event file is as following:

```
2
1 3 0 1 1 2 3 1
0 2 0 3 2 1
3
1 1 3 1
0 2 0 2 2 1
0 1 2 1
```

The TADM toolkit needs an events file which contains the feature distribution in our training dataset as input. The first part of an event file is a header, bracketed by lines containing "header" and "/". The header is optional and, if present, is ignored. The remaining of the file consists of one or more blocks. The first line of each block is the number of events (i.e. parse candidates) for the corresponding context (i.e. sentence). The example denotes 2 and 3 events for their corresponding context in the above example. The next lines are the event lines. Each event line has a marker which is either 1 (gold standard parse) or 0 (non-gold standard parse) in the beginning. There will be only one gold standard parse event (marked as 1) in any context in our study. After the marker is the number of feature-value pairs, then the pairs of feature ID and value (the number of times of its corresponding feature which has been observed in

[6] http://tadm.sourceforge.net

this event). Each feature can appear only once in an event and the ID are numbered starting with zero. Any feature with an expected value of zero is simply ignored.

Our feature extractor module analyses every sentence in our training dataset which contains active parse result and extract the feature sets based on section 4.3 as the input events file. The TADM produces an output file which contains the corresponding feature weight of our input feature for our discriminative model.

4.4.2 EVALB

We used the EVALB [7] parseval reference implementation (updated version by David Brooks, 2005), for bracket scoring. The EVALB uses the PARSEVAL measures [Black et al., 1992] to compare performance. It is a more exact measure, looking at single constituents in the parsed result instead of just calculate how many sentences are correctly parsed. It evaluates bracketing accuracy in a test-file against a gold-file. There are three criteria must be met for a trace to be "correct": (1) it must be an argument to the correct head-word; (2) it must be in the correct position in relation to that head word (preceding or following); (3) it must be dominated by the correct non-terminal label [Collins, 1997], [Collins, 2003].

It returns precision, recall, tagging accuracy by:

$$\text{Labeled Precision (LP)} = \frac{\textit{number of correct constituents in proposed parse}}{\textit{number of constituents in proposed parse}}$$

$$\text{Labeled Recall (LR)} = \frac{\textit{number of correct constituents in proposed parse}}{\textit{number of constituents in proposed parse}}$$

$$\text{F — score} = \frac{2 * LP * LR}{LP + LR}$$

By using this evaluator, we can get a more specific outcome of our model rather than the result of exact match. It goes deep into the structure of each parse tree to see if the

model suggests an approximate analysis tree and reports the harmonic mean of labeled precision and labeled recall – F score.

[7] http://nlp.cs.nyu.edu/evalb/

Chapter 5

Results and Analyses

This chapter presents the results of the experiments using our system which has been introduced in Chapter 4. In this work, we examine several different feature sets which extracted from the output of a statistical parser (derivation trees and phrase structure trees) in a Maximum Entropy Based discriminative ranking model to estimate their potentiality to be used for parse disambiguation.

As we talked in Section 4.1, our test data is the subset named ws13 in WeScience which contains 890 sentences. We first trained our model on our training dataset (17,225 sentences) and used development set (955 sentences) to optimize our model. The performance of disambiguation models based on different feature set and the evaluation methods we used will be presented in Section 5.1. We will report the exact match accuracy and the F1-scores based on the set of constituents of a parse. We conclude this chapter with a brief discussion that lights the strength and weakness of our approach.

5.1 Evaluation

Several measure methods can be applied to evaluate the performance of a parser. One of the example methods is *Exact Match*, which intuitively measures how much of the preferred reading matches the gold standard. Suppose that a parser produces a set of possible analyses $T = \{ t_1, t_2, t_3, \ldots, t_n \}$ of a sentence W. The gold tree τ' is one of the candidates in T. In case that the preferred parse result τ is the same as τ', we simply count this disambiguation task success. The exact match method is a very strict measurement that counts how often the classifier chooses the correct parse. The accuracy is the precision (in percentage) of the best pick, which was calculated by the equations below:

$$W \rightarrow T_w = \{t_1, t_2, t_3, ..., t_n\}$$
$$D(W, T_w) = \tau \in T_w$$
$$G(W, T_w) = \tau' \in T_w$$
$$ExactMatch = \frac{\left\| \{W \mid D(W, T_w) = G(W, T_w)\} \right\|}{\left\| T \right\|}$$

Exact match permits the parser only in the case it gets the parse tree exactly right. It is useful for parse selection evaluation, but it does not give part-credit for partially correct parses to observe how close the preferred parse was to the gold standard. Since parsing is a very detailed analysis, it is possible to get one piece wrong and still have useful information in the rest part. For this reason, the PARSEVAL measures have been defined. This measurement do not only evaluate a parse tree as a whole, but evaluate its component pieces — the set of constituents of a parse. Therefore, also trees which are partially correct are awarded. We use EVALB (as mentioned in Section 4.4.2); a widely used tool which is a bracketing scoring program based on the PARSEVAL measures for parsing performance evaluation.

5.2 Statistical significance

In statistical method, we need to test and verify the result is statistically significant to ensure that this result is unlikely to have occurred by chance. The amount of evidence required to accept that an event is not arose by chance is known as the *significance level* or *p-value*. In traditional statistical hypothesis testing, the p-value is the probability of observing data at least as extreme as that observed, given that the null hypothesis is true. Popular levels of significance are 10% (0.1), 5% (0.05), 1% (0.01), 0.5% (0.005) and 0.1% (0.001). The lower the significance level, the stronger the evidence required. In statistical parse disambiguation, we accept level of 5% (0.05) as a significance level (α-level). If the obtained p-value is lower than the α-level, the null hypothesis[1] is thus rejected.

[1] The null hypothesis typically corresponds to a general or default position. In this case, the hypothesis assumes that the two measured phenomena produce the observed results are the same. If we get a p-value which lower than 0.05, that means there is only 5% chance the hypothesis is true.

However, along with the development of statistical parsing application, a persistent problem is that the dubiety of the increases in evaluation metrics like labeled bracket recall and precision are statistically significant or not. To address this issue, Dan Bikel has develop a toolkit — Randomized Parsing Evaluation Comparator[2] that reads the output of EVALB on two different parsing runs and outputs p-values for whether observed differences in recall and precision are statistically significant. In this toolkit, the null hypothesis is tested by randomly shuffling individual sentences by employed a type of stratified shuffling, which is a type of compute intensive randomized test. The comparator scores between the two models we inputted and then re-compute the evaluation metrics. If the difference in a particular metric after a shuffling is equal to or greater than the original observed difference in that metric, then the p-value for that metric is incremented.

5.3 Results

We report parse disambiguation results on the dataset which extracted from LOGON and WeScience. These two corpora are the annotated corpus for either exactly one analysis was chosen as gold standard by the annotator, or none of the analyses was chose as correct. We select sentences in total 19,070 which have exactly one correct parse result as our training, development and test datasets in order to be able to evaluate the quality of our parse selection model. Refer to Section 4.1 for more detailed information.

Our test data contains 890 sentences in the WeScience corpus. We trained the feature sets on 17,225 sentences which comprised of both corpora. The training set has the average number of 251 reading per sentences and the average length per sentence is 14.46. The development set (subset tg2 in LOGON corpus) was used to fine-turn the methods, and the final results presented in this work are with respect to the test set. The average number of reading per sentences in the test dataset is 269 and the average length per sentence is 15.94, which is much higher than the training dataset.

Table 5.1 shows the accuracy for various models of our parse selection model obtained on development dataset. The exact match column shows the percentage of cases that the preferred parse tree ranked by the model is exactly the same with the

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

gold standard tree according to the treebank. The figures in label bracketing were calculated by the EVALB tool. From the result, we can see around 2.4% improvement of exact match accuracy were achieved by adding syntactic categories to the feature set. The different setting of feature template caused the expansion of features space about quarter of the original size. The p-value of Baseline model and Model 1 which calculated by the Randomized Parsing Evaluation Comparator is 3.7% in precision and 5% in recall. We can verify the refinement is statistically significant. That is to say the composite of grammar rules which were used during the parse phase for each sentence and their corresponding syntactic categories is advantageous indeed in parse disambiguation task.

Models	# features	Exact match accuracy	Label bracketing		
			Precision	Recall	F-score
Baseline	2,537,755	41.88%	86.29%	86.46%	86.37%
Model 1	3,309,888	44.24%	86.98%	87.12%	87.05%
Model 2	5,270,173	45.29%	87.29%	87.47%	87.38%
Model 3	3,310,465	43.19%	86.97%	87.01%	86.99%

Table 5.1: Accuracies obtained on development dataset using Baseline model (local configurations with grandparenting level up to 3), Model 1 (baseline model with syntactic categories), Model 2 (model 1 with sibling information) and Model 3 (model 1 with co-ordinate information).

The exact match accuracy from Baseline model to Model 2 got 3.41% improvement.
The increase from Baseline model to Model 2 got 3.41% in exact match accuracy and 1% in the f-score of labeled bracket. The reason why the improvement of labeled bracket is always lower than the exact match accuracy is that the position cause different analysis results often happen in the higher level of a parse tree. Sometimes only one different rule schema can cause several different parse results. But the syntactic inflection rules, basic schemas such as head-specifier are mostly been analyzed in a correct way. One small piece of correct selection leads the success in exact match but not much effect in labeled bracket evaluation. The p-value between the Baseline model and Model 2 is 3.4% in precision and 4.2% in recall. This result is accepted as statistically significant and proves the effectiveness of horizontal ancestor for a parse selection model.

After the verification of the effectiveness of our feature templates by experiment results in development dataset. We can use the same setting to run the experiment on the test dataset. Before we go further to the next step, the data shown in Table 5.1 indicates that a huge feature spaces will obstruct the efficiency of our disambiguation model. We simply remove the features which have no effect with the ranking model for the experiment of test dataset.

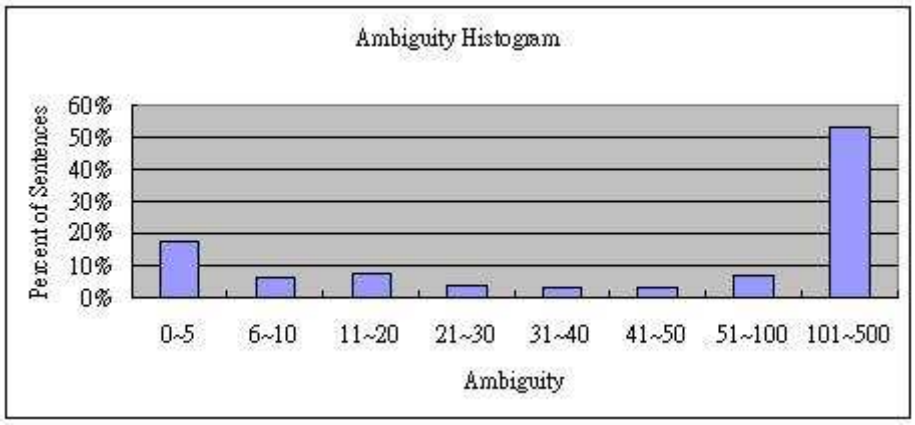

Figure 5.1: The distribution of ambiguity in test dataset.

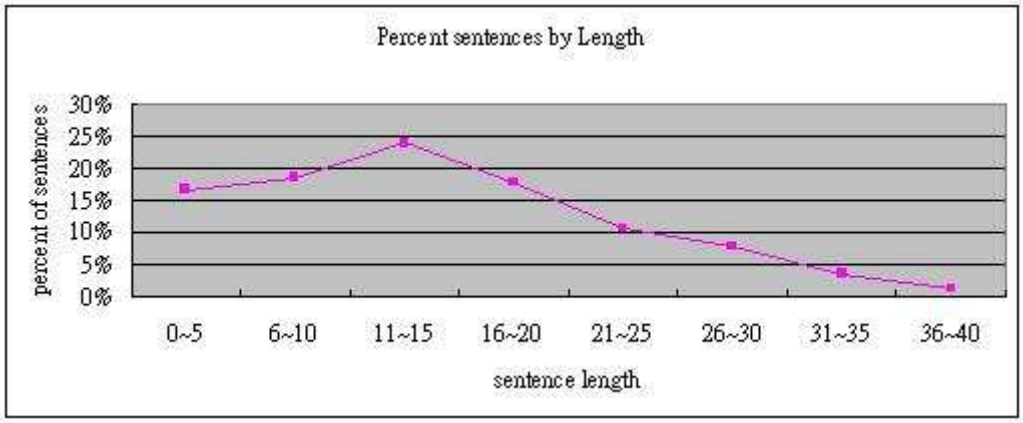

Figure 5.2: The distribution of sentence length in test dataset.

Figure 5.1 and Figure 5.2 show the ambiguity level and the distribution of sentence length in test dataset. There are more than 50% of sentences have more than 100 possible reading. The average length per sentence is 15.94, which is a quite high number compare to other relative work. This also shows the difficulty of selecting correct analysis in this kind of data.

Models	Exact match accuracy	Label bracketing		
		Precision	Recall	F-score
Random	6.49%	74.28%	75.84%	75.05%
Baseline	42.82%	86.99%	86.96%	86.97%
Model 1	44.82%	87.5 %	87.46%	87.48%
Model 2	45.19%	87.84%	87.75%	87.8 %
Model 3	44.94%	87.51%	87.44%	87.48%

Table 5.2: Accuracies obtained on test dataset using different models.

Table 5.2 shows the performance of various models in testing dataset. We added a random choose model here. The random model was executed by a random number generator, which chooses a random number in the range from 0 to the number of possible parse tree minus 1 (because of the serial number in parse candidates is starting from 0). This model got an extreme low accuracy in the exact match evaluation model. The reason is that the number of possible analyses per sentence is pretty high in our corpus (more than 50% of sentences have over 100 candidate trees). The succeed probability of blind guessing in such data environment is very low.

As mentioned before, we use a smaller feature sets which have moved all the non-effective features in each model in a more efficient purpose. We can see the actual improvement from the Baseline model to the other three models defined in this work. However, the outcome of Model 3 seems not much different with Model 1. As we talked in Section 4.3.2, the coordinate structure is common used in nature language. It is used to combine two or more similar phrases. We use the divergence in the size of coordination constituents as features to help the disambiguation model. There are 577 coordinate features be extracted from the training data and added to Model 3. Comparing to the feature set in Model 1, the number of new features is next to nothing. That is the reason why the coordinate feature did not give significant improvement. The effects are diluted by a mass number of other features. Moreover, the difference of the word length is not the only characteristic in coordination. We can add more features such as the similarity of their syntactic categories or semantic information of constituents.

5.4 Discussion

In parse disambiguation experiment, the decision is based on the candidate parse trees produced by the original parsers. We investigated the importance of the syntactic information to the task of ranking different parses. We examined four different approaches and estimate their possible contribution. They distinguished themselves by the features used for selecting: the local configuration with ancestor (the baseline), the baseline features with syntactic categories, the baseline information with syntactic categories and sibling information and the baseline features with syntactic categories and co-ordinate information. What all those approaches have in common is that they all based on the effectual feature template which proven by previous researches and adding extra information to examine the effectiveness of new feature set.

Our ultimate goal is to improve the performance of parse disambiguation model. We successfully demonstrate the effects of our parse disambiguation model. [Toutanova et al., 2005] claims that the information of phrase structure tree does not perform positive contributes on parse selection task. The result in this work shows that the phrase structure tree produced by a deep parser is not useless but need to be used in a correct way. However, as shown in Table 5.1, the number of feature increases substantially along with the syntactic categories and sibling information. The heavy feature sets will cause the low efficiency of our disambiguation model. Efficiency issues are important because the parsing task is a fairly large problem, often involving more then 500 possible parse trees depending on the parser. In the future work, we need to discard the features with trivial weight gradually and observe the change of performance in order to speed up the process.

Chapter 6

Conclusion and Outlook

In this work we present a new idea of integrating syntactical information with the rule schemata of HPSG which have been proved as useful in parse disambiguation task. We define several feature templates which captures the crucial differences between the correct analysis and a bunch of ill analyses. Those features can be applied in a Maxima Entropy Model easily and improve the disambiguation accuracy. We used the DELPH-IN collaborations set of tools, the grammar (the English Resource Grammar), parser (the PET parser) and classifier (the TADM maximum entropy model).

Using the probabilistic model as classifier, receiving the outputs of a parse as an input allows us to access non-local features easily and also provides the flexibility to include as much context as needed. In addition, it does not require extensive modification of the code of the PET parser, and allows us to examine different modules without needing to re-parse the data-set for each module. However, although convenient for research purposes, using the statistical models offline has some disadvantages. It depends on the set of candidates that we receive from the parser, and it does not contribute to the efficiency of the parser itself.

During our investigation, there is a question keeping perplex us. Whether the development of statistical parse disambiguation by features extracted from syntactic information (in a given treebank) has reached the limit amount of improvement? It raises other questions, where there is any extra or more information can be used to help the parse selection problem; and if so then how to extract and exploit those information. In order to approach the original question, we have done our best to investigate the possibly helpful feature sets in this specific field as we described in Chapter 4 and report the results in the previous sections. Pioneering works by [Collin, 2003] and [Toutanova et al., 2005] show that it seems quite hard to get dramatic

improvement by only using the syntactic information.

Traditionally, parse disambiguation has relied on structural features extracted from syntactic parse trees. In recent year, many researchers have done experiments that use semantic information in parse selection. [Toutanova et al., 2005], [Fujita et al., 2007], [Eneko Agirre et al., 2008] and [Ben-Gera et al., 2010] used various semantic related information such as semantic relation extracted from Minimal Recursion Semantics (MRS), manually annotated word sense and semantic class gathered from WordNet[3] or related corpora. Some of them got significant improvement but some did not. The main difference between them is that the former works used simpler semantic features without sense information, and got a far less dramatic improvement.

Recently, significant improvements have been made in combining symbolic and statistical approaches to various natural language processing tasks. Recall to the description of symbolic in Section 2.2, this approach uses manual analyzed constraints to restrict the possible ambiguous structures generated by the grammar. [Fujita et al., 2007] and [Eneko Agirre et al., 2008] explored the effect of lexical semantics on parse selection. The use of hand-crafted lexical resources sometimes restricted by the reason of such resources are hard to produce and scarce. However, it is worth to develop such lexicons for further research and many of them have already been created in various languages which have large treebanks.

To conclude, the reported results indicate a method of utilizing syntactical context information for parse disambiguation task. The positive outcome of this work regarding the success of improving parse disambiguation model by testing on various syntactic information. Although a significant improved results were obtained with the current modules, there is still place for improvement in several parts of the system.

[3] A lexical database for the English language which developed by Princeton University from 1985. http://wordnet.princeton.edu/

Bibliography

[Abney, 1997] Steven Abney. Stochastic attribute-value grammars. *Computational Linguistics*, 23:597–618.

[Ben-Gera et al., 2010] Danielle Ben-Gera, Yi Zhang and Valia Kordoni. Semantic Feature Engineering for Enhancing Disambiguation Performance in Deep Linguistic Processing. In Proceedings of LREC-2010, Valetta, Malta.

[Bender et al., 2002] Emily M. Bender, Dan Flickinger and Stephan Oepen. The Grammar Matrix: An Open-Source-Kit for the Rapid Development of Cross-Linguistically Consistent Broad-Coverage Precision Grammars. *In: Workshop on Grammar Engineering and Evaluation at the 19th International Conference on Computational Linguistics, Taipei, Taiwan*, pp. 8–14.

[Bresnan, 1982] Joan Bresnan, editor. *The Mental Representation of Grammatical Relation*. MIT Press, Cambridge, MA.

[Black et al., 1992] Ezra Black, Fred Jelinek, John Lafferty, David M. Magerman, Robert Mercer and Salim Roukos. Towards history-based grammars: Using richer models for probabilistic parsing. *In Proceedings of the Fifth DARPA Speech and Natural Language Workshop, Harriman, NY.*

[Bright, 1992] William Bright. *International encyclopedia of linguistics*. Oxford University Press.

[Callmeier, 2000] Ulrich Callmeier. PET – A platform for experimentation with efficient HPSG processing techiques. Natural Language Engineering.

[Charniak and Johnson, 2005] Eugene Charniak and Mark Johnson. Coarse-to-fine nbest-parsing and maxentdiscriminative reranking. *In Proceedings of ACL-05, pages 173–180, Barcelona.*

[Collins, 1997] Michael Collins. Three generative, lexicalised models for statistical parsing. *In Jonathan David Bobaljik and Tony Bures, editors, Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL), pages 16–23, NJ, USA.*

[Collins, 2003] Michael Collins. Head-Driven Statistical Models for Natural Language Parsing. *PhD Dissertation, University of Pennsylvania.*

[Copestake and Flickinger, 2000] Ann Copestake and Dan Flickinger. An open-source grammar development environment and broad-coverage English grammar using HPSG. *In Proceedings of the Second International Conference on Language Resources and Evaluation (LREC),* pages 591 – 600, Athens, Greece.

[Copestake, 1992] Ann Copestake. The ACQUILEX LKB: representation issues in semi-automatic acquisition of large lexicons. *Proceedings of the 3rd Conference on Applied Natural Language Processing, Trento, Italy:* 88–96.

[Copestake et al., 1995] Copestake, A., Flickinger, D.,Malouf, R., Riehemann, S., & Sag, I. *Translation using minimal recursion semantics*. In Proceedings of the sixth international conference on theoretical and methodological issues in machine translation. Leuven, Belgium.

[Cramer and Zhang, 2010] Bart Cramer and Yi Zhang. Constraining robust constructions for broad-coverage parsing with precision grammars. *In Proceedings of COLING-2010, Beijing, China.*

[Eneko Agirre et al., 2008] Eneko Agirre, Timothy Baldwin and David Martinez. *Improving Parsing and PP attachment Performance with Sense Information*. Proceedings of ACL-08: HLT, pages 317-325, Columbus, Ohio, USA.

[Fellbaum, 1998] Christiane Fellbaum. WordNet: An Electronic Lexical Database. MIT Press, Cambridge, USA.

[Fujita et al., 2007] Senae Fujita, Francis Bond, Stephan Oepen, and Takaaki Tanaka. *Exploiting semantic information for HPSG parse selection.* ACL-07, 100.

[Hutchins, 1992] John Hutchins. *Introduction of Machine Translation.*

[Johnson et al., 1999] Mark Johnson, Stuart Geman, Stephen Canon, Zhiyi Chi and Stefan Riezler. Estimators for stochastic unification based grammars, *In Proceedings of the 37th Annual Conference of the Association for Computational Lintuistics, San Francisco. Morgan Kaufmann.*

[Jurafsky and Martin, 2006] Daniel Jurafsky & James H. Martin. *Speech and Language Processing: An introduction to natural language processing, computational linguistics, and speech recognition*. 2006.

[Kaplan et al., 2004] Ronald M. Kaplan, Stefan Riezler, Tracy Holloway King, John T. Maxwell III, Alexander Vasserman and Richard Crouch. Speed and Accuracy in Shallow and Deep Stochastic Parsing. *In Proceedings of the Human Language Technology Conference and the 4th Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL'04)*, pages 97–104, Boston, MA.

[Klein and Manning, 2003] Dan Klein and Christopher Manning. Accurate unlexicalized parsing. *In Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics.*

[Levine and Meurers, 2006] Robert D. Levine and W. Detmar Meurers. Head-Driven Phrase Structure Grammar: Linguistic Approach, Formal Foundations, and Computational Realization. *In Encyclopedia of Language and Linguistics, 2nd edition.*

[Lønning et al., 2004] Jan Tore Lønning, Stephan Oepen, Dorothee Beermann, Lars Hellan, John Carroll, Helge Dyvik, Dan Flickinger, Janne Bondi Johannessen, Paul Meurer, Torbjørn Nordgård, Victoria Rosén, and Erik Velldal. LOGON. A Norwegian MT effort. *In Proceedings of the Workshop in Recent Advances in Scandinavian Machine Translation*, Uppsala, Sweden.

[Manning and Schutz, 1999] Christoopher D. Manning and Hinrich Schutze. *Foundations of Statistical Natural Language Processing*. The MIT Press.

[Malouf, 2002] Robert Malouf. A comparison of algorithms for maximum entropy parameter estimation, *In Proceedings of the Sixth Conference on Natural Language Learning (CoNLL-2002)*, pages 49–55.

[Miyao and Tsujii, 2002] Yusuke Miyao and Junichi Tsujii. *Maximum entropy estimation for feature forests*. In Proceedings of the Human Language Technology Conference. San Diego, CA.

[Oepen et al., 2002] Stephan Oepen, Ezra Callahan, Dan Flickinger, and Christopher D. Manning. LinGO Redwoods. A rich and dynamic treebank for HPSG, *In Beyond PARSEVAL. Workshop of the Third LREC Conference*, Las Palmas, Spain. 2002.

[Pollard and Sag, 1994 (1)] Carl Pollard and Ivan A. Sag. *Head-driven phrase structure grammar*, Chicago: University of Chicago Press.

[Pollard and Sag, 1994 (2)] Carl Pollard and Ivan A. Sag. Information-Based Syntax and Semantics, Volume 1: Fundamentals.

[Sag and Wasow, 1999] Ivan A. Sag and ThomasWasow. Syntactic Theory – A Formal Introduction, *CSLI Lecture Notes* no. 92. CSLI Publications.

[Schabes, 1992] Yves Schabes. Stochastic lexicalized tree-adjoining grammars, *In Proceedings of the 14th International Conference on Computational Linguistics*, pages 426–432.

[Steedman, 2000] Mark Steedman. *The Syntactic Process.* The MIT Press.

[Toutanova et al., 2005] Kristina Toutanova, Christopher D. Manning, Dan Flickinger, and Stephan Oepen. Stochastic HPSG parse disambiguation using the redwoods corpus. *Research on Language and Computation*, 3(1):83–105.

[Velldal, 2008] Erik Velldal. Empirical Realization Ranking. *Ph.D. thesis*, University of Oslo, Oslo, Norway.

[Ytrestøl et al., 2009] Gisle Ytrestøl, Dan Flickinger, and Stephan Oepen. Extracting and Annotating Wikipedia Sub-Domains. Towards a New eScience Community Resource. *In Proceedings of the Seventh International Workshop on Treebanks and Linguistic Theories*, Groningen, the Netherlands.

[Zhang et al., 2007] Yi Zhang, Stephan Oepen and John Carroll. Efficiency in unification-based N-best parsing. *In Proceedings of the 10th International Conference on Parsing Technologies (IWPT 2007)*, pages 48–59, Prague, Czech.