The Hong Kong Polytechnic University

Department of Computing

A Probabilistic Approach to Natural Language Disambiguation: Semantic Role Labeling and Dialogue Act Recognition

By

LAN, Cyrus Kwok-Cheung

A thesis submitted in partial fulfilment of the requirements for the Degree of Master of Philosophy

April, 2006

Certificate of Originality

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma except where due acknowledgement has been made in the text

____

(Signed)

LAN, Cyrus Kwok-Cheung

(Name of student)

To my past mother

Abstract

Resolving ambiguities has been a central problem in natural language processing. Most disambiguation tasks to date have focused on relatively low level processing such as morphological, lexical, and syntactic analysis. Their considerable success has stimulated research in higher level, but harder, disambiguation tasks.

This thesis addresses two disambiguation tasks, one is at semantic level and the other is at pragmatic level. The tasks are referred to as, respectively, semantic role labeling and dialogue act recognition. We address both tasks using a probabilistic framework, which is in the form of conditional distribution $p(ambiguity|expression, context)$. We estimate the distribution by conditional Maximum Entropy, which allows heterogeneous sources of information to be integrated in a unified model for disambiguation. Based on the principle of Maximum Entropy, the selected distribution is of the highest entropy, where no unjustified assumption is made on the training data while keeping easy for feature modeling. Maximum Entropy has been empirically proved useful in various applications, with moderately effective training time.

In the semantic role labeling task, we propose a three-phase labeling approach to the problem. The approach combines advantages from previously proposed methods, while addressing their weaknesses. The approach decomposes the problem of recognizing a complex structure into several local decisions, each recognizing a single piece of the structure. The decisions are determined by supervised learning techniques, by training algorithms from data for prediction. Evaluations on public benchmarks show that our recognition performance is competitive with the current best individual system.

In the dialogue act recognition task, we target at non-task oriented recognition. We study various types of features, including lexical, syntactic, and discourse, to evaluate the recognition performance. A feature selection method is used for systematically optimizing the feature set. Experimental results show that our system outperforms all the other approaches that use the same public data set.

Despite the high micro-average performance achieved in both tasks, the macro-average performance is unsatisfactory. This is due to the class-imbalance problem in the data sets, where the distribution of examples among the classes is highly skewed. We employ two methods to address this problem in each task. One is over-sampling and the other is error-based learning. Experimental results showed that both methods are effective in improving the macro-average performance in most cases.

Acknowledgements

Many sincere thanks are owed here. First, I thank my supervisor Edward Ho, who has gave me genuine support since my undergraduate study. He has always been gentle, and consistently gave me freedom in pursuing what I believed to be true. I also thank him for his invaluable advice about academics and life in general. As a mentor, he was keen on dealing with my personal and emotional matters.

I thank Robert Luk and Hong-Va Leong for their useful comments and helpful suggestions about my work. I also thank their emergent financial support during my thesis time.

I thank my colleagues and officemates Jing-Yang Zhou, Jin Yang, Cane Leung, Wing-Sze Wong, and Chi-Shing Wang, with whom I had many interesting discussions about research and religion, as well as many wonderful trips during holiday season. In particular, I am grateful to Chi-Shing Wang, who spent much time discussing with me about technical issues on computational linguistics.

I thank my friends Kwok-Leung Lam and Gheorghe Chen for their constant support over years. I particulary appreciate them sharing my joys and sorrows during the past three years.

I thank my churchmates for their prayer when I was in struggling circumstances. A special thanks goes to Mr. Liu, who gave me valuable advice on whether I should pursue the master degree. I also thank his wife for her sincere support when I was in my dark time.

I must give my deepest thanks to the my family for all of their encouragement and financial support throughout the years. I thank my parents for their sacrifices to support my interest in computer during my childhood. I also thank my parents for teaching me the importance of values at my young age. Particularly, I have to thank my memorable mother. I would like to dedicate this thesis to her. She never got to see my graduation, but I am sure she would have been proud.

My most gratitude is to my girlfriend, Sophia Soo, for being always with me, for her love and understanding. I especially thank for her patient of all the days and nights which I have put in studying and working.

Finally, and most importantly, I thank my Lord for all that He has done and all that He will do. I also thank my Lord to let me know who owns the true intelligence when I was dealing with artificial intelligence techniques for my work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Challenge

Natural language (human language) is the most expressive way for information access. As opposed to artificial language (e.g., mathematical notations) being specialized for certain purposes, natural language is most widely used for communication. Moreover, as human, without unnecessary effort, we can speak or write as freely as we can, using ordinary and unconstrained natural language. However, systems seldom fully take advantage of natural language to improve their applicability. Most system interfaces usually only allow users to employ keyword-based query or complicated command for retrieving the desired information. Often, the retrieved information is too general, which requires further manual processing to find the specific answer. A compromise is to limit the usage of language, by providing users with pre-defined templates or linguistic rules, to fill in their information need. This might solve the under-specification problem to some extent, but the interface is still rigid, hardly providing enough flexibility to users. The ideal interface is to *understand* what the user speaks, and satisfy the information need directly. Such interface is the most straightforward way that provides the highest degree of interaction with users, and adapts to ever-changing situations.

A long-standing goal in the field of *natural language processing* (NLP) is to enable computer understanding of natural language. Much progress has been made in achieving this goal. However, even much remains to be explored, particularly in semantic and pragmatic level. In 1950, Alan Turing proposed the Turing test (Turing, 1950) for testing the capability of a computer to perform human-like conversation. He believed that by the year 2000, computers would be able to pass the test. It has already been 50 years since Turing's proposal, but no computer has passed the test. Some simple chatbots such as ELIZA (Weizenbaum, 1966) or ALICE [1] did demonstrate some human-conversational behavior, but they were still far from reaching the ultimate goal of understanding natural language.

One main barrier is that natural language, either written or spoken, is inherently *ambiguous*. Ambiguities appear at all levels of language processing. They can be lexical, where words can be understood in more than one way. They can also be structural, where words can be grouped differently. Furthermore, ambiguities can be referential, where an expression is being referred by more than one entity. Even worse, they can be related to usage, where the use of an expression is different from its literal meaning. Consider, for example, the very simple sentence *I saw her fish*. Without contextual information, there could be at least three possible meanings:

1a. I perceived some female's animal.

1b. I perceived some female who was fishing.

1c. I cut some female's animal.

In meaning 1a, the sentence is syntactically grouped as (S (NP *I*) (VP *saw* (NP *her fish*))), where S is a clause, NP is a noun phrase, and VP is a verb phrase. Equivalently, the syntactic grouping can be represented by a tree structure, as depicted in Figure 1.1a. However, within the context-free grammar, the same sentence can also be grouped as (S (NP *I*) (VP saw (S (NP her) (VP fish)))), or hierarchically represented in Figure 1.1b. This syntactic ambiguity thus results in meaning 1b. Besides, lexical ambiguity can also affect the sentence interpretation, as shown in meaning 1c. Here, the word sense of *saw* is interpreted as *cut* rather than *perceived*, although it has the same tree structure as that in meaning 1a (see Figure 1.1c).

Disambiguation is the process of resolving ambiguities. Since ambiguity is a central problem in NLP, the ability to disambiguate becomes the core requirement in reaching the goal of understanding natural language. Previously, various levels of disambiguation tasks were defined, forming their own research sub-community, to help in reaching the goal in a finer-scale. For a very long time, most of the studies were dominated by

[1] http://www.alicebot.org/

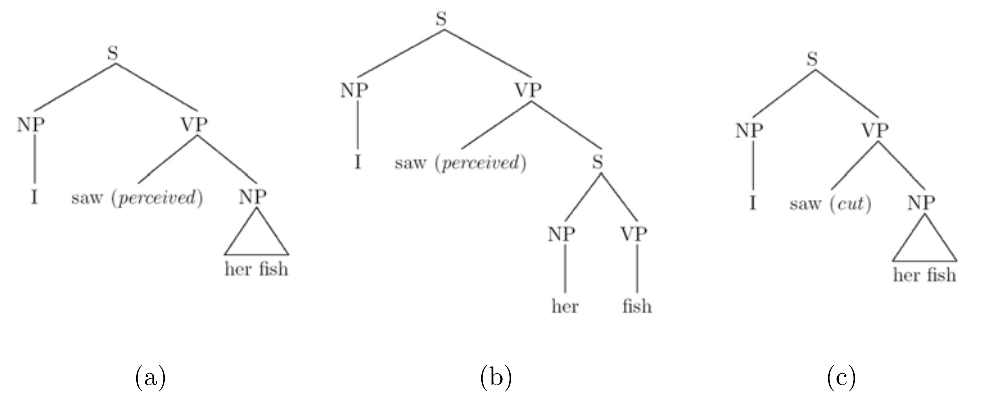


Figure 1.1: Alternative parse trees of the sentence *I saw her fish*

parsing, a disambiguation task at syntactic level. Various well-defined models for parsing have been proposed, and have achieved considerable success. In contrast, relatively few research work has been conducted on semantic and pragmatic levels, which are usually regarded as the indistinct area in NLP. However, they are far more important in achieving computer to understand natural language. One major reason for their under-studies is that there is less agreement on how to *represent* language at these two levels (Manning, 2004).

This thesis addresses two disambiguation tasks, one is for *semantic role* and one is for *dialogue act*. They are respectively at the semantic and pragmatic level. Recently, there is growing consensus that *semantic role* and *dialogue act* are the useful conceptual representations in supporting the deeper NLP analysis. Working with appropriate representation will be significant in advancing the progress for NLP. Although the taxonomies for semantic role and dialogue act are coarse-grained, their disambiguation tasks are very challenging.

Our first task is to resolve ambiguities in semantic roles, with respect to a target verb. Consider the following sentences:

2a. Peter opened the door with a key.

2b. The key opened the door.

2c. The door opened.

All the three sentences describe the same event, which is governed by the verb *opened*. Grammatically, although the three subjects (being underlined) are placed in the same syntactic position, they play different roles, semantically. In sentence 2a, the subject *Peter* is the person who opened the door, playing the semantic role AGENT. In sentence 2b, the subject *The key* is the thing used to open the door, playing the semantic role INSTRUMENT. In sentence 2c, the subject *The door* is the thing being opened, playing the semantic role PATIENT. In general, each verb in a sentence has more than one semantic role. Each role can be placed in *any* position in the sentence. Moreover, each role can consist of more than one word. Thus, there exists another ambiguity problem, which requires to resolve the boundary ambiguities of semantic roles. The collective task, resolving boundary ambiguities and semantic role ambiguities, is called *semantic role labeling*.

Our second task addressed in this thesis is to resolve pragmatic ambiguities in dialogue acts. Consider the utterance *Can you pass me the salt?* There are two possible interpretations:

3a. Asking whether the hearer have the ability to pass the salt.

3b. Requesting the hearer to pass the salt.

In interpretation 3a, the attention is restricted to what the words mean, whereas interpretation 3b is concerned with what the speaker means. This ambiguity on how to *use* the utterance results in two meanings. Assuming that the action behind the speaker is the intent for a request. Its concise abstraction can be represented by a dialogue act. In this case, the dialogue act of this utterance should be recognized as REQUEST rather than YES-NO QUESTION. The task to resolve dialogue act ambiguities is called *dialogue act recognition*.

1.2 Disambiguation as Statistical Classification

Many types of ambiguities are finite and definable, and thus can be tackled via classification. This makes it tractable for computer to perform disambiguation by treating it as a *classification* problem. In general, a classification problem is where one tries to assign instances to one of the predefined categories (also called classes). It is a generic type of problem in *pattern recognition*, and mainly addressed by the *machine learning* community. Here, a class being classified is referred to a possible ambiguity in a confusion set. The task is then to decide which of the candidates is more likely in the given context.

Classification is usually achieved by *inductive learning* techniques, which generalize previous observations from *examples*. Here, an example is referred to a paired set, containing an instance and its class. The type of learning is *supervised* because instances are all manually labeled with their classes. This serves like a teacher directing the learning process, where the correct answers are provided. To facilitate the induction during the learning, instances are usually represented by *features*. Features are the attribute-value pairs used to characterize instances. The value type of a feature can be nominal, ordinal or numerical. Due to its variability, features are often shared by all instances. The aim of learning is thus to find out the associative relation of each feature for each individual class. The learned model, called *classifier*, can then classify an unseen instance, based on how strong the input features are associated among classes. The class with the highest associative strength will be the outcome.

Formally, a classification problem is a mathematical-function approximation: Given a set of examples $\{(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)\}$, known as *training set*, the aim is to approximate a function (classifier) $h : \mathcal{X} \rightarrow \mathcal{Y}$ which maps an instance $x \in \mathcal{X}$ to its class label $y \in \mathcal{Y}$[2]. Here, $\mathcal{X}$ is the input domain of instances and $\mathcal{Y}$ is the finite set of class labels. After the approximation, the classifier can be used. The performance can be measured by evaluating the classifier on another set of examples. This is known as *test set*. Unlike the training set, the test set is separated into a set of answer labels $\{y'_1, y'_2, \ldots, y'_n\}$ and a set of instances $\{x'_1, x'_2, \ldots, x'_n\}$. Only the set of instances is provided to the classifier for prediction. The predicted results are then matched with the set of answer labels for evaluating the classification accuracy.

1.3 Thesis Scope, Initiative, and Methodology

This thesis takes a probabilistic classification approach to address two natural language disambiguations tasks. The ability of disambiguation is given by a probability distribution, which is in the form of $p(ambiguity|expression, context)$. It is a mapping from expression to ambiguity conditioned by context. The *context* is a part of text or expression that surrounds the given expression. Under this formulation, various contextual information is integrated, serving as the primary knowledge for disambiguation. The decision in disambiguation process is designed as: among the plausible ambiguities that

[2]Precisely, x should be the feature encoding of an instance.

belong to an expression, select the most probable one, based on the given information. Specifically, we employ conditional Maximum Entropy (ME) (Jaynes, 1957) to model the desired distribution. The involved classifiers are trained in a supervised learning way. Inside, a set of training examples is used for induction. Each example contains one of finite classes, each representing a possible ambiguity. The examples also contain the surrounding contexts about the class, which are represented in features. The examples are then supplied to the classifier, aiming at generalizing them and collecting the statistical relations between features and classes. Under the ME algorithmic supervision, the statistics are iteratively adjusted to derive the posterior knowledge for classification. The posterior knowledge is in the form of *conditional probability distribution* of classes, given that contextual features are extracted from ambiguities. The classifier is defined by estimating the probability distribution that is consistent with the observed distribution from the training data. In general, more than one distribution may be found. The principle of Maximum Entropy (Jaynes, 1957) helps in choosing the most appropriate distribution, without making any unnecessary assumptions on the training data. Moreover, under convex duality (Berger et al., 1996), the distribution modeling is equivalent to finding the Maximum Likelihood for exponential distributions (See Chapter 3), which eases the learning process by common statistical estimators. After training, the classifier can assign ambiguous instances to their most probable classes, based on the modeled distribution.

The two disambiguation tasks addressed in this thesis are context-dependent. They are at the semantic and pragmatic levels, respectively referred to *semantic role labeling* and *dialogue act recognition*. Both disambiguation tasks can potentially offer essential improvements to our previous prototype-system called FNDS, which stands for Financial News Dialogue System (Lan et al., 2003, 2005). The core of the system is the combination of two natural language applications: information extraction and question-answering. The two applications are pipelined, in which the output from information extraction is the input to question-answering. In the information-extraction module, the system automatically extracts important information from electronic financial news using predefined patterns. In the question-answering module, the system provides users with a dialogue-based interface to access the extracted entities by posing questions written in natural language. Both applications can be facilitated by semantic role labeling (Lan et al., 2004) and dialogue act recognition. On the one hand, semantic role labeling can enable a pattern-free approach for information extraction by providing verb-argument structures for extracting the desired entities (see Chapter 4). On the other hand, dialogue act recognition can provide question-answering with a coherent dialogue flow by interpreting the intention of user utterances (see Chapter 5).

In the semantic role labeling task, we propose a three-phase labeling approach to the problem. Evaluations on public benchmarks showed that our recognition performance with this approach is comparable with the current best individual system. In the dialogue act recognition task, we target at non-task oriented recognition. We employ various types of features, ranging from lexical, to syntactic, to discourse. A feature selection method is used for systematically optimizing the feature set. Experimental results showed that our system outperformed others using the same data set.

1.4 Problem Encountered in Classification: Class Imbalance

Like other classification tasks, semantic role labeling and dialogue act recognition suffer the *class imbalance* problem, which hinders the classifier's performance in terms of individual-class classification accuracy. This problem occurs when a data set is largely dominated by some usual classes with a significantly small percentage of unusual classes. In the other words, the distribution of examples among the classes is highly skewed. A classifier, trained with the skewed class distribution, often achieves high predictive performance over the usual classes but very poor predictions over the unusual classes. This will be problematic in practice because the unusual classes are often of greater value in the application. For example, in coreference resolution (Ng and Cardie, 2002), there are two kinds of relations (classes) between noun phrases: equivalent or non-equivalent. However, the equivalent relation between noun phrases is generally rare in real text. Failing to recognize the equivalent relation will affect the comprehension.

Class imbalance problem frequently occurs in binary-class classification problems. By convention, the class having more examples is called *majority class*, and the one having fewer examples is called *minority class*. Note that in the data mining literature, the majority class often refers to the *negative class* whereas the minority class refers to the *positive class*. Since our work is a multi-class classification problem, there is no clear distinction between majority classes and minority classes.

This thesis employs two methods to tackle the class-imbalance problem encountered in both semantic role labeling and dialogue act recognition. The first method is to re-sample the training data by replicating the examples of minority classes. The proportion of replications is heuristically computed. This makes the training data more balanced for training. Another method is to modify the learning mechanism of Maximum Entropy (ME) by weighting more the minority classes. More specifically, parameters are added to the updating rule of GIS algorithm. The parameters are determined according to the accuracy of each class , which are generated in each iteration of training. Both methods are shown effective in classifying minority class in the experiments conducted on publicly available data sets.

1.5 Contributions

- We address two challenging disambiguation tasks on natural language, using statistical classification techniques.
- We implement an advanced and robust classifier based on Maximum Entropy for disambiguations.
- We propose two methods at data and algorithmic level to tackle the class-imbalance problems encountered in the disambiguation tasks.
- We define an evaluation metric, in terms of the overall predictive rates among classes and their individuals, to measure the classifier performance.
- We perform extensive experiments on large data sets to verify the effectiveness of the proposed methods.

1.6 Thesis Outline

Chapter 2 reviews the literature relevant to our work. The advantages and disadvantage of previous approaches are discussed. We also compare them with our work.

Chapter 3 gives background materials on Maximum Entropy. It sketches the theoretical foundation to implementation details, with the inclusion of relevant derivations. The advantages and disadvantages of Maximum Entropy are also discussed.

Chapter 4 describes the first attempt at building a classification system to semantic role labeling using the word-by-word approach. The fairly satisfactory results lead us to the proposed three-phase labeling approach. Experimental setup, along with the data sets, classifiers, and evaluation metrics are discussed. Results on both performance of micro-averaging and macro-averaging are presented.

Chapter 5 describes the use of various features to improve the performance of dialogue act recognition. A feature selection method is presented. Beside the optimized results, micro-averaging and macro-averaging results are also presented.

Chapter 6 gives a conclusion to this thesis. Possible directions for future work are also outlined.

Chapter 2

Related Work

This thesis applies machine learning techniques to natural language disambiguations. Both areas have much to offer each other. On the one hand, machine learning can provide natural language with various powerful tools and algorithms to model human disambiguation. Among them, statistical classification (see Section 2.1) can offer a principled solution to computational disambiguation. On the other hand, natural language can provide machine learning with a range of challenging problems in morphology, syntax, semantics (see Section 2.2), and pragmatics (see Section 2.3). Moreover, these problems commonly share another set of problems such as class imbalance (see Section 2.4) and sparse feature space, that have to be resolved simultaneously.

2.1 Statistical Classification

2.1.1 Three Schools of Thought

Classification problem has a long tradition in pattern recognition, and has been well studied since 1960s (Duda et al., 2001). Over years, the study has been dominated by statistical approach, which has been further categorized into three main paradigms: generative, conditional, and discriminative (Rubinstein and Hastie, 1997; Johnson, 2001; Jebara, 2001, 2004). *Generative* approach estimates a joint probability distribution over all the variables (instances and classes) in question. As the name implies, the estimated distribution can "generate" examples of any variable combinations in probability. Inferential classification can then be performed by conditioning the joint distribution over the input variables. However, it is roundabout and sub-optimal to obtain a conditional distribution from a joint distribution, for a single classification task. Thus, *conditional* approach is advocated. It directly computes the conditional distribution instead of modeling all the variables as a whole. This makes the task more efficient and more effective in general. Compared with conditional approach, *discriminative* approach is even more minimalist. It focuses on a discriminant function between two classes, rather than their conditional distributions. The function is directly optimized, mapping from the input instances to the output classes. The optimized function then becomes the decision boundary for discriminative classification (e.g., either 0 or 1). Note that in some literature (Ng and Jordan, 2001; Ulusoy and Bishop, 2005), conditional learning is also regarded as discriminative approach. Here, we separate them, following the formalism in Jebara (2001, 2004)

Generative approach and discriminative approach are the two extremes. Generative approach allows a flexible learning by incorporating knowledge about the problem at hand. The knowledge includes Markov assumptions, prior distributions, latent variables and structural information (Box and Tiao, 1992; Jebara, 2004). These are all absent in discriminative approach. Moreover, generative approach is more robust to extend for multi-class problems, whereas discriminative approach requires extra work like using error-correcting codes (Dietterich and Bakiri, 1995) for multi-class extension. Another advantage over discriminative approach is that generative approach generally requires less time for training classifier. However, regarding to classification performance, discriminative approach has often outperformed generative approach in many areas. For example, Support Vector Machine (Vapnik, 1995), a discriminative classifier, achieved superior performance in face detection (Osuna et al., 1997), text categorization (Joachims, 1998), and speech recognition (Ganapathiraju and Picone, 2000).

2.1.2 Maximum Entropy: Generative and Conditional

This thesis employs Maximum Entropy (Jaynes, 1957) to model our classifier. Maximum Entropy (ME) is inherently generative (see Section 3.4), but can also be conditional (see Section 3.5). Both types of ME has been successfully applied in many useful tasks, particularly in natural language domain. Examples of generative ME include word-morphology discovery (Pietra et al., 1997), sentence modeling (Rosenfeld et al., 2001), and geographic-species modeling (Phillips et al., 2006). Meanwhile, examples of conditional ME include part-of-speech tagging (Ratnaparkhi, 1996), machine translation (Berger et al., 1996), and syntactic parsing (Ratnaparkhi, 1999).

Generative ME differs from bayesian generative classifier in estimating the joint distribution, and in handling the dependence of features. For examples, in typical Naive Bayes classifier (Mitchell, 1997), the joint distribution $p(x, y)$ is estimated by the product of the prior distribution $p(y)$ and the class-conditional distribution $p(x|y)$. Here, y is a class label, and x is an input instance which is usually represented by a feature vector. In practice, the computation for $p(x|y)$ is generally hard because there are numerous number of dependence between classes and features. For tractable computation, features in Naive Bayes are thus assumed conditionally *independent* of each other. It is achieved by using the chain rule to decompose $p(x|y)$ into a product of class-conditional probabilities (Mitchell, 1997). In resolving problems of highly structured natural language, these independent assumptions are clearly unjustified. On the other hand, generative ME expresses the joint distribution in an exponential parametric form rather than a product of distributions. Inside the exponential form, each feature is uniquely associated with its parameter. Thus, overlapping features can be combined without independent assumptions, at the expense of the numerical estimation of parameters (Ratnaparkhi, 1998). Furthermore, the "elasticity" of exponential form can resist to the introduction of irrelevant features by assigning marginal weight to them (Jin et al., 2003).

Conditional ME is closely connected to logistic regression, a common technique in statistics (Hosmer and Lemeshow, 1989). Logistic regression is designed to estimate a conditional distribution for binary-valued classes. In some literature, it is referred to as a generalized linear model (McCullagh and Nelder, 1989). Previously, Ratnaparkhi (1998) showed that logistic regression was a special case of conditional ME. Both are expressed in exponential form, but conditional ME is capable for multi-class problems.

In this thesis, we use conditional ME for our disambiguation tasks. Conditional ME share many of the properties as generative ME. For example, both ME can be parameterized in an exponential form (Jaynes, 1957), and can be trained with generic statistical estimator like maximum (conditional) likelihood (Chen and Rosenfeld, 2000). However, they differ in computing their normalization factors in the exponential form. In generative ME, the factor involves all the possibilities of variables. Its computation is thus more harder during training (Miller and Yan, 2000), and often requires simulation techniques such as the Monte Carlo method used in Pietra et al. (1997) and Metropolis Hastings used in Abney (1997). However, the factor computation in conditional ME is more expensive at run-time (Schofield, 2004). This is because the factor is dependent on the input context rather than being a global constant in generative ME.

2.1.3 From Individual Classification to Interdependent Classifications

Seldom disambiguation tasks involve a single classification outcome. Instead, they often require a sequence of classifications, whose outcomes are *interdependent* on each other. A typical example for modeling sequential data is part-of-speech tagging. In more complicated cases, disambiguation tasks require embedded classifications in a graphical structure, such as syntactic parsing. This section mainly reviews on the approaches for sequential data.

Hidden Markov Model (HMM) is one classical generative approach for *directed* sequential data, and has been applied in various disambiguation tasks such as part-of-speech tagging (Kupiec, 1992) and named entity recognition (Bikel and Weischedel, 1999). Suppose that a sequence of instances $\langle x_1, x_2, \ldots, x_n \rangle$, denoted as $x_{1,n}$, and a sequence of their classes $\langle y_1, y_2, \ldots, y_n \rangle$, denoted as $y_{1,n}$, are given. HMM can estimate a joint distribution $p(y_{1,n}, x_{1,n})$ by decomposing it into a likelihood distribution $p(x_{1,n}|y_{1,n})$ and a prior distribution $p(y_{1,n})$ (Rabiner, 1989). For trackable inference, three assumptions are made. Firstly, instances are independent of each other, resulting in $p(x_{1,n}|y_{1,n}) = \prod_{i=1}^{n} p(x_i|y_{i,n})$. Secondly, each instance is only dependent on its class, resulting in $p(x_i|y_{1,n}) = p(x_i|y_i)$. Thirdly, each class in the sequence is only dependent on its previous k class, resulting in $p(y_{1,n}) = \prod_{i=1}^{n} p(y_i|y_{i-1}, \ldots, y_{i-k})$. Thus, the final estimation for $k = 2$ (a.k.a. second order Markov chain or trigram model) can be represented as $p(y_{1,n}, x_{1,n}) = \prod_{i=1}^{n} p(x_i|y_i)p(y_i|y_{i-1}, y_{i-2})$. However, HMM hardly benefits from a rich representation of instances (McCallum et al., 2000), such as long-range dependency of features (Lafferty et al., 2001). It is mainly due to the first and second assumptions that are strictly made in HMM. For disambiguation tasks being context-dependent, these assumptions hardly allow surrounding contextual-features to be encoded into the model.

Several other frameworks for modeling sequential data have been introduced to alleviate HMM restrictions. For example, Ratnaparkhi (1996) proposed a *sequential tagging* method, based on a sole conditional ME classifier. During training, each instance in the sequence was encoded with the previous two classes and features of previous two and next two instances, based on a sliding window. At run-time, the best sequence is produced by beam search. It starts at the leftmost instance, producing the most-probable K classes, transiting to a next instance, producing another most-probable K classes, and until the rightmost instance is reached. Each production of K classes produces $K \times K$ sequence candidates. But only the top K candidates are used as features for the next class production. Formally, the sequence can be determined by the first order Markov chain, denoted as $p(y_{1,n}|x_{1,n}) = \underset{y_{1,n}}{\operatorname{argmax}} \prod_{i=1}^{n} p(y_i|x_i)$.

Maximum Entropy Markov Model (MEMM) (McCallum et al., 2000) is one conditional approach for modeling sequential data, and is almost similar to the sequential tagging method used in Ratnaparkhi (1996). MEMM builds a *separate* conditional ME classifier for each class, whereas sequential tagging method builds a *single* classifier for all the classes. However, both methods are susceptible to the label bias problem (Lafferty et al., 2001) in which decisions are biased to classes with few possible next classes. Conditional Random Field (CRF) (Lafferty et al., 2001) is one conditional approach that solves the label bias problem, and have achieved improvements in information extraction (Lafferty et al., 2001) and shallow parsing (Sha and Pereira, 2003). CRF can be viewed as an *undirected* graphical model for sequential data (Wallach, 2004). It retains all the advantages of MEMM (Lafferty et al., 2001), and benefits from global maximum likelihood (Zelenko et al., 2003). However, its parameter estimation is significantly slower than MEMM, at the same setting using iterative method (Lafferty et al., 2001).

Trading off performance and speed, we use sequential tagging method (Ratnaparkhi, 1996) in one experiment to address semantic role labeling. Details and research work on semantic role labeling can be found in Section 2.2, and our evaluation result can be found in Chapter 4.

2.2 Previous Work on Semantic Role Labeling

The goal of semantic role labeling is as follows: given a sentence, for each target verb, the system has to recognize the *arguments* governed by the verb, and label them with *semantic roles*. Recently, the task has gained much attention from the machine learning community, and has become a well-defined task with a collection of work. For example, there have been two years of CoNLL conference, holding shared task (Carreras and Màrquez, 2004, 2005) for semantic role labeling. Their general goal was to come forward with *learning strategies* to address the task in a principled way.

Over years, there have been three general approaches to the task: constituent-by-constituent, word-by-word, and chunk-by-chunk. Each of them has its own advantages and disadvantages. In general terms, the constituent-by-constituent approach is more accurate, but suffers of the alignment problem with arguments. In contrast, the word-by-word approach ensures that every word align with arguments, but is less accurate. The chunk-by-chunk approach is the compromise between them: it is moderately accurate and has only mild alignment problem.

We have built two systems based on different approaches. The first system is a preliminary work, which uses the word-by-word approach. In the second system, we collected the advantages from the three approaches mentioned above, and propose a three-phase labeling method.

2.2.1 Proposition, Predicate, Argument and Semantic Role

In linguistics, a sentence can be expressed as a *proposition* by characterized by a predicate. A proposition is a truth-conditional meaning. Thus, the same proposition can be expressed by sentences with different grammatical structures. For example, *John shoots Bobby* and *Bobby is shot by John* are propositionally the same. Predicates are usually presented as verbs. In the above example, *shoots* and *shot* are the predicates. In the succeeding sections, we will restrict our discussion on predicate as verb only.

The participants involved in a proposition are called *arguments*. They are syntactic phrases, or more generally a sequence of words. The role that an argument plays is called its *semantic role*. Examples of semantic roles include AGENT, PATIENT and RECIPIENT. In the above example, `John` is an AGENT, whether it occupies the subject or the object position. The concept of semantic roles has a long history in linguistics, and its similar notion includes case roles and thematic roles (Saeed, 1997).

The task of *semantic role labeling* is a propositional analysis. Specifically, given a verb in a sentence, the task recognizes the arguments that are set in relation to the verbs, and labels them with the appropriate semantic roles. A similar task to semantic role labeling is *function tagging* (Blaheta and Charniak, 2000), where functional tags are assigned to phrases. The assignment, however, is based on neighboring words rather than a target verb. Functional tags are thus defined wider. They may include grammatical functions, syntactic functions, and semantic functions.

2.2.2 Data Sets Commonly Used

Currently, two English corpora are commonly used to train systems for semantic role labeling: FrameNet (Baker et al., 1998) and PropBank (Kingsbury and Palmer, 2002). Comparisons between the two corpora can be found in (Gildea and Palmer, 2002). The strength of FrameNet is that target words are annotated with word senses. Moreover, besides verbs, adjectives and nominal phrases are annotated as predicates. However, the sentences in PropBank are longer and more complex. They are thus more suitable for real-world applications. Since PropBank is created from Penn TreeBank (Marcus et al., 1993), it contains richer syntactic information than FrameNet. Moreover, the sentences have a more even coverage of the verbs. So the data is less sparse, which helps classification performance.

2.2.3 Constituent-by-Constituent Approach

Automatic semantic role labeling was first empirically studied by (Gildea and Jurafsky, 2000, 2002) using FrameNet as the corpus (Baker et al., 1998). The work divided the task into two sub-problems: *argument recognition* and *semantic role assignment*. Both sub-problems are inherently classification problems: argument recognition is a binary-class problem whereas semantic role assignment is a multi-class problem. Two separate classifiers were built on likelihood probabilities of features, and maximized by linear interpolation. Most features were syntactic derived by *full parsing*[1]. The feature set included: (1) the phrase type expressing the semantic role, (2) the grammatical subject or object of the phrase, (3) the voice of the verb, (4) the head word of the phrase, (5) the phrase position before or after the verb, and (6) the traversed path from the phrase to the verb.

In the stage of argument recognition, all the *constituents* (i.e., words under parse

[1]The grammar formalism was in traditional phrase-structure grammer. Other formalisms were also tried, e.g, combinatory categorial grammar (Gildea and Hockenmaier, 2003) and tree adjoining grammar (Chen and Rambow, 2003).

tree nodes) generated by full parsing were considered as argument candidates. A binary-class classifier was then used to classify each argument candidate into recognizables or non-recognizables. In the stage of semantic role assignment, the recognized arguments were labeled with their semantic roles, using a multi-class classifier. The experiments in (Gildea and Jurafsky, 2002) found that the *path* and *head* features were most useful for the semantic role labeling task. The path feature was also useful in argument recognition.

Later, Gildea and Palmer (2002) tackled the task using PropBank (Kingsbury and Palmer, 2002) as the corpus. The same set of features were employed, but they were generated by *shallow parsing* (a.k.a. chunking (Abney, 1991)), which divided a sentence into non-recursive segments of words (i.e. chunks), rather than phrase-structured trees as in full parsing. In (Gildea and Palmer, 2002), the authors tested the hypothesis that the flattened representation of features could also do well in semantic role labeling. Their experimental settings were similar to (Gildea and Jurafsky, 2002), but differed in how argument candidates were chosen. More specifically, *single chunks* were chosen as argument candidates. However, a few arguments could actually correspond to single chunks. The performance was thus affected. This was revealed by the experimental results. They found that the shallow-parsing system was significantly worse than the full-parsing system, although it was computationally cheaper. Similar results were later confirmed by Punyakanok et al. (2005b).

However, the disadvantage of (Gildea and Jurafsky, 2002; Gildea and Palmer, 2002) is the *unmatching* problem of arguments. In (Gildea and Jurafsky, 2000), around 13% of arguments were unable to find the matching constituents while in (Gildea and Palmer, 2002), over 60% of arguments were unable to find the matching chunks. This was explained by two reasons (Surdeanu and Turmo, 2005): (a) the constituents were incorrectly generated by full parsing, and (b) the underlying representations of syntax and semantics were different. Since the performance of argument recognition was capped, so was the performance of role assignment.

2.2.4 Word-by-Word Approach

In view of the above-mentioned limitation, researchers (e.g., Hacioglu and Ward (2003)) attempted to recognize arguments word-by-word, rather than matching a complete syntactic constituent/chunk. This can ensure every word has a chance for matching.

This method was similar to that used by Ramshaw and Marcus (1995), who proposed chunking (Abney, 1991) as a tagging task. Semantic role labeling was formulated as a word-level classification, recognizing arguments and assigning roles *simultaneously*. It was achieved by defining *positional tags* over the original role labels. For example, they defined `B-X` tag for the first word in a role label of type `X`, `I-X` tag for the non-initial word(s) in a `X` role label, and `O` type for the word outside *any* role label. This `BIO` scheme was firstly proposed by Ramshaw and Marcus (1995) and its variants were extensively evaluated by Tjong Kim Sang and Veenstra (1999). Here, a single classifier was employed with input features of words, part-of-speech tags, and chunks. Each individual word was separately classified into one of the tags. A coherent role label could be re-combined by finding a consecutive tag with correct positional order. However, this approach has two drawbacks. Firstly, the number of role classes increases. It generally requires $2 \times N + 1$ classes, where N is the original number of classes. This makes classification harder. Secondly, since semantic roles generally involve more than one word, the number of classification steps per semantic role increases. Long-ranged semantic roles are thus difficult to recover. Therefore, this word-level approach was generally worse than the constituent approach (Punyakanok et al., 2005b), although it guaranteed that every word has a chance to match.

In this thesis, we address semantic role labeling in two different approaches. One of them is based on word-level classification, which has previously been presented in (Lan et al., 2004). The closest system to ours was the "Chunker-II" in Pradhan et al. (2003), who employed word-level classification inside the system. Both their work and ours conducted experiments on the PropBank corpus (Kingsbury and Palmer, 2002), with the same set of features. But we used the preliminary release of PropBank, where the number of role classes was much larger than theirs. Moreover, our classification task was performed by a conditional ME classifier whereas in (Pradhan et al., 2003), a SVM classifier was used, which was more accurate in general. Experimental results showed that their system performed slightly better than our system.

2.2.5 CoNLL-2004: Chunk-by-Chunk Approach

A better approach than word-level classification is to classify semantic roles at chunk-level. This approach has been used in CoNLL-2004 shared task (Carreras and Màrquez, 2004), which was a competition for semantic role labeling, strictly based on shallow parse information. In CoNLL-2004, participants were provided with data sets to build their learning systems. The feature set included words, part-of-speech tags, chunks, clauses, and named entities. Benchmark was also given for system evaluation. Most of the participants employed chunk-level classification to address the problem. The approach is basically the same as word-level classification, but it differs in the *granularity* at which the sentence elements are classified (Carreras and Màrquez, 2004). For example, in the work of Hacioglu et al. (2004), words were collapsed into chunks with the head words retained as lexical information. This collapsed representation has two advantages over the original one used in word-level classification. Firstly, chunks can span a larger segment of semantic roles. The number of classification steps are thus smaller. Secondly, lexical information is now replaced by head words rather than all the words in the chunk. It is thus less sparse for classification. In the CoNLL-2004 shared task, a SVM classifier has achieved the best performance (Hacioglu et al., 2004).

In the shared task, two participants (Baldewein et al., 2004; Lim et al., 2004) also used conditional ME to build their classifiers. But there were some underlying differences between their works and ours. Firstly, they employed chunk-level classification instead of word-level classification. This made classification more robust and less sparse. Secondly, their systems included a different feature set, which included clauses and named entities. These features were absent from our data set. Thirdly, the number of role classes in their data set was much smaller than ours. Their experimental results were thus hardly comparable to ours, although their systems achieved better performance.

A related approach to chunk-level classification is the *filtering* approach (Punyakanok et al., 2004; Carreras et al., 2004). The approach trains two classifiers, one to recognize the argument-start and the other to recognize the argument-end. A set of argument candidates can be constructed by combing *each* predicted-start with *each* predicted-end. Previously, a similar approach has been employed in chunking (Punyakanok and Roth, 2000). But this construction method would result in many possible candidates. They filtered out the unlikely candidates by a scoring function with dynamic programming (Punyakanok et al., 2004; Carreras et al., 2004). The remaining arguments were then labeled by a multi-class classifier. In the CoNLL-2004 shared task, systems with this approach achieved the second best performance (Punyakanok et al., 2004) and the third best performance (Carreras et al., 2004) respectively.

To further enhance performance, some participants in CoNLL-2004 employed global-optimization methods or post-processing steps for their systems. For example, beam search was employed by Baldewein et al. (2004) and Lim et al. (2004) to maximize the probability of the output sequence. Some work also defined global constraint functions with integer linear programming to ensure output coherence (Punyakanok et al., 2004). On the other hand, simple heuristic rules were used to post-process the output by correcting trivial errors (e.g., van den Bosch et al. (2004); Higgins (2004)). A more detailed comparison between approaches in CoNLL-2004 can be found in (Carreras and Màrquez, 2004).

2.2.6 CoNLL-2005: System Combination

Following the same initiative as CoNLL-2004, the CoNLL-2005 shared task (Carreras and Màrquez, 2005) was held again which was competition for semantic role labeling. A main difference between CoNLL-2004 and CoNLL-2005 was that in CoNLL-2005 two syntactic parse trees were given as additional input information for each sentence. The trees were generated by two parsers: one was from Charniak (2000) and the other was from Collins (1999). Most participants collected argument candidates, following the approach similar to that in (Gildea and Jurafsky, 2000). Constituents in the parse tree were considered as argument candidates. Many participants chose the Charniak's one because it generated fewer unmatching constituents (Liu et al., 2005). To reduce the number of negative examples (of argument candidates), most participants employed pruning (Xue and Palmer, 2004) to filter out the unlikely candidates. With this method, around 60% of argument candidates could be eliminated (Tsai et al., 2005). In general, systems usually performed role assignment after argument recognition. However, some systems (e.g Tsai et al. (2005)) did integrate two sub-problems into one single multi-class classification problem by adding a "null" category for unrecognized arguments. Since constituent nodes in a tree are embedded, arguments generated from these nodes are highly embedded too. Several heuristics were proposed to resolve the problem, by selecting a subset of the embedding arguments. For example, Liu et al. (2005) selected the argument with the largest probability score, whereas Ponzetto and Strube (2005) selected the argument with the largest span.

Remarkably, many participants used a combined system in CoNLL-2005, particularly in the best four systems (Punyakanok et al., 2005a; Haghighi et al., 2005; Màrquez et al., 2005; Pradhan et al., 2005). The aim was to increase syntactic coverage and robustness of the final system. The outputs to combine were obtained from different syntactic parses (full parses and shallow parse), n-best full parse candidates, or different learning algorithms. The combination methods included integer linear programming (e.g., Punyakanok et al. (2005a)), re-ranking (e.g. Haghighi et al. (2005)), greedy merging (e.g., Màrquez et al. (2005)), and stacking (Pradhan et al., 2005). A more detailed comparison between the approaches in CoNLL-2005 can be found in (Carreras and Màrquez, 2005).

2.2.7 Our Approach: Three-Phase Labeling

Our approach involves three phases. It includes the advantages from classifications at word, chunk, and constituent level. The classifiers involved are all trained with conditional ME. We performed the evaluation with the provided benchmark of CoNLL-2005. Thus, fair comparisons could be made. We followed the approach of Gildea and Jurafsky (2000) to divide the problem into two sub-problems: argument recognition and semantic role assignment. We also follow Punyakanok et al. (2004) to divide argument recognition into two sub-problems: argument-start recognition and argument-end recognition. Each problem is separately addressed in an individual phase. The outputs are pipelined, with the output of one phase used as the input of the next phase.

A pre-processing step is done on the original data set to collapse tokens into chunks. In case a chunk can not be aligned with an argument, the chunk is split back into separate tokens. Thus the new representation is mixed with chunks and tokens.

In the first phase, we use a classifier to recognize the argument-start at the chunk/token level. Here, we gain two advantages by this mixed representation. Firstly, the use of chunks can reduce the number of classification steps, unlike the word-by-word approach requires every step for each individual word. Secondly, the use of tokens can ensure no argument-start is unmatched, unlike the constituent-by-constituent approach aligns with arguments.

In the second phase, we recognize the argument-end by traversing the parse tree. The traverse starts at an argument-start, climbing bottom-up to find the lowest ancestor whose subtree can cover a complete syntactic constituent. The rightmost boundary of that constituent then becomes the argument-end. Here, we also gain an advantage by this indirect way of obtaining constituents. On the one hand, the use of these constituents can avoid the "brittleness" problem suffered by both word-level classification and chunk-level classification. On the other hand, the length of these constituents can be as long as those produced by the purist constituent approach. Thus, even long-ranged arguments can be recognized. This two-phase argument recognition approach is similar to that in (Punyakanok et al., 2004), who train two separate classifiers to recognize argument-start and argument-end respectively, and construct many possible argument candidates by combing each predicted-start with each predicted-end. Here, the problem of exponential grow of candidates suffered by Punyakanok et al. (2004) is avoided. We do not need a scoring function to filter out the unlikely candidates. Thus, our approach is computationally cheaper.

In the third phase, we use a classifier to label the recognized arguments with semantic roles. Our system with this three-phase labeling approach would rank the 9th of 20 systems in CoNLL-2005. Note that the first four and the 7th system were those combining individual systems, which were definitely better than the our single system. In CoNLL-2005, the best *single* system (Surdeanu and Turmo, 2005) ranked the 5th. Their system performed marginally better than our system. Their key advantage over ours was its complete analysis on constituent-argument mappings. They defined the mappings into three categories: (a) correct one-to-one mapping, (b) correct one-to-many mapping, and (c) one-to-many mapping due to incorrect syntax. They focused on modeling the first mapping with a AdaBoost classifier (Freund and Schapire, 1997), while resolving the second mapping by an argument expansion heuristic, namely, for arguments that mapped to more than one constituent, their boundaries were extended to the right, to include all other unlabeled constituents. Generally, AdaBoost is an ensemble method of combining weak classifiers using weighted voting. Thus, strictly speaking, their work is also a *combined* system rather than a single system.

2.2.8 Classification Approaches for Semantic Role Labeling

Over years, various learning algorithms were applied for building classifiers for semantic role labeling. They included rule-based learning (Higgins, 2004), decision-tree learning (Surdeanu et al., 2003), memory-based learning (van den Bosch et al., 2004), generative learning (Thompson et al., 2003), conditional exponential learning (Lim et al., 2004; Lan et al., 2004), and discriminative learning (Pradhan et al., 2004). Recently, some novel classification approaches have also been applied, e.g, relevant vector machine

(RVM) (Johansson and Nugues, 2005) and tree conditional random fields (T-CRF) (Cohn and Blunsom, 2005).

2.3 Previous Work on Dialogue Act Recognition

The goal of dialogue act recognition is to assign a *dialogue act* to a given utterance. Dialogue act, in general, is a concise abstraction of an utterance's intention. For example, the dialogue acts Greeting and Opinion should be assigned to the utterance *Good morning* and *I think it's great*, respectively.

Generally there are two approaches to dialogue act recognition (Jurafsky, 2004): plan-based approach and cue-based approach. While both approaches require inference processes, they differ in the way of inferencing. The plan-based approach, also known as BDI model (belief, desire, and intention), was proposed by Allen (1995). The approach makes use of *logical inference*, which employs techniques from classical artificial intelligence. The approach basically operates on predicate calculus for representing utterances. For example, *S believes a proposition P* is denoted as belief(S, P). The approach defines action schema for each dialogue act. Each schema contains three sets of parameters: (1) the conditions required to perform an action, (2) the states achieved by performing the action, and (3) the effects performed by the action. All of them are defined using predicate calculus. During recognition, logical inference is performed by pre-defined rules, moving from one state to another, until it meets an action's effect (dialogue act). The advantage of this approach is that it is easily understandable to human, and thus manageable for amendment. The disadvantage is its enormous effort in encoding the necessary knowledge for logical inference, resulted in relatively small-scaled uses. The system will also become very brittle if the domain is changed.

In contrast, the cue-based approach makes use of *statistical inference*, which employs techniques from machine learning community. In the approach, the recognition process is cast as a classification task, with cues as the input. In general, a cue is a surface feature that is usually associated with some dialogue act. Common cues include lexical feature, syntactic feature, and prosodic feature. The approach learns from a set of labeled examples by describing statistical relations of dialogue acts. The learned system then relies on the statistics to compute the most probable class of a dialogue act for a given untterance. The advantage of this approach is that it requires little expertise for handcrafting the knowledge, and it is thus more scalable to large systems.

We employ conditional ME to model a classifier for this task, and explore various types of features as input. We investigate the use of heterogeneous features to predict dialogue acts. A feature selection method is also employed to optimize the feature set.

2.3.1 The Origin of Dialogue Act

There is not much agreement with linguists on the definition of dialogue act. But it is uncontroversial that the main inspiration for dialogue act is from *speech act*. A comprehensive survey on the development from speech act to dialogue act can be found in Traum (1999).

The work on speech act was originated in the field of language philosophy. Its theory was developed from a need to give account for the pragmatic meaning for utterances, as opposed to the conditional-truth meaning for propositions. Austin (1962) explained that utterance was not simply the matter of either truth or false. It should be viewed as a kind of *action*. He distinguished the concept of speech act into three aspects: locutionary act, illocutionary act, and perlocutionary act. Searle (1969) extended Austin's work, focusing on the study on illocutionary acts.

Later, Cohen and Perrault (1979) introduced a plan-based theory of speech acts. They viewed speech acts are *plan operators* which affect the intention and belief of speakers. Litman and Allen (1990) tried to organize a dialogue in a hierarchical structure based on the sub-plans. This gave account not only for the functional meaning of utterances, but also for the structural dependencies between different plans and sub-plans in a dialogue.

However, traditional speech acts are still insufficient to explain the interactivity of human dialogues. Therefore, discourse management theories such as turn-taking, repair, reference, and attention are added into speech acts. Gradually, more new functions for speech acts were introduced. Therefore, new taxonomies for speech acts, namely dialogue acts, were formulated. Bunt (1994) emphasized that dialogue acts serve as a *function* that updates the dialogue context.

Note that the concept of speech acts had also been generalized to cover conversation and communication, resulting in conversational act (Traum and Allen, 1992) and communicative acts (Allwood, 2000) respectively. However, these kinds of acts are already out of our scope, and we do not discuss here.

2.3.2 The Cue-based Approach to DA Recognition

Most of the previous works on dialogue act recognition focused on *task-oriented* dialogues. For example, Wright (1998) used the Maptask corpus (Anderson et al., 1991) in which the dialogues were about a person giving instructions to guide another person through a route on a map. The appropriate sequence of dialogue acts (called move types in Anderson et al. (1991)) for labeling an utterance sequence U was determined by $M^* = \underset{M}{\operatorname{argmax}}\, p(M|I) = \underset{M}{\operatorname{argmax}}\, p(I|M)p(M)$, where I represented the sequence of suprasegmental features (i.e., intonational) observed in U. Here, $p(M)$ was estimated using 4-gram whereas three methods were used to calculate $p(I|M)$ separately: hidden markov models (HMMs), classification trees, and neural networks. The same task was also studied by Taylor et al. (1998), in which the appropriate move type sequence was determined by combining three probabilities: $p(M)$, $p(F|M)$, and $p(C|M)$, where $p(F|M)$ and $p(C|M)$ were likelihood probabilities of intonational and cepstral (i.e., speech) features respectively. Similar to the approach of Wright (1998), these probabilities were estimated based on HMMs and 4-gram. These two work mainly differ from ours in the form of input. They determined dialogue acts from speech input, whereas our work was based on hand-transcribed textual input.

Researchers (e.g., Reithinger et al. (1996); Samuel et al. (1998)) have also worked on the VERBMOBIL corpus, which contained dialogues related to appointment scheduling. For example, Reithinger et al. (1996) purely modeled the dialogue act sequence by n-gram approach. The most probable upcoming dialogue act d^* was modeled by conditioning its proceeding sequence, denoted as $d^* = \underset{d}{\operatorname{argmax}}(d|d_1, d_2, \ldots, d_{i-1})$. They shortened the sequence history as trigrams, which were smoothed by interpolation method. Samuel et al. (1998) applied transformation-based learning (Brill, 1995) whereby rules were learned to label utterances with dialogue acts. The features used were cue phrases appearing in the utterance, known as dialogue act cues, which were selected by mutual information. A drawback of the approach is that the decision rules may require *lookahead* information of the next utterance, which in practice is not available. It is thus not applicable for online dialogue act recognition.

Dialogues in other domains have been studied as well. For example, Kita et al. (1996) worked on the ATR Conference corpus that contained conversations between a secretary and a questioner at international conferences. Dialogue act recognition was achieved by modeling the proper dialogue act sequences, called illocutionary force types (Austin, 1962), using HMMs. Clark and Popescu-Belis (2004) also applied a conditional ME classifier to the ICSI MR corpus (Janin et al., 2003), which contained conversations recorded in meetings.

Compared with task-oriented dialogues, dialogue act recognition for *non-task oriented* dialogues is inherently more difficult since it involves more dialogue act types (to cater for the requirements of different tasks) and there is also a greater variance of *style* in the utterances. Besides, the corpus usually has a larger coverage in terms of the vocabulary's size. A representative work on non-task oriented dialogue act recognition was reported by Stolcke et al. (2000), where the Switchboard corpus (Godfrey et al., 1992) was employed, which contained dialogues spanning 70 topics. Global optimization was performed to find the dialogue act sequence D for labeling an utterance sequence U. This was achieved by maximizing the posterior probability $p(D|E)$, by applying the Viterbi search (Viterbi, 1967), where E denoted the evidence (i.e., features) observed in U. Various types of evidence were studied, including lexical, acoustic, and prosodic features. A model based on likelihood probabilities, similar to the one proposed by Taylor et al. (1998), was also applied to integrate different types of evidence for dialogue act recognition.

Our conditional ME method offers two advantages over other approaches. First, it allows heterogeneous features to be flexibly integrated in one classifier. This leads to a simpler design and it is also easier to assess the impact on performance of each type of features, facilitating feature selection. In other approaches, one often needs to assume that the heterogeneous features are independent, so that their likelihood probabilities can be modeled separately and then be combined for dialogue act recognition. Such assumption, however, is in many cases unwarranted. In our work, we study a wide range of features, ranging from lexical to syntactic to surface discourse features, and evaluate them empirically to select features that can lead to good recognition rate.

Another advantage is that the classification features are defined over the current and previous utterances only. This is in contrast to other approaches where either lookahead information of the next utterance is needed (Samuel et al., 1998) or the complete dialogue is required for global optimization to find the whole dialogue act sequence (Kita et al., 1996; Taylor et al., 1998; Wright, 1998; Stolcke et al., 2000). Our approach is thus more suitable for online dialogue act recognition.

Our work is different from that reported by Clark and Popescu-Belis (2004), who also applied conditional ME to dialogue act recognition. In their work, the features used were mainly words (selected based on occurrence frequency), and the limited contextual features adopted were also lexical in nature. Besides, they used a smaller tag set called MALTUS that consisted of 11 dialogue acts only, and a smaller data set that was task-oriented. We evaluate the approach with a large data set based on Switchboard (Godfrey et al., 1992) with a tag set of 44 dialogue acts. Our classification task was thus harder. Yet, we achieve better recognition result.

2.4 Previous Work on Class Imbalance Problem

A class-imbalance problem refers to a data set, where the distribution of examples among classes in the data set are highly skewed. This poses challenge to classifiers because they are typically designed for balanced data sets. The class-imbalance problem is encountered by many real-world applications. Previously, there have been methods to deal with imbalanced data sets in various domains, including oil-spill detection (Kubat and Matwin, 1997), credit-card fraud detection (Chan and Stolfo, 1998), premature-birth prediction (Grzymala-Busse et al., 2000), and text disambiguation such as coreference resolution (Ng and Cardie, 2002) and complement-adjunct distinction (Kermanidis et al., 2004).

There are two common approaches to the class-imbalance problem. One approach is at *data level*. The aim is to re-sample the original data to a more balanced distribution. The sampled data set is then provided to classifier for training. Methods of this approach mainly include under-sampling, over-sampling, and their variants and combinations. Another approach for this problem is at *algorithmic level*. It focuses on modifying the learning mechanism of classifiers. The aim is to reduce the statistical bias toward to the majority class. Methods of this approach mainly include cost-sensitive learning and ensemble learning.

We employ two methods to address the class-imbalance problem. They are at the data and algorithmic level respectively. Our first method is to employ over-sampling. But it differs from the prevalent approaches used previously. Traditional sampling techniques, either over-sampling or under-sampling, are mainly designed for binary-class problems. When these techniques deal with multi-class problems, some classes have to be collapsed or ignored (Provost et al., 1998; Chawla et al., 2002). This probably destroys the nature of the original problems. In contrast, our method can cope with problems with more than two classes. The amount of re-sampling is all determined by a sole heuristic function.

Our second method is quite similar to cost-sensitive learning. The aim is to intensively learn the minority class through some weighting scheme. However, our learning method focuses on minimizing misclassification error, whereas cost-sensitive learning targets at reducing misclassification cost.

In Section 5.1, we review both techniques on under-sampling and over-sampling, and present a comparison with our sampling method. In Section 5.1, we briefly survey a number of learning techniques currently used for the class-imbalance problem. Their differences with our learning method are also discussed.

2.4.1 Approach at Data Level

One method for balancing the class distribution is *under-sampling*. Its aim is to remove some examples from the majority classes while keeping the minority classes unchanged. Examples of this sampling research include (Kubat and Matwin, 1997; Japkowicz and Stepehn, 2002; Drummond and Holte, 2003). The known disadvantage of under-sampling is the removal of potentially useful information. This can be addressed by selectively removing the unnecessary majority-class examples through some heuristics. Two previous work in this direction are discussed.

Kubat and Matwin (1997) divided majority-class examples into four categories: noisy, borderline, redundant and safe. They selectively removed the noisy and borderline examples using the Tomek link technique (Tomek, 1976). They found that removing borderline examples could gain significant improvement.

Japkowicz and Stepehn (2002) evaluated the class-imbalanced problem on artificial data. The training sets were generated based on an one-dimensional input, with various combinations of class complexity (how classes are subdivided into clusters), size of training set, and level of imbalance. In one experiment, two under-sampling techniques were considered. One technique is *random under-sampling*. Inside, majority-class examples were randomly removed until their numbers matched the minority class. The other technique is *focused under-sampling*, which only removed the majority-class examples lying far away from the decision boundary. In another experiment, two over-sampling techniques were used with similar settings. Their work found that both under-sampling techniques were less effective than the over-sampling techniques.

On the other hand, *over-sampling* is another common method that can equalize the class distribution. Its aim is generally to replicate the examples for the minority classes while keeping unchanged for majority classes. Examples of this sampling research include (Ling and Li, 1998; Chawla et al., 2002; Batista et al., 2004). A disadvantage of over-sampling is that it increases the time to build the classifier because of the replicated examples.

The work by Ling and Li (1998) is most similar to our sampling method. In one of their experiments, they over-sampled the minority class with random replacement. The amount is arbitrarily set to 1 time, 2 times, 5 times, 10 times, and 20 times of the original size. The majority-class examples were respectively sampled to match with the number for each group. Here, our method differs from theirs by keeping all the majority-class examples unchanged, without reducing its number though sampling. Another difference from their method is the way in determining the amount of replacement. Ours is determined by a function instead of setting it arbitrarily. The function aims to re-scale the class distribution by flattening its shape proportionally. We also note that our method is capable for multi-class problems, unlike the traditional methods which often are limited to binary-class problem.

An inherent issue of over-sampling is that no new information is introduced to the system (Ling and Li, 1998). Technically, over-sampling hardly yields a clear decision-boundary by the replicated features. It just makes the original feature-region more crowded than before. Recently, a generation technique called SMOTE was proposed to counter this issue (Chawla et al., 2002). SMOTE operated in feature space to generate synthetic examples. Inside, each minority-class example is represented by a real-valued feature vector. SMOTE takes the difference between each vector and one of its nearest neighbor. The difference is multiplied by a random number between 0 to 1. A new feature vector (i.e. new example) is thus created by adding the difference to the original vector. However, when examples involve binary-valued features (e.g., a word whether exists or not exists), the difference between vectors is no longer a simple mathematical substraction. It often requires a more sophisticated metric to account for the difference (Cost and Salzberg, 1993). In this thesis, both disambiguation tasks involve binary-valued features. For simplicity, our over-sampling technique thus omits the synthesis process like the one used in SMOTE.

Regarding to the performance, previous work have not reached any conclusive results about whether under-sampling or over-sampling is better. Often, reported results were conflicted between literatures. Some claimed that under-sampling was better (Domingos, 1999; Drummond and Holte, 2003), but some suggested over-sampling should be preferred (Japkowicz and Stepehn, 2002; Batista et al., 2004). Nevertheless, research work combining both sampling methods have been made. For example, Solberg and Solberg (1996) over-sampled the minority class up to a number of examples, and under-sampled the majority class down to the same number, to generate an equal class distribution.

2.4.2 Approach at Algorithmic Level

One method to the class-imbalance problem is to apply *cost-sensitive learning* (Pazzani et al., 1994; Domingos, 1999; Elkan, 2001) to classifier. Cost here means some prior weighting that measures how serious is the consequence if an example is misclassified. A wider definition of cost, and its various types were introduced in Turney (2000). For computation purpose, cost is quantified in numerical value. Higher values are usually assigned to the cases that minority-class examples are misclassified. It is because, for example, misclassifying an ill patient (minority class) as a healthy one (majority class) is rarely tolerable. Conversely, it is relatively acceptable if a health patient is misclassified to be ill for a just further checking. By convention, the value in each possible case is stored in a matrix, whose rows and columns respectively represent the predicted and correct classes. The diagonal cells in the matrix are all set to zero because they refer to correct classifications. During training, the objective is to minimize the overall misclassification cost instead of the total number of errors. One disadvantage of this approach is that the information about the cost is hardly available in practice. Approximation (Sahami et al., 1998) or smoothing (Zadrozny and Elkan, 2001) is thus required, but probably unreliable. Despite its drawback, cost-sensitive learning was often reported as a better choice than simple sampling methods (Domingos, 1999; Japkowicz and Stepehn, 2002).

Compared with cost-sensitive learning, our learning method is inherently error-based. We aim to reduce the number of errors, instead of minimizing the misclassification cost. We modify the learning mechanism of Maximum Entropy by adding weighted parameters to the updating rule of GIS algorithm. In each iteration of the learning method, each training example is predicted by the ME classifier. The predicted label and the correct label form a pair. The occurrence frequency of each pair is recorded in a matrix, whose rows and columns respectively represent the predicted and correct classes. By taking the diagonal values, we can obtain the accuracy distribution among classes. To predict better in next iteration, more weight is assigned to the features whose class (probably minority class) is predicted poorly. Conversely, less weight is assigned to the features whose class (probably majority class) is predicted correctly. The amount of weight is determined by a inverse function about accuracy. Here, whenever a new iteration begins, the matrix value changes. The weight assignment is re-computed too. This differs from the fixed value pre-defined in cost-sensitive learning. Moreover, we use the matrix to store the information predicted by the classifier rather than the misclassification cost.

Another algorithmic method that addresses the class-imbalance problem is *ensemble learning*. Its aim is to use a collection of classifiers to reduce the prediction variance. This can minimize the bias toward the majority class. While cost-sensitive learning modifies the internal mechanism of a classifier, ensemble learning modifies the external mechanism by classifier combination.

MetaCost (Domingos, 1999) uses a variant of bagging (Breiman, 1996) as the ensemble method. MetaCost is a wrapper technique for making classifier cost-sensitive. In MetaCost, multiple replicates of training set are created by taking samples from the original training set with replacement. Each replicate is used to train an individual classifier. The original training set is then relabeled by the averaged prediction from all classifiers. A final classifier is formed by training on the relabeled training set.

Chan and Stolfo (1998) used stacking (Wolpert, 1992) to train multiple classifiers. They divided the training data into several subsets for each classifier. The prediction of each classifier was concatenated with the original training set. A meta-classifier was then formed by training on the concatenated set.

Boosting (Freund and Schapire, 1997) is an another ensemble-learning technique. It combines the outputs of many weak classifiers to produce a strong committee using weighted voting. Its various enhancements for resolving class-imbalance problem have been proposed, such as AdaCost (Fan et al., 1999) and RareBoost (Joshi et al., 2001).

Some work also combined boosting with sampling methods, such as SMOTEBoost (Chawla et al., 2003) and DataBoost (Guo and Viktor, 2004).

Chapter 3

Classification by Maximum Entropy

3.1 Introduction

Maximum Entropy (ME) offers a statistically appealing way to approximate an unknown probability distribution, with a few assumptions made on sample data. It originated from statistical mechanics (Jaynes, 1957) and has been widely used in various applications (Pietra et al., 1997; Ratnaparkhi, 1999; Phillips et al., 2006). In a machine learning perspective, ME can be viewed as Maximum Likelihood (ML) training for exponential distributions. This chapter, based on this perspective, gives an introduction to ME, with focus on modeling a classifier whose decision boundary is computed by probability distribution. Some relevant ME derivations and computation issues are also included.

ME offers more advantages than disadvantages. Firstly, ME can be modeled for various uses, such as density estimation via unconditional distribution (see Section 3.4), and classification via conditional distribution (see Section 3.5). Secondly, it makes few unjustified assumptions on training data to model the target distribution (see Section 3.6). Thirdly, it can combine heterogeneous features for decision making, while keeping easy for feature encoding (see Section 3.7). Fourthly, the parameter estimation in ME can be computed by iterative methods such as GIS algorithm (see Section 3.8) or gradient methods. Lastly, bayesian regularization can be employed, in case of overfitting to training data (see Section 3.10).

Some drawbacks are noted. ME training is generally a time-consuming process, especially when there are a lot of features (see Section 3.9). On the other hand, it is expressed as an exponential form, which is inherently unbounded above. Parameter value might thus be incredibly large, resulted in inaccurate prediction.

3.2 Notation

- $\mathcal{X}$: the input space
- $\mathcal{Y}$: the finite set of class labels
- x: an instance over the input space $\mathcal{X}$
- y: a class label in the finite set $\mathcal{Y}$
- $\xi = \mathcal{X} \times \mathcal{Y}$: the joint event space
- $S \in \xi$: the training sample data (a subset of the event space)
- (x, y): a possible event over ξ
- (x_i, y_i): an i-th event over S for $i = 1, 2, \ldots, n$
- $\hat{p}$: the true underlying probability distribution over $\mathcal{X} \times \mathcal{Y}$
- p: the model probability distribution that we want to estimate
- $\tilde{p}$: the observed probability distribution over S
- H: the entropy function of a probability distribution
- $f_j : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$: a j-th binary-valued function (also known as *feature function*) that returns either 0 or 1 to indicate whether a feature of x co-occurs with y
- $E_p[f_j]$: the model expectation of a j-th function with respect to p
- $E_{\tilde{p}}[f_j]$: the observed expectation of a j-th function with respect to $\tilde{p}$
- w_j: a j-th Lagrange multiplier (variously known as *model parameter* or *weight*)
- γ: a Lagrange multiplier

- $\mathbf{f}(x, y) \in \mathbb{R}^m$: the m-dimensional sparse vector that stores the value of each j-th function for $j = 1, 2, \ldots, m$
- $\mathbf{w} \in \mathbb{R}^m$: the m-dimensional real-valued vector that stores the value of each j-th multiplier
- $\mathbf{c} \in \mathbb{R}^m$: the m-dimensional real-valued vector that stores the model counts of each j-th function
- $\tilde{\mathbf{c}} \in \mathbb{R}^m$: the m-dimensional real-valued vector that stores the observed counts of each j-th function
- $||\mathbf{w}||_1$: l_1-norm, the sum of magnitudes of each elements in $\mathbf{w}$, defined as $\sum\limits_{i=1}^{m} |w_i|$
- $||\mathbf{w}||_2$: l_2-norm, the Euclidean length of $\mathbf{w}$, defined as $\sqrt{\sum\limits_{i=1}^{m} w_i^2}$
- L: the log-likelihood function of S with respect to $\mathbf{w}$
- L': the log-posterior function of S with respect to $\mathbf{w}$

3.3 Preliminaries

Consider a supervised classification problem. We are given a set $S = \{(x_i, y_i)\}_{i=1}^{n}$ of n training examples which are drawn independently from some distribution $\hat{p}$. Here, x_i is the input and y_i is the class label. For each x_i, a set of feature functions f_j's are defined for classification.

Given these ingredients, a classifier of assigning x to y can be implemented with a conditional probability distribution $p(y|x)$ by choosing the class y with the highest probability. By definition, $p(y|x)$ can be computed by the ratio of the joint probability distribution $p(x, y)$ and the marginal probability distribution $p(x)$. Here, $p(x, y)$ is such a target distribution that can be approximated with Maximum Entropy (see Section 3.4). On the other hand, $p(y|x)$ can also be computed directly based on a conditional version of ME (see Section 3.5).

3.4 Exponential Parametric Form

The probability distribution modeled by Maximum Entropy belongs to the family of exponential distributions, which is parameterized in an exponential form. This section shows how the parametric form is formulated from the theoretical foundation of Maximum Entropy.

The principle of Maximum Entropy (Jaynes (1957); Ratnaparkhi (1998)) states that among the probability distributions that satisfy our partial known information about the data, the best distribution to choose is the one that maximizes the entropy (i.e. closest to uniform distribution with the least bias) while remaining consistent with the available information.

For a classification problem modeled with this principle, the aim is to look for a probability distribution $p(x, y)$ by (1) maximizing the entropy H for each i-th event (x_i, y_i) in the sample data S, and (2) satisfying the equality constraints which are encoded with feature functions f_j's, and an axiom of the probability theory:

$$\underset{p}{\text{maximize}} \qquad H(p) = -\sum_{x,y} p(x,y) \log p(x,y) \tag{3.1}$$

$$\text{subject to} \qquad E_p[f_j] - E_{\tilde{p}}[f_j] = 0 \quad \forall j \tag{3.2}$$

$$\sum_{x,y} p(x,y) - 1 = 0 \tag{3.3}$$

Here, $E_p[f_j]$ is the expectation of a j-th feature function with respect to the model distribution p being estimated, and $E_{\tilde{p}}[f_j]$ is the expectation of a j-th feature function with respect to the training data. The notation for a feature function f_j is in short of $f_j(x, y)$. Both expectations are respectively defined as:

$$E_p[f_j] = \sum_{x,y} p(x,y) f_j(x,y) \tag{3.4}$$

$$E_{\tilde{p}}[f_j] = \sum_{i}^{n} \tilde{p}(x_i, y_i) f_j(x_i, y_i) \tag{3.5}$$

Equivalently, this formulation becomes a constrained optimization problem: find the *extrema* of an *objective function* which is subject to a set of *constraints*. Here, the objective function is Equation 3.1, the constraints are the ones in Equation 3.2 and in Equation 3.3 respectively, and the extrema we want to find is the maximized entropy of the joint probability distribution $p(x, y)$. To reduce the constrained problem to an unconstrained problem, Lagrangian method (see, for example, Arfken (1985) for a review) is employed. Inside, a scalar variable for each constraint, namely Lagrange multiplier, is introduced so that a Lagrangian function F can be defined:

$$F(p, \mathbf{w}, \gamma) = H(p) - \sum_{j}^{m} w_j \Big(E_p[f_j] - E_{\tilde{p}}[f_j] \Big) - \gamma \Big(\sum_{x,y} p(x, y) - 1 \Big) \tag{3.6}$$

Here, finding the maxima of H is equivalent to finding the maxima of F. Since F is converse, its maximum value can be found by setting its gradient equal to 0. By taking the partial derivative of F with respect to $p(x, y)$, Equation 3.6 becomes:

$$\frac{\partial F}{\partial p(x, y)} = -(1 + \log p(x, y)) - \sum_{j}^{m} w_j f_j(x, y) - \gamma \tag{3.7}$$

Setting the derivative equal to 0 and solving $p(x, y)$, we have:

$$\log p(x, y) = -1 - \sum_{j}^{m} w_j f_j(x, y) - \gamma \tag{3.8}$$

$$= -1 - \mathbf{w} \cdot \mathbf{f}(x, y) - \gamma \tag{3.9}$$

Returning to the base-10 logarithm of $p(x, y)$, it becomes:

$$p(x, y) = e^{\gamma - 1} e^{\mathbf{w} \cdot \mathbf{f}(x,y)} \tag{3.10}$$

Summing all over the event space, we get:

$$\sum_{x,y} p(x, y) = e^{\gamma - 1} \sum_{x,y} e^{\mathbf{w} \cdot \mathbf{f}(x,y)} \tag{3.11}$$

Because $\sum_{x,y} p(x, y)$ in Equation 3.3 equal to 1, $e^{\gamma-1}$ in Equation 3.11 thus equals to:

$$e^{\gamma - 1} = \frac{1}{\sum_{x,y} e^{\mathbf{w} \cdot \mathbf{f}(x,y)}} \tag{3.12}$$

Finally, substituting Equation 3.12 into 3.10, we get the parametric form of ME in terms of parameter $\mathbf{w}$ for the joint probability distribution $p(x, y)$:

$$p(x, y) = \frac{e^{\mathbf{w} \cdot \mathbf{f}(x,y)}}{\sum_{x,y} e^{\mathbf{w} \cdot \mathbf{f}(x,y)}} \tag{3.13}$$

3.5 Conditional Maximum Entropy

For the classification problem with large instance space, the conditional probability distribution $p(y|x)$ is preferable to be directly computed. This is because evaluating the denominator in Equation 3.13 requires summing all over possible combinations of both classes and input, which is often infeasible in practice. Unless a good and fast approximation (Schofield, 2004) can be made, one may neglect to model the input x, but focus on determining which class y fits x best. This gives us the parametric form for the conditional Maximum Entropy (e.g., Berger et al. (1996)), defined as:

$$p(y|x) = \frac{e^{\mathbf{w}\cdot\mathbf{f}(x,y)}}{\sum\limits_{y} e^{\mathbf{w}\cdot\mathbf{f}(x,y)}} \tag{3.14}$$

This parametric form differs from the original form mainly in the denominator, which is technically called partition function in the literature of statistical mechanics. The modification on the denominator not only reduces the unnecessary burden imposed to the model, but also makes the computation tractable for tremendous sample data.

3.6 As a Special Case of Minimum Relative Entropy

Maximum Entropy (ME) is a special case of Minimum Relative Entropy (MRE), assuming that the target distribution is uniform. Suppose that some sample data is given, and its true underlying distribution $\hat{p}(x, y)$ is also known. The aim is to look for a distribution $p(x, y)$, approximated to $\hat{p}(x, y)$, subject to a set of constraints. The MRE principle (Kullback, 1959; Shore and Johnson, 1980) states that among the probability distributions that satisfy the constraints, the best distribution to choose is the one with the minimum entropy relative to $\hat{p}(x, y)$. Here, the relative entropy between $\hat{p}(x, y)$ and $p(x, y)$ is known as Kullback-Leilber (KL) divergence, defined as:

$$KL(p||\hat{p}) = \sum_{x,y} p(x,y)\log\frac{p(x,y)}{\hat{p}(x,y)} \tag{3.15}$$

$$= \sum_{x,y} p(x,y)\log p(x,y) - \sum_{x,y} p(x,y)\log\hat{p}(x,y) \tag{3.16}$$

ME is a special case of MRE when assuming $\hat{p}(x, y)$ to be a uniform distribution u, defined as:

$$u(x,y) = \frac{1}{|\mathcal{X} \times \mathcal{Y}|} \tag{3.17}$$

Now, the KL divergence becomes:

$$KL(p||\hat{p}) = \sum_{x,y} p(x,y)\text{log}p(x,y) - \sum_{x,y} p(x,y)\text{log}\frac{1}{|\mathcal{X}\times\mathcal{Y}|} \tag{3.18}$$

$$= -H(p) - \sum_{x,y} p(x,y)\text{log}\frac{1}{|\mathcal{X}\times\mathcal{Y}|} \tag{3.19}$$

$$= -H(p) + \text{log}|\mathcal{X}\times\mathcal{Y}|\sum_{x,y} p(x,y) \tag{3.20}$$

$$= -H(p) + \text{log}|\mathcal{X}\times\mathcal{Y}| \geq 0 \tag{3.21}$$

As depicted, the entropy $H(p)$ is bounded by the logarithm of the cardinality $|\mathcal{X}\times\mathcal{Y}|$:

$$H(p) \leq \text{log}|\mathcal{X}\times\mathcal{Y}| \tag{3.22}$$

The equality only holds when $p(x,y) \approx \hat{p}(x,y) = u(x,y)$. Thus, maximizing entropy is equivalent to minimizing relative entropy at the assumption that the target distribution is uniform.

3.7 Feature Encoding

Features in Maximum Entropy are encoded as binary functions to indicate their existence. Given a feature set $\{v_1, v_2, \ldots, v_i, \ldots, v_n\}$, their values are generated by a function $\text{GEN}(x, v_i)$, where x is an input instance. Assuming the value of a class label y is LABEL, a *feature function* $f_j(x,y)$ can be encoded as:

$$f_j(x,y) = \begin{cases} 1 & \text{if } y\text{=LABEL and } v_i\text{= GEN}(x,i) \\ 0 & \text{otherwise} \end{cases} \tag{3.23}$$

Suppose that the feature set generates m possible feature functions. Each function output (either 0 or 1) is used to indicate if a feature v_i with a particular value co-exists with a class label. All the binary outputs are then gathered to form a m-dimensional feature vector $\mathbf{f}(x,y)$ for classifier input.

3.8 Parameter Estimation

3.8.1 Maximum Likelihood

Parameter estimation is a classic problem in statistics, and can be approached in several ways. Maximum likelihood (ML) (see, for example, Bickel and Doksum (1977) for a review) is one statistical method that estimates the parameters $\mathbf{w}$ for a (joint) probability distribution. Given a training data $S = \{(x_1, y_1), (x_2, y_2), \dots (x_n, y_n)\}$, the data likelihood function DL is defined as:

$$DL(\mathbf{w}) = p(S|\mathbf{w}) \tag{3.24}$$

$$= \prod_{i=1}^{n} p(y_i, x_i|\mathbf{w}) \tag{3.25}$$

The best estimate for the parameters $\mathbf{w}_{ML}$ is defined to be the one that maximizes the likelihood from observed data, denoted as:

$$\mathbf{w}_{ML} = \operatorname*{argmax}_{\mathbf{w}\in\mathbb{R}} \prod_{i=1}^{n} p(y_i, x_i|\mathbf{w}) \tag{3.26}$$

The ML method can be used for estimating parameters for *generative* Maximum Entropy.

3.8.2 Maximum Conditional Likelihood

Similarly, maximum conditional likelihood (MCL) is another statistical estimator consistently designed for conditional distribution. The data conditional likelihood function DCL is defined as:

$$DCL(\mathbf{w}) = p(S|\mathbf{w}) \tag{3.27}$$

$$= \prod_{i=1}^{n} p(y_i|x_i, \mathbf{w}) \tag{3.28}$$

The best estimate for the parameters $\mathbf{w}_{MCL}$ is defined to be the one that maximizes the conditional likelihood from observed data, denoted as:

$$\mathbf{w}_{MCL} = \operatorname*{argmax}_{\mathbf{w}\in\mathbb{R}} \prod_{i=1}^{n} p(y_i|x_i, \mathbf{w}) \tag{3.29}$$

Given the parametric form of *conditional* maximum entropy, the parameters $\mathbf{w}$ can be estimated by MCL method. The conditional likelihood function L for conditional maximum entropy is defined as:

$$L(\mathbf{w}) = \prod_{i=1}^{n} \frac{e^{\mathbf{w}\cdot\mathbf{f}(x_i,y_i)}}{\sum_{y'\in\mathcal{Y}} e^{\mathbf{w}\cdot\mathbf{f}(x_i,y')}} \tag{3.30}$$

Because of the monotonicity of the logarithm, the value of L is symmetrically the same as the one after taking logarithm, denoted as:

$$L(\mathbf{w}) = \sum_{i=1}^{n} \mathbf{w}\cdot\mathbf{f}(x_i, y_i) - \sum_{i=1}^{n} \log \sum_{y'\in\mathcal{Y}} e^{\mathbf{w}\cdot\mathbf{f}(x_i,y')} \tag{3.31}$$

By taking the partial derivative of L with respect to $\mathbf{w}$, it becomes:

$$\frac{\partial L}{\partial \mathbf{w}} = \sum_{i=1}^{n} \mathbf{f}(x_i, y_i) - \sum_{i=1}^{n} \frac{\sum_{y' \in \mathcal{Y}} \mathbf{f}(x_i, y') e^{\mathbf{w} \cdot \mathbf{f}(x_i, y')}}{\sum_{z' \in \mathcal{Y}} e^{\mathbf{w} \cdot \mathbf{f}(x_i, z')}} \tag{3.32}$$

$$= \sum_{i=1}^{n} \mathbf{f}(x_i, y_i) - \sum_{i=1}^{n} \sum_{y' \in \mathcal{Y}} \mathbf{f}(x_i, y') \frac{e^{\mathbf{w} \cdot \mathbf{f}(x_i, y')}}{\sum_{z' \in \mathcal{Y}} e^{\mathbf{w} \cdot \mathbf{f}(x_i, z')}} \tag{3.33}$$

$$= \sum_{i=1}^{n} \mathbf{f}(x_i, y_i) - \sum_{i=1}^{n} \sum_{y' \in \mathcal{Y}} \mathbf{f}(x_i, y') p(y'|x_i) \tag{3.34}$$

or, in short:

$$\frac{\partial L}{\partial \mathbf{w}} = \tilde{\mathbf{c}} - \mathbf{c} \tag{3.35}$$

Once setting the derivative equal to 0, the parameters $\mathbf{w}$ are the optimal ones (or value maximized) for which $\mathbf{c}$ and $\tilde{\mathbf{c}}$ are equal:

$$\tilde{\mathbf{c}} = \mathbf{c} \tag{3.36}$$

where $\tilde{\mathbf{c}}$ is the observed counts of every j-th feature function with respect to S, and $\mathbf{c}$ is the model counts of every j-th feature function with respect to p. As depicted in Equation 3.33, calculating $\mathbf{c}$ requires the involvement of $\mathbf{w}$ again. There is thus no closed-form solution for Equation 3.35. Alternatively, this can be solved with a numeric root-finding algorithm, such as Generalized Iterative Scaling (GIS) (Darroch and Ratcliff, 1972; Ratnaparkhi, 1998). Figure 3.1 shows the main GIS algorithm, with two procedures shown in Figure 3.2 and 3.3 respectively. At each iteration t, as depicted in Figure 3.1, the parameters $\mathbf{w}$ are successively adjusted, based on divergence between $\tilde{\mathbf{c}}$ and $\mathbf{c}$. The algorithm will converge to $\mathbf{w}_{MCL}$ after a number of t iterations. Generally, using 100 iterations is good enough in this thesis, because the change in log-likelihood L is almost negligible.

3.9 Complexity for Generalized Iterative Scaling

The Generalized Iterative Scaling (GIS) algorithm for estimating parameters is generally slow. The algorithm involves a main loop, performing successive iterations (see Figure 3.1). At each iteration, it involves another loop over *all* the examples in the training data, which is the most time-consuming part in the algorithm (see Figure 3.2).

```
GIS($S, \tau, \eta$)
1.  $\mathbf{w} \leftarrow 0$, $\mathbf{c} \leftarrow 0$, $T \leftarrow 0$;
2.  $C \leftarrow \max_i |x_i|$
3.  for each $i \in n$
4.      $\tilde{\mathbf{c}} \leftarrow \tilde{\mathbf{c}} + \mathbf{f}(x_i, y_i)$;
5.  repeat
6.      $(L^{(t)}, \mathbf{c}) =$ COUNTS($S, \mathbf{w}$);
7.      $\mathbf{w} \leftarrow \mathbf{w} + \frac{1}{C}\log\frac{\tilde{\mathbf{c}}}{\mathbf{c}}$;
8.      $t \leftarrow t + 1$;
9.  until $(|L^{(t)} - L^{(t-1)}| \leq \tau)$ or $(t < \eta)$, where $\tau$ and $\eta$ are thresholds
```

Figure 3.1: `GIS` algorithm

```
COUNTS($S, \mathbf{w}$)
1.  $L \leftarrow 0$, $\mathbf{c} \leftarrow 0$;
2.  for each $i \in n$
3.      $p \leftarrow$ PROBABILITY($x_i, \mathbf{w}$)
4.      $L \leftarrow L + \log p[y_i]$;
5.      for each $y \in \mathcal{Y}$
6.          $\mathbf{c} \leftarrow \mathbf{c} + p[y] \times \mathbf{f}(x_i, y_i)$;
7.  return $(L, \mathbf{c})$;
```

Figure 3.2: `COUNTS` procedure

```
PROBABILITY($x_i, \mathbf{w}$)
1.  $Z \leftarrow 0$, $s \leftarrow 0$, $p \leftarrow 0$;
2.      for each $y \in \mathcal{Y}$
3.          $s[y] = e^{\mathbf{w} \cdot \mathbf{f}(x_i, y)}$;
4.          $Z \leftarrow Z + s[y]$;
5.      for each $y \in \mathcal{Y}$;
6.          $p[y] \leftarrow \mathbf{s}_y / Z$;
7.  return $p$;
```

Figure 3.3: `PROBABILITY` procedure

Here, line 3 involves another inner loop to compute the normalized probability distribution among the classes (see Figure 3.3). Most importantly, line 5 and 6 involves the computation for *all* the modeled counts of feature functions, and each feature function is bounded by its type of implementation. This computation thus dominates the running time of each iteration. The complexity of each iteration is $O(|N||\mathcal{Y}|A)$, where $|N|$ is the number of training examples, $|\mathcal{Y}|$ is the set of classes, and A is the average number of feature functions for an example.

3.10 Regularized Estimation: Maximum A Posteriori

Regularization generally prevents overfitting to the training data by penalizing the fitted parameters by a certain quantity, which is controlled by a hyperparameter α and a function $R(\mathbf{w})$. For the regularization for Maximum Entropy training, maximum a posteriori (MAP) can be used. MAP is a statistical method for parameter estimation. The best estimate is defined to maximize the posterior probability from observed data. By Bayes' theorem, the posterior distribution $p(\mathbf{w}|S)$ can be computed by combining the likelihood distribution $p(S|\mathbf{w})$ with a prior distribution $p(\mathbf{w})$. The choice for the prior distribution $p(\mathbf{w})$ depends on the problem domain. Gaussian prior (Berger and Miller, 1998; Chen and Rosenfeld, 2000) and Laplace prior (Goodman, 2004) have respectively been used for natural language classification problems. In case of Gaussian distribution, the MAP method estimates $\mathbf{w}$ by taking the maximum value of $p(\mathbf{w}|S)$:

$$\mathbf{w}_{MAP} = \underset{\mathbf{w}\in\mathbb{R}}{\operatorname{argmax}}\, p(\mathbf{w}|S) \tag{3.37}$$

$$= \underset{\mathbf{w}\in\mathbb{R}}{\operatorname{argmax}}\, p(S|\mathbf{w})p(\mathbf{w}) \tag{3.38}$$

$$= \underset{\mathbf{w}\in\mathbb{R}}{\operatorname{argmax}} \prod_{i=1}^{n} \frac{e^{\mathbf{w}\cdot\mathbf{f}(x_i,y_i)}}{\sum\limits_{y'\in\mathcal{Y}} e^{\mathbf{w}\cdot\mathbf{f}(x_i,y')}} \prod_{j=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{w_j^2}{2\sigma^2}\right) \tag{3.39}$$

where σ^2 is the variance of parameters.

To ease the numerical calculations, the estimation criterion for Equation 3.39 is now given by a logarithmic function L':

$$L'(\mathbf{w}) = \sum_{i=1}^{n} \log \frac{e^{\mathbf{w}\cdot\mathbf{f}(x_i,y_i)}}{\sum\limits_{y'\in\mathcal{Y}} e^{\mathbf{w}\cdot\mathbf{f}(x_i,y')}} + \sum_{j=1}^{m} \log \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{w_j^2}{2\sigma^2}\right) \tag{3.40}$$

$$= L(\mathbf{w}) - \sum_{j=1}^{m} \frac{w_i^2}{2\sigma^2} - \log\sqrt{2\pi\sigma^2} \tag{3.41}$$

$$= L(\mathbf{w}) - \frac{1}{2\sigma^2}||\mathbf{w}||_2^2 - \text{const.} \tag{3.42}$$

$$= L(\mathbf{w}) - \alpha R(\mathbf{w}) - \text{const.} \tag{3.43}$$

As depicted, the new function $L'(\mathbf{w})$ is equal to the log-likelihood function $L(\mathbf{w})$ with a penalty term, which is the product of a hyperparameter $\frac{1}{2\sigma^2}$ and a function $R(\mathbf{w}) = ||\mathbf{w}||_2^2$. In the context of machine learning, this is equivalent to l_2^2-norm regularization. There have also been some work using Laplace prior (Goodman, 2004) for l_1-norm regularization, formulating $R(w)$ as $||\mathbf{w}||_1$.

For computation, integrating Gaussian prior into the GIS algorithm requires subtracting an extra term from the derivative in Equation 3.35:

$$\frac{\partial L}{\partial \mathbf{w}} = \tilde{\mathbf{c}} - \mathbf{c} - \frac{\mathbf{w}}{\sigma^2} \tag{3.44}$$

or equivalently denoted as:

$$\tilde{\mathbf{c}} = \mathbf{c} + \frac{\mathbf{w}}{\sigma^2} \tag{3.45}$$

Chapter 4

Semantic Role Labeling

4.1 Introduction

[1] Semantic role labeling is a simplified task for deep semantic analysis. Its aim is to label sequences of words of a sentence with their semantic roles with respect to a verb in the sentence. Examples of semantic roles include AGENT, PATIENT, TEMPORAL, LOCATIVE, etc. Recently, semantic role labeling research has been facilitated by the release of semantically annotated corpora, which allow researchers to apply corpus-based learning techniques to estimate probability distribution of semantic roles. The corpus statistics obtained may then be used to assign semantic roles to unseen text.

Automatic semantic role labeling provides benefits to many natural language applications such as information extraction, summarization, question answering, etc. Among these applications, the advantage to Information Extraction (IE) is described. Traditional approaches to information extraction systems are heavily dependent on hand-crafted verb-based patterns where the slots are used to fill in extracted entities. However, encoding such patterns into systems is often tedious, and it requires the expertise of linguists familiar with the underlying domain. In contrast, verb-argument structure generated by semantic role labeling enables a pattern-free approach to IE systems, which is more flexible and portable.

This chapter presents machine learning techniques on semantic role labeling. Two

[1] The preliminary version of the first part of this chapter has been published in (Lan et al., 2004). In the second part of this chapter, we propose a new labeling approach to the same task, at the lesson learnt from the previous work.

classifier systems are implemented, both based on the conditional maximum entropy approach (Ratnaparkhi, 1998). Each system employs a different approach. In the first system, the task is formulated as a sequential tagging problem (Ramshaw and Marcus, 1995) and experiments are conducted on the preliminary release of PropBank (Kingsbury and Palmer, 2002). In the second system, we propose a new approach, where the task is decomposed into a number of decision sub-problems. Experiments are conducted on the benchmark data set provided by the CoNLL-2005 shared task (Carreras and Màrquez, 2005). Moreover, we also employ two methods to address the class-imbalance problem encountered in the semantic role labeling task. Evaluation reveals that the methods are effective in improving the macro-average performance.

4.2 Task Description

4.2.1 Problem Definition

The goal of semantic role labeling is to group sequences of words of a sentence together into arguments and label them with their semantic roles, based on a particular verb in the sentence. A verb and its set of arguments form a proposition specifying a truth-conditional meaning of the sentence. In general, a sentence may contain a number of propositions. For example, there are two verbs in the following sentence, each governing a proposition, as shown in 1a and 1b, respectively (in each case, the verb concerned is italicized):

1a. The state *gave* CenTrust 30 days to sell the Rubens.

1b. The state gave CenTrust 30 days to *sell* the Rubens.

By grouping the words sequentially, non-embedded and non-overlapping arguments are formed, as shown in 2a and 2b below (arguments are bracketed):

2a. (The state) *gave* (CenTrust) (30 days to sell the Rubens).

2b. (The state) gave CenTrust (30 days) to *sell* (the Rubens).

Upon successful labeling, semantic roles are assigned to arguments, as shown in 3a and 3b (semantic roles are small-capitalized):

3a. (Agent The state) *gave* (Recipient CenTrust) (Theme 30 days to sell the Rubens).

3b. (Agent The state) gave CenTrust (Temporal 30 days) to *sell* (Theme the Rubens).

As illustrated, the task can generally be viewed as the following two sub-problems.

- Argument recognition - For each targeted verb, find all the argument candidates by locating their boundaries without embedding and overlapping.
- Semantic role assignment - For each identified argument, label it by a semantic role with respect to the verb.

Various approaches for addressing these problems have been reviewed in Section 2.2.

4.2.2 Evaluation Metrics

The evaluation is performed on unseen sentences. For each sentence, the verb is first given. The system then recognizes arguments and semantic roles with respect to the verb. An argument is said to be correctly *recognized* if both the words spanning the argument and its semantic role are correctly predicted.

The performance is measured in terms of precision, recall and F_1 measure. Precision (p) is the percentage of predicted recognitions that are correct. Recall (r) is the percentage of target recognitions that are correctly predicted by the system. F_1 measure (F_1) is the weighted harmonic mean of precision and recall, defined as $F_1 = 2pr/(p+r)$.

All these three measures can be calculated by two different averaging methods: micro-averaging and macro-averaging. In micro-averaging, the average value is calculated over all recognitions. We use the official script provided by the CoNLL-2004 and CoNLL-2005 conference to obtain the micro-average result. The three micro-average formulae for precision, recall, and F_1 measure are shown as follows:

$$\text{micro-precion} = \frac{\text{total number of correctly recognized arguments}}{\text{total number of recognized arguments}} \tag{4.1}$$

$$\text{micro-recall} = \frac{\text{total number of correctly recognized arguments}}{\text{total number of target arguments}} \tag{4.2}$$

$$\text{micro-F}_1 = \frac{2 \times \text{micro-precion} \times \text{micro-recall}}{\text{micro-prec.} + \text{micro-recall}} \tag{4.3}$$

In macro-averaging, the average value is calculated for each category and then averaged by the number of classes N. The three macro-average formulae for precision, recall, and F_1 measure are shown as follows:

$$\text{macro-prec.} = \frac{1}{N}\sum_{i=1}^{N} \frac{\text{number of correctly recognized arguments in class } i}{\text{number of recognized arguments in } i} \tag{4.4}$$

$$\text{macro-recall} = \frac{1}{N}\sum_{i=1}^{N} \frac{\text{number of correctly recognized arguments in class } i}{\text{number of target arguments in } i} \tag{4.5}$$

$$\text{macro-F}_1 = \frac{2 \times \text{macro-prec.} \times \text{macro-recall}}{\text{macro-prec.} + \text{macro-recall}} \tag{4.6}$$

The two averaging methods bias the results differently. Micro-averaging tends to emphasize the performances on the largest classes, while macro-averaging emphasizes the performances on the smallest ones. Unless specified, precision, recall and F_1 measure refer to micro-precision, micro-recall and micro-F_1 measure, respectively.

4.3 System I

System I was originally designed to complement our prototype system, called FNDS (Financial News Dialogue System) (Lan et al., 2003), with semantic role labeling, to improve its performance in information extraction. It uses the word-by-word approach for the labeling task[2].

4.3.1 Data Set

We use the preliminary release of PropBank as our data set[3]. Inside, some of the sentences of the Penn TreeBank (Marcus et al., 1993) are labeled with verb-argument structures as propositions. We apply a procedure on the propositions to check for consistency and duplication. Totally, 53,022 propositions are compliant with our checking, with each proposition consist of about 27 tokens (words and punctation marks).

This preliminary release contains 220 semantic roles[4], with 190 of them for core arguments, 29 of them for adjunct arguments and the remaining 1 for annotation errors. They occupy 74.6%, 25.4%, 0.02% of the data respectively.

[2]This work was our earlier effort which has been presented in Lan et al. (2004)

[3]http://www.cs.rochester.edu/∼ gildea/PropBank/

[4]Every semantic role can be extended by a prepositional tag, resulting in this large number of classes.

token	**POS**	**chunk**	**roles**$_{set}$
Consumer	NN	NP	A0
stocks	NNS		
once	RB	ADVP	AM-TMP
again	RB		
set	VBD	VP	rel
the	DT	NP	A1
pace	NN		
for	IN	PP	AM-PRP
blue-chip	JJ	NP	
issues	NNS		
.	.	O	O

(a) Before BIO format

token	**POS**	**chunk**	**roles**$_{set}$
Consumer	NN	B-NP	B-A0
stocks	NNS	I-NP	I-A0
once	RB	B-ADVP	B-AM-TMP
again	RB	I-ADVP	I-AM-TMP
set	VBD	B-VP	rel
the	DT	B-NP	B-A1
pace	NN	I-NP	I-A1
for	IN	B-PP	B-AM-PRP
blue-chip	JJ	B-NP	I-AM-PRP
issues	NNS	I-NP	I-AM-PRP
.	.	O	O

(b) After BIO format

Figure 4.1: Illustration of an annotated proposition. Annotations are presented in columns, where the BIO format is applied on both labels of chunks and semantic roles. Appendix A shows the description of tags for POS and chunk. Appendix B shows the description of tags for semantic role.

To support the labeling task, two linguistic annotations are included in the data set. One annotation is part-of-speech (POS) tags, which are directly extracted from the Penn TreeBank. The other annotation is bare-phrase chunks, which are computed by a perl script contributed by Tjong Kim Sang and Buchholz (2000). To reduce the search space mapping from tokens to chunks/arguments, the BIO format (Ramshaw and Marcus, 1995) is applied to both annotations, where B (**B**egin) tag denotes the first token in a chunk/an argument, I (**I**nside) tag denotes the non-initial tokens in a chunk/an argument, and O (**O**utside) tag denotes the tokens outside any chunks/arguments. As such, every semantic role label is then categorized into a finer *positional* group (i.e., B, I, O). This thus increases the total number of classes. Totally, there are 438 role labels after employing the BIO format[5]. Figure 4.1 shows an annotated example.

[5]Theoretically, the total number of roles should be $2\times220+1=441$. However, there are some semantic roles, say `NEG`, occupying one word only. Thus, only B tag is needed but not I tag.

Feature	Description	Type
Token	Current token	nominal
Part-of-speech	Part-of-speech of the current token	nominal
Bare-phrase chunk	Syntactic category of token	nominal
Path	Chain of syntactic categories from the current token to verb	ordinal
Verb	Lemmatized verb	nominal
Voice	Active or passive voice of the verb	nominal
Position	Whether the current token is before or after the verb	nominal
Previous role	Last two semantic role predictions from classifier	nominal

Table 4.1: Features used in tagging for semantic role labeling.

4.3.2 Semantic Role Labeling as Sequential Token Classification

As discussed in Section 4.2.1, semantic role labeling involves two-sub problems: argument recognition and semantic role assignment. Here, we address them in a unified way, formulating the labeling task as a sequential tagging problem (Ramshaw and Marcus, 1995).

Argument recognition is equivalent to the typical segmentation problem, which requires finding the boundaries of arguments. By using positional tags to denote the boundaries, the segmentation problem can be resolved by token-level classification. On the other hand, semantic role assignment is also a multi-class classification of role labels. By concatenating the positional tag with the role label, both problems can simultaneously be resolved by a multi-class classifier at the token level.

For classification purpose, each outcome (i.e., positional tag and role label) is characterized by linguistic features from a fixed size of context. This is achieved by using a 5-token sliding window (Pradhan et al., 2003), scanning through the entire proposition. Each sliding collects various features, based on the feature set shown in Table 4.1. The sliding notion is illustrated in Figure 4.2. Excluding feature duplications of verb and voice, each sliding can have 29 features for characterizing an outcome.

A self-implemented classifier based on conditional Maximum Entropy (ME) (Ratnaparkhi, 1998) is used throughout the labeling process. The conditional exponential form of ME for the decision class y given an input instance x (i.e., a vector containing the 29 features) is:

$$p(y|x) = \frac{e^{\mathbf{w}\cdot\mathbf{f}(x,y)}}{\sum\limits_{y} e^{\mathbf{w}\cdot\mathbf{f}(x,y)}} \tag{4.7}$$

token	**POS**	**chunk**	**path**	**verb**	**voice**	**position**	role$_{set}$
Consumer	NN	B-NP	NN→NP→NP→ADVP→ADVP→VP→VBD	set	active	before	B-A0
stocks	NNS	I-NP	NNS→NP→ADVP→ADVP→VP→VBD	set	active	before	I-A0
once	RB	B-ADVP	RB→ADVP→ADVP→VP→VBD	set	active	before	B-AM-TMP
again	RB	I-ADVP	RB→ADVP→VP→VBD	set	active	before	I-AM-TMP
set	VBD	B-VP	-	set	active	-	rel
the	DT	B-NP	DT→NP→VP→VBD	set	active	after	B-A1
pace	NN	I-NP	NN→NP→ NP→VP→VBD	set	active	after	I-A1
for	IN	B-PP	IN→PP→ NP→NP→VP→ VBD	set	active	after	B-AM-PRP
blue-chip	JJ	B-NP	JJ→NP→ PP→NP→NP→ VP→ VBD	set	active	after	I-AM-PRP
issues	NNS	I-NP	NNS→NP→NP→PP→NP→NP→VP→VBD	set	active	after	I-AM-PRP
.	.	O	.→O→NP→NP→PP→NP→NP→VP→VBD	set	active	after	O

Figure 4.2: Illustration of the contextual features captured by a 5-token sliding window, where its center is now placed at the word `the`. Appendix A shows the description of tags for POS and chunk. Appendix B shows the description of tags for semantic role.

Here, $\mathbf{f}$ is a m-dimensional sparse vector for storing the values of the feature functions $f_j(x,y)$'s, and $\mathbf{w}$ is another m-dimensional real-valued vector for storing the values of the parameters. Thus, $\mathbf{w} \cdot \mathbf{f}$ is an inner product. Every feature function $f_j(x,y)$ is binary-valued, returning either 0 or 1, to indicate whether a feature of x co-occurs with y. For example, with reference to Figure 4.2, with the current class label being `B-A1`, a feature function for the POS tag at previous two position can be encoded as:

$$f_j(x,y) = \begin{cases} 1 & \text{if } y\text{=B-A1 and } POS_{i-2}\text{=RB} \\ 0 & \text{otherwise} \end{cases} \tag{4.8}$$

If the feature exists, its corresponding parameter will contribute weight to the probability $p(y|x)$. The parameters $\mathbf{w}$ can be found by an iterative algorithm called Generalized Iterative Scaling (GIS) (Ratnaparkhi, 1998; Darroch and Ratcliff, 1972).

The outcome prediction is determined by the trained ME classifier. During prediction, the tokens in a sentence are tagged one by one in a left-to-right manner. Given a token t, an input instance x_t is generated. The outcome y_t can be found:

$$y_t = \operatorname*{argmax}_{y \in Y} p(y|x_t) \tag{4.9}$$

Here, out of 438 outcomes, only one is locally assigned to each token. It is thus difficult to form a coherent solution from the individual outcomes. Trading off speed and search space, we formulate the coherent solution by finding the outcome sequence with the local maxima of probability. Given a proposition t_1,t_2,...,t_n, a sequence candidate

	Precision	Recall	F$_1$
1	65.42%	50.54%	57.03%
2	49.74%	64.80%	56.28%
3	64.78%	50.05%	56.47%
4	64.90%	50.00%	56.48%
5	64.75%	49.80%	56.30%
6	64.50%	49.67%	56.12%
7	64.86%	49.84%	56.37%
8	64.61%	49.73%	56.20%
9	64.38%	49.68%	56.08%
10	64.78%	49.91%	56.38%
average	63.27%	51.40%	56.37%

Table 4.2: Results of 10 folds and its average

y_i,y_2,...,y_n has the conditional probability:

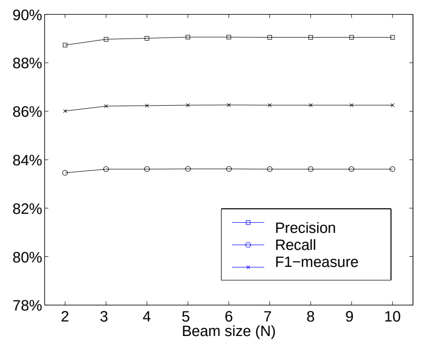
$$p(y_1, y_2, ..., y_n | t_1, t_2, ..., t_n) = \prod_{i=1}^{n} p(y_i | x_i) \tag{4.10}$$

where x_i is the input instance containing two previous predictions. Following Ratnaparkhi (1998), we employ beam search for searching the desired sequence. By parameterizing the beam size M, the complexity can be reduced from $O(N^{|Y|})$ to $O(MN|Y|)$, where N is the length of a proposition, and Y is the set of all possible outcome.

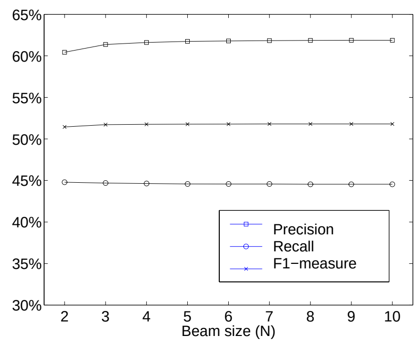
4.3.3 Experimental Evaluation

Out of 53,022 propositions, 50,000 propositions were randomly selected. They were shuffled and divided into two sets equally: 25,000 propositions for training and 25,000 propositions for testing. The training set was first used for estimating the probability distribution p, using the GIS algorithm. The number of iterations was set to 100. The beam size was set to 3. The resulting classifier was then evaluated, by measuring its performance in assigning semantic roles to unseen sentences in the test set. With 10-fold cross validation, our system achieved 63.3% precision, 51.4% recall and 56.37% F$_1$, as shown in Table 4.2. The classification performance for different semantic roles varied.

The effect of the beam size has been studied. A set of 10,000 propositions were randomly selected for training the classifier. Another set of 10,000 propositions were selected from the remaining 40,000 propositions for testing. Classification performance



(a) Training sentences



(b) Testing sentences

Figure 4.3: Effect of the beam size on the performance of the semantic role classifier.

was evaluated using both training and testing set. The results in Figure 4.3 indicate that the beam size has no significant impact on performance.

To further evaluate our system, we compare its performance with other approaches. The results are shown in Table 4.3. Compared with the two concurrent work in (Baldewein et al., 2004; Lim et al., 2004) reported in the CoNLL-2004 shared task that also used the maximum entropy approach, our result is slightly worse than theirs. We argue that it is mainly due to the larger number of classes involved. In our work, there are totally 438 classes, in contrast with 75 classes in their work. Hence, our classification task is inherently more difficult. Moreover, they employed additional features, such as clauses and named entities, which were absent from our data set. This definitely reduces our classification performance. Also, they adopted the chunk-by-chunk approach, whereby bare-phrase chunks instead of words were labeled. In our approach, each word is separately labeled and a chunk is correctly labeled only if all its words are correctly assigned to semantic role classes. Definitely, this makes our task more difficult.

We are also interested in studying the training time of the ME classifier for our experiments. For using 10,000 propositions, our system required about 39 hours in training[6]. The training time increased to around 109 hours as the number of propositions is increased from 10,000 to 25,000.

[6]We ran the experiments on 4 $\times$ 900Mhz UltraSPARC processors. We used a Java implementation.

System	Data set	Method	Granularity	Precision / Recall
Our system	Replicated PropBank	Maximum Entropy	word-by-word	63.3% / 51.4%
Gildea and Palmer (2002) System II	PropBank (Dec. 2001)	Linear Interpolation	constituent-by-constituent	49.5% / 35.5%
Pradhan et al. (2003) W-by-W Chunker-II	PropBank (Jul. 2002)	Support Vector Machine	word-by-word	66.2% / 54.9%
Baldewein et al. (2004) system	CoNLL-2004 PropBank (Feb. 2004)	Maximum Entropy	chunk-by-chunk	65.7% / 42.6%
Lim et al. (2004) system	CoNLL-2004 PropBank (Feb. 2004)	Maximum Entropy	chunk-by-chunk	68.4% / 61.5%

Table 4.3: Performance comparison with other approaches

4.3.4 Lessons Learned

We have formulated semantic role labeling as a sequential tagging problem, and resolved it by the word-by-word approach using a ME classifier. The features of the classifier are collected by a 5-token sliding window. Its disadvantage is its reliance on a small fixed context, resulting in its inability to resolve long-range tagging dependence and ambiguities. Moreover, the use of positional tags increases the number of classes of the ME classifier. This makes the classification task inherently harder. Therefore, evaluation reveals that it has only satisfactory performance on the preliminary release of PropBank.

4.4 System II

There are four motivations for this work. Firstly, a new labeling approach is designed, targeting at improving the labeling performance by addressing the weaknesses of the first system. Secondly, a cleaner and larger PropBank is formally released, allowing us to evaluate its scalability on our ME classifier. Thirdly, full syntactic information is employed, enriching the original feature sets. Finally, a sampling technique and a learning mechanism is proposed, trying to resolve the class-imbalance issue.

4.4.1 Data Set

The CoNLL-2005 shared task freely provided the data set to the public. We use it for training and testing our classifier, although we did not actually participate in the shared task. The data set consists of several sections from the Wall Street Journal (WSJ)

Data	sections	#sents.	#tokens	#props.	#unique verbs	#args.
WSJ Train	02-21	39,832	950,028	90,750	3,101	239,858
WSJ Devel.	24	1,346	32,853	3,248	860	8,346
WSJ Test	23	2,416	56,684	5,267	982	14,077

Table 4.4: Counts on CoNLL data set

part of the Penn TreeBank (Marcus et al., 1993), together with their verb-argument structures from the released PropBank (Kingsbury and Palmer, 2002). Sections 2 through 21 are used as the training set, section 24 is used as the development set, and section 23 is used as the test set.

Table 4.4 summarizes the number of sentences, tokens (words and punctation marks), propositions, unique verbs and arguments in the data set. With respect to the WSJ training set, each sentence consists of about 23 tokens and 2 propositions. Each proposition contains about 2 arguments.

In the data set, there are totally 35 semantic roles, including 7 roles for core arguments, 14 roles for adjunct arguments and 14 roles for referential arguments (see Appendix A for definition). They occupy 71.7%, 25.0% and 3.3% of data, respectively.

The data set contains a number of linguistic annotations to support the labeling task. These annotations include part-of-speech (POS) tags, base-phrase chunks, embedding clauses, two full syntactic trees (Collins, 1999; Charniak, 2000), and named entities (NE). In our experiments, we choose Charniak's as our main syntactic information. Figure 4.4 shows an example of a sentence annotated with the START-END format, which was previously defined in the CoNLL-2001 shared task (Tjong Kim Sang and Déjean, 2001).

4.4.2 Three-Phase Labeling Approach

A three-phase approach is proposed to address the labeling task in which two problems are involved: argument recognition and semantic role assignment. Argument recognition is further decomposed into two sub-problems, namely, argument-start recognition and argument-end recognition. Thus, there are totally three problems and each of them is addressed in a separate phase. The classifiers involved in the first and third phase are built by the Maximum Entropy (ME) approach (Ratnaparkhi, 1998).

A pre-processing phase is first done on the original data set to collapse tokens into

token	**POS**	**chunk**	**clause**	**full parse**$_{Charniak}$	**NE**	**verb**	**role**$_{give}$	**role**$_{sell}$
The	DT	(NP*	(S*	(S1(S(NP*	*	-	(A0*	(A0*
state	NN	*)	*	*)	*	-	*)	*)
gave	VBD	(VP*)	*	(VP*	*	give	(V*)	*
CenTrust	NNP	*	*	(NP*)	(ORG*)	-	(A2*)	*
30	CD	(NP*	*	(NP(NP*	*	-	(A1*	(AM-TMP*
days	NNS	*)	*	*)	*	-	*	*)
to	TO	(VP*	(S*	(SBAR(S(VP*	*	-	*	*
sell	VB	*)	*	(VP*	*	sell	*	(V*)
the	DT	(NP*	*	(NP*	*	-	*	(A1*
Rubens	NNP	*)	*)	*)))))))	(PER*)	-	*)	*)
.	.	*	*)	*))	*	-	*	*

Figure 4.4: Example of annotated sentence in CoNLL-2005 shared task. The sentence contains two verbs: *give* and *sell*. Each line corresponds to a token of the sentence and its associated annotations. Six columns of linguistic elements (part-of-speech (POS), chunk, clause, full parse (by Charniak's parser (Charniak, 2000)), NE, role$_{give}$ and role$_{sell}$) are annotated with the START-END format. The tag (X*, *), and * denote that a token starts, ends, and neither starts nor ends at an linguistic element of type X, respectively. The first two tags can be used in combination with each other. For example, the tag (S1(S(NP* indicates a token which starts a noun phrase and two clauses; the tag (A2*) marks a token where an A2 argument starts and ends, forming a complete argument. Appendix A shows the description of tags for POS, chunk, and clause. Appendix B shows the description of argument tag.

token	**POS**	**chunk**	**clause**	**full parse**$_{Charniak}$	**NE**	**verb**	**role**$_{give}$	**role**$_{sell}$
The state	DT NN	(NP**)	(S**	(S1(S(NP**)	* *	-	(A0**)	(A0**)
gave	VBD	(VP*)		(VP*	*	give	(V*)	*
CenTrust	NNP	*	*	(NP*)	(ORG*)	-	(A2*)	*
30 days	CD NNS	(NP**)	**	(NP(NP**	* *	-	(A1**	(AM-TMP**)
to	TO	(VP*	(S*	(SBAR(S(VP*	*	-	*	*
sell	VB	*)	*	(VP*	*	sell	*	(V*)
the Rubens	DT NNP	(NP**)	**)	(NP**)))))))	* (PER*)	-	**)	(A1**)
.	.	*	*)	*))	*	-	*	*

Figure 4.5: Chunked data format. Tokens are collapsed into base-phrase chunks.

bare-phrase chunks, to be used as the input of the first phase. Figure 4.5 shows the new representation. In case a chunk cannot be aligned with an argument, the chunk is split back into separate tokens. For example, a phrasal-verb chunk is split into a verb and its particle.

In the first phase, argument-start recognition is formulated as a binary classification, in which each chunk/token in a proposition is classified into either the start of an argument or not a start of any argument, i.e., a non-start. This binary decision is determined by a trained ME classifier. The features used in this phase are outlined in Table 4.5. The features are divided into four groups, in which three of them, abbreviated as G, P and X respectively, were defined in (Gildea and Jurafsky, 2000; Pradhan et al., 2004; Xue and Palmer, 2004). These features have been proved useful to improving the classification performance. An additional group of features, abbreviated as A, are also proposed and shown at the bottom of the table.

In the second phase, the end of an argument is identified by traversing the parse tree beginning at the argument start where its terminal node in parse tree is climbed, to the lowest ancestor whose subtree covers a complete syntactic constituent. The traversal is subject to two constraints:

1. One cannot proceed beyond the start of the next argument.

2. One cannot proceed beyond the target verb.

Otherwise, the traversal backtracks to the original node. The word corresponding to that node is then the argument end. A complete argument can be formed by combining a start and its corresponding end. Figure 4.6 illustrates the traversal method and the formation of arguments.

In the third phase, semantic role assignment is formulated as a multi-class classification, in which each argument is assigned a semantic role. A multi-class ME classifier is employed. The features used are the same as those in the first phase, but they are now applied to *arguments* instead of chunks. Four extra features introduced by (Gildea and Jurafsky, 2000; Pradhan et al., 2004; Xue and Palmer, 2004) are also used in this phase. They are bolded in Table 4.6. Also, another group of additional features, similarly abbreviated as A, are proposed, and they are shown at the bottom of the table.

Feature		Description	Type
(G1)	Verb	Lemmatized verb	nominal
(G2)	Phrase type	Syntactic category of chunk	nominal
(G3)	Path	Path from chunk to verb	ordinal
(G4)	Head	Head word of chunk	nominal
(G5)	Voice	Active or passive voice of verb	nominal
(G6)	Subcategorization	Parse tree nodes immediately expanded from verb	nominal
(P1)	Head POS	Part-of-speech of chunk's head	nominal
(P2)	PP head/POS	If chunk's parent is a Prepositional Phrase, return its head word and part-of-speech of that head word	nominal
(P3)	First word/POS	First word of chunk and its part-of-speech	nominal
(P4)	Last word/POS	Last word of chunk and its part-of-speech	nominal
(P5)	Parent type/head/POS	Syntactic category, head word and part-of-speech of chunk's parent	nominal
(P6)	Left-sibling type/head/POS	Syntactic category, head word and part-of-speech of chunk's left sibling	nominal
(P7)	Right-sibling type/head/POS	Syntactic category, head word and part-of-speech of chunk's right sibling	nominal
(P8)	Horizontal distance	Horizontal position of chunk from verb	numerical
(P9)	Vertical distance	Vertical position of chunk from verb in parse tree	numerical
(P10)	Partial path	Path from chunk to lowest common ancestor of position from verb	ordinal
(X1)	Verb& Phrase type	Conjunction of two features	nominal
(X2)	Verb & Path	Conjunction of two features	nominal
(X3)	Verb & Head	Conjunction of two features	nominal
(X4)	Verb & PP head	Conjunction of two features	nominal
(A1)	Verb POS	Part-of-speech of verb	nominal
(A2)	Verb phrase type	Syntactic category of verb	nominal
(A3)	Partial path variant	Add syntactic nodes in partial path with their depths relative to verb	ordinal
(A4)	Path variant I	A chain of clause nodes with their depths relative to verb, concatenated with the depth, the horizontal position and the clause level of the chunk	ordinal
(A5)	Path variant II	A chain of clause nodes with their depths relative to verb, concatenated with the depth and the horizontal position of the chunk	ordinal
(A6)	Path variant III	A chain of clause nodes with their depths relative to verb, concatenated with a chain of chunk types	ordinal
(A7)	Path variant IV	A chain of commas and coordinating conjunctions	ordinal
(A8)	Path variant V	Path concatenated with head word suffixes of length 2, 3 and 4	ordinal
(A9)	Previous chunk	Whether previous chunk is the beginning of sentence, the verb, the coordinating conjunction, the comma or the parenthesis	nominal
(A10)	Next chunk	Whether next chunk is the beginning of sentence, the verb, the coordinating conjunction, the comma or the parenthesis	nominal
(A11)	NEG	Active if chunk is `n't` or `not`	nominal
(A12)	MOD	Active if chunk is a model verb	nominal
(A13)	Path context	Path of chunk within a within a window size of -2/+2	

Table 4.5: Features used in argument-start identification

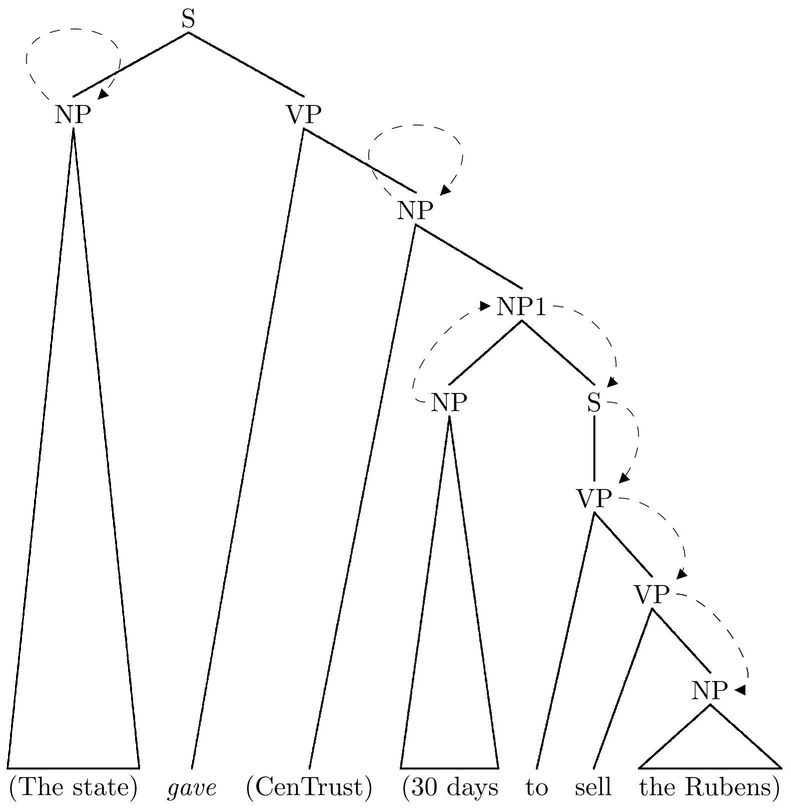


Figure 4.6: Illustration of finding the argument ends by traversing the parse tree. The traversal for the argument start `The state` cannot proceed beyond the verb ***gave*** due to the first constraint. Hence, it backtracks to the original node. Similarly, the proceeding of the argument start `CenTrust` is restricted by the next start at `30 days` due to the second constraint. Thus, it returns to the original node. On the other hand, the traversal for the argument start `30 days` succeeds to reach the chunk `the Rubens`, finding the argument end heuristically, through the path `NP↑NP1↓S↓VP↓VP↓NP`. Note that `NP1` is such the lowest ancestor which can constitute a correct argument. Appendix A shows the description of phrase tags.

Feature		Description	Type
(G1)	Verb	Lemmatized verb	nominal
(G2)	Phrase type	Syntactic category of argument	nominal
(G3)	Path	Path from argument to verb	ordinal
(G4)	Head	Head word of argument	nominal
(G5)	Voice	Active or passive voice of verb	nominal
(G6)	Subcategorization	Parse tree nodes immediately expanded from verb	nominal
(G7)	**Position**	Whether argument is before or after verb	ordinal
(P1)	Head POS	Part-of-speech of argument's head	nominal
(P2)	PP head/POS	If argument's parent is a Prepositional Phrase, return its head word and part-of-speech of that head word	nominal
(P3)	First word/POS	First word of argument and its part-of-speech	nominal
(P4)	Last word/POS	Last word of argument and its part-of-speech	nominal
(P5)	Parent type/head/POS	Syntactic category, head word and part-of-speech of argument's parent	nominal
(P6)	Left-sibling type/head/POS	Syntactic category, head word and part-of-speech of argument's left sibling	nominal
(P7)	Right-sibling type/head/POS	Syntactic category, head word and part-of-speech of argument's right sibling	nominal
(P8)	Horizontal distance	Horizontal position of argument from verb	numerical
(P9)	Vertical distance	Vertical position of argument from verb in parse tree	numerical
(P10)	Partial path	Path from argument to lowest common ancestor of position from verb	ordinal
(P11)	**Named entities**	Whether argument contains named entities	nominal
(P22)	**Temporal cue words**	Whether argument contains temporal words that are defined in a lexicon	nominal
(X1)	Verb & Phrase type	Conjunction of two features	nominal
(X2)	Verb & Path	Conjunction of two features	nominal
(X3)	Verb & Head	Conjunction of two features	nominal
(X4)	Verb & PP head	Conjunction of two features	nominal
(X5)	**Verb & Position**	Conjunction of two features	ordinal
(A1)	Verb POS	Part-of-speech of verb	nominal
(A2)	Verb phrase type	Syntactic category of verb	nominal
(A3)	Syntactic pattern	Chunk nodes within the argument chained with their depth	ordinal
(A4)	Head suffixes	Argument head suffixes of length 2, 3 and 4	nominal
(A5)	Head & Position	Conjunction of two features	ordinal
(A6)	Head & Voice	Conjunction of two features	nominal
(A7)	Head POS & Position	Conjunction of two features	ordinal
(A8)	Head POS & Voice	Conjunction of two features	nominal
(A9)	Phrase type & Position	Conjunction of two features	ordinal
(A10)	Phrase type & Voice	Conjunction of two features	nominal
(A11)	PP head & Position	Conjunction of two features	ordinal
(A12)	PP head & Voice	Conjunction of two features	nominal
(A14)	Word next to argument	Whether word next to argument is the beginning of sentence, the verb, `n't`, `not` or model verb	nominal
(A15)	Word previous to argument	Whether word previous to argument is the beginning of sentence, the verb, `n't`, `not` or model verb	nominal

Table 4.6: Features used in semantic role assignment. Bolded features are only used here instead of argument-start identification.

The three-phase labeling approach with the new representation has gained advantages from the word-by-word approach, chunk-by-chunk approach, and constituent-by-constituent approach:

- Token is used to avoid the alignment problem suffered by the constituent-by-constituent approach.
- Chunks are being classified instead of tokens in most of the time. The number of classifications can be reduced since chunks generally span a larger segment of a sentence.
- Constituent can be generated by traversal heuristic. Long-ranged argument can also be recovered.

4.4.3 Experimental Evaluation

We did not participate the CoNLL2005 shared task, but we use the their date set for evaluation. Following the same settings of the shared task, sections 2 through 21 were used for training, section 24 for development and section 23 for testing. The number of iterations for GIS algorithm was set to 100. A simple feature-selection technique was used, in which a feature that occurred less than 5 times was discarded (as a rule-of-thumb in this experiment). The evaluations were performed using the perl script officially provided by the CoNLL-2005 shared task.

In our experiment, each phase was individually evaluated, and its output was piped into the next phase as input. Table 4.7 shows the individual result for each phase. After processing three phases, our system achieved 73.50 F_1 on the development set and 75.43 F_1 on the test set.

To study the upper bound performance of the labeling task, we evaluated its classification accuracy on semantic role assignment. Accuracy is defined as the percentage of total recognitions that are correct. During evaluation, the correct argument boundaries were given, without employing the predications from the first two phases. The accuracy was 88.32% on the development set and 89.27% on the test set. The results revealed that incorrect argument identifications were the key bottleneck of the labeling performance.

Experiments have also been performed to evaluate the impact of our proposed features on performance, which are described in Section 4.4.2. As shown in Table 4.8,

development set	precision	recall	F_1
argument-start identification	91.52	87.54	89.49
argument-end identification	83.46	79.68	81.52
semantic role assignment	75.12	71.95	73.50

test set	precision	recall	F_1
argument-start identification	92.62	88.72	90.63
argument-end identification	84.47	80.79	82.59
semantic role assignment	76.93	73.99	75.43

Table 4.7: The result for each phase on the development and test set.

F_1 was consistently improved by using the proposed features, yielding 1.49 and 1.37 points of improvements on the development set and test set, respectively.

Our system was also compared with those systems participated in the CoNLL-2005 shared task (Carreras and Màrquez, 2005). Based on the F_1 measure on the WSJ test set, our system ranked the 9th of 20 systems, with the best system obtaining 79.44 F_1 and the worst one obtaining 66.73 F_1. Note that the first four (Punyakanok et al., 2005a; Haghighi et al., 2005; Màrquez et al., 2005; Pradhan et al., 2005) and the 7th system (Tsai et al. (2005)) are those combining individual systems, which are definitely better than the those using a single system. It is because combining systems reduces the variance of classifiers. The best single system approach, ranked the 5th (Surdeanu and Turmo, 2005), achieved 76.46 F_1, which was only around 1 points higher than ours. Their key advantage over ours is its enhanced constituent-by-constituent approach, where a post-processing step is designed for resolving the one-to-many mapping problem of arguments to syntactic constituents. Similarly, the 6th system (Liu et al., 2005) also pro-processed its results, but focused on the arguments which had no matching constituents. Both issues, however, are not addressed in our system. On the other hand, the 8th system (Moschitti et al., 2005) employed a SVM classifier with a tree kernel for structural feature processing, which is certainly better than our linear feature representation method.

4.4.4 Time Complexity Evaluation

We attempt to compare our training time with other CoNLL-2005 systems. Since the shared task focused on evaluating the recognition performance, most participants did

development set	precision	recall	F_1
G + P + X + A	75.12	71.95	73.50
G + P + X	73.67	70.42	72.01
difference	+1.45	+1.53	+1.49

test set	precision	recall	F_1
G + P + X + A	76.93	73.99	75.43
G + P + X	75.45	72.71	74.06
difference	+1.48	+1.28	+1.37

Table 4.8: Improvements of using our proposed features on the development and test set.

not publish their training time results. As a general rule, we can compare the time complexity instead. Since SVM classifiers were widely used among the top performance systems, it is used as a reference here. In general, SVM training involves quadratic programming to solve its constrained optimization problem. Thus, it requires $O(N^3)$ time complexity, where N is the size of the training set. However, the GIS algorithm employed in ME training only requires $O(|N||\mathcal{Y}|A)$ time complexity, where $|N|$ is the number of training examples, $|\mathcal{Y}|$ is the set of classes, and A is the average number of feature functions for an example (see Section 3.9 for details). Clearly, the complexity of the GIS algorithm is much lower than that of SVM. Thus, we believe that our system can be trained faster than SVM.

Moreover, many participants use the combined system approach where multiple labelings are generated using different learning models and/or alternative parses of the sentence, which are then combined (e.g., by means voting) to give the final prediction (Carreras and Màrquez, 2005). To further improve performance, various global optimization and post-processing techniques are often applied as well. Such approaches/techniques can lead to higher recognition rate, but it is achieved at the expense of longer training time and/or labeling time. In contrast, our three-phase approach adopts the *single* system approach without involving any post-processing steps, so it can be trained more efficiently than other systems. Yet, it outperforms a lot of other approaches that apply more complicated techniques.

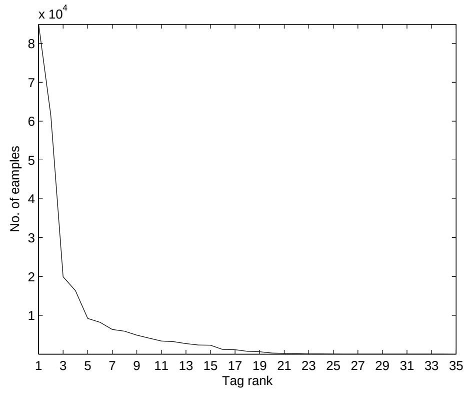


Figure 4.7: Distribution of examples among the semantic roles in the data set used in one training run. The semantic role labels are ranked according to their frequencies of occurrence.

4.4.5 Improving Macro-average Performance

So far, all performance measures were calculated by micro-average methods. Generally, micro-average measures give equal weight to each individual test case. Thus, they are usually dominated by the most frequent classes. In contrast, macro-average measures assign equal weight to each class. Thus, they are better indicators to show the classification performance across all classes. For a more *comprehensive* evaluation, performance here is additionally evaluated by macro-average methods. Thus, there are totally six measures, which include micro-precision, micro-recall, micro-F_1, macro-precision, macro-recall, and macro-F_1.

As shown in Figure 4.7, the distribution of examples among the semantic roles in the data set is heavily skewed (in the figure, the higher the rank of a semantic role, the lower its frequency of occurrence in the data set). We addressed this class-imbalance problem by two methods, one is over-sampling and the other is error-based learning. During evaluation, the first two phases of our proposed approach remained the same. We only applied these two methods on the third phase, which is a multi-class classification.

In the over-sampling method, we replicate examples by randomly selecting a number of examples from each semantic role type. For a semantic role type s (where s is from 1 to $|S|$, $|S|$ being the number of semantic role types), the number of examples replicated,

test set	micro-prec.	micro-recall	micro-F_1	macro-prec.	macro-recall	macro-F_1
sampling	76.93	73.93	75.40	54.51	48.83	51.15
original	76.93	73.99	75.43	58.52	48.67	52.48
difference	0.00	-0.06	-0.03	-4.01	**+0.16**	-1.33

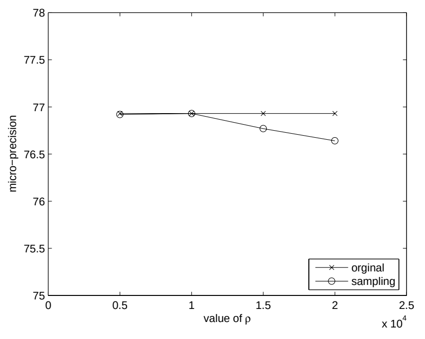
Table 4.9: Results using our sampling technique (ρ=10,000) on the test set.

n_s, is defined as:

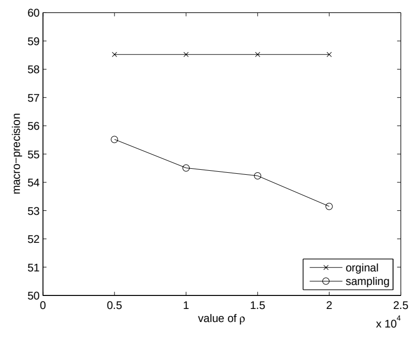
$$n_s = \frac{(N_s - N_{|S|})}{(N_1 - N_{|S|})} \times (N_1 - \rho) + \rho - N_s \tag{4.11}$$

Here, N_s is the number of examples of the semantic role type s in the original training set. The degree of balance is controlled by the parameter $\rho > 0$. In general, the larger the value of ρ, the more balanced distribution. In our experiment, the parameter ρ was set to 10,000. Table 4.9 shows the results. As depicted, only macro-recall was insignificantly improved. Others were worsened. In general, example replication can increase macro-average measure. However, meanwhile, it decreases micro-average measure as well. In our case, the lack of improvement for macro-average measure was probably due to two reasons: (a) the replication was still not enough or (b) the learning algorithm overfitted with the replicated data, particularly the rare features whose occurrences in the data set were due to replication. A simple verification for the first reason is to graph the number of example replications as a function of performance measures. As shown in Figure 4.8 (in the figure, the larger the ρ value, the more the replications), nearly all the measures were consistently decreased for all ρ values from 5,000 to 20,000. The results implied that the replication number insignificantly affect our over-sampling method. Thus, the first proposed reason was untenable. On the other hand, the second reason was verified by graphing the number of feature cutoffs as a function of performance measures. The smaller the value of cutoffs, the larger the number of rare features. As shown in Figure 4.9, for almost all of the measures, the over-sampling method hardly yield improvements at any cutoff values from 1 to 10. The results implied that the second reason was also untenable, that the number of rare features is an uncritical control factor to our over-sampling method. Further exploratory study is required to find out the reason why our over-sampling was ineffective in improving the macro-average performance in this task.

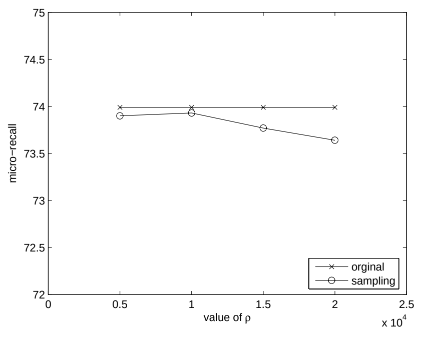
In the error-based learning method, we aim to reduce the number of errors introduced by the minority class. We modify the GIS algorithm by adding a parameter r_y



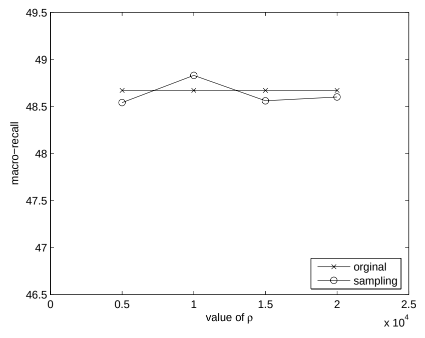
(1a) micro-precision



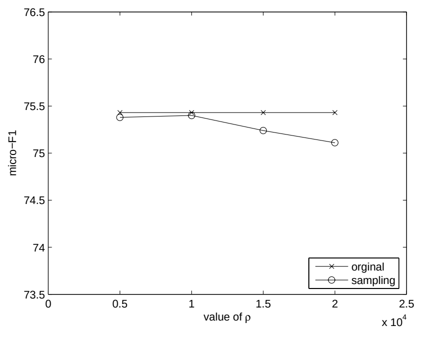
(2a) macro-precision



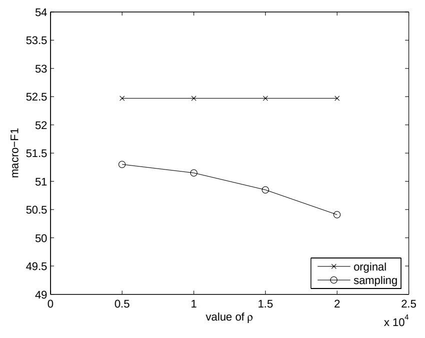
(1b) micro-recall



(2b) macro-recall

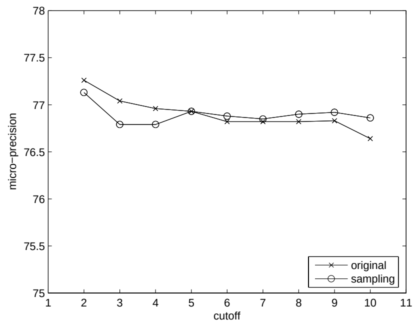


(1c) micro-F_1

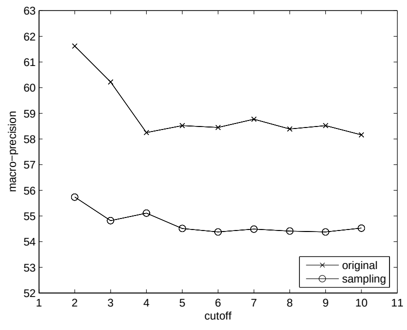


(2c) macro-F_1

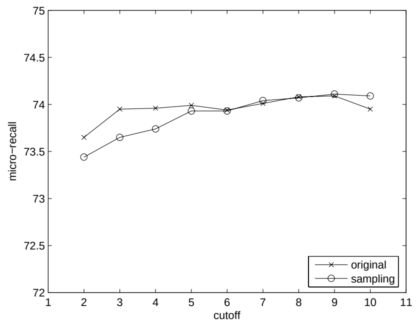
Figure 4.8: Effect of the number of replications on the micro-average and macro-average performance



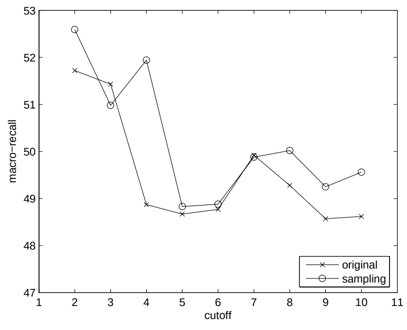
(1a) micro-precision



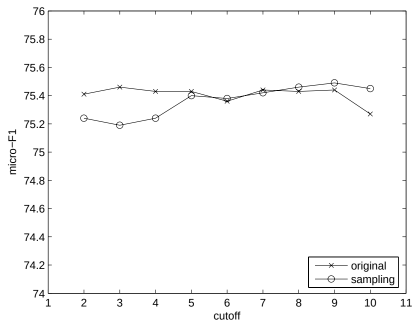
(2a) macro-precision



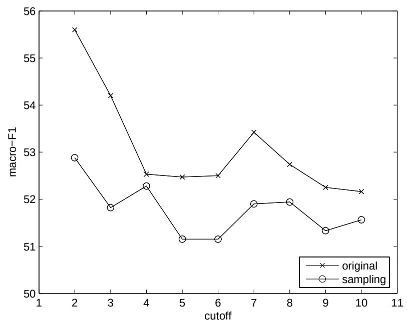
(1b) micro-recall



(2b) macro-recall



(1c) micro-F_1



(2c) macro-F_1

Figure 4.9: Effect of the number of cutoffs on the micro-average and macro-average performance

test set	micro-prec.	micro-recall	micro-F_1	macro-prec.	macro-recall	macro-F_1
m.s. learning	76.95	74.00	75.44	60.57	50.39	54.31
original	76.93	73.99	75.43	58.52	48.67	52.48
difference	+0.02	+0.01	+0.01	+2.05	+1.72	+1.83

Table 4.10: Results using our error-based learning method on the test set.

to the original updating rule. In an iteration t, the modified updating rule for each feature estimates its associated weight w as follows:

$$w_t \leftarrow w_{t-1} + r_y \cdot \frac{1}{C} \log \frac{\tilde{c}}{c} \tag{4.12}$$

Here, w is estimated by the divergence between the observed feature count $\tilde{c}$ and the modeled feature count c, with C being a constant. In each iteration, each training example is predicted by the classifier. The number of correct predictions for each class y is recorded, for generating the accuracy distribution among classes. To achieve better prediction in next iteration, larger r_y is assigned to the features whose class y (probably minority class) is predicted poorly. Conversely, smaller r_y is assigned to the features whose class y (probably majority classes) is predicted correctly. The value of r_y is determined by an inverse function about the accuracy of a class y. Here, whenever a new iteration begins, each value of r_y is re-computed too. We used the macro-averaging performance by our proposed mistake-sensitive mechanism. Table 4.10 shows six results, using the new updating rule. All macro-averaging performance were improved, particularly yielding a significant improvement of 1.83 points in macro-F_1.

To take into account of both micro-F_1 (F_{mic}) and macro-F_1 (F_{mac}), we propose a new measure, namely FF_1, defined as $FF_1 = 2F_{mic}F_{mac}/(F_{mic} + F_{mac})$. The notion of our formula is similar to original F_1 measure. It takes the harmonic mean between micro-average performance and macro-average performance. We calculated macro-F_1 and FF_1 for the CoNLL-2005 systems. All the systems were then re-ranked, based on FF_1. As shown in Table 4.11, our system ranks the 4th, achieving 63.16 FF_1. Note that under this new ranking, the 1st and the 3rd system (Haghighi et al., 2005; Tjong Kim Sang et al., 2005) were again the combined systems. On the other hand, the 2nd system (Moschitti et al., 2005) was the one ranked the 8th before, which employed a SVM classifier to the task.

rank		micro-F_1	macro-F_1	FF_1
1	Haghighi et al. (2005)	78.45	56.15	**65.45**
2	Moschitti et al. (2005)	75.89	**56.79**	64.96
3	Tjong Kim Sang et al. (2005)	75.37	55.51	63.93
4	Our system	75.44	54.31	63.16
5	Liu et al. (2005)	76.44	52.41	62.18
6	Ozgencil and McCracken (2005)	74.44	52.66	61.68
7	Tsai et al. (2005)	76.38	51.69	61.66
8	Pradhan et al. (2005)	77.37	50.91	61.41
9	Yi and Palmer (2005)	75.17	51.21	60.90
10	Punyakanok et al. (2005a)	**79.44**	48.63	60.33
11	Johansson and Nugues (2005)	74.30	50.20	59.92
12	Surdeanu and Turmo (2005)	76.46	48.39	59.27
13	Park and Rim (2005)	72.68	49.53	58.91
14	Màrquez et al. (2005)	77.97	45.90	57.78
15	Mitsumori et al. (2005)	71.08	45.32	55.35
16	Bharati et al. (2005)	69.40	44.65	54.34
17	Cohn and Blunsom (2005)	73.10	41.93	53.29
18	Ponzetto and Strube (2005)	69.56	33.13	44.88
19	Lin and Smith (2005)	67.91	33.05	44.46
20	Sutton and McCallum (2005)	66.73	35.19	46.08

Table 4.11: 20 system performance are sorted by FF_1 on the WSJ test set. The micro-F_1 performance is taken from the result of CoNLL-2005 shared task (Carreras and Màrquez, 2005), whereas the macro-F_1 is calculated by ourselves, based on the output files provided by the CoNLL-2005 homepage (http://www.lsi.upc.es/∼srlconll/st05/st05.html).

4.5 Summary

In view of lesson learnt from our previous work using the word-by-word approach, we proposed the three-phase labeling approach to the semantic role labeling task. Our approach gains advantages from previous approaches while addressing their weaknesses. Our results were comparable with the current state-of-art individual system, as based on large PropBank data set. We also applied two different methods to this task to evaluate the performance in terms of macro-averaging. Evaluations with our over-sampling method showed that only macro-recall was improved, but to an insignificant extent. Investigations discovered that the lack of improvement was not related to the number of replications nor the number of cutoffs. Experiments were also conducted using our error-based learning method. Results showed that all macro-averaging performance were significantly improved. In terms of our proposed FF_1 measure, our system ranked 4th, when compared with 19 other systems, which is even better than some approaches that employ combined systems.

Chapter 5

Dialogue Act Recognition

5.1 Introduction

Dialogue Act (DA) is the concise abstraction of an utterance's intention, which serves a similar function as illocutionary force (Austin, 1962) and speech act (Searle, 1969). Examples of DAs include Greeting, Suggestion, Appreciation, Acknowledgment, etc. Recognizing DA is useful in many applications. For example, it helps dialogue systems in generating appropriate response to users (Allen et al., 1996; Reithinger et al., 1996) and helps speech recognizer in resolving speech ambiguities (Stolcke et al., 2000).

DA recognition is challenging. Often, DA cannot be directly inferred from the literal context. Consider the utterance *My soup is cold.* Literally, it looks like a simple statement of fact. However, its underlying DA should be recognized as Indirect Request for a replacement when situational context is also considered. Definitely, accurate recognition requires comprehensive and in-depth analysis in linguistics, where diverse pieces of evidence have to be considered carefully.

Maximum Entropy (ME) (Jaynes, 1957) is such a statistical approach, allowing heterogeneous evidence to be integrated into a probability model of decisions. The ME approach has previously been applied to various natural language processing and information retrieval problems such as part-of-speech tagging (Ratnaparkhi, 1998) and text classification (Kazama and Tsujii, 2005).

This chapter presents a maximum entropy approach to train a classifier for *non-*

task oriented DA recognition. The classifier is defined by estimating the probability distribution $p(d|h)$. Its aim is to assign an utterance u to a dialogue act d, based on the "history" h (or context) of the utterance u. The probability distribution found can then be used for labeling unseen utterances. Given a history h' of an utterance u', $p(d|h')$ is estimated for each $d \in \mathcal{D}$ ($\mathcal{D}$ being the set of all possible d's), and u' is assigned to $d' = \underset{d \in \mathcal{D}}{\operatorname{argmax}}\, p(d|h')$.

We use the Switchboard corpus to evaluate the performance. Despite the high overall recognition rate achieved, the individual recognition rates of some DAs are unsatisfactory. This is caused by the heavily skewed distribution of examples among the DAs in the data set, that some DAs have extremely low occurrence frequencies. Due to the lack of examples in the training set, the ME classifier is weak in recognizing some of these DAs. Previously, various approaches have been put forth to tackle this data sparseness problem (Chen and Rosenfeld, 2000; Kazama and Tsujii, 2005). These approaches were mostly *model-oriented*, as they involved modification of the ME formulation to reduce the bias of the model toward the observed data. In this chapter, we apply the error-based learning method, as discussed in Section , to tackle the problem. Besides, a *data-oriented* method (see Section) is applied to replicate examples for the DAs. A DA with lower occurrence frequency will have more examples replicated for it. The distribution of examples among the DAs thus becomes more balanced. Preliminary experiments reveal that both methods are effective in improving the recognition performances of those DA types that have very low occurrence frequencies in the original data set.

5.2 Data Set

Our data set is based on Switchboard (Godfrey et al., 1992), which is a collection of spontaneous telephone conversations. The collection contains 2,438 dialogues spanning 70 different topics, which are fully transcribed with annotations of speaker turns, disfluencies, filled pauses, and discourse markers. The whole transcription contains about 3 million words, hand-segmented into utterances. Each dialogue is participated by two speakers, who take turns to speak. In each turn, one or more utterances are produced, each containing one or more words. A total of 1,155 dialogues have been annotated using the DAMSL tag set (Dialogue Act Markup in Several Layers) (Core and Allen, 1997), where each utterance is categorized into one of 220 DAs. The annotated corpus is called SWBD. DAMSL has later been collapsed into SWBD-DAMSL, which contains only 42 DAs (Jurafsky et al., 1997; Stolcke et al., 2000). We collapse the tag set in the same way done by Stolcke et al. (2000). Our resulting tag set contains 44 tags [1], which has 2 more tags than that in Stolcke et al. (2000). One of them is the SEGMENT DA of DAMSL, which has a high occurrence frequency in SWBD ($\approx$8%) but was ignored in the experiments by Stolcke et al. (2000). Also, we distinguish between ABANDONED and UNINTERPRETABLE, which were combined into one DA by Stolcke et al. (2000).

We enrich SWBD with part-of-speech (POS) tags, by combining SWBD with the tagged Switchboard which is a part of Phase III of the Penn Treebank corpus (Marcus et al., 1993). Figure 5.2 shows an example. The segmentation of utterances in SWBD differs from the one in TreeBank. There are 1,706 differences between them. We follow the segmentation of TreeBank. The resulting data set contains 1,125 dialogues, which involve 112,881 turns and 216,292 utterances. A dialogue has 16 to 379 turns, with the average number of turns per dialogue being equal to 109.2. The length of a turn varies from 1 to 30 utterances. On average, however, each turn contains 1.8 utterances only, so most of the turns are short. An utterance consists of 1 to 102 words. The average number of words per utterance is 9.7. In general, the shorter the utterance, the larger the number of possible DAs. Intuitively, short utterances are more ambiguous as they contain only a few words, thus providing limited clues for the classifier to determine its DA.

5.3 Features for Dialogue Act Recognition

Various pieces of information can be useful to DA recognition. Here, we study and evaluate seven types of features, as summarized in Table 5.1.

5.3.1 Lexical Features

A lexical feature, also known as dialogue act cue (Samuel et al., 1998), is a sequence of words that frequently appears in certain DA(s), such as *how about* and *thank you.* For example, an utterance containing *how about* is likely a BACKCHANNEL QUESTION. In

[1] We used the same data set as in Stolcke et al. (2000), who reported 42 tags in their journal. However, after pre-processing, we found 44 tags in the data set.

DA	Speaker Turn	Utt. No.	Utterance
h	A_117	utt1:	I/PRP do/VBP n't/RB know/VB ./. E_S
bh	A_117	utt2:	What/WP can/MD we/PRP do/VB about/IN it/PRP ?/. E_S
ba	B_118	utt1:	[Goo-/JJ ,/, + good/JJ] question/NN <laughter> ./. E_S
x	A_119	utt1:	<Laughter> ./.
%	B_120	utt1:	Probably/RB ,/, E_S
sv	A_121	utt1:	[Money/NN ,/, +
%	B_122	utt1:	Taking/VBG ,/, E_S
+	A_123	utt1:	Money/NN] is/VBZ not/RB the/DT answer/NN ./. E_S

Figure 5.1: Example of annotated dialogue in SWBD. The fragment consists of 7 turns, numbering from 117 to 123, switching between speaker A and speaker B. In turn 117, there are two utterances, namely utt1 and utt2. The first column is the dialogue act (see Appendix C). Each token in the utterance is annotated with a part-of-speech tag (see Appendix A). Some dialogue phenomenon are annotated with special mark-ups. For example, non-utterance event such as `laughter` is angle-bracketed. Restarts with substitution such as saying `Goo` followed by `good` is bracketed with the symbol "+". Detailed description on these special mark-ups can be found in (Taylor, 1995).

our work, two methods are used for selecting lexical features: mutual information and n-grams.

Mutual Information

A word sequence s is a lexical feature if its mutual information with a dialogue act d is sufficiently large and s occurs *frequently enough* in the data set. The set of lexical features is defined as: $C = \{s \in S \,|\, \exists\, d \; s.t. \; I(s,d) > \theta_1 \wedge \#(s) > \theta_2\}$, where S is the set of all possible word sequences and $\#(s)$ is the number of occurrences of s in the data set (θ_1 and θ_2 being thresholds). To minimize the size of S, only sequences containing three or fewer words are considered. Here, $I(s,d) = \log \frac{p(s,d)}{p(s)p(d)}$ represents the mutual information between s and d. Due to data sparseness, some DAs are not covered at all by the lexical features in C. Hence, another set of lexical features $C' = \{s \in S \,|\, I(s;D) > \theta_1 \wedge \#(s) > \theta_2\}$ is constructed, where $I(s;D) = \sum\limits_{d \in D} p(d,s) \log \frac{p(s,d)}{p(s)p(d)}$ is the mutual information of s with the set D of DAs that co-occur with s in the data set.

Bigram and Trigram

The bigram and trigram language models have previously been applied in DA recognition (Reithinger and Klesen, 1997; Stolcke et al., 2000). Using the interpolation method for smoothing (Jelinek and Mercer, 1980), they can be defined as in (5.1) and (5.2) respectively (where w_i, w_{i-1}, and w_{i-2} denote words and N is the total number of words in the data set):

$$p(w_i|w_{i-1}) = \lambda_1 \frac{\#(w_i)}{N} + \lambda_2 \frac{\#(w_{i-1}, w_i)}{\#(w_{i-1})} \tag{5.1}$$

$$p(w_i|w_{i-2}, w_{i-1}) = \lambda_1 \frac{\#(w_i)}{N} + \lambda_2 \frac{\#(w_{i-1}, w_i)}{\#(w_{i-1})} + \lambda_3 \frac{\#(w_{i-2}, w_{i-1}, w_i)}{\#(w_{i-2}, w_{i-1})} \tag{5.2}$$

To train the two language models, the data set is randomly divided into three parts: 80% for training, 10% for testing, and the remaining 10% for estimating the weight parameters λ_i's using Expectation Maximization (in each case, $\sum\limits_i \lambda_i = 1$). The perplexity measure $P = 2^{-\frac{1}{N}\sum_{i=1}^{N} \log p(w_i)}$ is used to evaluate the language models. For bigram, $\lambda_1 = 0.29$ and $\lambda_2 = 0.71$; for trigram, $\lambda_1 = 0.27$, $\lambda_2 = 0.38$, and $\lambda_3 = 0.35$. The perplexities are equal to 90.50 and 77.96 respectively, so it is *sufficiently informative* to use the two language models as features for DA recognition.

5.3.2 POS Features

POS features can be defined in a way similar to lexical features, by selecting POS sequences that have strong association with certain DA(s), based on mutual information or bigram/trigram. We refer to them as POS cues and POS bigrams/trigrams respectively. For example, YES NO QUESTION utterances often start with the POS sequence ⟨`VBP PRP`⟩ (i.e. subject auxiliary inversion), as in *Do*/`VBP` *you*/`PRP` *know*/`VB` *Peter*/`NNP` *?*/`.`".

5.3.3 Disfluency Features

In the data set, there are a lot of word fragments, filled pauses, repetitions, and self-repairs. These types of incoherent phenomena, known as disfluencies (Shriberg (1996)), are annotated with special mark-ups in the corpus. Hence, six features are designed to detect their existence in the utterance.

5.3.4 Grammatical Features

These features identify some special grammatical structures that are frequently used to convey certain DAs. For example, a YES ANSWER or NO ANSWER often does not contain a verb.

5.3.5 Punctuation Features

The use of certain punctuation marks provides hints for determining the DA. For example, an utterance with a question mark is likely (though not always) a question.

5.3.6 Historical Features

Historical features encode the relationship between the current utterance and the previous utterances by the same speaker and by the other speaker, which may reveal information useful to the interpretation of the current utterance. For example, if the current utterance repeats a large part of the previous utterance by the other speaker, it is probably a REPEAT PHRASE.

5.3.7 Surface Discourse Features

As mentioned in Section 5.2, the average utterance length for each DA type varies. Hence, the utterance's length may help the classifier to filter out some irrelevant DA types. Similarly, the utterance's position may also be useful. For example, if an utterance is at the beginning of a dialogue, it is likely a greeting message (which is a Conventional opening).

	Feature	Meaning of the feature	Domain
Lexical	(L1) C	Word sequences s selected based on $I(s,d)$	String
	(L2) C'	Word sequences s selected based on $I(s;D)$	String
	(L3) Bigram	Word sequences selected based on bigram	String
	(L4) Trigram	Word sequences selected based on trigram	String
POS	(POS1) POS cues T	POS sequences t selected based on $I(t,d)$	String
	(POS2) POS cues T'	POS sequences t selected based on $I(t;D)$	String
	(POS3) POS bigram	POS sequences selected based on bigram	String
	(POS4) POS trigram	POS sequences selected based on trigram	String
Disfluency	(D1) Non-linguistic element	Whether the utterance is a non-linguistic element	Boolean
	(D2) Last token	Last token of the utterance	String
	(D3) First token	First token of the utterance	String
	(D4) Bracket completion	Whether there are an odd number of brackets	Boolean
	(D5) Close bracket	Whether the first bracket is a close bracket	Boolean
	(D6) Restarts frequency	Number of restarts the utterance contains	Integer
Grammatical	(G1) Verb	Whether a verb exists in the utterance	Boolean
	(G2) Starts with verb	Whether the utterance starts with a verb	Boolean
	(G3) Polarity	Whether the utterance's condition is positive or negative	Boolean
Punctuation	(P1) Question mark	Whether the utterance ends with a question mark	Boolean
	(P2) Period	Whether the utterance ends with a period	Boolean
	(P3) Number of commas	Number of commas the utterance contains	Integer
Historical	(H1) Last token of previous utterance	Last token of the previous utterance by the same speaker	String
	(H2) Last token of counterpart utterance	Last token of the previous utterance by the other speaker	String
	(H3) Repetition count	Number of tokens of the counterpart utterance repeated in the current utterance	Integer
	(H4) Turn size	Number of turns in the counterpart utterance	Integer
Surface Discourse	(SD1) Length	Length of the utterance	Integer
	(SD2) Position	Position of the utterance in the dialogue	Integer

Table 5.1: Features used for dialogue act recognition

DA	**Speaker Turn**	**Utt. No.**	**Utterance**
h	A_117	utt1:	I/PRP do/VBP n't/RB know/VB ./. E_S
bh	A_117	utt2:	What/WP can/MD we/PRP do/VB about/IN it/PRP ?/. E_S
ba	B_118	utt1:	[Goo-/JJ ,/, + good/JJ] question/NN <laughter> ./. E_S
x	A_119	utt1:	<Laughter> ./.
%	B_120	utt1:	Probably/RB ,/, E_S
sv	A_121	utt1:	[Money/NN ,/, +
%	B_122	utt1:	Taking/VBG ,/, E_S
+	A_123	utt1:	Money/NN] is/VBZ not/RB the/DT answer/NN ./. E_S

Figure 5.2: Illustration of the contextual features captured by a history of tuple $\langle$`A_121`, `A_122`, `A_123`$\rangle$, where `A_123` is the current utterance.

5.4 Modeling Dialogue Act Recognition using ME

5.4.1 Notations

A subset of dialogues are randomly selected from the data set. This set, called the training set, is for modeling the ME classifier. We define some notations for the training set. Each speaker turn t_j in a dialogue consists of one or more utterances. Each utterance u_i in t_j has its dialogue act d_i. A "history" h_i of tuple $\langle t_{j-2}, t_{j-1}, u_i \rangle$, together with the dialogue act d_i, serves a training pair for $i = 1, 2, \ldots, n$. Here, t_{i-2} is the previous turn for the same speaker, and t_{i-1} is the previous turn for the counterpart. We use $\mathcal{H}$ to denote the set of all possible histories, and $\mathcal{D}$ to denote the set of dialogue acts.

5.4.2 Encoding Feature Vector

In ME modeling, each feature is represented using an *feature function*, denoted as $f : \mathcal{H} \times \mathcal{D} \rightarrow \mathbb{R}$. Specifically, its output is either 0 or 1 to indicate the existence of a feature in an utterance. For example, with reference to Figure 5.2, in turn `A_123`, the dialogue act label d is + (i.e. SEGMENT). Suppose that the value of a feature bigram in the utterance is `the answer`. A feature function f_j can then be encoded as:

$$f_j(d, h) = \begin{cases} 1 & \text{if } d\text{=+ and } \textit{bigram}\text{= How old} \\ 0 & \text{otherwise} \end{cases} \tag{5.3}$$

Feature name	**Feature value(s)**	**Feature function value**
L1	is not	1
L2	NULL	0
L3	Money is; is not; not the; the answer	1
L4	Money is not; is not the; not the answer	1
POS1	VBZ RB; DT NN	1
POS2	NULL	0
POS3	NN VBZ; VBZ RB; RB DT; DT NN	1
POS4	NN VBZ RB; VBZ RB DT; RB DT NN	1
D1	yes	1
D2	E_S	1
D3	Money	1
D4	Yes	1
D5	No	1
D6	0	1
G1	Yes	1
G2	No	1
G3	Yes	1
P1	No	1
P2	No	1
P3	0	1
H1	+	1
H2	E_S	1
H3	0	1
H4	1	1
SD1	6	1
SD2	123	1

Figure 5.3: Encoding result of turn `A_123`

Figure 5.3 shows the encoding result of turn `A_123`, using the 22 feature sets in Table 5.1. By enumerating all training examples, m possible feature functions can be formed. A m-dimensional *feature vector*, denoted as $\mathbf{f}(h, d) \in \mathbb{R}^m$, can be *sparsely* formed, in terms of 1 and 0, by the output value of each feature function. Let each component in the vector be $\mathbf{f}_k(h, d)$ for $k = 1, 2, \ldots, m$. By enumerating all training pairs of size n, the empirical count of the k-th feature is equal to:

$$\#(\mathbf{f}_k) = \sum_{i=1}^{n} \mathbf{f}_k(h_i, d_i) \tag{5.4}$$

where $\#(\mathbf{f}_k)$ refers to the number of occurrences of the k-th feature in the training set.

5.4.3 Training the Classifier

Given a set of history-act pairs, the aim of training is to find a *weight vector* **w** for the parametric form of ME (Jaynes, 1957):

$$p(d|h) = \frac{e^{\mathbf{w}\cdot\mathbf{f}(h,d)}}{\sum\limits_{d\in\mathcal{D}} e^{\mathbf{w}\cdot\mathbf{f}(h,d)}} \tag{5.5}$$

which is subject to the equality constraint :

$$\#(\mathbf{f}_k) = \tilde{\#}(\mathbf{f}_k) \quad \text{for } k = 1, 2, \ldots, m \tag{5.6}$$

Here, $\tilde{\#}(\mathbf{f}_k)$ is the expected count of the k-th feature, which can be estimated by the probability distribution $p(d|h)$:

$$\tilde{\#}(\mathbf{f}_k) = \sum_{i=1}^{n} \sum_{d'\in\mathcal{D}} \mathbf{f}_k(h_i, d')p(d'|h_i) \tag{5.7}$$

However, simply substituting Equation 5.5 into Equation 5.7 does not yield a closed form solution for solving the weight vector **w**. Alternatively, it must be proceeded analytically by using either iterative methods or gradient methods. In our experiments, we solved it by an iterative algorithm called Generalized Iterative Scaling (GIS) (Ratnaparkhi, 1998; Darroch and Ratcliff, 1972).

5.4.4 Dialogue Act Recognition

After solving the weight vector **w**, the classifier can be used to classify unseen utterances into DA types. Given a history h of an utterance u, the *most appropriate* DA of u is:

$$d_u = \operatorname*{argmax}_{d\in\mathcal{D}} p(d|h) \tag{5.8}$$

5.5 Experimental Evaluation

Experiments were performed to find a good combination of features for the ME classifier. In each run of the experiments, 1,000 dialogues were randomly picked up from the data set for training the classifier. Out of the remaining 125 dialogues, 100 dialogues were randomly selected as the testing set. The number of iterations of the GIS algorithm was set to 100. After training, the classifier was evaluated by measuring its performance in processing the testing set. The micro-average recognition rate was

Mutual information	Features set (L1-θ_1 / L2-θ_1)	θ_1	Average no. of lexical features	Average no. no. of events	Average no. feature functions	Mean recognition rate
$I(s,d)$	L1-0	0	37,067	103,583	7,043	64.87%
	L1-0.5	0.5	29,433	103,883	7,045	64.87%
	L1-1	1	20,025	101,113	6,245	64.33%
	L1-1.5	1.5	14,088	89,322	5,113	60.88%
	L1-2	2	10,466	67,114	4,223	42.96%
	L1-2.5	2.5	7,595	44,413	3,305	30.17%
$I(s;D)$	L2-0	0	10,573	103,583	7,044	64.87%
	L2-0.00005	0.00005	5,760	101,817	4,921	65.20%
	L2-0.0001	0.0001	4,160	100,685	3,881	65.21%
	L2-0.0002	0.0002	3,094	99,132	2,979	65.16%
	L2-0.0003	0.0003	2,630	98,242	2,588	65.14%
	L2-0.0005	0.0005	2,139	97,224	2,123	64.96%

Table 5.2: Recognition result using lexical features selected by mutual information

used as the metric, which is defined as the *mean* recognition rate of all testing cases, independent of the DA type each case belongs to.

5.5.1 Lexical Features

The two sets of lexical features L1 and L2 were generated using the two mutual information measures respectively. For each set, 6 subsets of experiments were performed, each using a different value of θ_1 with the value of θ_2 fixed at 5. In each subset, 3 runs were performed, each using a different training set and test set, randomly drawn from the date set. The mean recognition rate for each subset was then computed. As depicted in Table 5.2, the results obtained with $I(s;D)$ are consistently better than those obtained with $I(s,d)$.

On the other hand, lexical features may also be selected using bigram/trigram to select lexical features. Another 3 runs of experiments were performed in each case with the mean recognition rate being computed. As shown in Table 5.3, using bigram/trigram to select lexical features achieves higher recognition rate than that obtained with mutual information. However, it requires more events and feature functions. The training time is thus longer. As shown, the result obtained by using both bigram and trigram (i.e., L3 + L4) is only marginally better than that achieved with bigram alone (i.e., L3), but it requires more than twice the number of feature functions generated with bigram.

Feature set	Average no. of unique bi-/tri-grams	Average no. of events	Average no. of feature functions	Mean recognition rate
Bigram (L3)	196,859	123,131	14,100	68.24%
Trigram (L4)	547,839	106,825	14,413	66.30%
Bigram + Trigram (L3 + L4)	744,698	123,163	28,513	68.50%

Table 5.3: Recognition result using lexical features selected by bigram and trigram

Feature set	θ_1	Average no. of lexical features	Average no. of POS cues	Average no. of events	Average no. of feature functions	Mean recognition rate
L1-0.5	0.5	29,433	-	103,883	7,045	64.87%
L1-0.5 + POS1	0.5	29,433	9,501	107,827	9,246	66.40%
L2-0.0001	0.0001	4,160	-	100,685	3,881	65.21%
L2-0.0001 + POS2	0.0001	4,160	1,525	106,783	5,241	66.50%

Table 5.4: Recognition result using lexical features and POS cues selected by mutual information

5.5.2 POS Features

POS1 and POS2 were then constructed. For POS1, $\theta_1 = 0.5$ while for POS2, $\theta_1 = 0.0001$, which were the values that led to the best results for L1 and L2 respectively (L1-0.5 was chosen instead of L1-0 because it generated fewer lexical features). Two feature sets were formed, namely, POS1 $\cup$ L1 and POS2 $\cup$ L2, to train and evaluate the classifier separately. For each set, 3 runs of experiments were performed. As shown in Table 5.4, adding POS cues improves the performance for both L1 and L2.

On the other hand, performance is always improved by using both lexical and POS bigrams/trigrams as features than using lexical bigrams/trigrams alone (see Table 5.5). One can see that the recognition rate achieved with L3 is only marginally lower than the best-case result, but it requires the fewest feature functions and the shortest training time. To strike a balance between performance and efficiency, L3 was selected as the baseline feature. We then selected features from the remaining feature types to be used with L3.

5.5.3 Combining Features

To select a good combination of features, a greedy approach was used (see Figure 5.4). In each step, a separate feature set was formed using the baseline feature set "Baseline" and one of the features in the set "Features", which was used to train and evaluate a classifier. We run the evaluation 10 times, each using a different training set and

Feature set	Avg. no. of unique bi-/tri-grams	Avg. no. of unique POS bi-/tri-grams	Avg. no. of events	Avg. no. of feature functions	Mean recog. rate
L3	196,859	-	123,131	14,100	68.24%
L3 + POS3	196,680	2,337	125,669	15,183	69.23%
L4	547,839	-	106,825	14,413	66.30%
L4 + POS4	546,976	20,026	124,155	21,039	68.96%
L3 + L4	744,698	-	123,163	28,513	68.50%
L3 + L4 + POS3 + POS4	743,656	22,363	125,671	36,211	69.82%

Table 5.5: Recognition result using both lexical and POS bigrams/trigrams as features

```
1.   Baseline $\leftarrow$ { L3 };
2.   Train and evaluate an ME classifier $\mathcal{C}[0]$ for 10 runs using Baseline as feature set;
         Calculate $\mu[0]$ and $\sigma[0]$, mean and variance of recognition rate of $\mathcal{C}[0]$;
3.   Features $\leftarrow$ { POS, D, G, P, H, SD };
4.   repeat
5.       $i \leftarrow 1$;
6.       for each $f \in$ Features do
7.           $F[i] \leftarrow$ Baseline $\cup \{f\}$;
8.           Train and evaluate an ME classifier $\mathcal{C}[i]$ for 10 runs using $F[i]$ as feature set;
                 Calculate $\mu[i]$ and $\sigma[i]$, mean and variance of recognition rate of $\mathcal{C}[i]$;
9.           $i \leftarrow i + 1$;
10.      $m \leftarrow \underset{k}{\text{argmax}}\, \mu[k]$, where $1 \le k < i$;
11.      if $(\mu[m] > \mu[0])$ then
12.          Calculate $t$: $t$-value with respect to the recognition rate achieved with Baseline;
13.          if $(t > t_\beta)$ then
14.              Baseline $\leftarrow F[m]$; $\mu[0] \leftarrow \mu[m]$; $\sigma[0] \leftarrow \sigma[m]$; Features $\leftarrow$ Features $\setminus F[m]$;
15.  until ((Features $= \emptyset$) or $(\mu[0] > \mu[m])$ or $(t_\beta \le t)$);
```

Figure 5.4: Algorithm for feature selection

testing set, randomly drawn from the data set. The feature in “Features” that led to the greatest improvement in recognition rate was identified. If the improvement was significant with respect to the recognition rate achieved with the baseline feature set, the feature would be added to “Baseline” (and removed from “Features”).[2] The process was repeated until there was no further significant improvement in recognition rate (see Table 5.6). The combined feature set consists of L3, H, D, SD, and P. The mean recognition rate is 74.07%, with the best-case performance being equal to 75.03%. The result is better than that achieved by Stolcke et al. (2000), who used the same data set as ours and achieved a recognition rate of 71.0% only.[3] Moreover, as shown in Table 5.7, our performance is comparable to that achieved by the other task-oriented approaches, despite the fact that they use far smaller data sets and testing sets, and they involve significantly smaller number of DAs.

5.6 Improving Macro-average Recognition Rate

Despite the high micro-average recognition rate achieved, the individual recognition rates of some DA types are low. Some DAs even have zero recognition rate. This might be problematic when encountered in real-world applications. For example, the dialogue act MAYBE/ACCEPT PART (e.g., *Something like that*) is one of dialogue acts having zero recognition rate. However, it is crucially important when dealing with some uncertain situations. Failing to recognize such a dialogue act will definitely affect the interaction with the user, leading to an incoherent dialogue *flow*.

Currently, the macro-average recognition rate, defined as the average of the recognition rates of individual DA types, is unsatisfactory. For example, using the feature set {L3, H, D, SD, P}, the ME classifier achieves a mean macro-average recognition rate of 32.35% only. The cause of the problem is that the distribution of examples among the DAs in the data set is heavily skewed, as depicted in Figure 5.5 (in the figure, the higher the rank of a DA, the lower its frequency of occurrence in the original data set). Out of the 44 DAs, only 11 of them appear in more than 1% of the utterances in the

[2]The t-value was calculated for this purpose: $t = \frac{\mu[m]-\mu[0]}{\sqrt{\frac{\sigma[m]^2+\sigma[0]^2}{2}}\cdot\sqrt{\frac{2}{N}}}$, where $N = 10$. If the confidence level corresponding to the t-value exceeded 50% (i.e. $t > t_\beta = 0.6885$), the improvement in recognition rate was considered to be significant.

[3]As claimed by Clark and Popescu-Belis (2004), human accuracy for the DA recognition task is estimated to be 84%, based on the inter labeler agreement rate in labeling Switchboard.

Step	Feature set	Mean	Variance	Best-case	t-value	Confidence
1	Baseline: L3	68.48%	0.4576	69.42%	-	-
	L3 + POS	69.23%	0.3568	70.10%	1.74	$\approx$ 90%
	L3 + D	69.82%	0.3572	70.84%	3.13	$>$ 95%
	L3 + G	68.18%	0.4446	69.16%	-0.67	$\approx$ 49%
	L3 + P	68.30%	0.3639	69.08%	-0.43	$\approx$ 33%
	L + H	**71.34%**	**0.4519**	**72.15%**	**6.34**	$>$ **95%**
	L3 + SD	68.75%	0.4516	69.63%	0.59	$\approx$ 44%
2	Baseline: L + H	71.34%	0.4519	72.15%	-	-
	L3 + H + POS	71.86%	0.4346	72.87%	1.17	$\approx$ 74%
	L + H + D	**73.30%**	**0.4381**	**74.44%**	**4.40**	$>$ **95%**
	L3 + H + G	71.95%	0.5090	72.87%	1.30	$\approx$ 79%
	L3 + H + P	72.13%	0.3930	72.88%	1.82	$\approx$ 91%
	L3 + H + SD	72.20%	0.5625	73.10%	1.80	$\approx$ 91%
3	Baseline: L + H + D	73.30%	0.4381	74.44%	-	-
	L3 + H + D + POS	72.95%	0.4460	73.94%	-0.80	$\approx$ 57%
	L3 + H + D + G	73.44%	0.4105	74.50%	0.30	$\approx$ 23%
	L3 + H + D + P	73.40%	0.5182	74.48%	0.21	$\approx$ 16%
	L + H + D + SD	**73.70%**	**0.4444**	**74.84%**	**0.91**	$\approx$ **63%**
4	Baseline: L + H + D + SD	73.70%	0.4444	74.84%	-	-
	L3 + H + D + SD + POS	73.22%	0.4418	74.16%	-1.10	$\approx$ 70%
	L3 + H + D + SD + G	73.80%	0.4353	74.97%	0.20	$\approx$ 16%
	L + H + D + SD + P	**74.07%**	**0.3960**	**75.03%**	**0.83**	$\approx$ **59%**
5	Baseline: L + H + D + SD + P	74.07%	0.3960	75.03%	-	-
	L3 + H + D + SD + P + POS	73.56%	0.3808	74.38%	-1.22	$\approx$ 76%
	L + H + D + SD + P + G	74.15%	0.4073	75.18%	0.18	$\approx$ 14%

Table 5.6: Experimental results for selection features

			No. of	No. of dialogues		No. of utterances		Recognition
	Task	Tag set	DAs	Training	Testing	Training	Testing	rate
Our approach	Switchboard (70 topics)	SWBD-DAMSL	44	1,000	100	192,365	18,962	74.07%
Stolcke et al. (2000)	Switchboard (70 topics)	SWBD-DAMSL	42	1,115	19	198,000	4,000	71.00%
Samuel et al. (1998)	Appointment scheduling	VERBMOBIL	18	143	20	2,701	328	75.12%
Reithinger and Klesen (1997)	Appointment scheduling	VERBMOBIL	18	143	20	2,701	328	74.70%
Wright (1998)	Map routing	Maptask	12	20	5	3,726	1,061	64.00%
Clark and Popescu-Belis (2004)	Meeting recording	MALTUS	11	40	5	?	6,771	73.20%

Table 5.7: Performance comparison with previous works on DA recognition

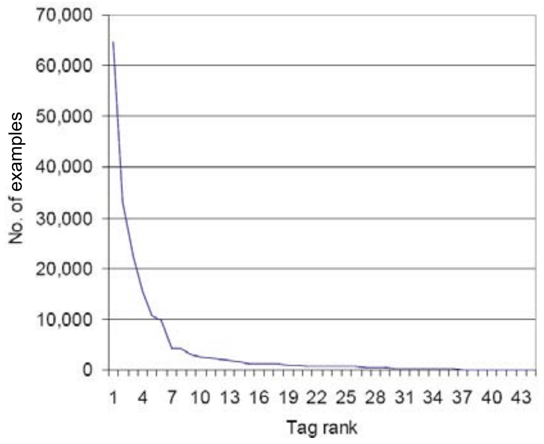


Figure 5.5: Distribution of exmples among the DAs in the original data set. The DA tags are ranked according to their frequencies of occurrence.

data set. Yet, nearly 34% of the utterances in the data set are Statement. Due to the lack of examples in the training set, the classifier is relatively weak in recognizing some DAs.

In practice, it is very difficult to find a data set that has an even distribution of examples among the DAs, since some DAs are rarely used in conversation that it is hard to increase their numbers drastically. Hence, we address this class-imbalance problem by two methods: over-sampling method and error-based learning method.

5.6.1 Over-sampling Method

In the over-sampling method, we create training examples via replication. Figure 5.6 shows an example. The distribution of examples in the original training set (the one labeled as "no replication") is modified to give the new distribution (the one labeled as "with replication"), by randomly selecting a certain number of examples from each DA type and replicating them. For a DA type d (where d is from 1 to $|D|$, $|D|$ being the number of DA types), the number of examples replicated, n_d, is defined as:

$$n_d = \frac{(N_d - N_{|D|})}{(N_1 - N_{|D|})} \times (N_1 - \rho) + \rho - N_d \tag{5.9}$$

Here, N_d is the number of example of the DA type d in the original training set. As depicted, more examples are replicated for DAs with higher ranks, i.e. those occur less frequently in the original training set. In fact, the extra examples replicated for the

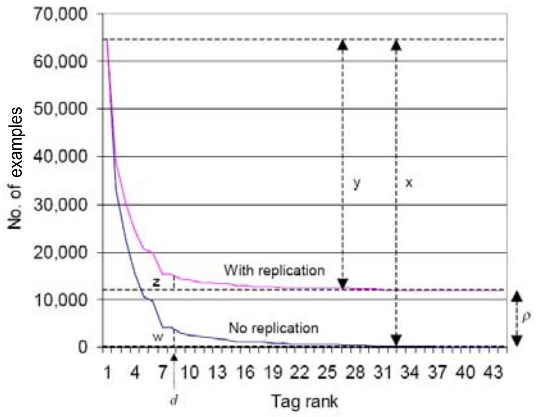


Figure 5.6: Determining the number of examples to replicate for a DA type d (where d is from 1 to $|D|$, $|D|$ being the number of DA types). Here, $\mathsf{x} = N_1 - N_{|D|}$, $\mathsf{w} = N_d - N_{|D|}$, $\mathsf{v} = N_1 - \rho$, and $\mathsf{z} = \frac{\mathsf{w}}{\mathsf{x}} \cdot \mathsf{v}$. The number of examples replicated is $n_d = \mathsf{z} + \rho - N_d$. Note

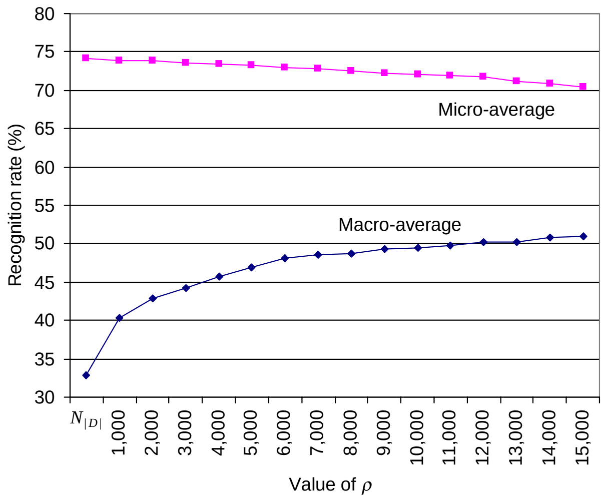


Figure 5.7: Experimental results of example replication. The feature set used is {L3, H, D, SD, P, G}. Note that $\rho = N_{|D|}$ corresponds to the case where the original data set is used (i.e. no replication).

DAs with the lowest ranks are almost negligible. The new distribution of examples among the DAs is thus more balanced than the original one. The degree of balance is controlled by the parameter $\rho > 0$. In general, for a larger ρ, the distribution will be more balanced.

Preliminary experiments have been performed to assess the approach for different values of ρ, using the features L3, H, D, SD, P, and G. For each value of ρ, 3 runs have been performed, using different training and testing sets, and the mean recognition rates are reported. As depicted in Figure 5.7, with $\rho = 1,000$, the macro-average recognition rate is boosted from 32.83% to 40.36%, at the expense of only a slight decrease of the micro-average recognition rate from 74.15% to 73.87%. The macro-average recognition rate further improves to 50.90% as ρ is being increased from 1,000 to 15,000. The micro-average recognition rate has decreased to 70.48% at the same time. Yet, the extent is moderate as compared with the gain in the macro-average recognition rate, and it is still maintained at a relatively high level.

By using replication of examples, improvement in recognition rate is observed in many of the DAs with higher ranks, which used to have very low frequencies of occurrence in the original data set. Particularly, the number of DA types with zero recognition rate is greatly reduced. A higher macro-average recognition rate can thus be achieved. For example, in the experiment set that generates the best recognition rate of 75.18% using the original data set (see Table 5.6), a total of 10 DA types have zero recognition rate, and the macro-average recognition rate achieved is only 32.40% (see Figure 5.8, the dotted line). Whereas by using example replication with $\rho = 10,000$ (see Figure 5.8, the solid line), there are no DA types with zero recognition rate, and the macro-average recognition rate achieved is 51.78%. As depicted in Figure 5.9, using replication with $\rho = 1,000$, the average number of DA types with zero recognition rate drops from 9.3 to 4, which is further reduced to about 1.33 as ρ is being increased from 1,000 to 15,000.

5.6.2 Error-Based Learning Method

In the error-based learning method, we target at reducing the number of errors introduced by the minority class. We achieve this by modifying the learning mechanism of Maximum Entropy. Inside, we add a parameter r_y to the original updating rule in the GIS algorithm. In an iteration t, the modified updating rule for each feature estimates

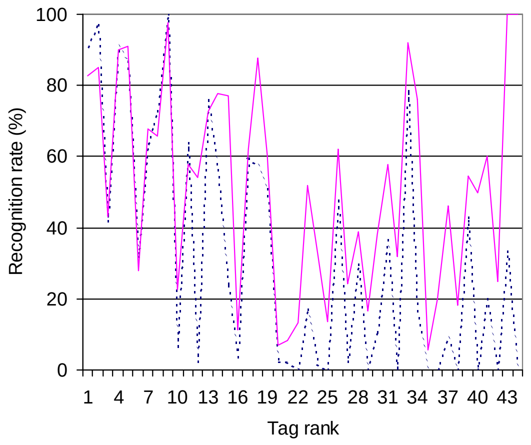


Figure 5.8: Recognition rates of individual DAs before replication (dotted line) and after replication (solid line, $\rho = 10,000$). The ranks of the DA tags are the same as in Figure 5.5.

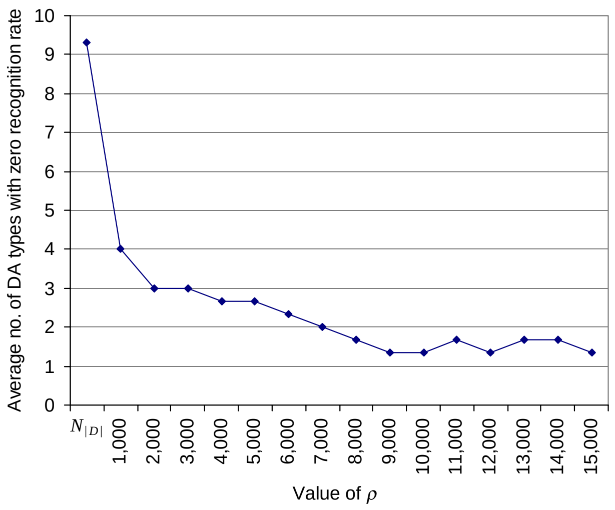


Figure 5.9: The replication of the training examples reduces the number of DA types with zero recognition rate.

its weight w as follows:

$$w_t \leftarrow w_{t-1} + r_y \cdot \frac{1}{C} \log \frac{\tilde{c}}{c} \tag{5.10}$$

Here, w is estimated by the divergence between the observed feature count $\tilde{c}$ and the modeled feature count c, with C being a constant. In each iteration, the number of correct predictions for each class r_y is recorded, estimating the accuracy distribution among classes. The value of r is determined by an inverse function about the accuracy of a class y.

Preliminary experiments have been performed to evaluate this method, using the features L3, H, D, SD, P, and G. We performed the experiments 3 times, using different training and testing sets, and the mean recognition rates are reported. The macro-average recognition rate is boosted from 32.83% to 35.60%, with a slight decrease of the micro-average recognition rate from 74.15% to 74.04%. Moreover, the average number of DA types with zero recognition rate drops from 9.3 to 7.

5.7 Summary

We have built a maximum entropy classifier for DA recognition. The approach allows heterogeneous sources of features to be integrated in one classifier flexibly, without assuming that they are independent. Compared with other approaches, our classifier is more suitable for online DA recognition, since the classification features are defined over the current utterance and its previous context only, and it does not require the whole dialogue to be available in order to label the utterances. Evaluation using a non-task oriented data set reveals that our approach achieves a best-case recognition rate of over 75%, outperforming other approaches that employ the same data set. The result is also comparable to other works on task-oriented DA recognition. To improve the recognition performance of those DAs with very low frequencies of occurrence in the data set, two methods have been applied to address the problem. One is over-sampling method and one is error-based learning method. Preliminary evaluation reveals that both methods are effective in boosting the macro-average recognition rate.

Chapter 6

Conclusion

To conclude the thesis, we summarize the findings and results of this work, and we discuss some possible directions in which this work might benefit from future investigation.

6.1 Summary and Results

This thesis has studied how to model natural language disambiguation by proven classification techniques. The classifiers were built using Maximum Entropy, which allows combination of features from heterogeneous sources, and is highly resistant to irrelevant features. This capability of feature integration is robust to various levels of disambiguation in natural language, which require extensive contextual information in general.

We have empirically evaluated this capability by conducting experiments on two challenging tasks. They are semantic role labeling and dialogue act recognition, respectively, at the semantic and pragmatic level. Both tasks are useful in improving our prototype dialogue-system (Lan et al., 2003, 2005), which contains two parts. One is to extract information by filling slots in pre-defined patterns, and the other is to provide user with extracted information through a dialogue interface. Semantic role labeling can benefit the extraction part by providing arguments to the slots, while dialogue act recognition can help the interface part to be more robust for interacting with users.

In the semantic role labeling task, we have proposed a three-phase labeling approach to the problem. The first two phases are used for recognizing arguments, that provides candidates to the last phase for semantic role assignment. The success in recognizing arguments, as shown in previous literatures, was often the key in achieving high system performance. We have found that previous approaches in recognizing arguments have their unique advantages, together with their own weaknesses. In view of this, our approach combines their advantages, while addressing their weaknesses by their complementary supports. Firstly, our approach takes advantage from word-level classification, to guarantee no argument-alignment problem, which is frequently encountered in constituent-level classification. Secondly, our approach benefits by chunk-level classification in reducing the numerous classification steps, which are involved in word-level classification. Thirdly, our approach gains advantage from constituent-level classification, to ensure recognizing long-ranged arguments, which is hardly achieved in word-level and chunk-level classification. Experimental results show that the performance of our system is comparable with the current best *single* system using the same data set, despite the fact that they employ weighted ensemble classifiers.

In the dialogue act recognition task, we have studied various types of features to evaluate system performance. Our feature set is only defined over the current utterance and its pervious context, so our system can readily be applied to online dialogue act recognition. Experiments were conducted using a large public data set, which is non-task oriented. Experimental results show that our system outperforms others that used the same data set. Moreover, our result is comparable to that performed on task-oriented data set, despite the fact that their training size and number of dialogue acts are both smaller than ours.

We have also addressed the class-imbalance problem encountered in both disambiguation tasks. We have employed two methods for each task. One is over-sampling and the other is error-based learning. For over-sampling, unlike traditional approaches that more usually being restricted to binary-class problems, our sampling method is fully applicable to multi-class problems. On the other hand, for the error-based learning, our method can also be regarded as a wrapper technique, which can be portable to other scoring-based classifiers. Experimental results show that both methods are effective in improving the macro-average performance.

In summary, this research has demonstrated the feasibility to model automatic disambiguation, with two substantial tasks that are significant for dialogue systems. This research holds promise for constructing natural language interfaces to information systems, making the wealth amount of information easily accessible by users.

6.2 Directions for Future Work

We introduce three possible directions for future work, and we believe that they might enhance the current system.

6.2.1 From Local to Global Inference

In the three-phase labeling approach for semantic role labeling, we employ the "divide-and-conquer" strategy to decompose the task into several sub-problems. In every phase, local decisions are directly assigned, without considering other possible assignments that can be recognized later. This pipelined architecture is thus very greedy, probably leading to sub-optimal results. To address the issue, we can maintain all plausible decisions in each phase, and consider the competing candidates to achieve the final decisions. To achieve this, we require an extra phase for global inference. Inside, an objective scoring function is to be designed. The aim is then to maximize the scores among the competing solutions.

6.2.2 Cached Learning for Maximum Entropy

The main objective of Maximum Entropy learning is to equalize the observed feature counts and the modeled feature counts. In each iteration, the algorithm visits each example, updating the modeled counts, estimating the parameters by the divergence between the observed and the modeled counts, and visiting again the example at next iteration, until the divergence is minimized. This procedure is clearly inefficient. In general, some of modeled counts, after a number of iterations, will become stable. We can thus skip the computation for these counts. Alternatively, we can improve it by incorporating a cache. We can store the indexes for the counts whose divergences are far from their expected counts. We can then focus on learning those cached counts instead of all the counts.

6.2.3 Synthesis of Examples at Feature Level

In our over-sampling method, minority-class examples are replicated rather than synthesized. In the literature, Chawla et al. (2002) proposed SMOTE to synthesize examples at feature space. In their method, a new example is synthesized by taking the difference between the feature vector at hand and its nearest neighbor, multiplying the difference by a random number, and adding the difference to the original vector for creating a new vector. Their feature vectors are real-valued ones. However, most of our features are nominal-valued, in which we face difficulties to perform feature vector substraction. To counter this issue, we can use WordNet (Fellbaum, 1998) to perform feature substitution instead of feature substraction. For example, for word-string feature *car*, we can find its hypernym *vehicle* as substitute. However, this method is only applicable to lexical word, but not other complex nominal features, such as syntactic path. A more robust method is to use voting method (Chawla et al., 2003). Inside, majority vote is taken between the feature vector and its k nearest neighbors. The voted feature-vector thus becomes the new example.

Appendix A

Penn TreeBank Tag Set

Clause tag	Description
S	simple declarative clause
SBAR	clause introduced by a subordinating conjunction
SBARQ	direct question introduced by *wh*-word or a *wh*-phrase
SINV	inverted declarative sentence
SQ	inverted yes/no question

Phrase tag	Description
ADJP	adjective phrase
ADVP	adverb phrase
CONJP	conjunction phrase
FRAG	fragment
INTJ	interjection
LST	list marker
NAC	not a constituent
NP	noun phrase
NX	complex noun phrase
PP	prepositional phrase
PRN	parenthetical
PRT	particle
QP	quantifier phrase
RRC	reduced relative clause
UCP	unlike coordinated phrase
VP	vereb phrase
WHADJP	*wh*-adjective phrase
WHAVP	*wh*-adverb phrase
WHNP	*wh*-noun Phrase
WHPP	*wh*-prepositional phrase
X	unknown, uncertain, or unbracketable

Part-of-speech tag	Description
CC	coordinating conjunction
CD	cardinal number
DT	determiner
EX	existential there
FW	foreign word
IN	preposition or subordinating conjunction
JJ	adjective
JJR	adjective, comparative
JJS	adjective, superlative
LS	list item marker
MD	modal
NN	noun, singular or mass
NNS	noun, plural
NNP	proper noun, singular
NNPS	proper noun, plural
PDT	predeterminer
POS	possessive ending
PRP	personal pronoun
PRP$	possessive pronoun
RB	adverb
RBR	adverb, comparative
RBS	adverb, superlative
RP	particle
SYM	symbol
TO	infinitive *to*
UH	interjection
VB	verb, base form
VBD	verb, past tense
VBG	verb, gerund or present participle
VBN	verb, past participle
VBP	verb, non-3rd person singular present
VBZ	verb, 3rd person singular present
WDT	*wh*-determiner
WP	*wh*-pronoun
WP$	possessive *wh*-pronoun
WRB	*wh*-adverb

Appendix B

PropBank Tag Set

Argument tag	Description
A0, A1, A2, A3, A4, A5, AA	each role definition depends on the verb sense from VerbNet (Kipper et al. (2000))
AM	bare adjunct argument
AM-ADV	general purpose argument
AM-CAU	cause argument
AM-DIR	direction argument
AM-DIR	discourse marker argument
AM-EXT	extent argument
AM-LOC	location argument
AM-MNR	manner argument
AM-MOD	model verb argument
AM-NEG	negation marker argument
AM-PNC	purpose argument
AM-PRD	predication argument
AM-REC	reciprocal argument
AM-TMP	temporal argument
R-A0, R-A1, R-A2, R-A3, R-A4, R-AA	each definition is the same as its referenced argument
R-AM-ADV	general purpose reference argument
R-AM-CAU	cause reference argument
R-AM-DIR	direction reference argument
R-AM-EXT	extent reference argument
R-AM-LOC	location reference argument
R-AM-MNR	manner reference argument
R-AM-PNC	purpose reference argument
R-AM-TMP	temporal reference argument

Appendix C

SWBD-DAMSL Tag Set

Dialogue act tag	Description
%	uninterpretable
%-	abandoned/turn taking
ˆ2	collaborative completion
ˆg	tag question
ˆh	reject
ˆq	quotation
+	segment
aa	agreement/accept
aap, am	self talk
ad	action directive
ar	negative non no answers
arp, nd	dispreferred answers
b	backchannel/acnowledge
bˆm	repeat phrase open question
ba	appreciation
bd	downplayer
bf	summarize/refumulate
bh	backchannel question
bk	response acknowledgment
br	other answers
fa	apology
fc	conventional closing
fpno	conventional opening
ft	thanking
h	hedge
na, nyˆe	affimative non yes answers

Dialogue act tag	Description
ng, nnˆe	signal non understanding
nn	no answers
ny	yes answers
o, fo, bc, by, fw	other
oo, cc, co	offers, options or, commits
qh	hold before answer/agreement
qo	rhetorical questions
qrr	or clause
qw	*wh*-question
qwˆd	declarative wh question
qy	yes-no question
qyˆd	declarative yes-no question
sd	statement
sv	opinion
t1	maybe/accept part
t3	3rd party talk
x	non verbal

Bibliography

Abney, S. (1991). Parsing by chunks. In Berwick, R., Abney, S., and Tenny, C., editors, *Principle-Based Parsing: Computation and Psycholinguistics*, pages (257–278. Kluwer, Dordrecht.

Abney, S. (1997). Stochastic attribute value grammars. *Computational Linguistics*, 23(4):597–618.

Allen, J. (1995). *Natural Language Understanding.* John Benjamins, CA.

Allen, J., Miller, B., Ringger, E., and Sikorski, T. (1996). Robust understanding in a dialogue system. In *Proceedings of the 34th Annual Conference on Association for Computational Linguistics (ACL-1996)*, pages 62–70, Santa Cruz, CA.

Allwood, J. (2000). *Abduction, Belief and Context in Dialogue: Studies in Computational Pragmatics*, chapter An Activity based Approach to Pragmatics, pages 47–80. John Benjamins, Philadelphia.

Anderson, A., Bader, M., Bard, E. G., Boyle, E., Doherty, G., Garrod, S., Isard, S., Kowtko, J., McAllister, J., Miller, J., Sotillo, C., Thompson, H., and Weinert, R. (1991). The HCRC Map Task corpus. *Language and Speech*, 34(4):351–366.

Arfken, G. (1985). *Mathematical Methods for Physicists.* Academic Press, CA.

Austin, J. L. (1962). *How to do Things with Words.* Clarendon Press, Oxford.

Baker, C., Fillmore, C., and Lowe, J. (1998). The berkeley framenet project. In *Proceedings of the 17th International Conference on Computational Linguistics (COLING-1998)*, pages 86–90, Montreal, Canada.

Baldewein, U., Erk, K., Padó, S., and Prescher, D. (2004). Semantic role labeling with chunk sequences. In *Proceedings of the 8th Conference on Computational Natural Language Learning (CONLL-2004)*, pages 98–101, Boston, MA.

Batista, G., Prati, R., and Monard, M. (2004). A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explorations*, 6(1):20–29.

Berger, A. and Miller, R. (1998). Just-in-time language modelling. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-1998)*, pages 705–708, Seattle, WA.

Berger, A., Pietra, S. D., , and Pietra, V. D. (1996). A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

Bharati, A., Venkatapathy, S., and Reddy, P. (2005). Inferring semantic roles using sub-categorization frames and maximum entropy model. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CONLL-2005)*, pages 165–168, Ann Arbor, MI.

Bickel, P. and Doksum, K. (1977). *Mathematical Statistics : Basic Ideas and Selected Topics*. Holden-Day, CA.

Bikel, D. and Weischedel, R. S. R. (1999). An algorithm that learns what's in a name. *Machine Learning Journal*, 34(1-3):211–231.

Blaheta, D. and Charniak, E. (2000). Assigning function tags to parsed text. In *Proceedings of the 1st Annual Meeting of the North American Chapter of the Association for Computational Linguistics (NACACL-2000)*, pages 234–240, Seattle, WA.

Box, G. and Tiao, G. (1992). *Bayesian Inference in Statistical Analysis*. Wiley, New York.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

Brill, E. (1995). Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. *Computational Linguistics*, 21(4):543–566.

Bunt, H. (1994). Context and dialogue control. *Think Quarterly*, 3(1):19–31.

Carreras, X., Màrquez, L., , and Chrupala, G. (2004). Hierarchical recognition of propositional arguments with perceptrons. In *Proceedings of the 8th Conference on Computational Natural Language Learning (CONLL-2004)*, pages 106–109, Boston, MA.

Carreras, X. and Màrquez, L. (2004). Introduction to the conll-2004 shared task: Semantic role labeling. In *Proceedings of the 8th Conference on Computational Natural Language Learning (CONLL-2004)*, pages 89–97, Boston, MA.

Carreras, X. and Màrquez, L. (2005). Introduction to the conll-2005 shared task: Semantic role labeling. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CONLL-2005)*, pages 152–164, Ann Arbor, MI.

Chan, P. and Stolfo, S. (1998). Towards scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD-1998)*, pages 164–168, New York, NY.

Charniak, E. (2000). A maximum-entropy-inspired parser. In *Proceedings of the 1st Conference of North American Chapter of the Association for Computational Linguistics (NAACL-2000)*, pages 132–139, Seattle, WA.

Chawla, N., Bowyer, K., Hall, L., and Kegelmeyer, P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.

Chawla, N., Lazarevic, A., Hall, L., and Bowyer, K. (2003). SMOTEBoost: Improving prediction of the minority class in boosting. pages 107–119, Cavtat-Dubrovnik, Croatia.

Chen, J. and Rambow, O. (2003). Use of deep linguistic features for the recognition and labeling of semantic arguments. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP-2003)*, pages 41–48, Sapporo, Japan.

Chen, S. and Rosenfeld, R. (2000). A survey of smoothing techniques for ME models. *IEEE Transactions on Speech and Audio Processing*, 8(1):37–50.

Clark, A. and Popescu-Belis, A. (2004). Multi-level dialogue act tags. In *Proceedings of the Workshop on Discourse and Dialogue at 5th SIGdial*, pages 163–170, Cambridge, MA.

Cohen, P. and Perrault, R. (1979). Elements of a plan-based theory of speech acts. *Cognitive Science*, 3(3):177–212.

Cohn, T. and Blunsom, P. (2005). Semantic role labelling with tree conditional random fields. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CONLL-2005)*, pages 169–172, Ann Arbor, MI.

Collins, M. (1999). *Head-driven Statistical Models for Natural Language Parsing.* PhD thesis, University of Pennsylvania.

Core, M. and Allen, J. (1997). Coding dialogs with the DAMSL annotation scheme. In *Proceedings of the AAAI Fall Symposium on Communicative Action in Humans and Machines*, pages 28–35, Cambridge, MA.

Cost, S. and Salzberg, S. (1993). A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, 10(1):57–78.

Darroch, J. and Ratcliff, D. (1972). Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics*, 43(5):1470–1480.

Dietterich, T. and Bakiri, G. (1995). Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286.

Domingos, P. (1999). MetaCost: A general method for making classifiers cost sensitive. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-1999)*, pages 155–164, San Diego, CA.

Drummond, C. and Holte, R. (2003). C4.5, class imbalance, and cost sensitivity: Why undersampling beats over-sampling. In *Proceedings of the Workshop on Learning from Imbalanced Data Sets II at the 20th International Conference on Machine Learning (ICML-2003)*, Washington, DC.

Duda, R., Hart, P., and Stork, D. (2001). *Pattern Classification.* Wiley, New York.

Elkan, C. (2001). The foundations of cost-sensitive learning. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-2001)*, pages 973–978, Seattle, WA.

Fan, W., Stolfo, S., Zhang, J.-X., and Chan, P. (1999). AdaCost: Misclassification cost-sensitive boosting. In *Proceedings of the 16th International Conference on Machine Learning (ICML-1999)*, pages 99–105, Morgan Kaufmann, CA.

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

Freund, Y. and Schapire, R. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139.

Ganapathiraju, A. and Picone, J. (2000). Hybrid SVM/HMM architectures for speech recognition. In *Proceedings of the 6th International Conference on Spoken Language Processing (ICSLP-2000)*, pages 504–507, Beijing, China.

Gildea, D. and Hockenmaier, J. (2003). Identifying semantic roles using combinatory categorial grammar. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP-2003)*, pages 57–64, Sapporo, Japan.

Gildea, D. and Jurafsky, D. (2000). Automatic labeling of semantic roles. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics (ACL-2000)*, pages 512–520, Hong Kong, China.

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

Gildea, D. and Palmer, M. (2002). The necessity of syntactic parsing for predicate argument recognition. In *Proceedings of the 40th Annual Conference of the Association for Computational Linguistics (ACL-2002)*, pages 239–246, Philadelphia, PA.

Godfrey, J., Holliman, E., and McDaniel, J. (1992). SWITCHBOARD: Telephone speech corpus for research and development. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-1992)*, pages 517–520, San Francisco, CA.

Goodman, J. (2004). Exponential priors for maximum entropy models. In *In Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2004)*, pages 305–312, Boston, MA.

Grzymala-Busse, J., Goodwin, L., and Grzymala-Busse, W. (2000). An approach to imbalanced data sets based on changing rule strength. In *In Technical Report of the Workshop on Learning from Imbalanced Data Sets at AAAI'2000*, pages 69–74, Austin, TX.

Guo, H. and Viktor, H. (2004). Learning from imbalanced data sets with boosting and data generation: The databoost-im approach. volume 6, pages 30–39.

Hacioglu, K., Pradhan, S., Ward, W., Martin, J., and Jurafsky, D. (2004). Semantic role labeling by tagging syntactic chunks. In *Proceedings of the 8th Conference on Computational Natural Language Learning (CoNLL-2004)*, pages 110–113, Boston, MA.

Hacioglu, K. and Ward, W. (2003). Target word detection and semantic role chunking using support vector machines. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL-2003)*, pages 25–27, Edmonton, Canada.

Haghighi, A., Toutanova, K., and Manning, C. (2005). A joint model for semantic role labeling. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 173–176, Ann Arbor, MI.

Higgins, D. (2004). A transformation-based approach to argument labeling. In *Proceedings of the 8th Conference on Computational Natural Language Learning (CONLL-2004)*, pages 114–117, Boston, MA.

Hosmer, D. and Lemeshow, S. (1989). *Applied Logistic Regression*. Wiley, New York.

Janin, A., Baron, D., Edwards, J., Ellis, D., Gelbart, D., Morgan, N., Peskin, B., Pfau, T., Shriberg, E., Stolcke, A., and Wooters, C. (2003). The ICSI meeting corpus. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-2003)*, pages 364–367, Hong Kong, China.

Japkowicz, N. and Stepehn, S. (2002). The class imbalance problem: A systematic study. *Intelligent Data Analysis Journal*, 6(5):429–449.

Jaynes, E. (1957). Information theory and statistical mechanics. *Physical Review*, 3:620–630.

Jebara, T. (2001). *Discriminative, Generative and Imitative Learning.* PhD thesis, Media Laboratory, MIT.

Jebara, T. (2004). *Machine Learning: Discriminative and Generative.* Kluwer Academic Publishers, Boston.

Jelinek, F. and Mercer, R. (1980). Interpolated estimation of Markov source parameters from sparse data. In *Proceedings of the Workshop on Pattern Recognition in Practice*, pages 381–397, North Holland, Amsterdam.

Jin, R., Yan, R., Zhang, J., and Hauptmann, A. (2003). A faster iterative scaling algorithm for conditional exponential model. In *Proceedings of the 20th International Conference on Machine Learning (ICML-2003)*, pages 282–289, Washington, DC.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning (ECML-1998)*, pages 137–142, Chemnitz, Germany.

Johansson, R. and Nugues, P. (2005). Sparse bayesian classification of predicate arguments. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 177–180, Ann Arbor, MI.

Johnson, M. (2001). Joint and conditional estimation of tagging and parsing models. In *Proceedings of the 39th Annual Conference on Association for Computational Linguistics (ACL-2001)*, pages 314–321, Toulouse, France.

Joshi, M., Kumar, V., and Agarwal, R. (2001). Evaluating boosting algorithms to classify rare classes: Comparison and improvements. In *Proceedings of the 1st IEEE International Conference on Data Mining (ICDM-2001)*, pages 257–264, San Jose, CA.

Jurafsky, D. (2004). *The handbook of Pragmatics*, chapter Pragmatics and Computational Linguistics, pages 578–604. Blackwell, MA.

Jurafsky, D., Shriberg, E., and Biasca, D. (1997). Switchboard SWBD-DAMSL shallow-discourse-function annotation coders manual. Technical Report 97–02, Institute of Cognitive Science, University of Colorado.

Kazama, J. and Tsujii, J. (2005). Maximum entropy models with inequality constraints: A case study on text categorization. *Machine Learning*, 60(1-3):159–194.

Kermanidis, K., Maragoudakis, M., Fakotakis, N., and Kokkinakis, G. (2004). Learning greek verb complements: Addressing the class imbalance. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-2004)*, pages 1065–1071, Geneva, Switzerland.

Kingsbury, P. and Palmer, M. (2002). From treebank to propbank. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, pages 1989–1993, Canary Islands, Spain.

Kipper, K., Dang, H.-T., and Palmer, M. (2000). Class-based construction of a verb lexicon. In *Proceedings of 17th National Conference on Artificial Intelligence and 12th Conference on Innovative Applications of Artificial Intelligence*, pages 691–696, Austin, TX.

Kita, K., Fukui, Y., Nagata, M., and Morimoto, T. (1996). Automatic acquisition of probabilistic dialogue models. In *Proceedings of the 4th International Conference on Spoken Language Processing (ICSLP-1996)*, pages 196–199, Philadelphia, PA.

Kubat, M. and Matwin, S. (1997). Addressing the curse of imbalanced training sets: One-sided selection. In *Proceedings of the 14th International Conference on Machine Learning (ICML-1997),*, pages 179–186, Nashville, Tennesse.

Kullback, S. (1959). *Information Theory and Statistics*. Wiley, New York.

Kupiec, J. (1992). Robust part-of-speech tagging using a hidden markov model. *Computer Speech and Language*, 6(3):225–242.

Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning (ICML-2001)*, pages 282–289, Williams, MA.

Lan, K.-C., Ho, K.-S., and Luk, R. W.-P. (2003). Accessing financial news using dialogues. In *Proceedings of the 8th International Conference on Applications of Natural Language to Information Systems (NLDB-2003)*, pages 155–167, Burg, Germany.

Lan, K.-C., Ho, K.-S., Luk, R. W.-P., and Leong, H.-V. (2004). Semantic role labeling using maximum entropy. In *Proceedings of the 1st International Symposium* *on Computational and Information Sciences (CIS-2004)*, pages 954–961, Shanghai, China.

Lan, K.-C., Ho, K.-S., Luk, R. W.-P., and Leong, H.-V. (2005). Fnds: A dialogue-based system for accessing digested financial news. *Journal of Systems and Software*, 78(2):180–193.

Lim, J.-H., Hwang, Y.-S., Park, S.-Y., and Rim, H.-C. (2004). Semantic role labeling using maximum entropy model. In *Proceedings of the 8th Conference on Computational Natural Language Learning (CONLL-2004)*, pages 122–125, Boston, MA.

Lin, C.-S. and Smith, T. C. (2005). Semantic role labeling via consensus in pattern-matching. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CONLL-2005)*, pages 185–188, Ann Arbor, MI.

Ling, C. and Li, C.-H. (1998). Data mining for direct marketing: Problems and solutions. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD-1998)*, pages 73–79, New York, NY.

Litman, D. and Allen, J. (1990). *Intentions in Communication*, chapter Discourse Processing and Commonsense Plans, pages 365–388. MIT Press, Cambridge.

Liu, T., Che, W.-X., Lia, S., and nd Huai-Jun Liu, Y.-X. H. (2005). Semantic role labeling system using maximum entropy classifier. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CONLL-2005)*, pages 189–192, Ann Arbor, MI.

Manning, C. (2004). Language learning: Beyond thunderdome. In *Proceedings of the 8th Conference on Computational Natural Language Learning (CONLL-2004)*, pages 138–138, Boston, MA.

Marcus, M., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2):313–330.

Màrquez, L., Comas, P., Giménez, J., and Català, N. (2005). Semantic role labeling as sequential tagging. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 193–196, Ann Arbor, MI.

McCallum, A., Freitag, D., and Pereira, F. (2000). Maximum entropy markov models for information extraction and segmentation. In *Proceedings of the 17th International Conference on Machine Learning (ICML-2000)*, pages 591–598, Stanford, CA.

McCullagh, P. and Nelder, J. (1989). *Generalized linear models*. Chapman and Hall, London.

Miller, D. and Yan, L. (2000). Approximate maximum entropy joint feature inference consistent with arbitrary lower-order probability constraints: Application to statistical classification. *Neural Computation*, 12(9):2175–2207.

Mitchell, T. (1997). *Machine Leaning*. McGraw Hill, New York.

Mitsumori, T., Murata, M., Fukuda, Y., Doi, K., and Doi, H. (2005). Semantic role labeling using support vector machines. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 197–200, Ann Arbor, MI.

Moschitti, A., Giuglea, A.-M., Coppola, B., and Basili, R. (2005). Hierarchical semantic role labeling. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 201–204, Ann Arbor, MI.

Ng, A. and Jordan, M. (2001). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Proceedings of 14th Conference on Advances in Neural Information Processing Systems (NIPS-2001)*, pages 841–848, Cambridge, MA.

Ng, V. and Cardie, C. (2002). Combining sample selection and error-driven pruning for machine learning of coreference rules. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-2002)*, pages 55–62, Philadelphia, PA.

Osuna, E., Freund, R., and Girosi, F. (1997). Training support vector machines: An application to face detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR–97)*, pages 130–136, San Juan, Puerto Rico.

Ozgencil, N. E. and McCracken, N. (2005). Semantic role labeling using libSVM. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 205–208, Ann Arbor, MI.

Park, K.-M. and Rim, H.-C. (2005). Maximum entropy based semantic role labeling. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CONLL-2005)*, pages 209–212, Ann Arbor, MI.

Pazzani, M., Merz, C., Ali, P., Hume, T., and Brunk, C. (1994). Reducing misclassification costs. In *Proceedings of the 11th International Conference of Machine Learning (ICML-1994)*, pages 217–225, New Brunswick, Morgan Kaufmann.

Phillips, S., Andersonb, R., and Schapired, R. (2006). Maximum entropy modeling of species geographic distributions. *Ecological Modelling*, 190(3-4):231–259.

Pietra, S. D., Pietra, V. D., and Lafferty, J. (1997). Inducing features of random fields. *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393.

Ponzetto, S. P. and Strube, M. (2005). Semantic role labeling using lexical statistical information. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CONLL-2005)*, pages 213–216, Ann Arbor, MI.

Pradhan, S., Hacioglu, K., Krugler, V., Ward, W., Martin, J., and Jurafsky, D. (2003). Support vector learning for semantic argument classification. Technical Report TR-CSLR-2003-03, The Center for Spoken Language Research, University of Colorado. Later appeared in Machine Learning, 60(1):11-39, 2005.

Pradhan, S., Hacioglu, K., Ward, W., Martin, J., and Jurafsky, D. (2005). Semantic role chunking combining complementary syntactic views. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 217–220, Ann Arbor, MI.

Pradhan, S., Ward, W., Hacioglu, K., Martin, J., and Jurafsky, D. (2004). Shallow semantic parsing using support vector machines. In *Proceedings of the 2nd Human Language Technology Conference/North American chapter of the Association for Computational Linguistics annual meeting (HLT/NAACL-2004)*, pages 233–240, Boston, MA.

Provost, F., Fawcett, T., and Kohavi, R. (1998). The case against accuracy estimation for comparing induction algorithms. In *Proceedings of the 15th International Conference on Machine Learning (ICML 1998)*, pages 445–453, Madison, WI.

Punyakanok, V., Koomenand, P., Roth, D., and Yih, W.-T. (2005a). Generalized inference with multiple semantic role labeling systems. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 181–184, Ann Arbor, MI.

Punyakanok, V. and Roth, D. (2000). The use of classifiers in sequential inference. In *Proceedings of 13th Conference on Advances in Neural Information Processing Systems (NIPS-2000)*, pages 995–1001.

Punyakanok, V., Roth, D., and Yih, W.-T. (2005b). The necessity of syntactic parsing for semantic role labeling. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-2005)*, pages 181–184, Edinburgh, Scotland, UK.

Punyakanok, V., Roth, D., Yih, W.-T., Zimak, D., and Tu, Y.-C. (2004). Semantic role labeling via generalized inference over classifiers. In *Proceedings of the 8th Conference on Computational Natural Language Learning (CoNLL-2004)*, pages 130–133, Boston, MA.

Rabiner, L. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–285.

Ramshaw, L. and Marcus, M. (1995). Text chunking using transformation-based learning. In *Proceedings of the 3rd Workshop on Very Large Corpora (WVLC-3)*, pages 82–94, Cambridge, MA.

Ratnaparkhi, A. (1996). A maximum entropy model for part-of-speech tagging. In *Proceedings of the 1st Conference on Empirical Methods in Natural Language Processing (EMNLP-1996)*, pages 133–142, Philadelphia, PA.

Ratnaparkhi, A. (1998). *Maximum Entropy Models for Natural Language Ambiguity Resolution*. PhD thesis, University of Pennsylvania.

Ratnaparkhi, A. (1999). Learning to parse natural language with maximum entropy models. *Machine Learning*, 34(1-3):151–175.

Reithinger, N., Engel, R., Kipp, M., and Klesen, M. (1996). Predicting dialogue acts for a speech-to-speech translation system. In *Proceedings of the 4th International Conference on Spoken Language Processing (ICSLP-1996)*, pages 654–657, Philadelphia, PA.

Reithinger, N. and Klesen, M. (1997). Dialogue act classification using language models. In *Proceedings of the 5th European Conference on Speech Communication and Technology (EuroSpeech-1997)*, pages 2235–2238, Rhodes, Greece.

Rosenfeld, R., Chen, S., and Zhu, X.-J. (2001). Whole-sentence exponential language models: a vehicle for linguistic-statistical integration. *Computer Speech and Language*, 15(1):55–73.

Rubinstein, D. and Hastie, T. (1997). Discriminative vs. informative learning. In *Proceedings of the 3th International Conference on Knowledge Discovery and Data Mining (KDD-1997)*, pages 49–53, NewPort Beach, CA.

Saeed, J. (1997). *Semantics*. Blackwell Publishers, Oxford.

Sahami, M., Dumais, S., Heckerman, D., and Horvitz, E. (1998). A bayesian approach to filtering junk e-mail. In *Proceedings of the Workshop on Text Categorization at AAAI-1998*, pages 55–62, Madison, WI.

Samuel, K., Carberry, S., and Vijay-Shanker, K. (1998). Dialogue act tagging with transformation-based learning. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics (COLING/ACL-1998)*, pages 1150–1156, Montreal, Canada.

Schofield, E. (2004). Fast parameter estimation for joint maximum entropy language models. In *Proceedings of the 8th International Conference on Spoken Language Processing (ICSLP-2004)*, pages 2241–2244, Jeju, Korea.

Searle, J. R. (1969). *Speech Acts*. Cambridge Univ. Press, New York.

Sha, F. and Pereira, F. (2003). Shallow parsing with conditional random fields. In *Proceedings of the 2003 Human Language Technology Conference and North American Chapter of the Association for Computational Linguistics (HLT/NAACL-2003)*, pages 213–220, Edmonton, Canada.

Shore, J. and Johnson, R. (1980). Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy. *IEEE Transaction on Information Theory*, 26(1):26–36.

Shriberg, E. (1996). Disfluencies in SWITCHBOARD. In *Proceedings of the 4th International Conference on Spoken Language Processing (ICSLP-1996)*, pages 11–14, Philadelphia, PA.

Solberg, A. and Solberg, R. (1996). A large-scale evaluation of features for automatic detection of oil spills in ers sar images. In *Proceedings of International Geoscience and Remote Sensing Symposium (IGARSS-1996)*, pages 1484–1486, Lincoln, NE.

Stolcke, A., Ries, K., Coccaroa, N., Shriberg, E., Bates, R., Jurafsky, D., Taylor, P., Martin, R., Ess-Dykema, C. V., and Meteer, M. (2000). Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):339–373.

Surdeanu, M., Harabagiu, S., Williams, J., and Aarseth, P. (2003). Using predicate-argument structures for information extraction. In *Proceedings of the 41th Annual Meeting on Association for Computational Linguistics (ACL-2003)*, pages 8–15, Sapporo, Japan.

Surdeanu, M. and Turmo, J. (2005). Semantic role labeling using complete syntactic analysis. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 221–224, Ann Arbor, MI.

Sutton, C. and McCallum, A. (2005). Joint parsing and semantic role labeling. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 225–228, Ann Arbor, MI.

Taylor, A. (1995). Dysfluency annotation stylebook for the switchboard corpus. Technical report, Department of Computer and Information Science, University of Pennsylvania.

Taylor, P., King, S., Isard, S., and Wright, H. (1998). Intonation and dialogue context as constraints for speech recognition. *Language and Speech*, 41(3):493–512.

Thompson, C., Levy, R., and Manning, C. (2003). A generative model for semantic role labeling. In *Proceedings of the 14th European Conference on Machine Learning (ECML-2003)*, pages 397–408, Dubrovnik, Croatia.

Tjong Kim Sang, E. and Buchholz, S. (2000). Introduction to the conll-2000 shared task: Chunking. In *Proceedings of the 4th Conference on Computational Natural Language Learning (CONLL-2000)*, pages 127–132, Lisbon, Portugal.

Tjong Kim Sang, E., Canisius, S., denBosch, A., and Bogers, T. (2005). Applying spelling error correction techniques for improving semantic role labelling. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CONLL-2005)*, pages 229–232, Ann Arbor, MI.

Tjong Kim Sang, E. and Déjean, H. (2001). Introduction to the conll-2001 shared task: Clause identification. In *Proceedings of the 5th Conference on Computational Natural Language Learning (CONLL-2001)*, pages 53–57, Toulouse, France.

Tjong Kim Sang, E. and Veenstra, J. (1999). Representing text chunks. In *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics (ECAL-1999)*, pages 173–179. Bergen, Norway.

Tomek, I. (1976). Two modifications of cnn. *IEEE Transactions on Systems, Man and Cybernetics*, 6:769–772.

Traum, D. (1999). *Foundations of Rational Agency*, chapter Speech Acts for Dialogue Agents, pages 169–202. Kluwer Academic Publishers, Boston.

Traum, D. and Allen, J. (1992). A speech acts approach to grounding in conversation. In *Proceedings of the 2nd International Conference on Spoken Language Processing (ICSLP-1992)*, pages 137–140, Banff, Canada.

Tsai, T.-H., Wua, C.-W., Lin, Y.-C., and Hsu, W.-L. (2005). Exploiting full parsing information to label semantic roles using an ensemble of me and svm via integer linear programming. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 233–236, Ann Arbor, MI.

Turing, A. (1950). Computing machinery and intelligence. *Mind*, 59(236):433–460.

Turney, P. (2000). Types of cost in inductive concept learning. In *Proceeding of the Workshop on Cost-Sensitive Learning at the 17th International Conference on Machine Learning (ICML-2000)*, pages 15–21, Stanford, CA.

Ulusoy, I. and Bishop, C. (2005). Generative versus discriminative methods for object recognition. In *Proceedings of the 2005 IEEE International Conference on Computer Vision and Pattern Recognition (CVPR-2005)*, San Diego, CA.

van den Bosch, A., Canisius, S., Daelemans, W., Hendrickx, I., and Sang, E. T. K. (2004). Memory-based semantic role labeling: Optimizing features, algorithm, and output. In *Proceedings of the 8th Conference on Computational Natural Language Learning (CONLL-2004)*, pages 102–105, Boston, MA.

Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.

Viterbi, A. (1967). Error bounds for convolutional codes and asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269.

Wallach, H. (2004). Conditional random fields: An introduction. Technical Report MS-CIS-04-21, Department of Computer and Information Science, University of Pennsylvania.

Weizenbaum, J. (1966). Eliza - a computer program for the study of natural language communication between man and machine. *Communications of the Association for Computing Machinery (ACM)*, 9(1):36–45.

Wolpert, D. (1992). Stacked generalization. *Neural Network*, 5(2):241–259.

Wright, H. (1998). Automatic utterance type detection using suprasegmental features. In *Proceedings of the 5th International Conference on Spoken Language Processing (ICSLP-1998)*, pages 1403–1406, Sydney Australia.

Xue, N. and Palmer, M. (2004). Calibrating features for semantic role labeling. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP-2004)*, pages 88–94, Barcelona, Spain.

Yi, S.-T. and Palmer, M. (2005). The integration of syntactic parsing and semantic role labeling. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CONLL-2005)*, pages 237–240, Ann Arbor, MI.

Zadrozny, B. and Elkan, C. (2001). Learning and making decisions when costs and probabilities are both unknown. In *Proceedings of the 7th ACM SIGKDD Inter-*

national Conference on Knowledge Discovery and Data Mining (KDD-2001), pages 204–213, San Francisco, CA.

Zelenko, D., Aone, C., and Richardella, A. (2003). Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106.