NAIST-IS-DD0361217

**Doctoral Dissertation**

# Unknown Word Identification for Chinese Morphological Analysis

Chooi-Ling Goh

September 29, 2006

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

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

Chooi-Ling Goh

Thesis Committee:

| | |
|---|---|
| Professor Yuji Matsumoto | (Supervisor) |
| Professor Kiyohiro Shikano | (Co-Supervisor) |
| Professor Shunsuke Uemura | (Member) |
| Associate Professor Kentaro Inui | (Member) |

# Unknown Word Identification for Chinese Morphological Analysis*

Chooi-Ling Goh

**Abstract**

Since written Chinese does not use blank spaces to indicate word boundaries, segmenting Chinese texts becomes an essential task for Chinese language processing. Besides word segmentation, we also need to identify the part-of-speech (POS) tags of the words. The segmentation and POS tagging process are denoted as morphological analysis. During the process of word segmentation, two main problems occur: segmentation ambiguities and unknown word occurrences. There are basically two types of segmentation ambiguities: covering ambiguity and overlapping ambiguity. These ambiguities are dealt with known words. For the unknown word problem, we need to detect them from the text based on the context. In this report, we have focused on the problem of unknown words and proposed some machine-learning based methods towards solving it. Besides, we also face the ambiguity problem with POS tagging because a single word can hold multiple POS tags and it depends on the context to decide which one is the correct answer. Furthermore, if the word is unknown, then we need to guess the POS tag based on the word components and contexts.

At the end of the research, we have built a practical morphological analyzer which can be freely used by anyone for research purpose. In order to build a practical system, a reasonable size dictionary is needed. The initial dictionary is built from the Penn Chinese Treebank corpus v4.0 and contains only 33,438 entries. Since the initial dictionary is quite small, the unknown word detection method is applied to huge raw texts in order to extract new words to be added into the system dictionary. We have successfully constructed a dictionary with 120,769 entries. Finally, we propose a two-layer morphological analysis to cater for two sets of outputs. The first layer produces the minimal segmentation unit

*Doctoral Dissertation, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD0361217, September 29, 2006.

defined by us, and the second layer transforms the first layer output to the original segmentation unit defined by Penn Chinese Treebank.

**Keywords:**

Chinese, segmentation, POS tagging, unknown words, morphological analysis, dictionary, machine learning

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Chinese Language Processing

The very first problem faced in Chinese language processing is said to be word segmentation. It is because there is no indicators such as blank spaces to show the word boundaries in Chinese text. The same phenomenon does not happen only to Chinese language but also many other Asia languages such as Japanese, Arabic and Thai. Therefore, in order to understand the Chinese text, the first thing that we need to do is to cut the sentences into word segments. Although it sounds easy to cut a sentence into a word sequence, however, from the past experience, we know that it is not a trivial task.

The characteristics of Chinese language have made the segmentation problem more difficult than other languages. In Japanese, characters are divided into three types, which are hiragana, katakana and kanji. These different types of characters can help in telling where are the word boundaries. In languages such as Arabic, the form changes according to the location of the character in a word. Generally, there are three forms for each character which show the location at the first, in the middle and at the last positions in a word. These different forms can become some clues to show the word boundaries too. However, there is no clue to indicate where the word boundaries are in Chinese texts as there is only one single type of characters that is the hanzi and only one single form for each word. There are only some punctuation marks which can tell the sentence or phrase boundaries.

Chinese word segmentation has been put into focus in the past decade, along with the high demand on various natural language processing systems, such as machine translation and information retrieval. Researchers realize the importance

of word segmentation in order to develop high performance systems. In the word segmentation task, segmentation ambiguity and unknown word are the two main problems. Together with word segmentation, part-of-speech tagging is also an important task. We will discuss these two problems deeply in this dissertation.

## 1.2 Chinese Word Segmentation Bakeoff

The first Chinese word segmentation bakeoff was carried out in Second SIGHAN[1] Workshop in year 2003[2]. The purpose is to compare the accuracy of various methods [37]. As far as we know, there is no standard definition of Chinese word segmentation. A text can be segmented differently depending on the linguists who decide on the rules and also the purpose of the segmentation. Sproat et al. [38] described the importance of segmentation for a text-to-speech system and Wu and Tseng [47] discussed the role of segmentation for information retrieval. Each of them has defined the segmentation in their own standard. Therefore, it is always difficult to compare the results obtained with different methods as the data used in experiments are different. Therefore, this bakeoff intended to standardize the training and testing corpora, so that a fair evaluation could be made. There are two tracks in the bakeoff: open and closed. In the open track, the participants are allowed to use any other resources such as dictionaries or more training data in their system besides the training materials provided. However, in the closed track, the condition is somehow strict, no other material other than the training data provided is allowed to train the system.

The details of the training materials are shown in Table 1.1. There are four tracks of data provided by different institutions. PKU stands for the Peking University Corpus, CTB stands for the Penn Chinese Treebank, AS stands for Academia Sinica Corpus and HK stands for Hongkong City University Corpus. PKU and CTB are simplified Chinese texts (in GB code) while AS and HK are traditional Chinese texts (in Big5 code). The sizes of the training and testing data vary by each track. Therefore, the unknown word (also referred as out-of-vocabulary words, OOV) rates are also different. We can assume that the higher the unknown word rate, the harder the task of segmentation. In this bakeoff, CTB has the highest unknown word rate and AS has the lowest.

[1] A Special Interest Group of the Association of Computational Linguistics, http://www.sighan.org/.

[2] The second bakeoff was carried out in year 2005 but most of the experiments conducted in this research use only the data provided in the first bakeoff.

| Corpus | # of train words | # of test words | Unknown word rate (token) |
|---|---|---|---|
| PKU | 1.1M | 17,194 | 6.9% |
| CTB | 250K | 39,922 | 18.1% |
| AS | 5.8M | 11,985 | 2.2% |
| HK | 240K | 34,955 | 7.1% |

Table 1.1. SIGHAN bakeoff data

The results of SIGHAN bakeoff are evaluated in five measurements: recall, precision and F-measure for overall segmentation, and recall for unknown words and known words, as shown in the equations below.

$$
\begin{aligned}
Recall &= \frac{number\ of\ correctly\ segmented\ words}{total\ number\ of\ words\ in\ gold\ data} \\
Precision &= \frac{number\ of\ correctly\ segmented\ words}{total\ number\ of\ words\ segmented} \\
F\text{-}measure &= \frac{2 \times Recall \times Precision}{Recall + Precision} \\
Recall(OOV) &= \frac{number\ of\ correctly\ segmented\ unknown\ words}{total\ number\ of\ unknown\ words\ in\ gold\ data} \\
Recall(IV) &= \frac{number\ of\ correctly\ segmented\ known\ words}{total\ number\ of\ known\ words\ in\ gold\ data}
\end{aligned}
$$

The bakeoff results are summarized in Table 1.2. We show only the results on overall segmentation F-measure and unknown word recall for both open and closed tracks. Following our assumption, one gets better results if there are less unknown words in the test data, such as AS. We also observe that if one can get good recall for unknown words, the overall segmentation is better too.

| | Closed | | Open | |
|---|---|---|---|---|
| Corpus | F-measure (seg) | Recall (OOV) | F-measure (seg) | Recall (OOV) |
| PKU | 0.894–0.951 | 0.159–0.763 | 0.886–0.959 | 0.503–0.799 |
| CTB | 0.732–0.881 | 0.076–0.705 | 0.829–0.912 | 0.578–0.766 |
| AS | 0.938–0.961 | 0.043–0.729 | 0.872–0.904 | 0.236–0.426 |
| HK | 0.901–0.940 | 0.243–0.670 | 0.879–0.956 | 0.579–0.788 |

Table 1.2. Bakeoff results

In the following chapters, we will refer to this bakeoff as SIGHAN bakeoff. We will use the bakeoff data in some of our experiments so that we can make a comparison with the others.

## 1.3 Chinese Morphological Analysis

Morphological analysis of a language is more complicated than what we think. For a language with morphological changes such as inflection, we might want to restore their original forms by the process of stemming. Then, of course we want to identify the part-of-speech (hereafter POS) tag for each word in the text. For a language written without word boundaries like Chinese and Japanese, the first thing that we have to do is to segment the sentence into a word sequence. Then only we POS tag the words[3]. In Japanese, the original form is restored if the word is inflected.

In Tseng and Chen [40], a morphological analyzer for Chinese is designed. Their task is to automatically analyze the morphological structures of compounds words. The morphological structures of compound words contain essential information regarding their syntactic and semantic characteristics. According to their study, this is the primary step for predicting the categories of unknown words. The system takes a compound word as an input and produces the morphological structure of the word. The major steps are: (1) to segment the word into a sequence of morphemes, (2) to tag the POS of morphemes, and (3) to identify the morpho-syntactic relation between the morphemes.

Besides the known words in the dictionary, there are five types of highly productive words (cf. unknown words). The abbreviations and proper names are without semantic transparency and are hardly to be identified based on their internal components. However, morphological derived words and compound words are semantically transparent. In other words, their meanings or categories can be interpreted by their internal morpheme components. The last type is numeric type compounds, which can be easily identified using regular expression and is not really a big issue.

1. abbreviation (acronym): e.g. '中日韩' (China/Japan/Korea).

2. proper names (person name, place name, company name): e.g. 江泽民(Jiang Zemin (person name)), 槟城(Penang, an island in Malaysia (place

[3]Segmentation and POS tagging can also be performed simultaneously.

name)), 微软(Microsoft (company name)).

3. derived words (those words with affixes): e.g. 总经理(General Manager), 电脑化(computerized).

4. compounds: e.g. 获允(receive permission), 泥沙(mud), 电脑桌(computer desk).

5. numeric type compounds: e.g. １８．７％(18.7%), 三千日圆(3 thousands Japanese yen), ２００３年(year 2003).

There are also other lower productive word pattern such as reduplication and parallel words. There are a lot of Chinese words which can be reduplicated to form new words. There are basically seven types of reduplication patterns [51, 40].

1. A to AA: eg. 走走/v (to walk), 听听/v (to listen), 厚厚/z (thick), 尖尖/z (sharp)

2. AB to AAB: eg. 挥挥手/v (to wave hand), 试试看/v (to try)

3. AB to ABB: eg. 孤单单/z (alone, lonely), 一阵阵/m (classifier for wind)

4. AB to AABB: eg. 整整齐齐/z (tidily), 比比划划/v (to compete), 日日夜夜/d (days and nights)

5. AB to A(X)AB: eg. 马里马虎/z (careless), 相不相信/v, (believe or not), 漂不漂亮/z (pretty or not)

6. AB to ABAB: eg. 比划/v 比划/v (to compete), 很多/m 很多/m (a lot), 一个/m 一个/m (each of them), 哗啦/o 哗啦/o (onomatopoeia, the sound of rain)

7. A(X*)A: eg. 谈/v 一/m 谈/v (to discuss), 想/v 了/u 想/v (to think), 读/v 了/u 一/m 读/v (to read)

Normally, the form A or AB are known words, but the newly generated patterns are unknown words. Out of these seven types of patterns, only the pattern numbers 6 and 7 are easily recognized as they are further segmented into the dictionary units. However, the rest cannot be detected easily as they are considered as one single unit. This type of unknown words probably can only be detected by introducing some morphological rules.

The parallel pattern of words are as ABC, which is actually formed by AC and BC, or AB and AC. For example, "中小学" (secondary and primary school) is composed by "中学" (secondary school) and "小学" (primary school), "国内外" (domestic and abroad) is composed by "国内" (domestic) and "国外" (abroad). However, this type of words do not appear so frequently in text and the words that can be used to form parallel words are also limited.

In Chinese, the definition of words is somehow arbitrary. A character (or morpheme) can be a word. A group of characters can also form a word. The very first Chinese concordance system proposed by Uemura [44] used character as the basic for retrieving concordances in Chinese. Compared to its Japanese version in [43], the concordances for Chinese do not work on words with more than one character. The various types of characters in Japanese, such as hiragana, katakana and kanji, have provided some clues to segment a sentence in Japanese into words. However, these clues cannot be applied to Chinese as Chinese has only one type of characters. This history of Chinese language processing has shown the needs of defining words in Chinese and the needs to segment Chinese texts into words.

Linguistically, a word is defined as a minimal unit that can function independently. However, in the real life, different group of people will interpret a word differently, according to their definition, usage and pratice. Therefore, it is quite difficult to know what should be done in Chinese morphological analysis. The first question is, what should be the size of a word? How to deal with compound words? Besides segmentation and POS tagging, do we need to do more? Do we want to know the components of a word? How is the word formed? Is it a compound word or a morphological derived word? These are some of the questions that may arise when one talks about morphological analysis. Currently the most important issue is only segmentation and POS tagging that based on some predefined rules. Therefore, although we call our system a morphological analyzer, we actually only want to segment and POS tag the text, without knowing the morpho-syntactic structure of the words. However, our system can definitely be expanded to cater for the needs of "real" morphological analysis in the future.

In Chapter 5, we propose a two-layer morphological analyzer for Chinese. Our initial intension is to build a system that will analyze the texts into minimal unit segmentation based on a dictionary in the first layer and combine the minimal unit to form larger unit such as named entities, compound words, etc in the second layer. However, it is still not clear what is the minimal unit segmentation. What should be included in a dictionary? What kind of words are considered as

compound words? Therefore, our initial stage is to collect as much as possible words to register in our dictionary. Then in a later stage, we can decide whether or not to break a larger unit word into smaller ones.

Therefore, the first layer will not handle regular pattern such as numbers and foreign words. These are the words that can be easily detected using regular expression. The dictionary only contains the minimal set of characters of numbers and alphabets. Secondly, the combination of Chinese person names (also Japanese and Korean names) is almost uncountable. In most of the corpus provided, the names are as one unit. However, following our minimal unit definition, we want to break up a name into a family name and a given name, which is easier to control and also easier to be combined in the second layer.

In the future we would like to adopt the analyzer to be able to analyze morphologically derived words and compound words. For example, the first layer will produce "朋友/NN 们/M" , "副/JJ 首相/NN" and "研究/NN 室/NN", and the second layer will combine those into "朋友们/NN" (friends), "副首相/NN" (deputy prime minister) and "研究室/NN" (research laboratory).

Currently even linguists have the difficulty in deciding what should be contained in a dictionary. For example, if we say that "牛肉" (beef) should be consider a word, how about "鹿肉" (deer's meat) and "猴肉" (monkey's meat)? All these words have similar structure (an animal name plus meat), but "牛肉" is commonly used and with high frequency, "鹿肉" is sometimes used with moderate frequency and "猴肉" is seldom used but is a possible word in real text. Therefore, the consistency of segmentation unit is yet to be defined more precisely in the future.

Besides the problem of the definition of words in Chinese, we also face the problem of assigning a POS tag to a word. In Chinese, there is no morphological changes. There is no inflection on words to show their functionalities. A word can be a noun or a verb without any changes. For example, CTB defines "爱国" (patriotic) with four POS tags: adjective, noun, verb, and person name[4]. Therefore, we can only decide the POS tag of a word on the text level, meaning by looking at the context.

[4]A common word in Chinese can be used as proper name as well.

## 1.4 Problem Setting

Our purpose for this research is to build a Chinese morphological analyzer. We define the morphological analysis as word segmentation and POS tagging only. We leave the analysis of the structure of words as future work. Prior to doing this, we need to study the problems occurring in word segmentation, unknown word identification, and POS tagging. We will follow this direction and analyze each problem in detail. Finally, we propose a framework of morphological analyzer based on the Penn Chinese Treebank standard.

### 1.4.1 Word Segmentation and POS Tagging

Before one can work on unknown word detection, the first thing that we need is a model that can correctly segment and POS tag known words. Since these are the known words, they can be found in the system dictionary. Although it sounds simple to cut a sentence into words but unfortunately there exist ambiguities during this operation. A string of characters may be segmented into different words according to the contexts. A word can hold more than one POS tag based on the usage. Chapter 3 discusses about this problem and proposes some solutions towards it. Two methods are proposed. The first one uses Support Vector Machines to label each character with position tags. These position tags tell the word boundaries. We propose using the information from a dictionary as the features in the training of Support Vector Machines which is a new idea along this line. The second method is based on Hidden Markov Models. The word and tag sequence is determined by Viterbi algorithm, where the highest probability path is selected. The results of this chapter serve as a baseline for unknown word detection which is the main research topic of this dissertation.

### 1.4.2 Unknown Word Extraction and POS Tag Guessing

After we have the initial segmentation and POS tagging for known words, in Chapter 4, we will tackle the problem of unknown words. Unknown words are words not found in the system dictionary. As a language evolves, a fixed entry dictionary will never be complete. Therefore, we always need to collect some new words from the text, from time to time, in order to keep up-to-date the words in the dictionary. There are a few approaches to detect unknown words. We can either do it word-based or character-based. Word-based approach normally gives us higher precision with character-based approach gives us better recall.

Secondly, should unknown word detection be part of the whole process, meaning it is processed together with known word disambiguation, or should it be done before or after known word segmentation. Again, if unknown word detection is done together with known word segmentation, the recall of unknown word will be higher but false unknown word will deteriorate the accuracy of known word segmentation. Finally, the distribution of types of unknown words is different depending on whether we use a proper dictionary or not in our system. Our conclusion is that a proper dictionary is a more natural way to morphological analysis. Therefore, we suggest to enlarge the system dictionary using unknown word detection methods to be used in our final system. Our unknown word detection methods are based on character-based tagging, with suitable set of features, using machine-learning-based methods such as Support Vector Machines and Maximum Entropy Models.

### 1.4.3 Two-layer Morphological Analyzer

At the end of the research, we have built a proper dictionary using unknown word detection methods, with only valid words in it. We have enlarged the initial dictionary from 33,438 entries to 120,769 entries. The construction of the dictionary is still an ongoing process. In Chapter 5, we propose a two-layer morphological analyzer which caters for two sets of outputs with different level of segmentation units. Currently the two-level outputs only work on CJK person names, numbers, time nouns, and alphabet words only. However, the design enables us to further apply to compound words and morphological derived words. The first layer of the analysis is using Hidden Markov Models and the second layer is using Support Vector Machines.

## 1.5 Organization of the Dissertation

The organization of the dissertation is as below. Chaper 2 gives a brief introduction to all the machine-learning methods that we apply to our research. Chapter 3 introduces the problems of word segmentation in Chinese and proposes some methods towards solving them. Chapter 4 focuses on the problems of unknown word identification, including detection and POS tag guessing. A few approaches are discussed in this chapter. Chapter 5 introduces our approach towards building a Chinese morphological analyzer based on the previous studies. Finally, Chapter 6 concludes the work and suggests some future work for improvement.

# Chapter 2

# Machine Learning-based Methods

In this chapter, we will describe some probabilistic models that will be used throughout the research. The four current state-of-the-art models, i.e., Hidden Markov Models, Support Vector Machines, Maximum Entropy Models and Conditional Random Fields are briefly described. They are all capable of labeling sequential data and classification, which solve a lot of problems in natural language processing, such as segmentation, POS tagging, base-phrase chunking and named-entity recognition.

## 2.1 Hidden Markov Models

Markov Models have been applied in part-of-speech tagging for English texts. Since English texts consist of blank spaces to indicate the word boundaries, the only problem is to assign the POS tags. However, for languages such as Chinese and Japanese, having no spaces to mark the word boundaries, segmentation of words and identification of POS tags must be done simultaneously. We need to modify the original model to suit for this purpose. We will now describe the Hidden Markov Models (hereafter HMM) in detail in the following.

Let $S$ be the given sentence (sequence of characters) and $S(W)$ be the sequence of characters that composes the word sequence $W$. POS tagging is defined as the determination of the POS tag sequence, $T = t_1, \ldots, t_n$, if a segmentation into a word sequence $W = w_1, \ldots, w_n$ is given. The goal is to find the POS sequence $T$ and word sequence $W$ that maximize the following probability:

$$
\begin{aligned}
W, T &= \arg \max_{W,T,S(W)=S} P(T, W|S) \\
&= \arg \max_{W,T,S(W)=S} P(W, T) \\
&= \arg \max_{W,T,S(W)=S} P(W|T)P(T)
\end{aligned}
$$

We make the following approximations that the tag probability, $P(T)$, is determined by the preceding tag only and that the conditional word probability, $P(W|T)$, is determined by the tag of the word. HMMs assume that each word is generated a hidden state which is the same as the POS tag of the word. A tag $t_{i-1}$ transits to another tag $t_i$ with the probability $P(t_i|t_{i-1})$, and outputs a word with the probability $P(w_i|t_i)$. Then the approximation for both probabilities can be rewritten as follows.

$$
\begin{aligned}
P(W|T) &\triangleq \prod_{i=1}^{n} P(w_i|t_i) \\
P(T) &\triangleq \prod_{i=1}^{n} P(t_i|t_{i-1})
\end{aligned}
$$

The probabilities are estimated from the frequencies of instances in a tagged corpus using Maximum Likelihood Estimation. $F(X)$ is the frequency of instances in the tagged corpus, $\langle w_i, t_i \rangle$ shows the co-occurrences of a word and a tag, and $\langle t_i, t_{i-1} \rangle$ shows the co-occurrences of two tags.

$$
\begin{aligned}
P(w_i|t_i) &= \frac{F(\langle w_i, t_i \rangle)}{F(t_i)} \\
P(t_i|t_{i-1}) &= \frac{F(t_i, t_{i-1})}{F(t_{i-1})}
\end{aligned}
$$

The possible segmentation of a sentence can be represented by a lattice, as shown in Figure 2.1. The nodes in the lattice show possible word segments together with the POS tags. With the estimated parameters, the most probable tag and word sequence are determined using the Viterbi algorithm. In practice, negated log likelihood of $P(w_i|t_i)$ and $P(t_i|t_{i-1})$ is calculated as the cost. Maximizing the probability is equivalent to minimizing the cost. In this example, the correct path is marked by bold lines.

This POS tagger is only able to segment and POS tag known words that can be found in the dictionary. If some words are not found in the dictionary, they

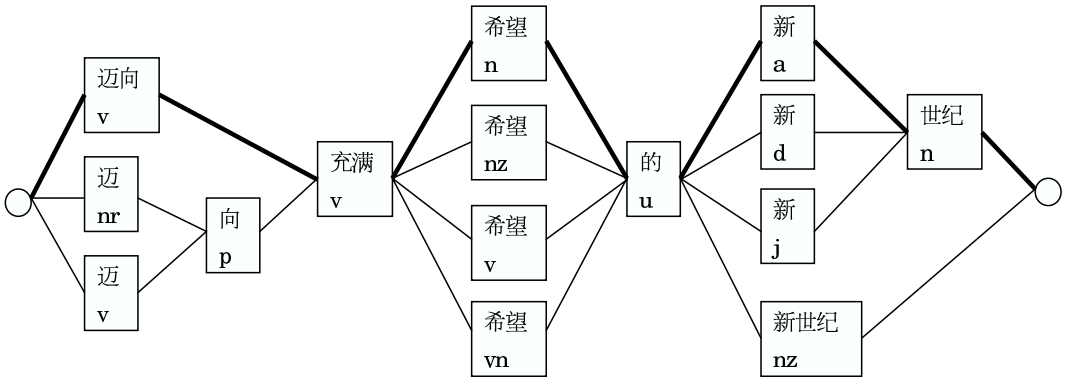



Figure 2.1. Example of a lattice using HMM

will be segmented accordingly, depending on the parts of words that can be found in the dictionary. Therefore, the unknown words detection need to be done in a separate process.

*ChaSen*[1] is a widely used morphological analyzer for Japanese texts [28] based on Hidden Markov Models. It achieves over 97% precision for newspaper articles. We customize it to suit our purpose for Chinese segmentation and POS tagging. We will describe the application of *ChaSen* for Chinese in more detail in Section 5.

## 2.2 Support Vector Machines

Support Vector Machines (hereafter SVM) [45] are binary classifiers that search for hyperplanes with the largest margin between positive and negative samples. Suppose we have a set of training data for a binary classification problem: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$, where $\mathbf{x}_i \in R^n$ is a feature vector of the $i$ th sample in the training data and $y_i \in \{+1, -1\}$ is the label of the sample. The goal is to find a decision function which accurately predicts $y$ for an unseen $\mathbf{x}$. An SVM classifier gives a decision function $f(\mathbf{x})$ for an input vector $\mathbf{x}$ where

$$f(\mathbf{x}) = sign\left(\sum_{\mathbf{z}_i \in SV} \alpha_i y_i K(\mathbf{x}, \mathbf{z}_i) + b\right).$$

$f(\mathbf{x}) = +1$ means that $\mathbf{x}$ is a positive member, and $f(\mathbf{x}) = -1$ means that $\mathbf{x}$

[1] http://chasen.naist.jp

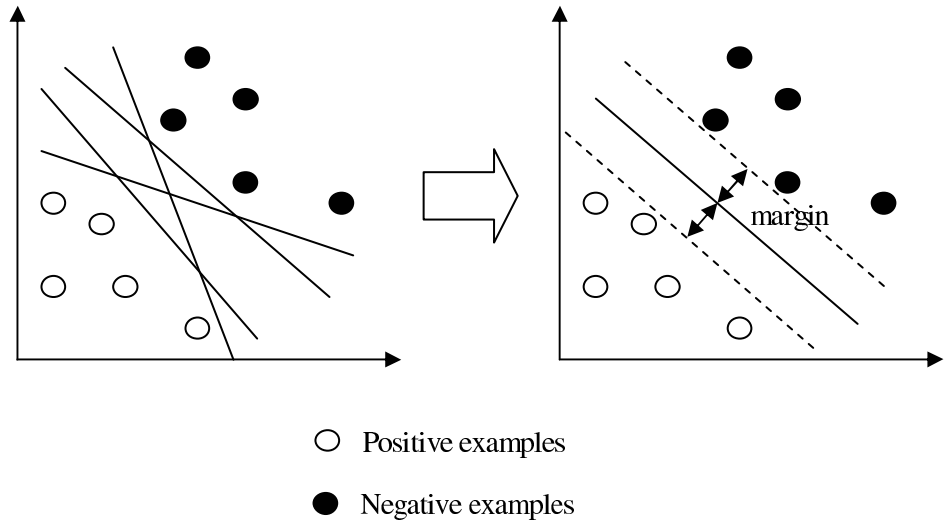



Figure 2.2. Maximize the margin in SVM

is a negative member. The vectors $\mathbf{z}_i$ are called support vectors, which receive a non-zero weight $\alpha_i$. Support vectors and the parameters are determined by solving a quadratic programming problem. $K(\mathbf{x}, \mathbf{z})$ is a kernel function which maps vectors into a higher dimensional space. We use a polynomial kernel of degree 2 given by $K(\mathbf{x}, \mathbf{z}) = (1 + \mathbf{x} \cdot \mathbf{z})^2$.

*YamCha*[2] [21] is a multi-purpose chunker. It extends binary classification to n-class classification because for natural language processing purposes, we would normally want to classify into several classes, such as in the case for POS tagging or base phrase chunking. Mainly two straightforward methods are used for this extension, the "one-vs-rest method" and the "pairwise method". In the "one-vs-rest method", $n$ binary classifiers compare one class with the rest of the classes. In the "pairwise method", we use $\binom{n}{2}$ binary classifiers, between all pairs of classes. Details of the system can be found in [21], used for base phrase chunking. We will use *YamCha* as our chunker for various purposes that will be explained later whenever used (Section 3.2.2, 4.4, 5.4.2).

## 2.3 Maximum Entropy Models

The Maximum Entropy Models (hereafter ME) that we use in our research is similar to the one proposed by Ratnaparkhi [34] for POS tagging of English. ME

[2]http://chasen.org/~taku/software/yamcha/

models have been widely used in many tasks in natural language processing and proved to be effective in these tasks.

In ME, the joint probability of a history $h$ and a tag $t$ is defined as:

$$p(h,t) = \pi \prod_{j=1}^{k} \alpha_j^{f_j(h,t)}$$

where $\pi$ is a normalization constant, $\alpha_1, \ldots, \alpha_k$ are the positive model parameters and $f_1, \ldots, f_k$ are known as "feature funtions", where $f_j(h,t) \in \{0,1\}$. Each parameter $\alpha_j$ corresponds to a feature function $f_j$. Given a large scale POS tagged corpus as a training data, the parameters $\{\alpha_1, ..., \alpha_k\}$ are chosen to maximize the likelihood of the training data using $p$:

$$L(p) = \prod_{i=1}^{n} p(h_i, t_i) = \prod_{i=1}^{n} \pi \prod_{j=1}^{k} \alpha_j^{f_j(h_i,t_i)}$$

In practice, the parameters can be estimated using Generalized Iterative Scaling (GIS) or Improved Iterative Scaling (IIS) algorithms. In this implementation, limited memory quasi-Newton method [32] is used because it is able to find the optimal parameters for the model much faster than the iterative scaling methods. The word and the tag context available to the features are as in the following definition of a history $h_i$:

$$h_i = \{t_{i-2}, t_{i-1}, w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}\}$$

For example,

$$f_j(h_i, t_i) = \begin{cases} 1, & \text{if } t_{i-1} = \text{“n”} \\ 0, & \text{otherwise} \end{cases}$$

The above feature actually says that if the previous tag equals to "n" (noun), then it is true, otherwise, false. In practice, we need to define the feature templates to be used in scanning each pair of $(h_i, t_i)$ in the training data. A possible feature template can be:

1. $w_x$ $(x = i-2, i-1, i, i+1, i+2)$
2. $t_{i-2}$, $t_{i-1}$
3. $w_x w_{x+1}$ $(x = i-2, i-1, i, i+1)$

4. $t_{i-2}t_{i-1}$

5. length($w_i$) - length of the word

6. prefix($w_i$) - prefix of the word (In English, it can be the first/first two/first three character(s) of the word. In Chinese, only the first character is considered)

7. suffix($w_i$) - suffix of the word (In English, it can be the last/last two/last three character(s) of the word, In Chinese, only the last character is considered)

8. punct($w_i$) - whether the word is a punctuation mark

These parameters and features are used to calculate the probability of testing data. Given a word $w$ and the history $h$, the tagger searches for the tag $t$ with the highest conditional probability

$$\begin{aligned} p(t|w) &= p(t|h) \\ &= \frac{p(h,t)}{\sum_{t' \in T} p(h,t')} \end{aligned}$$

where $T$ is the set of all possible POS tags. The ME is used as MEMM (Maximum Entropy Markov Model) where the conditional probability is applied step by step fo getting the best sequence of tags.

## 2.4 Conditional Random Fields

Conditional Random Fields [22] (hereafter CRF) are undirected graphical models trained to maximize a conditional probability of the whole graph structure. A common case of a graph structure is a linear chain, which corresponds to a finite state machine, and is suitable for sequence labeling. A linear-chain CRF with parameters $\Lambda = \{\lambda_1, ..., \lambda_K\}$ defines a conditional probability for a label sequence $\mathbf{y} = y_1...y_T$ given an input sequence $\mathbf{x} = x_1...x_T$ to be:

$$P_\Lambda(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_x} \exp\left(\sum_{t=1}^{T} \sum_k \lambda_k f_k(y_{t-1}, y_t, \mathbf{x})\right)$$

where $Z_x$ is the normalization factor that makes the probability of all state sequences sum to one; $f_k(y_{t-1}, y_t, \mathbf{x})$ is a feature function, and $\lambda_k$ is a learned weight

associated with feature $f_k$. The feature function measures any aspect of a state transition, $y_{t-1} \rightarrow y_t$, and the entire observation sequence, $\mathbf{x}$. Large positive values for $\lambda_k$ indicate a preference for an event, and large negative values make the event unlikely.

The most probable label sequence for an input $\mathbf{x}$,

$$\mathbf{y}^* = \operatorname*{argmax}_y P_\Lambda(\mathbf{y}|\mathbf{x})$$

can be efficiently determined using the Viterbi algorithm.

CRFs are trained using maximum likelihood estimation, i.e., maximizing the log-likelihood $L_\Lambda$ of a given training set $T = \langle x_i, y_i \rangle_{i=1}^N$,

$$\begin{aligned} L_\Lambda &= \sum_i \log P_\Lambda(\mathbf{y}_i|\mathbf{x}_i) \\ &= \sum_i \left( \sum_{t=1}^{T} \sum_k \lambda_k f_k(y_{t-1}, y_t, \mathbf{x}) - \log Z_{x_i} \right) \end{aligned}$$

In this implementation, quasi-Newton method is used as the learning algorithm for parameter optimization, which has been shown to converge much faster. To avoid over-fitting, log-likelihood is penalized with Gaussian prior.

CRFs are discriminative models and can capture many correlated features of the inputs. Therefore, it is suitable in many tasks in NLP for sequence labeling. Since they are discriminatively-trained, they are often more accurate than the generative models, even with the same features. CRF++[3] is a customizable implementation of linear-chain CRFs for labeling sequential data. We use this package in some of our experiments.

## 2.5 Summary

This chapter described four state-of-the-art machine-learning methods that are applied in this research, namely Hidden Markov Models, Support Vector Machines, Maximum Entropy Models and Conditional Random Fields.

[3] http://chasen.org/~taku/software/CRF++/

# Chapter 3

# Chinese Word Segmentation

During the process of segmentation, two main problems are encountered: segmentation ambiguities and unknown word occurrences. This chapter focuses on solving the segmentation ambiguity problem. Segmentation ambiguities are dealt with known words, i.e. words found in the dictionary. Usually, before one can solve the problem of unknown word occurrences, one need to accurately segment the known words in the text first. This known word segmentation will become the foundation of unknown word detection. In this chapter, we assume that there is no unknown words in the text, and we only need to correctly segment the known words found in the dictionary. Then, based on the research output of this chapter, we will focus on the unknown word detection in Chapter 4.

## 3.1 Covering Ambiguities and Overlapping Ambiguities

There are basically two types of segmentation ambiguity: covering ambiguity and overlapping ambiguity. The definitions are given below.

Let $x, y, z$ be some strings which could consist of one or more Chinese characters. Assuming that $W$ is a given dictionary, the covering ambiguity is defined as follows: For a string $w = xy$, $x \in W$, $y \in W$, and $w \in W$. As almost any single character in Chinese can be considered as a word, the above definition reflects only those cases where both word boundaries $.../xy/...$ and $.../x/y/...$ can be found in sentences. On the other hand, overlapping ambiguity is defined as follows: For a string $w = xyz$, both $w_1 = xy \in W$ and $w_2 = yz \in W$ hold. Although most of the time, one form of segmentation is preferred over the other,

we still need to know about the contexts in which the other form is used. Both types of ambiguity require that the context be considered to decide which is the correct segmentation form given a particular occurrence in the text.

(1a) and (1b) show examples of covering ambiguity. The string "一家" is treated as a word in (1a) but as two words in (1b).

(1a) 胡/ 世庆/ 一家/ 三/ 口/

Hu/ Shiqing/ whole family/ three/ member

(All three members of Hu Shiqing's family)

(1b) 在/ 巴黎/ 一/ 家/ 杂志/ 上/

in/ Paris/ one/ company/ magazine/ at/

(At one magazine company in Paris)

On the other hand, (2a) and (2b) are examples of overlapping ambiguity. The string "不可以" is segmented as "不/ 可以" in (2a) and as "不可/ 以" in (2b), according to the context in each sentence.

(2a) 不/ 可以/ 淡忘/ 远在/ 故乡/ 的/ 父母/

not/ can/ forget/ far away/ hometown/ DE/ parents/

(Cannot forget parents who are far away at home)

(2b) 不可/ 以/ 营利/ 为/ 目的/

cannot/ by/ profit/ be/ intention/

(Cannot have the intention to make a profit)

Solving the ambiguity problems is a fundamental task in Chinese segmentation process. Although many previous researches have focused on segmentation, only a few have reported on the accuracy achieved in solving ambiguity problems. Li et al. [23] proposed an unsupervised method for training Naïve Bayes classifiers to resolve overlapping ambiguities. They achieved 94.13% accuracy in 5,759 cases of ambiguity. An alternative form of TF.IDF weighting was proposed for solving the covering ambiguity problem in [26]. They focused on 90 ambiguous words and achieved an accuracy of 96.58%.

Without considering the unknown word problem, we will try to solve the ambiguity problem in this chapter.

## 3.2 Solving Segmentation Problem with Minimum Resources

We propose a method that uses only minimum resources, meaning that only a segmented corpus is required. The underlying concept of our proposed method

is as follows. We regard the problem as a character classification problem. We believe that each character in Chinese tends to appear in certain positions in words. A character can be used at the beginning of a word, in the middle of a word, at the end of a word, or as a single-character word. It can appear at different positions in different words. By looking at the usage of the characters, we can decide on their position tags using machine-learning based models, such as Support Vector Machines and Maximum Entropy Models. Our method employs a model to solve the ambiguity problem and, at the same time, embeds a model to detect unknown words (to be described in Section 4.5.1). The method will be described in more detail in the following section.

### 3.2.1 Maximum Matching Algorithm

We intend to solve the ambiguity problem by combining a dictionary-based approach with a statistical model. The Maximum Matching (MM) algorithm is regarded as the simplest dictionary-based word segmentation approach. It starts from one end of a sentence and tries to match the first longest word wherever possible. It is a greedy algorithm, but it has been empirically proved to achieve over 90% accuracy if the dictionary used is large. However, the ambiguity problem cannot be solved effectively, and it is impossible to detect unknown words because only those words existing in the dictionary can be segmented correctly. If we look at the outputs produced by segmenting the sentence forwards (FMM), from the beginning of the sentence, and backwards (BMM), from the end of the sentence, we can determine the places where overlapping ambiguities occur. For example, FMM will segment the string "即将来临时" (when the time comes) into "即将/ 来临/ 时/" (immediately/ come/ when), but BMM will segment it into "即/ 将来/ 临时/" (that/ future/ temporary).

Let $O_f$ and $O_b$ be the outputs of FMM and BMM, respectively. According to Huang [17], for overlapping cases, if $O_f = O_b$, then the probability that both the MMs will be the correct answer is 99%. If $O_f \neq O_b$, then the probability that either $O_f$ or $O_b$ will be the correct answer is also 99%. However, for covering ambiguity cases, even if $O_f = O_b$, both $O_f$ and $O_b$ could be correct or could be wrong. If there exist unknown words, they normally will be segmented as single characters by both FMM and BMM. Based on the differences and contexts created by FMM and BMM, we apply a machine learning based model to re-assign the position tags which indicate character positions in words.

In traditional character tagging approach [49, 33], only the information on

each character is used as features. We are the first who has thought of using information from a dictionary as the features. Later, in [30] and [25], the same idea is also been used in their approach. However, they have used it in a different manner. They search for a longest matching word in a dictionary for the current character, and include the word length of the matched word and the position of the current character in the word as the features. By using the information from a dictionary in the machine-learning based method, we can improve the learning if we found a larger size dictionary, even though the training corpus is not changed.

### 3.2.2 Classification of Characters

We intend to classify the characters using the Support Vector Machines as described in Section 2.2. To do this, first we need to prepare the feature sets to be used for training. Xue and Converse [48] proposed to regard the word segmentation problem as a character tagging problem. Instead of segmenting a sentence into word sequences directly, characters are first assigned with position tags. Later, based on these position tags, the characters are converted into word sequences. The basic features used are the characters. However, the number of examples per feature will be small if there is only character information and no other information is provided. Since there are always more known words than unknown words in a text, it is advantageous if we can segment known words beforehand. Therefore, we supply the outputs from FMM and BMM as some of the features. In this case, the learning is guided by a dictionary for known word segmentation. The similarities and differences between FMM and BMM are used for training in solving the segmentation ambiguity problem.

First, we convert the output of the MMs into a character-wise form, where each character is assigned a position tag as described in Table 3.1. The BIES tags are as described in [42] and [35] for named entity extraction. These tags show possible character positions in words. For example, the character "本" is used as a single character word in "一/ 本/ 书/" (a book), at the end of a word in "剧本" (script), at the beginning of a word in "本来" (originally), or in the middle of a word in "基本上" (basically).

The solid box in Figure 3.1 shows the features used to determine the tag of the character "春" at location $i$ using SVM. In other words, our feature set consists of the characters, the FMM and BMM outputs, and the previously tagged outputs. The context window is two characters on both the left and right sides of the current character. Based on the output position tags, finally, we get the

| Tag | Description |
|---|---|
| S | one-character word |
| B | first character in a multi-character word |
| I | intermediate character in a multi-character word (for words longer than two characters) |
| E | last character in a multi-character word |

Table 3.1. Position tags in a word (BIES tags)

segmentation "迎/ 新春/ 联谊会/ 上/" (welcome/ new year/ get-together party/ at/).

| Position | Char. | FMM | BMM | Output | Answer |
|---|---|---|---|---|---|
| *i - 2* | 迎 | B | S | S | S |
| *i - 1* | 新 | E | B | B | B |
| *i* | 春 | B | E | ? | E |
| *i + 1* | 联 | E | B | | B |
| *i + 2* | 谊 | S | E | | I |
| *i + 3* | 会 | B | B | | E |
| *i + 4* | 上 | E | E | | S |

Figure 3.1. An illustration of classification process applied to "At the New Year gathering party"

This character-based tagging method resembles the idea from [48] for Chinese word segmentation. They have tagged the characters with one of the four tags, LL, RR, MM and LR, depending on their positions within a word. The four tags are equivalent to what we have as B, E, I and S.

### 3.2.3 Experiment with PKU Corpus

The corpus used for this experiment was provided by Peking University (PKU)[1] and consists of about 1.1 million words. It is a segmented and POS-tagged corpus, but we only used the segmentation information for our experiments. We divided the corpus randomly into two parts consisting of 80% and 20% of the corpus,

[1]Downloadable from Institute of Computational Linguistics, Peking University, http://www.icl.pku.edu.cn/

for training and testing, respectively. Since our purpose in this experiment was only to solve the ambiguity problem, not the unknown word detection problem, we assumed that all the words could be found in the dictionary. We created a dictionary with all the words from the corpus, which had 55,310 entries. This experiment was conducted to evaluate the performance of the method in solving the ambiguity problem.

It is difficult to determine how many ambiguities appear in a sentence. For example, in the sentence shown in Figure 3.1, "迎新" (welcome the new year), "新春" (new year), "春联" (a strip of red paper that is pasted beside a door; on it is written some greeting words to celebrate the new year in China), "联谊" (get-together), "联谊会" (get-together party), "会上" (at the meeting) and "上" (at) are all possible words. A word candidate may cause more than one ambiguities with the alternative word candidates. Therefore, we try to represent the ambiguities by means of character units since our method is character-based. We assign each character to one of these six categories.
Let,

$O_f$ = Output of FMM,
$O_b$ = Output of BMM,
$Ans$ = Correct answer,
$Out$ = Output from our system.

| Category | Conditions | No. of Char. | % |
|---|---|---|---|
| *Allcorrect* | $O_f = O_b = Ans = Out$ | 330220 | 96.35% |
| *Correct* | $O_f \neq O_b$ and $Ans = Out$ | 7663 | 2.23% |
| *Wrong* | $O_f \neq O_b$ and $Ans \neq Out$ | 658 | 0.19% |
| *Match* | $O_f = O_b$ and $O_f \neq Ans$ and $Ans = Out$ | 1876 | 0.55% |
| *Mismatch* | $O_f = O_b$ and $O_f \neq Ans$ and $Ans \neq Out$ | 1738 | 0.51% |
| *Allwrong* | $O_f = O_b = Ans$ and $Ans \neq Out$ | 571 | 0.17% |
| Total | | 342726 | 100.00% |

Table 3.2. Disambiguation results obtained with the PKU Corpus

Table 3.2 shows the conditions for each category together with the results obtained with the method for solving the ambiguity problem. The categories

*Allcorrect*, *Correct*, and *Match* have correct answers, whereas the categories *Wrong*, *Mismatch*, and *Allwrong* have wrong answers. We can roughly say that the categories *Correct* and *Wrong* contain overlapping ambiguities, and that the categories *Match*, *Mismatch*, and *Allwrong* contain covering ambiguities. We can also say that *Match* and *Mismatch* categories refer to cases where words should be split, whereas *Allwrong* category refers to cases where words should not be split but the system mistakenly splits them.

Overall, we could correctly tag 99.13% of the characters. If we only consider the overlapping cases (*Correct* and *Wrong*), 92.09% of the characters were correctly tagged. As for covering cases, if we look at only those cases where we need to split the words (*Match* and *Mismatch*), then 51.91% of them were successfully split.

| | FMM | BMM | SVM (char. only) | FMM + SVM | BMM + SVM | FMM + BMM + SVM |
|---|---|---|---|---|---|---|
| Recall | 96.9 | 97.1 | 94.0 | 98.7 | 98.7 | **98.9** |
| Precision | 97.7 | 97.9 | 94.3 | 98.9 | 99.0 | **99.1** |
| F-measure | 97.3 | 97.5 | 94.1 | 98.8 | 98.9 | **99.0** |

Table 3.3. Segmentation results obtained with the PKU Corpus

Table 3.3 shows overall word segmentation results. Compared with the baseline models, namely, FMM, BMM, and SVM (using only characters as features), our proposed method can achieve higher accuracy with an F-measure of 99.0. This means that our method is able to solve the ambiguity problem given information about locations where ambiguities occur by looking at the outputs of FMM and BMM.

## 3.3 Word Segmentation and POS Tagging using HMM

The method described in the previous Section (3.2) is only able to segment the texts without assigning POS tags. In morphological analysis, we also need to assign POS tags. One can either separate the word segmentation and POS tagging as two separate processes, or carry out segmentation and POS tagging simultaneously. The former has the advantages that the complexity of each process can

be reduced but the latter gives us the flexibility to use the information of POS tags in segmentation. We choose the latter approach here. We will describe a model that can do segmentation and POS tagging simultaneously using Hidden Markov Models[2] (hereafter HMM).

In Chinese, many words hold a few POS tags in the dictionary, without changing of the form. For example, "保险" (to insure, insurance, insured) can be used as a verb, a noun or an adjective, "必然" (certainly, certain) can be used as an adverb, a noun or a noun-modifier. Therefore, our POS tagger must be able to select the most suitable tag sequence based on the sentence given.

The HMM model is the one described in Section 2.1. In order to train the model, we need two resources: a tagged corpus and a system dictionary. Then by using these training materials, we calculate the word probability $P(w_i|t_i)$ and the connection probability $P(t_i|t_{i-1})$.

### 3.3.1 Preparation of System Dictionary

There are two ways to prepare the system dictionary. First, we can get a proper dictionary that holds the same segmentation standard with the tagged corpus. We managed to get such a dictionary from Peking University. The dictionary contains 88,910 entries (5.9% unknown word/POS pairs and 4.8% unknown words exist in the test data).

In the case where we do not hold a proper dictionary, we can actually create a dictionary from the tagged corpus. The same Peking University Corpus is used in this experiment with the same division of training and testing. If we take all words from the corpus, we get 62,030 entries (no unknown word exist in the testing data). If we just take from the training data part, then we get 55,409 entries (4.5% unknown word/POS pairs and 4.0% unknown words).

The size of the dictionary used and the number of unknown words in the test data influence the accuracy tremendously. We run the experiments using all these dictionaries to see the effects.

### 3.3.2 Experiments and Results

Table 3.4 shows the result of the system. Although a proper dictionary should give us better result because the vocabulary used is larger, it suffers from the

[2]The system will be used again in other experiments for unknown word detection. Moreover, it is the basic model that will be used to build the Chinese ChaSen as described in Chapter 5.

existence of high productive unknown words such as numeral words (numbers and time nouns) and proper names. Of course, the best result was obtained if there exist no unknown words in the text. However, this case has never happened in real world. Since the training data and the testing data are with the same style, same genre, same period of time, the number of unknown words in the test set is much less if we create the dictionary from the training data. However, for a real world system, a proper dictionary will certain be more advantageous if the texts that need to be analyzed are from difference sources.

| | | Recall | Precision | F-measure |
|---|---|---|---|---|
| Segmentation | Proper Dict | 93.9 | 87.2 | 90.4 |
| | Corpus Dict (All) | 98.4 | 98.7 | 98.5 |
| | Corpus Dict (Training Only) | 94.7 | 90.5 | 92.5 |
| POS Tagging | Proper Dict | 87.9 | 81.7 | 84.7 |
| | Corpus Dict (All) | 93.1 | 93.4 | 93.2 |
| | Corpus Dict (Training Only) | 89.1 | 85.2 | 87.1 |

Table 3.4. Segmentation and POS tagging results obtained using PKU Corpus

From these results, we realize that if unknown words exist in the testing data, the results obtained are not satisfactory. From these experiments, we know that unknown word detection is necessary in order to improve the accuracy of segmentation and POS tagging. In the next chapter, we will focus on the processing of unknown words.

## 3.4 Summary

There are mainly two problems in Chinese word segmentation: segmentation ambiguities and unknown word occurrences. This chapter focused on the problem of segmentation ambiguities. We proposed two methods towards solving it. The first method used the outputs of forward-backward maximum matching algorithms as the features in the classification using Support Vector Machines. The second method used Hidden Markov Models, with the POS tags as the hidden states to solve the ambiguity problem. The first method provides the segmentation only as the output but the second method provides the segmentation together with the

POS tags. Both methods obtained high accuracy for word segmentation provided unknown words do not exist in the text.

# Chapter 4

# Unknown Word Identification

## 4.1 Definition of Unknown Words

An unknown word is defined as a word that is not found in the system dictionary. In other words, it is an out-of-vocabulary word. For any languages, even the largest dictionary we may think, will not be capable of registering all geographical names, person names, organization names, technical terms etc. In Chinese too, all possibilities of derivational morphology cannot be foreseen in the form of a dictionary with a fixed number of entries. Therefore, proper solutions are necessary for unknown word detection.

Our goal in this research is to detect unknown words in the texts and to increase the accuracy of word segmentation. As a language grows, there are always some new terms being created. With the expansion of Internet, the possibilities of getting new words are increasing. Furthermore, Chinese language is used throughout the world. The people who speak Chinese, are not coming only from the mainland China, which has the highest population in the world, but also from Taiwan, Hong Kong, Malaysia, Singapore, Vietnam and also other countries. Although 2/3 of this population share the same language, Mandarin, the standard based on the pronunciation of Peking, there are always some terms which are used only locally. For example, there are transliterated terms from Malay language like "拿督斯里" [1], "巴冷刀" [2], "巴刹"[3] etc, which are used only in Malaysia. Therefore, a proper solution for detecting unknown words is necessary.

[1]Datuk Seri, an honorific title awarded by the king

[2]Parang, a kind of knife

[3]Pasar, a market

## 4.2 Previous Work on Unknown Word Detection

Along traditional methods, unknown word detection has been done using rules for guessing their location. This can ensure a high precision for the detection of unknown words, but unfortunately the recall is not quite satisfactory. It is mainly due to the Chinese language, as new patterns can always be created, that one can hardly efficiently maintain the rules by hand. Since the introduction of statistical techniques in NLP, research has been done on Chinese unknown word detection using such techniques, and the results showed that statistical model could be a better solution. The only resource needed is a large annotated corpus. Fortunately, to date, more and more Chinese tagged corpora have been created for research purpose.

In ([6], [36], [13], [52]), statistical models were used for unknown word detection. Chiang et al. [6] used the length of an unknown word for maximizing the probability. If there is a region where an unknown word is suspected to occur, the following probability is used, $score \approx \cdots \times P(w_u|l_{k-1}) \times P(w_{k+1}|l_u) \times \cdots$, where $w_u$ is an unknown word and $l_u$ is the length of the unknown word. This equation calculates the probabilty of a word given the length of the previous word. The reduction in error rates amounts to 7-9%. In Shen et al. [36], local statistic information is used. They assumed that the frequency of an unknown word is high in a certain cache. For example, if the article is talking about Israel, then the word Israel "以色列", will occur frequently. This happened normally with place names, person names or foreign names. Let $W = AB$, $A$ and $B$ are two strings. If frequency of $A$, $F(A)$, equals to frequency of $B$, $F(B)$, then $W$ should be a word (an unknown word). If $F(A)$ is not equal to $F(B)$, then possibly $A$ and $B$ are two separate words. This method works only if the frequency of the unknown words are high, but not for low frequency unknown words. They have achieved a 54.9% recall for the detection. Fu and Wang [13] used an unsupervised method for unknown word identification. They proposed using word formation power of a character $c$, which can be defined as the division of the frequency of $c$ in multi-word form and $c$ as a single-character word. The formation power can the be applied to prefix, suffix, or middle characters. About 80% of accuracy was reported. Another recent research on unknown word detection was reported by Zhang et al. [52]. Instead of using POS tags which are sparse, they proposed using lexical role tags as a substitution. For example, B as family name, F as prefix in a name, K as previous context before a name, etc. The unknown word recognition consists of 3 steps: (1) automatic acquisition of roles knowledge from

the corpus. (2) role tagging with Viterbi algorithm. (3) unknown word recognition through maximum pattern matching. A model is created individually for each type of unknown words such as person names, place names, transliteration names, etc. They reported an F-measure of 79.30 for person names and 84.69 for transliteration names.

In ([4], [5], [27]), instead of rules written by hand, rules are created automatically from a very large corpus. It is a better solution for rule based models as the maintenance of the rules is eased. They assumed that unknown words are formed by monosyllabic words. First, they tried to identify the location of unknown words by using two properties: (1) a proper-character should not be a bound morpheme, and (2) the context of a proper-character should be grammatical. Then, they create rule patterns that can represent the "proper-characters". Their rules can be represented with unigram, bigram or trigram. These are some examples of the rules, "的", "就(VH)" and "(Na)(Dfa)高"[4]. If the sequence of characters do not apply to any rule, then there is a high possibility that it is an unknown word.

Research has also been done on hybrid approaches which combines statistical and rule based models ([31], [54]). Nie et al. [31] used maximum-matching algorithm to first segment the text, and then used some heuristic rules for identification of words with fixed morphology. In their study, strings containing determiners, ordinal-number markers, cardinal numbers and classifiers are considered in this category. Then an unknown word detection component is added in a later stage. Their unknown word detection is based on both heuristic knowledge about word formation and statistical information on the occurrence rates of various character strings. First, N-gram grouping is done from isolated characters. Then, noise elimination is done by checking the word formation power (how likely the character is used to form a word). Most of the time, N-grams that contain a functional word will be eliminated as they are most likely to be noise. In the third step, heuristic rules are used for candidate word suggestion, such as a family name should exist in a 3-gram name, those characters with bad meaning cannot be used in a name, and affixes. Finally, if a shorter N-gram is part of a longer N-gram, then the shorter N-gram is said to have overlapping, and hence can be eliminated. They achieved about 96% accuracy for overall segmentation including unknown word detection. The second research is from Zhou and Lua [54], which is quite similar to the previous report. They used 4 steps for unknown word

[4]The {} shows a real character and () shows a POS tag. VH - stative intransitive verb, Na - common noun, Dfa - Post-stative V degree adverbial.

detection. (1) Word formation tagging by using HMM and Viterbi algorithm. (2) N-gram grouping. (3) N-gram overlapping. (4) Phrase elimination by heuristic rules. Their method yielded a very high precision of 92%, and a recall of 70% for unknown word detection.

There are some previous methods reported on the accuracy for overall segmentation, solving segmentation ambiguity and unknown word detection at the same time. Recently, many researches are done by combining multiple models. Furthermore, most people have realized that working on character-based is more efficient than word-based for Chinese word segmentation. In Xue and Converse [48], two classifiers are combined for Chinese word segmentation. First, a Maximum Entropy model is used to segment the text, then an error driven transformation model is used to correct the word boundaries. Similarly, they also use character-based tagging on the position of characters in words. They achieved an F-measure of 95.17. Another recent report is by Fu and Luke [11], where hybrid models for integrated segmentation is proposed. Modified word juncture models and word-formation patterns are used to find the word boundaries and at the same time to identify the unknown words. They achieved 96.1 points of F-measure.

## 4.3 Problem Setting

One can detect the unknown words using two different approaches. The first approach is that, the sentence is first segmented into words found in a dictionary, i.e. known words. Then, from the output, one tries to combine some known words to form new words, i.e. unknown words. The limitation of this approach is that the creation of new words can only be done from known words but not part of known words, therefore, the recall is low. The second approach is more arbitrarily. During the process of segmentation, ones can combine any number of characters freely, based on the word formation power of the characters. In this approach, the segmentation of known words and unknown words is carried out simultaneously. The merit of this approach is that more unknown words can be detected and therefore the recall could be higher. However, it has the drawback of over generation. It generates more false unknown words than the previous approach.

Furthermore, if we have a proper dictionary, then the types and numbers of unknown words will be different from without a proper dictionary. We can create

a dictionary from a tagged training corpus, but depending on the size of the corpus, we may not be able to create a dictionary large enough for use in the system. We refer to this dictionary as a not-proper dictionary. Normally if we have a proper dictionary, then the types of unknown words will be more towards proper nouns and numeral type words. If the dictionary is created from a corpus, then the types of unknown words will be more diversified. Therefore, whether or not to get a proper dictionary in the system is also remaining as a question. Some previous researches used one in their systems and some did not use any in their systems. However, it is not obvious that whether using a proper dictionary or without using a dictionary would generate higher accuracy.

In this chapter, we try to experiment in various approaches and see what will be the best solution for our morphological analyzer.

## 4.4 Unknown Word Detection and Extraction

In this Section, we will describe a few methods to solve the unknown word problem. Some methods give us better precision while others give better recall. We will give some explanation on the strength and weakness on each method.

### 4.4.1 Detection based on the Output of Morphological Analysis

Assume that we already have a morphological analyzer as described in Section 3.3. If a word is not found in the system dictionary, then its occurrence will be segmented wrongly. Based on the errors from the morphological analyzer, we want to train a model that can detect the unknown words and reassign the word boundaries.

The method can be summarized into the following three steps.

1. A Hidden Markov Model-based morphological analyzer is used to analyze Chinese texts. It produces the initial segmentation and POS tags for each word found in the dictionary.

2. Each word produced by the analyzer is broken into characters. Each character is annotated with a POS tag together with a position tag. The position tag shows the position of the character in the word.

3. A Support Vector Machine-based chunker is used to label each character with a tag based on the features of the character. The unknown words are detected by combining sequences of characters based on the output labels.

### 4.4.2 Detection without a Proper Dictionary

#### Preparation of System Dictionary for HMM

We did not used other resources rather than the tagged corpus (Peking University corpus) in this approach. The dictionary used was created from the tagged corpus. The initial dictionary created contains all words extracted from the corpus, including training and testing data (62,030 words). As we wanted to create unknown word occurrences in this corpus, all words that occurred only once in the corpus (both training and testing data) were deleted from the dictionary, and are thus treated as unknown words. This means that the unknown words in the testing data have not been seen in the training data. A total of 25,271 (20,876 in training data/4,845 in testing data) unknown words were created under this condition. Then we deleted these words from the dictionary. After the deletion, the final dictionary contains only 36,309 entries. In other words, about 42% of the words in the original dictionary, 2.25% of the corpus, are unknown. In fact, with this setting, we have created a strict condition for unknown word detection as our dictionary is considered very small. Furthermore, the unknown words are of low frequency. This dictionary is used in the training of HMM.

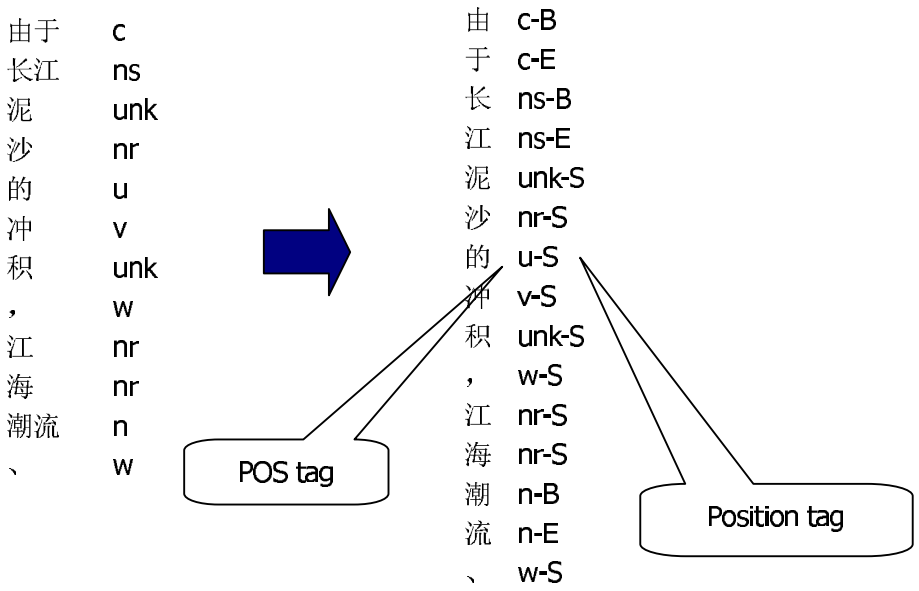
#### Word-based vs Character-based Features

From the output of the morphological analyzer, a sentence is segmented into words together with their POS tags. We can actually use the direct output from the morphological analyzer, which is the word-based for detecting the unknown words. In this case, the features used in the chunking process consist only of the words and the POS tags, as shown on the left hand side of Figure 4.1.

Here, we propose to break the segmented words further into characters and provide the characters with more features. Character-based features allow the chunker to detect the unknown words more effectively. This is especially true when the unknown words overlap with the known words. For example, the morphological analyzer will segment the phrase "邓颖超生前..." (Deng Yingchao before death) into "邓/颖/超生/前/..." (Deng Ying before next life). If we use word-based features, it is impossible to detect the unknown person name "颖

超” (Yingchao) because it does not break up the overlapped word “超生” (next life). Breaking words into characters enables the chunker to look at the characters individually and to identify the unknown words more effectively.

From the output of morphological analysis, each word receives a POS tag. This POS tag information is subcategorized to include the position of the character in the word. We use SE chunking tag set [42], as shown in Table 3.1, to indicate the position. Although there are other chunking tag sets, we choose this tag set because it can represent the positions of characters in Chinese in more details[5]. For example, if a word contains two or more characters, then the first character is tagged as ⟨POS⟩-B, the intermediate characters are tagged as ⟨POS⟩-I and the last is tagged as ⟨POS⟩-E. A single character word is tagged as ⟨POS⟩-S. Figure 4.1 shows an example of conversion from word-based to character-based features.




‘Because of the accumulation of mud from Changjiang, the current between sea and river ...’

Figure 4.1. Conversion from word-based to character-based features

The difference between this feature and the one described in Section 3.2.2 is that we use the paired tags, ⟨POS⟩-⟨position⟩, as the features but the previous one use only the position tags as the features in the model. Therefore, this feature contains more information than the previous one.

[5]The other chunking tag sets such as IOB and IOE use only two tags to indicate the begin and the end of a chunk.

**Chunking with Support Vector Machines**

We regard the unknown word detection problem as a chunking process. Unknown words are detected based on the output of the morphological analysis after converting into character-based features. SVMs are known for the capability of coping with many features, which are suitable for unknown word detection as we need a larger set of features.

We only need to classify the characters into 3 categories, B (beginning of a chunk), I (inside a chunk) and O (outside a chunk). A chunk is considered as an unknown word in this case. This tagging is similar to the notation used in [35] for base-phrase chunking which is called IOB2. These tags are slightly different from the position tags used in character tagging as in Table 3.1. The position tags are used to mark the location of characters in a word, while the IOB2 tags are used to mark chunk boundaries. Therefore, these simpler labels are sufficient to indicate the boundaries of unknown words. SVM is a binary classifier, where only two classes are considered. As we need more than two classes, we have chosen pairwise method to cater for multi-class classification. In each classifier, there are $\binom{n}{2}$ binary classifiers, where $n$ is the number of classes. In this case, $n$ equals to 3.

We can either parse a sentence forwards, from the beginning of the sentence, or backwards, from the end of the sentence. It depends on the formation of a word, whether the head or the tail that are more meaningful. For example, "江" (family name) can be used as the head of a person name, and "人" (person) can be used as the tail of a noun for persons in charge of certain job. We assume that by looking at the more meaningful part of a word first, the word can be detected more correctly.

There are always some relationships between the unknown words and their contexts in the sentence. Tentatively, we use two characters on the left and right sides as the context window for chunking (Figure 4.2). We assume that this window size is reasonable enough for making correct judgment. As we need to classify the characters into 3 categories, we chose "pairwise method" in this experiment because it is more efficient during the training.

The training data of SVM is generated from the output of the morphological analyzer. First, the original training data is input as raw texts into the morphological analyzer. Then the outputs which are words and POS tags, are converted into character-based features as described. Each character is labeled with IOB2 tagset to show the chunks of unknown words. Finally, this data is served as the

| Position | Char. | POS-position | Chunk | Answer |
|---|---|---|---|---|
| *i - 4* | 由 | c-B | O | O |
| *i - 3* | 于 | c-E | O | O |
| *i - 2* | 长 | ns-B | O | O |
| *i - 1* | 江 | ns-E | O | O |
| *i* | 泥 | unk-S | ? | B |
| *i + 1* | 沙 | nr-S | | I |
| *i + 2* | 的 | u-S | | O |
| *i + 3* | 冲 | v-S | | B |
| *i + 4* | 积 | unk-S | | I |

Figure 4.2. An illustration of the features used for chunking
'Because of the accumulation of mud from Changjiang', Char. - Chinese character, POS-position - POS tag plus position tag, Chunk - label for unknown word

training data for the SVM model. By doing this, the unknown words are first segmented and POS tagged by the morphological analyzer. Later, the output labels of the unknown words are learned by SVM based on the error output of the morphological analyzer.

Figure 4.2 illustrates a snapshot of the chunking process with forward parsing. To guess the unknown word tag "B" at position *i*, the chunker uses the binary features appearing in the solid box. This means that we have maximum 12 active features for use to classify a single character. The *Chunk* column is the output labels of SVM where we can identify the unknown words. The last column shows the correct answers for the output. If the chunker could label the tags correctly, then we could get "泥沙" (mud) and "冲积" (accumulation) as unknown words.

### Experiments and Results

We run the experiments using word-based and character-based features. For word-based features, only the words and POS tags are used. For character-based features, there are the characters, POS tags and position tags.

We present the results of our experiments in recall, precision and F-measure, which are defined in the equations below, as usual in such experiments.

$$recall = \frac{\#\ of\ correctly\ extracted\ unknown\ words}{total\ \#\ of\ unknown\ words}$$

$$precision = \frac{\#\ of\ correctly\ extracted\ unknown\ words}{total\ \#\ of\ recognized\ as\ unknown\ words}$$

$$F\text{-}measure = \frac{2 \times recall \times precision}{recall + precision}$$

The results are shown in Table 4.1. Around 60 points of F-measure is achieved for unknown word detection. The first two rows show the results using word-based features and the next two rows using character-based features. As shown in this table, character-based features have made an improvement. The reason of improvement is that the character-based tagging provided better features in combining sequence of characters during the chunking process. As each character carried its own features, they could be freely combined with the adjacent characters to form new words. Therefore, the recall obtained was higher.

| | Recall (%) | Precision (%) | F-measure |
|---|---|---|---|
| Word-based/F | 51.33 | 64.36 | 57.11 |
| Word-based/B | 53.02 | 63.60 | 57.83 |
| Character-based/F | 56.78 | **64.49** | **60.39** |
| Character-based/B | **58.27** | 63.82 | 59.87 |

F - forward chunking, B - backward chunking

Table 4.1. Results for unknown word detection

Until this stage, the unknown words detected still do not have POS tags associated with them. In order to get a rough idea on how well the model has done for each type of POS tags, we just made a calculation based on the original answers. Table 4.2 shows the distribution for the POS tags with frequency more than 1000. This model was able to detect numbers and person names quite well, and was moderate for place names and nouns. On the other hand, the worst was with collocations and idioms. This is because collocations and idioms have no standard morphological pattern for detection and therefore the accuracy was low.

The detected unknown words were combined with the initial segmentation to get the final segmentation. The combination is simple. For example, if we have the output from SVM such as in Figure 4.2, then we just replace the original

| | All | Testing | Correct | Recall |
|---|---|---|---|---|
| Noun (n) | 7902 | 1618 | 901 | 56% |
| Person name (nr) | 4535 | 605 | 463 | 77% |
| Number (m) | 2959 | 522 | 422 | 81% |
| Verb (v) | 2691 | 457 | 199 | 44% |
| Place name (ns) | 1641 | 372 | 239 | 64% |
| Idiom (i) | 1122 | 235 | 72 | 31% |
| Collocation (l) | 1098 | 203 | 49 | 24% |

Table 4.2. Distribution of detected unknown words by their POS tags

words with the new detected words, and the final segmentation is like "由于/c 长江/ns 泥沙/unk 的/u 冲积/unk", where "unk" is the unknown POS tag.

We made no effort to determine whether the unknown words detected were correct words or not. We gave priority to the SVM output. There were also some cases where the initial segmentation was correct but then was incorrectly detected as unknown word, and this caused the undesired errors in the final segmentation.

| | Recall | Precision | F-measure |
|---|---|---|---|
| Only using HMM | 96.53 | 93.75 | 95.12 |
| HMM+Word-based+SVM/F | **96.81** | 96.45 | 96.63 |
| HMM+Word-based+SVM/B | 96.76 | 96.49 | 96.62 |
| HMM+Character-based+SVM/F | 96.78 | 96.72 | **96.75** |
| HMM+Character-based+SVM/B | 96.63 | **96.76** | 96.70 |

Table 4.3. Results for word segmentation

Before the unknown word detection, the F-measure of segmentation from the HMM only achieved 95.12. After the unknown word detection using character-based features, the F-measure increased to 96.75, an improvement of 1.63. From Table 4.3, we observed that the improvement has taken place in precision, an increment of about 2.97%, from 93.75% to 96.72%. The result also shows that the character-based features generated slightly better results than the word-based features by F-measure. The segmentation recall using the word-based features is slightly higher than the character-based features because even more unknown words have been detected in character-based model, but at the same time there exists more incorrectly detected unknown words as well.

### 4.4.3 Detection with a Proper Dictionary

The number of unknown words is depending on the size of the dictionary used. Certainly, the larger the dictionary, the less the unknown word occurrences in the texts. One can create a dictionary from a tagged corpus as in the previous setting but that will not be a proper dictionary. Furthermore, if all words in the tagged corpus are used to create the dictionary, then there will be no unknown word in the texts. Therefore, it is important to define the meaning of unknown words properly. In the previous experiment, those words that occur only once in the corpus are treated as unknown words in the experiment. However, some people argue that this is not really true because even low frequency words are actually words in some dictionaries but those person names even with high frequencies could not be found in a dictionary. A more natural way is by having a proper dictionary. We can consider those words that are not in a proper dictionary to be unknown words. In this case, some words in the corpus are not found in the dictionary and can be used as training data for unknown word detection [4, 11]. As far as we know, the definitions of words are different by institutions, such as Peking University Corpus, Penn Chinese Treebank and Academia Sinica Corpus. Therefore, the dictionary and the tagged corpus used must be consistent. We use the dictionary and tagged corpus provided by Peking University. The dictionary contains 88,910 entries and the corpus has about 1.1 million words.

#### Preparation of System Dictionary for HMM

The dictionary used in this approach is a proper dictionary obtained from Peking University. This dictionary contains 88,910 entries. It consists of almost all common words in Chinese.

From our survey in the corpus, about 4.5% of the words are unknown. According to the part-of-speech tags (POS), 29% of the unknown words are numbers (m), 20% are time nouns (t), 17% are person names (nr), and 34% for other types. That is to say, almost 50% of the unknown words are made up from number types (numbers and time nouns). The detection of number types is a trivial task although the production is high. As for Chinese person names, normally they consist of family names and given names, which somehow have similar patterns for recognition. And for foreign names, the characters used are limited to a set of characters which is used to spell the words by pronunciation in the foreign language.

### New Features and New Classification Approaches

Besides the features used in the previous approach, we also introduce new features in this approach. We define character type as a new feature. Strictly saying, there is no character type in Chinese language, but we can group them according to their usage, such as possible family names and transliteration characters (although they still can be used in other places). Currently we have collected 436 family names[6] and 160 transliteration characters[7]. A character is assigned with one of these four types: SURNAME (a family name), FOREIGN (a transliteration character), BOTH (can be used as both family name or transliteration character), or OTHER (not in any type). Finally, a character will have a POS tag with its position tag and a character type to be used as features during classification.

For the output of classification, we only need 3 basic tags to identify the location of unknown words, namely tag "B" (the beginning of an unknown word), tag "I" (inside of an unknown word), or tag "O" (outside of any unknown word). Two characters at both sides of the character are used as context window. Figure 4.3 shows an illustration of the classification process. The solid box shows the features used to determine the class of the character at location *i*. The characters tagged with "B" and "I" compose an unknown word "秀兰" (Xiulan), a person name.

| Loc. | Char. | POS + position tag | Char. Type | Class |
|---|---|---|---|---|
| *i-2* | 周 | nr-S | SURNAME | O |
| *i-1* | 秀 | Vg-S | OTHER | B |
| *i* | 兰 | Ng-S | BOTH | I |
| *i+1* | 夫 | n-B | FOREIGN | O |
| *i+2* | 妻 | n-E | OTHER | O |

Figure 4.3. An illustration of classification process - 'Zhou Xiulan couple'

We have chosen pairwise method to cater for multi-class classification using SVM. In each classifier, there are $\binom{n}{2}$ binary classifiers, where $n$ is the number of classes. By using the method described above, we now define 3 approaches of classification. Note that we regard the $\binom{n}{2}$ binary classifiers as one multi-class

[6]Chinese family names are almost a fix set, where new family names are rarely created.

[7]Although the characters used for transliteration words are also limited, but they can be increased easily if there exist new pronunciations of new words.

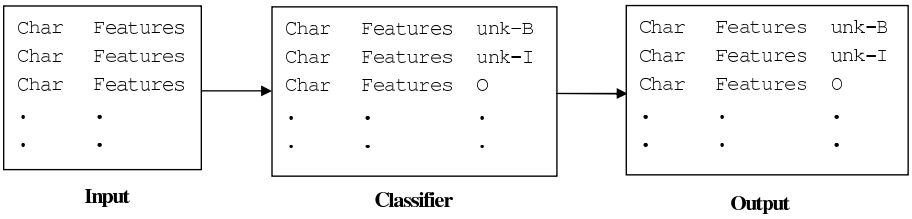



Figure 4.4. One-Classifier-One-Type classification

classifier in the following section.

One-Classifier-One-Type Classification

In the first approach, we regard all the unknown words as one single type of words and we only need to classify the characters into 3 classes, namely unk-B, unk-I or O. The output will be the unknown words without knowing the types, as shown in Figure 4.4.

One-Classifier-Multi-Type Classification

From our survey in the corpus, about 66% of the unknown words are numbers, time nouns and person names. If we straightaway classify these three types during unknown word detection process, then it will be grateful that we do not need to guess the category for these types anymore. Therefore, in the second approach, instead of only 3 classes, we define 9 classes for classification, namely nr-B, nr-I (for person names), m-B, m-I (for numbers), t-B, t-I (for time nouns), unk-B, unk-I (for others) and O. Figure 4.5 shows the classification process for this multi-type method.

Multi-Classifier-Multi-Type Classification

The third approach comes from the idea in [53], where a hierarchical model is used for different types of unknown words. If only one classifier is used for all types of unknown words, the same features, same parameters must be used for all of them. From our past experiments, we realized that different types of unknown words need different feature sets and parameters. For example, numbers are best detected using only the POS+position tag as features, without the character

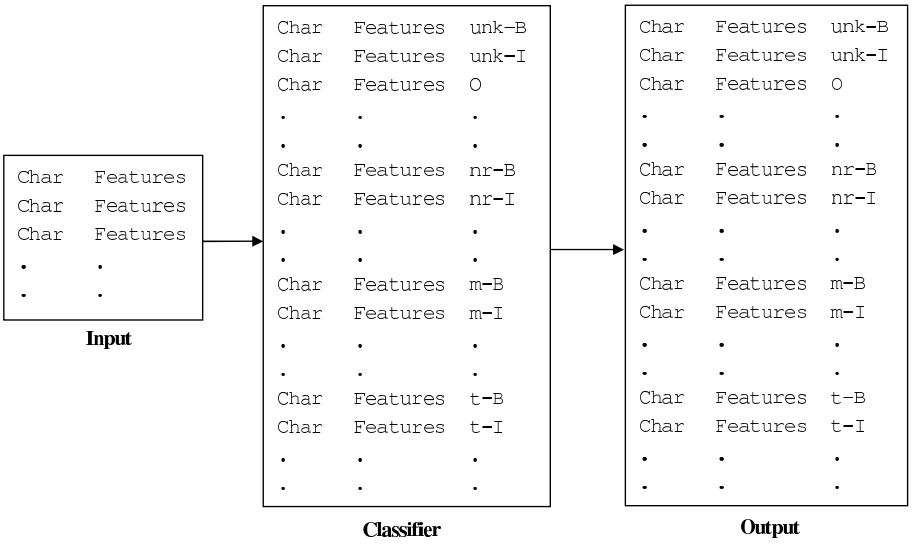



Figure 4.5. One-Classifier-Multi-Type classification

type, and with forward parsing. Therefore, if one classifier is created for each type of unknown words, and the best fitted features and parsing direction are used, then, optimal results may be obtained for all of them. At last, the outputs from each classifier are combined to generate the final output. This approach is shown in Figure 4.6. We make no effort to combine the result, but just give priority to the type with higher precision in case there are any conflicts where a character receives more than one tags. As a result, the sequence of priority is "time nouns > numbers > person names > others". In fact, there are not so many overlapping cases, more often with numbers and time nouns. Usually, time nouns are more preferred than numbers. We leave the more intelligent way to combine the outputs for the future work.

### Experiments and Results

We use the Peking University (about 1.1 million words) corpus for our experiment. The corpus is randomly divided into a proportion of 80%/20% for training and testing respectively. The dictionary contains 88,910 entries. Based on this dictionary, there are about 4.5% unknown words in the texts, which spread evenly between training and testing data. The distribution of unknown words is as shown in Table 4.4.

Table 4.5 shows the individual results produced by each classifier in Multi-Classifier-Multi-Type approach. The first two columns show the results where

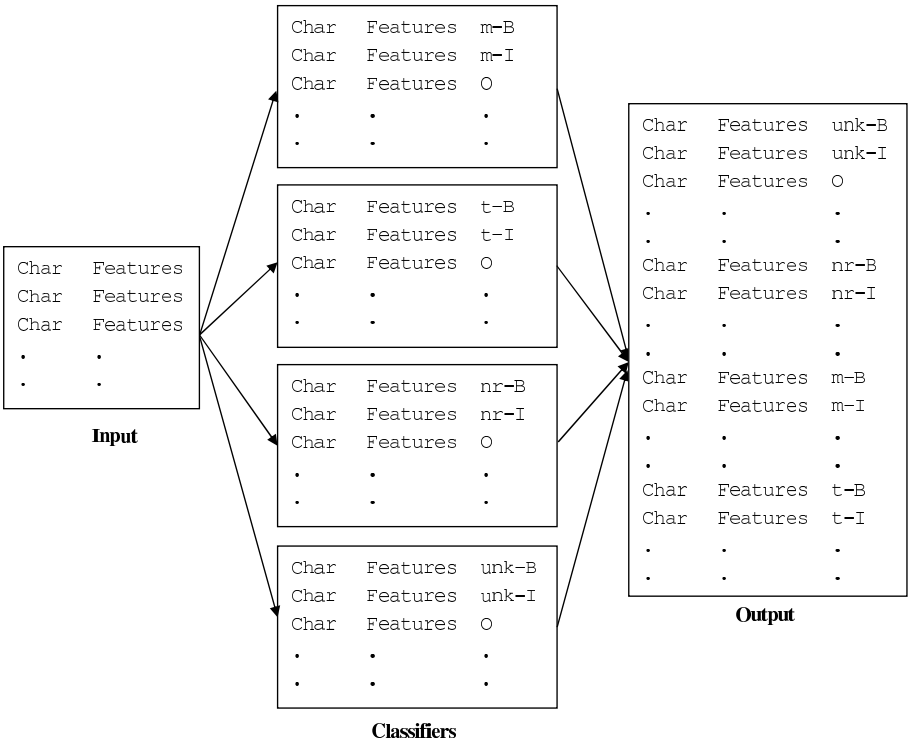



Figure 4.6. Multi-Classifier-Multi-Type classification

| | # of words | # of unknown words | # of distinct unknown words | unknown word rate |
|---|---|---|---|---|
| Training data | 911,551 | 40,733 | 17,027 | 4.47% |
| Testing data | 209,896 | 10,033 | 5,201 | 4.78% |
| Total | 1,121,447 | 50,766 | 20,424 | 4.53% |

Table 4.4. Experimental corpus

character types are not used as the features, and the second two columns include character types as the features. Forward and Backward represent the parsing directions (read from the beginning of the sentence or reverse) during the SVM classification. This table shows that each type of unknown words needs different feature sets and parsing directions. Our final output is composed by choosing the best result from each classifier (as indicated in bold face).

Table 4.6 shows the overall unknown word detection results. We realize that the Multi-Classifier-Multi-Type approach has done slightly better than others by

| | POS+position tag | | POS+position tag & Char. Type | |
|---|---|---|---|---|
| | Forward | Backward | Forward | Backward |
| Person Name | 82.43 | 84.18 | 84.25 | **86.04** |
| Number | **97.06** | 96.55 | 96.99 | 96.33 |
| Time noun | 95.84 | 97.30 | 95.79 | **97.36** |
| Others | 58.68 | **61.97** | 58.92 | 61.61 |

Table 4.5. Individual F-measure of Multi-Classifier-Multi-Type approach

| | | POS+position tag | | POS+position tag & Char. Type | |
|---|---|---|---|---|---|
| | | Forward | Backward | Forward | Backward |
| Recall | One-C-One-T | 76.92 | 79.34 | 77.19 | **79.38** |
| | One-C-Multi-T | 75.94 | 78.38 | 76.63 | 78.61 |
| | Multi-C-Multi-T | 77.56 | | | |
| Precision | One-C-One-T | 85.94 | 85.44 | 85.90 | 85.24 |
| | One-C-Multi-T | 87.09 | 87.15 | 86.80 | 86.51 |
| | Multi-C-Multi-T | **88.91** | | | |
| F-measure | One-C-One-T | 81.18 | 82.28 | 81.31 | 82.20 |
| | One-C-Multi-T | 81.14 | 82.53 | 81.40 | 82.37 |
| | Multi-C-Multi-T | **82.85** | | | |

Table 4.6. Unknown word detection results

F-measure. Although the recall is worse compared with One-Classifier-One-Type, the improvement on the precision is significant (at 5% level).

In Fu and Luke [12], a class-based language model is introduced for Chinese unknown word identification. A hybrid model which composes of class-based word juncture models and class-based word formation patterns is proposed. The classes refer to the POS tags, which is similar to our method of dividing the unknown words into 4 types. Their method handles both internal word formation features and external contextual information which are important to identify the word boundaries. Since we are using the same corpus, namely the Peking University corpus, we have the same segmentation standard. However, their lexicon is smaller, only contains about 65,000 words (with 6.81% unknown words in the test data). They report the accuracy of unknown word detection of 81.8, 80.8 and 82.5 for F-measure, recall and precision respectively, and we have 82.85, 77.56

and 88.9, respectively. They have higher recall while we have better precision.

In One-Classifier-Multi-Type and Multi-Classifier-Multi-Type approaches, there are possibilities that a number is detected as a time noun, or a person name is detected as other, and so on. Therefore, the overall accuracy drops a bit when we evaluate our results by types. Although we do not know the types by One-Classifier-One-Type approach, we just do the calculation by recall for comparison. We could not calculate the precision for One-Classifier-One-Type approach as the types of unknown word are not known. As shown in Table 4.7, the recall is better by One-Classifier-One-Type approach. However, we get high precision with Multi-Classifier-Multi-Type approach for time nouns (99.24%), numbers (98.29%) and person names (89.09%), and reasonable for others (72.87%).

In Zhang et al. [52], role tagging on characters is used for person name detection. Instead of POS tags, they define role tags according to their linguistic features for words related to unknown person names. For example, a role set for person name extraction could be context, suffix, tokens in a Chinese person name and etc. They use Hidden Markov Models to assign the role tags to the words. Finally, unknown person names are recognized through maximum pattern matching on role sequence. They reported an F-measure of 79.30 for Chinese person name detection and 84.96 for transliteration name detection. We do not discriminate between Chinese and transliteration person names. We get 86.04 for both types, which is better then theirs. Fu and Luke [12], in which a class-based language model is used, gets 86.4 for person name detection, slightly better than ours.

The training of our models requires a dictionary and a tagged corpus. Since the dictionary and the corpus are two different data sources, it also means that not all words in the training corpus are in the dictionary. Some people argued that although the unknown words are not in the dictionary, they probably have been seen in the training corpus. In this case, it is not a surprise that they can be detected correctly. Therefore, we also make an evaluation on those unknown words that occur only in the testing data but not in the training corpus. We refer to these unknown words as real unknown words. There are 4,427 (44%) real unknown words in the testing data. Table 4.8 shows the results for real unknown words. We get about 60% recall with all approaches. The distribution of real unknown words are as below: person names (20%), numbers (13%), time nouns (1%) and others (66%). Originally the numbers and the time nouns have the highest unknown word distribution but they are not real. Most of them have been seen in the training data, therefore the detection is easier. The most

| | | Person Name | Number | Time Noun | Others | Overall |
|---|---|---|---|---|---|---|
| Recall | One-C-One-T | **(86.78)** | **(97.19)** | **(96.44)** | **(59.09)** | **(79.34)** |
| | One-C-Multi-T | 80.25 | 96.48 | 95.70 | 56.26 | 77.45 |
| | Multi-C-Multi-T | 83.20 | 97.00 | 95.55 | 53.95 | 76.97 |
| Precision | One-C-One-T | n.a. | n.a. | n.a. | n.a. | n.a. |
| | One-C-Multi-T | 85.82 | 96.26 | 99.24 | 70.74 | 86.11 |
| | Multi-C-Multi-T | **89.09** | **98.29** | **99.24** | **72.87** | **88.22** |
| F-measure | One-C-One-T | n.a. | n.a. | n.a. | n.a. | n.a. |
| | One-C-Multi-T | 83.13 | 96.37 | **97.44** | **62.67** | 81.56 |
| | Multi-C-Multi-T | **86.04** | **97.64** | 97.36 | 62.00 | **82.21** |

We show results of POS+position tag as features, with backward parsing for One-C-One-T and One-C-Multi-T as they have the best F-measures overall. On the contrary, the best result from each classifiers is chosen to compose the final results for Multi-C-Multi-T.

Table 4.7. Results by types of unknown words

difficult one is with the type others, which has the highest real unknown word distribution. We need more attention on this type in the future.

| | | POS+position tag | | POS+position tag & Char. Type | |
|---|---|---|---|---|---|
| | | Forward | Backward | Forward | Backward |
| Recall | One-C-One-T | 58.69 | 63.27 | 59.27 | **63.43** |
| | One-C-Multi-T | 57.40 | 61.44 | 58.69 | 61.73 |
| | Multi-C-Multi-T | 60.18 | | | |

Table 4.8. Results by recall of real unknown words

### Error Analysis

Although we could obtain high accuracy for numbers and time nouns, but they are not a surprise. If the numbers are made up of character numbers, and time nouns are made up of character numbers with suffixes such as year, month, day, etc, then we should be sure that they can be detected correctly. Those words that could not be detected are not made up of numbers such as “些许” (a little

bit), “一艘艘” (counting of ship) and “双双” (both). Examples for time nouns are such as “去冬” (last year winter), “今宵” (tonight), “岁初” (the beginning of a year) and “丑牛” (the year of ox).

For person name detection, we have problems on detecting Japanese name, as they are not made up from either Chinese family names nor transliteration characters. We will need a different approach for detecting Japanese person names. We also could not detect person names which contain affixes such as “贝老” (the old man Bei), “红嫂” (a lady named Hong) and “叶氏” (the person Ye). The occurrences of these person names are quite low in the corpus, therefore, we still need more examples for training. Normally they are made up from a family name (or given name) with an affix, therefore, we may detect them by using some rules instead of statistics-based method.

We get quite satisfactory precision (88.91%) using the proposed method. As there is no single standard definition of words in Chinese, we could hardly say that the gold data is perfectly correct. Therefore, human judgment is necessary. Since there are not so many incorrectly detected words, we have gone through all the errors to examine what kind of mistakes has been made.

Surprisingly, there are quite a number of words in the error list which are said to be acceptable by human judgment. Out of 971 incorrect words, 380 words are acceptable. Appendix A shows some examples of these words. Some of these errors happen because of the non-standardization of segmentation. For example, “艺术史” (the history of art) is segmented as one word and “京剧/ 史/” (the history of Peking opera) is segmented as two words. There are also human errors like “史/ 泰龙/” where the name is segmented into a family name and a given name but our system has extracted it as one segment which is a correct one [8]. There are also some collocation phrases such as “大肚佛” (big stomach Buddha) and “百鸟朝凤” (hundred birds facing the phoenix), which to some people they can be considered as words too. If we consider these errors to be correct ones, then our method has achieved 93.24% precision. Again, we can conclude that our method can achieve high precision for unknown word detection.

### Effects on Overall Segmentation

By replacing the new detected words with the original segmentation, we get the final segmentation. We get only 90.40 points F-measure using solely HMM. After

[8] “史泰龙” (Stallone - an American actor) in fact is a transliteration foreign name but not a Chinese name although the first character can be a family name.

unknown word detection by using Multi-Classifier-Multi-Type approach, we get 96.59 points, an improvement of 6.19 points.

The segmentation results of the open test[9] in SIGHAN bakeoff for Peking University dataset are ranging from 88.6–95.9 of F-measure, and the recalls for unknown words are 50.3–79.9%. We did not re-train our model with their training materials, but just what we have on hand to run on the testing data. There are 1,253 (7.3%) unknown words in the test data based on our dictionary. We get an F-measure of 88.32 for segmentation by using only HMM, and 95.11 after unknown word detection. The unknown word recall is 75.74% and precision is 89.19% according to our dictionary and the recall is 80.2% according to bakeoff dictionary. Compared to the result, we would get a 3rd place in the bakeoff with the highest unknown word recall.

Regarding unknown word detection as a chunking process has also been used in [55]. In their approach, a sentence is first pre-segmented into a sequence of word atoms using maximum matching algorithm. Then, a chunking model is applied to detect unknown words by chunking one or more word atoms together according to the word formation patterns of the word atoms. The concept behind is similar to our word-based features. They adopted a discriminative Markov model, namely Mutual Information Independence Model in chunking. Besides, a maximum entropy model is applied to integrate various types of contexts and resolve the data sparseness problem. Moreover, an error-driven learning approach is used to learn useful contexts in the maximum entropy model. Their evaluation on the PK and CTB corpora in SIGHAN bakeoff gave the best results on unknown word recall, which is 80.5% and 77.6% respectively.

## 4.5 Relation between Unknown Word Detection and Segmentation

The methods described above detect unknown words after the initial segmentation. The methods make use of the output segmentation and POS tagging as part of the features. In SIGHAN bakeoff, only segmented corpus are provided as the training data. In the closed test, we cannot use any other resources for the training. Therefore, in order to compete with other participants in the closed test, we need to find a better way for word segmentation using only minimum

[9]We compare the results with open test because we have used extra resources such as the tagged corpus and the dictionary.

resources, meaning only a tagged corpus.

### 4.5.1 Embedding Unknown Word Detection during Segmentation

Section 3.2 described a method that use only minimum resources to solve the segmentation ambiguity problem. In this section, the method is extended so that it can also handle unknown word segmentation at the same time. The fundamental method is briefly described. Basically, FMM and BMM are used to first segment the text based on a dictionary. Then, SVM is used to classify each character into BIES tag categories based on the features given by FMM and BMM. In this section, we modify the dictionary used in FMM and BMM, so that the model can cater for unknown word detection as well.

#### Accuracy in Solving the Unknown Word Problem

The method used in this experiment is the same as described in Section 3.2, but the setting is different. In this round, the corpusbis divided into three sets, referred to as Set 1, Set 2, and Set 3. Set 1 plus Set 2 (80%) is used for training, and Set 3 (20%) is used for testing, the same test data as in the previous experiment. The difference is in the preparation of the dictionary. It is prepared in two ways. In the first case, all the words from Set 1 and Set 2 are used to create the dictionary. There are 49,433 entries in the dictionary and 8,346 (4.0%) unknown words exist in the testing data (referred to as Experiment 2). This experiment is conducted to investigate the performance of the method when unknown words exist. In the second case, only the words from Set 1 are used to create the dictionary, resulting in a situation where unknown words exist in the training data (referred to as Experiment 3). The top part of Table 4.9 shows the proportions of Set 1 and Set 2, along with the sizes of the dictionaries and the numbers of unknown words in Set 2 and Set 3 (the testing data). Set 2 serves as a learning model for unknown word detection[10]. When we segment Set 2 using FMM and BMM, most of the unknown words are segmented into single characters (namely tag 'S'). Based on these tags and contexts, the SVM-based chunker is trained to change

[10]It is possible to create unknown word phenomena in a training corpus by collecting all the words from the corpus but dropping some words like compounds, proper names, numbers etc. However, since we assume that out target corpus is only a segmented corpus, without other information like POS tags, it is difficult to determine what words that should be dropped and be treated as unknown words.

the tags into the correct answers. The last experiment (referred to as Experiment 4) is the opposite of Experiment 2; nothing is used to create the dictionary. All the words are considered to be unknown words. Only the characters are used as features during the classification phase, meaning that no information from FMM and BMM is available. Experiment 1 is the result obtained from Section 3.2, which is the perfect case with this method (unknown words do not exist).

| | Exp 1 | Exp 2 | Exp 3 | | | Exp 4 |
|---|---|---|---|---|---|---|
| Set1(%)/ Set2(%) | | 80/0 | 60/20 | 40/40 | 20/60 | 0/80 |
| # of words in Dict. | 62,030 | 49,433 | 41,582 | 33,355 | 22,363 | 0 |
| # of unk-words in Set 2 | 0 | 0 | 10,927 | 25,297 | 53,353 | All |
| # of unk-words in Test (Set 3) | 0 | 8,346 | 9,768 | 11,924 | 17,115 | All |
| Recall | 98.9 | 95.3 | **95.8** | 95.7 | 95.2 | 94.0 |
| Precision | 99.1 | 90.7 | 93.5 | 94.5 | **94.7** | 94.3 |
| F-measure | 99.0 | 92.9 | 94.7 | **95.1** | 94.9 | 94.1 |
| OOV (recall) | – | 8.0 | 41.2 | 54.9 | 63.3 | **69.3** |
| IV (recall) | 98.9 | **98.9** | 98.1 | 97.4 | 96.5 | 95.0 |

Table 4.9. Different settings and segmentation results with unknown words (PKU Corpus)

The bottom part of Table 4.9 shows the results obtained in these experiments. Our method in fact worked quite well in solving both the segmentation ambiguity and unknown word detection problems. However, while the accuracy for unknown word detection improved, the performance in solving the ambiguity problem worsened. This is because the precision in unknown word detection was not one hundred percent. False unknown words caused the accuracy of known word segmentation to deteriorate. The highest recall rate that we could get for known words was 98.9% (as in model 80/0) and that for unknown words was 69.3% (as in model 80/0). However, the best overall segmentation result was achieved by dividing the training corpus into half (as in model 40/40), and the result was an F-measure of 95.1. This is the optimal point where a balance is found between detecting unknown words and at the same time maintaining accuracy in the segmentation of known words. Figure 4.7 shows the F-measure results for segmentation and recall results for unknown words and known words, when

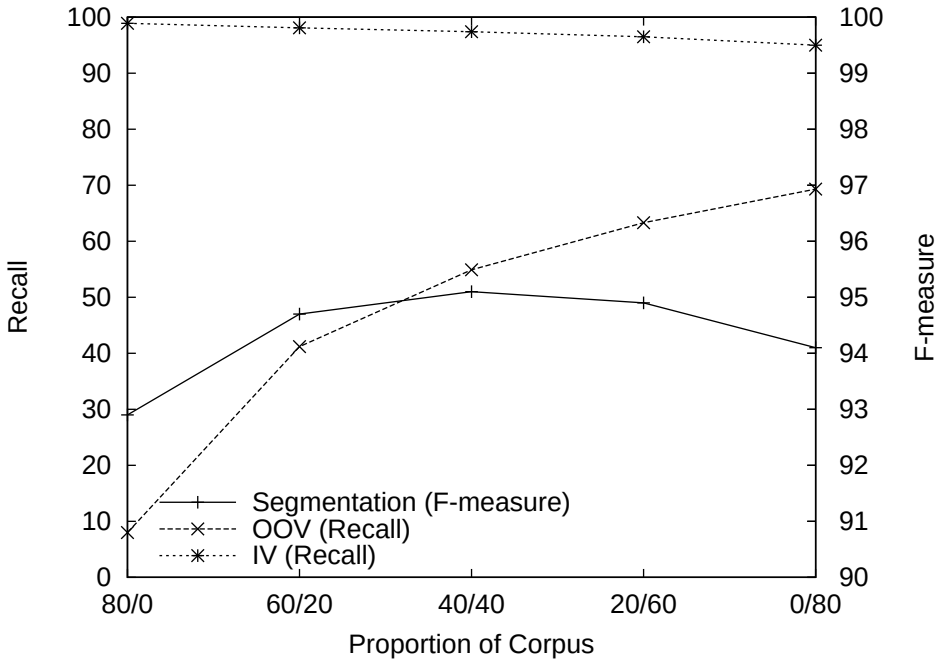



Figure 4.7. Accuracy of segmentation (F-measure), OOV (Recall) and IV (Recall)

different proportions of the training corpus were used to create the dictionary.

**Experiment with SIGHAN Bakeoff Data**

In the SIGHAN bakeoff closed test, only the training data were allowed to be used for training and no other material. Under this strict condition, it is possible to create a lexicon from the training data, but, of course, unknown words will exist in the testing data. We also conducted an experiment using the bakeoff data. Since our system works only on two-byte coding, some ascii code in the data, especially numbers and letters, are converted to GB code or Big5 code prior to processing. The original dictionaries consisted of all the words extracted from the training data. Some of the unknown words automatically became known words after ascii code was converted to GB/Big5 code. The conversion step reduced the number of unknown words. For example, if the number "１９９８" written in GB code existed in the training data but it was written in ascii code as "1998" in the testing data, then it was treated as an unknown word at the first location. Following conversion, it became a known word.

The experimental setup was similar to that in Experiment 3 above. In Experiment 3, based on our previous experiments, using half of the training corpus to

create the dictionary generated the best F-measure result. Therefore, only about 50% (first half) of the training corpora were used to create the dictionaries[11]. As a result, the new dictionaries contained fewer entries than the original dictionaries. Table 4.10 shows the details of the sizes of the dictionaries used.

| Corpus | Size of original dictionary | Size of dictionary used |
|---|---|---|
| PKU | 55,226 | 36,830 |
| CTB | 19,730 | 12,274 |
| AS | 146,226 | 100,161 |
| HK | 23,747 | 17,207 |

Table 4.10. Bakeoff dictionary

| Corpus | *Recall* | *Precision* | *F-measure* | $Recall_{unknown}$ | $Recall_{known}$ |
|---|---|---|---|---|---|
| PKU | 95.5 | 94.1 | 94.7 | 71.0 | 97.3 |
| CTB | 86.0 | 83.5 | 84.7 | 57.7 | 92.2 |
| HK | 95.4 | 92.1 | 93.7 | 65.5 | 97.7 |
| AS | 97.0 | 94.8 | 95.9 | 69.0 | 97.6 |

Table 4.11. Segmentation results obtained with bakeoff data

As observed in [37], none of the participants of the bakeoff could get the best results for all four tracks. Therefore, it is quite difficult to compare accuracy across different methods. Our results are shown in Table 4.11. Comparing with the bakeoff results, one can see that our results are not the best, but they are among the top three best results, as shown at the top of Figure 4.8. During the bakeoff, only two participants took part in all four tracks in the closed test. We obtained better results than one of them [1], where a similar method was used to re-assign word boundaries. The difference is that words are first categorized into 5 or 10 classes (which are assumed to be equivalent to POS tags) using the Baum-Welch algorithm, and then the sentence is segmented into word sequences using a Hidden Markov Model-based segmenter. Finally, the same Support Vector Machine-based chunker is trained to correct the errors made by the segmenter. Our method which simply uses a forward and backward Maximum Matching

[11]Since the size of the training data is too big for the AS dataset, we had difficulty training the SVM as the time required was extremely long. Therefore, we divided it into five classifiers and finally combined the results through simple voting.

algorithm, achieved better results than theirs when complicated statistics-based models were involved. On the other hand, compared to the results obtained by [53], we only obtained better results for two datasets and worse results for the other two datasets. They used hierarchical Hidden Markov Models to segment and POS tag the text. Although it was a closed test, they used extra information, such as class-based segmentation and role-based tagging models [52], which gave better results for unknown word recognition. The bottom of Figure 4.8 shows the results of unknown word detection. Again, our method performed comparatively well in detecting unknown words.

Regarding Chinese word segmentation problem as character tagging problem has previously been seen in Xue and Converse [48]. The difference in our method is that we supply FMM and BMM outputs as a control for the final output decision. However, only words from half of the training corpus are controlled. Since false unknown words are the main cause of errors with known words, our method tries to maintain accuracy for known words while at the same time detecting new words. As Xue and Converse [48] used a different corpus than ours, namely, the Penn Chinese Treebank, it is difficult to make a fair comparison. They also participated in the bakeoff for the HK and AS tracks only [49]. They obtained segmentation F-measures of 91.6 and 95.9, respectively, while we achieved 93.7 and 95.9, which are quite comparable. They did a bit better in unknown word recall, achieving 67.0% and 72.9% recall rates, whereas ours were 65.5% and 69.0%. On the other hand, we obtained much better results in known word recall, 97.7% and 97.6%, compared to their recall rates of 93.6% and 96.6%. Usually a piece of text contains more known words than unknown words; therefore our method, which controls the outputs of known words, is a correct choice. Furthermore, our method can also detect unknown words with comparable results.

In conclusion, our results did not surpass the best results in the bakeoff for all datasets. However, our method is simpler. We only need a dictionary that can be created from a segmented corpus, FMM and BMM modules, and a classifier, without the use of human knowledge. We can get quite comparable results for both known words and unknown words. The results are worse when the training corpus is small and there exist a lot of unknown words, such as in CTB testing data. Therefore, we still need to investigate the relationship between the size of the training corpora and the proportion of the corpora used to create the dictionaries in the training for solving ambiguity problems and performing unknown word detection. We are also looking into the possibility of designing an ideal model, where optimal results for known words, as in Experiment 2, and unknown

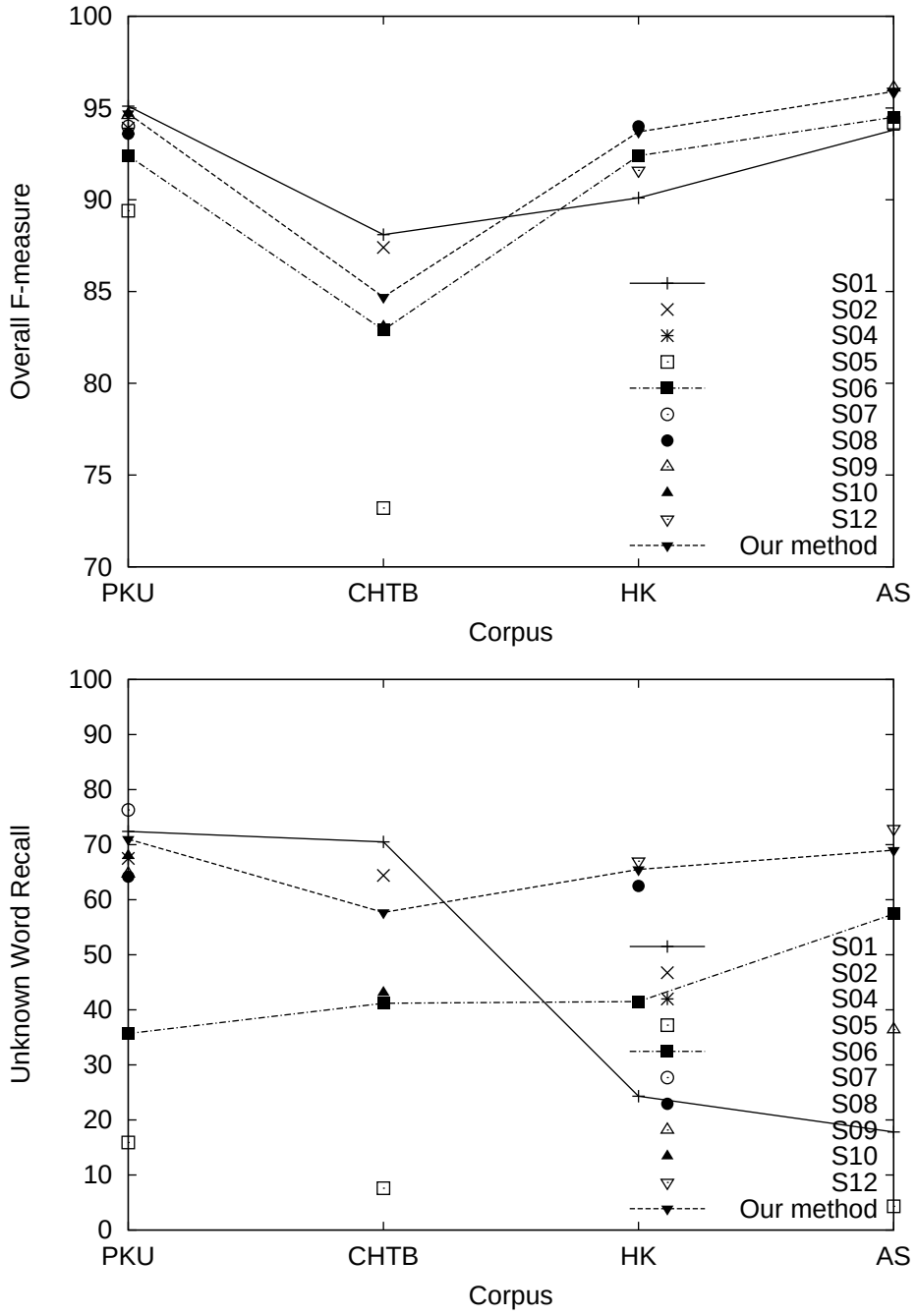



Figure 4.8. Comparison of bakeoff results (overall F-measure and unknown word recall)

words, as in Experiment 4, can be obtained.

### 4.5.2 Unknown Word Detection and Segmentation as Separate Phases

The SIGHAN bakeoff results show that combining word segmentation and unknown word detection in one process produces reasonable result. Without unknown word detection, we get worse result if there are a lot of unknown words in the text. However, while the recall for unknown words increases, the recall for known words decreases. This is because those mistakenly detected unknown words cause the errors in known word segmentation. Our idea relies on the following findings. Introducing one valid unknown word creates one correct word. However introducing one invalid unknown word will possibly make (at least) two words incorrect (one unknown and one known). On the other hand, deleting one valid unknown word makes one word incorrect but deleting one invalid unknown word will possibly make two known words correct. If we can delete as many invalid words as possible, we will be able to increase the accuracy of known words and the overall segmentation.

Furthermore, the same unknown word found in one context may be missed out at another context. Therefore, after unknown word detection, we could rerun the overall segmentation again to include those missing unknown words. In short, our approach is to separate the word segmentation (disambiguation) and unknown word detection into two independent processes, so that we could focus on each problem more thoroughly and more specifically, as to overcome the weakness of the previous approach.

#### Proposed Method

The new proposed method is based on the method as described in Section 4.5.1, where a maximum matching algorithm (MM) combining with Support Vector Machines (SVM) model is proposed to solve the ambiguity problem and unknown word detection at the same time. In that method, if the model focuses on solving ambiguity problem, then the accuracy for known words is higher; and on the contrary if it focuses on unknown word detection, then the recall for unknown words is higher but the accuracy for known words drops. Although there is a balance point for both problems, it is quite difficult to further improve on the accuracy. Two problems are observed. First, since only half of the words from

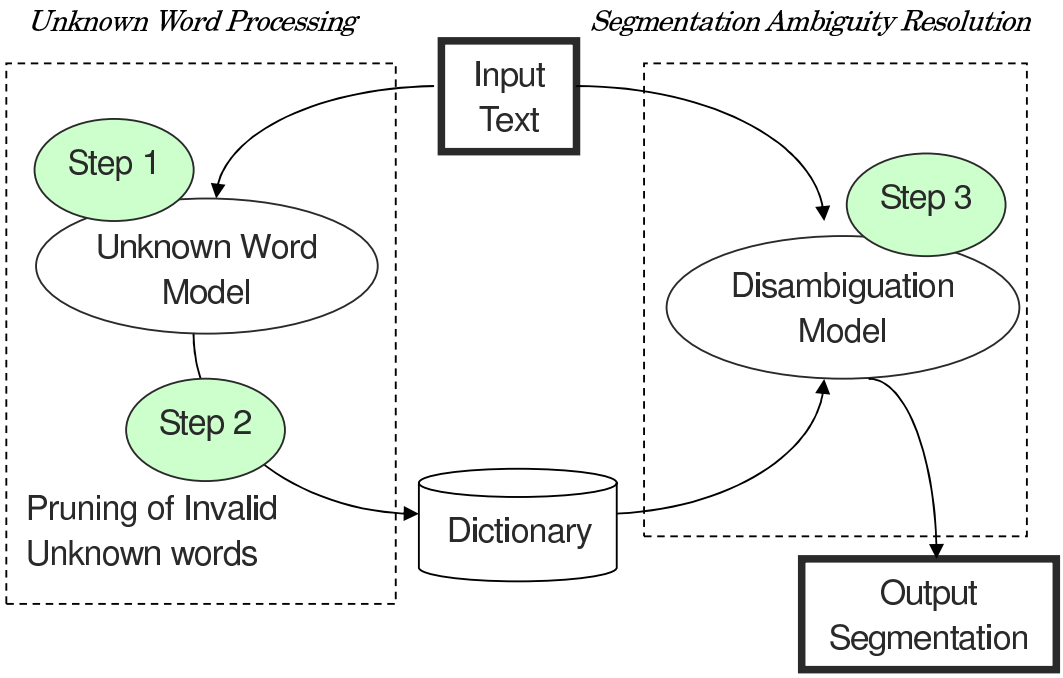



Figure 4.9. Two-phase segmentation flow

the training data are used in the dictionary, some of the known words cannot be segmented correctly as they are not found in the dictionary. Second, only part of the words in the training data are used for the unknown word detection training. In other words, the training of word patterns is not thorough too. The new method intends to make full use of the training data for both problems, so that we can increase the recall for unknown words while at the same time maintains the accuracy for known words.

Figure 4.9 shows the flow of our process. We refer to our two models as the unknown word model and the disambiguation model. First, we use the unknown word model to extract unknown word candidates from the input text and apply a pruning process to eliminate false unknown words. Next, the new words are registered to the disambiguition model's dictionary and the final segmentation is done with the new dictionary. We will describe each step in more detail.

### Unknown Word Processing

The unknown word processing consists of two steps. First, we extract unknown word candidates with the unknown word model. Since not all extracted unknown words are valid, we then apply the second step to eliminate those invalid unknown

words.

### Unknown Word Model

In fact, the unknown word model itself is a complete word segmentation model. It could handle both disambiguation and unknown word detection in one single process. However, while the recall for unknown word increases, the accuracy for known words is affected. Since this model can get optimal result for unknown word detection, we would like to extract the unknown words in this model, meaning those words not found in the dictionary[12]. We then apply a pruning process to the unknown word candidates before registering the new words to the dictionary used in the disambiguition model for final segmentation.

The probability model used is the maximum entropy (ME) model as described in Section 2.3. The ME model is similar to the one described in [48] with different feature templates. Lets $c_i$ be the current character that we want to tag and $i$ stands for the focus position. We use characters (represented by $c_{i-2}$, $c_{i-1}$, $c_i$, $c_{i+1}$, $c_{i+2}$), character types (represented by $y_{i-2}$, $y_{i-1}$, $y_i$, $y_{i+1}$, $y_{i+2}$) and previously estimated tags (represented by $t_{i-2}$, $t_{i-1}$) as the feature templates. We define four character types in our model, digits, alphabets, symbols (including punctuation marks) and hanzi (other Chinese characters). The task is to estimate the tag $t_i$.

1. Characters. Unigram ($c_{i-2}$, $c_{i-1}$, $c_i$, $c_{i+1}$, $c_{i+2}$). Bigram ($c_{i-2}c_{i-1}$, $c_{i-1}c_i$, $c_{i-1}c_{i+1}$, $c_ic_{i+1}$, $c_{i+1}c_{i+2}$).

2. Character types. Unigram ($y_{i-2}$, $y_{i-1}$, $y_i$, $y_{i+1}$, $y_{i+2}$). Bigram ($y_{i-2}y_{i-1}$, $y_{i-1}y_i$, $y_{i-1}y_{i+1}$, $y_iy_{i+1}$, $y_{i+1}y_{i+2}$).

3. Previously estimated tags. ($t_{i-2}$, $t_{i-1}$).

We also regard the problem as a character tagging problem. The ME model will tag each character into one of the 4 possible tags, BIES, as shown in Table 3.1 based on these feature templates.

The outputs of ME model are then converted back to word segments based on the position tags. The conversion becomes complicated when there exists inconsistency in consecutive tags. For example, it is possible that ME model assigns "SE" to two continuous characters, which is logically not allowed. Therefore, we made a slight correction to the output tags as shown in Table 4.12. We look at the current tag or the next tag to decide whether to make a change on previous

[12]The initial dictionary contains all words from the training data.

tag or current tag. The correction does not cover all possible mistakes but only those that are seen in the outputs. The intuition behind is quite simple. We assume that when there is an "I", then is must end with an "E". Alternatively, we may trust the next coming tag, and try to change the former tag. After the correction of inconsistency tags, we convert the characters back to words. We put a word separater (a blank space) in every place that begins with either "B" or "S".

| Condition | Correction |
|---|---|
| prevtag = "I" and curtag = "S" | curtag = "E" |
| prevtag = "B" and curtag = "S" | prevtag = "S" |
| prevtag = "S" and curtag = "E" | prevtag = "B" |
| prevtag = "S" and curtag = "I" | prevtag = "B" |
| prevtag = "I" and curtag = "B" and nexttag = "B" | curtag = "E" |
| prevtag = "B" and curtag = "B" and nexttag = "E" | prevtag = "S" |
| prevtag = "I" and curtag = "B" and nexttag = "S" | curtag = "E" |
| prevtag = "B" and curtag = "B" and nexttag = "B" | curtag = "E" |
| prevtag = "B" and curtag = "E" and nexttag = "E" | curtag = "I" |

prevtag: previous tag, curtag: current tag, nexttag: next tag

Table 4.12. Correction on output tags

From the output word segmentation, those words that are not in the dictionary will be treated as unknown word candidates, which will go through the pruning process as decribed below.

### Pruning of Invalid Unknown Words

We apply two levels of pruning for the detected unknown word candidates. First, pruning by using adjacent words and internal components. Second, pruning by using word formation power.

The first level of pruning is by using adjacent words and internal components. Let $w_{i-1}$, $w_i$, $w_{i+1}$ be three continuous words in the text where $w_i$ is an unknown word candidate and $w_i = e_{i,1}e_{i,2}...e_{i,n}$ where $e_{i,j}$ is a character and $n$ is the length of the word. We assume that if the unknown word forms a known word with adjacent characters or words, then it is not a valid unknown word. Therefore, if any one of the following words exists in the dictionary, then the unknown word is deleted from the list:

1. $e_{i-1,n}e_{i,1}$ - the last character of previous word and the first character of unknown word
2. $w_{i-1}e_{i,1}$ - the previous word and the first character of unknown word
3. $e_{i-1,n}w_i$ - the last character of previous word and the unknown word
4. $w_{i-1}w_i$ - the previous word and the unknown word
5. $e_{i,n}e_{i+1,1}$ - the last character of unknown word and the first character of next word
6. $e_{i,n}w_{i+1}$ - the last character of unknown word and the next word
7. $w_ie_{i+1,1}$ - the unknown word and the first character of next word
8. $w_iw_{i+1}$ - the unknown word and the next word

For those unknown words with length greater than 4 characters, it is possible that it includes a known word inside, especially an idiomatic phrase. Therefore, if either $e_1e_2e_3e_4$ (the first 4 characters) or $e_{n-3}e_{n-2}e_{n-1}e_n$ (the last four characters) exists in the dictionary (except those words that are numbers, alphabets or symbols), then the unknown word candidate is deleted from the list.

The second level of pruning is by using word formation power [31, 11]. We define the word formation power (WFP) as below, where the *pattern* is either S, B, I or E, introduced in Table 3.1.

$$pattern(e) = \frac{count(pattern(e))}{count(e)}$$

$$WFP(w) = B(e_1)\prod_{i=2}^{n-1} I(e_i)E(e_n)$$

Previous researches used a predefined threshold to eliminate the unknown words but we generate the threshold from the training corpus. The threshold is defined as the minimum WFP of words of the same length with the unknown word. Therefore, if the WFP falls in any one of the conditions below, then the unknown word candidate is deleted. However, any unknown word of one character is accepted.

1. $WFP(w)$ is less than the minimum $WFP(x)$ where $length(x) = length(w)$

2. The WFP is less than the total production of every single character in the word, $WFP(w) < S(e_1)S(e_2)...S(e_n)$

3. There exists high probability of single character in the word. Currently we run only on words where $length(w) = 4$, $WFP(w) < S(e_1)S(e_2)B(e_3)E(e_4)$ or $WFP(w) < B(e_1)E(e_2)S(e_3)S(e_4)$

4. Any one of the character in the word appears only as single character word, $S(e_j) = 1$

After the two-level pruning, the unknown word candidates are registered in the dictionary for used in the disambiguation model.

**Segmentation Ambiguity Resolution**

We assume that there is no unknown words in the disambiguition model. If all word candidates can be found in the dictionary, we just need to solve the ambiguity problem here. Similar to the previous method, we use maximum matching algorithm to first segment the text forwards (FMM) and backwards (BMM), but instead of using SVM, we apply maximum entropy (ME) models for classification of characters. This is because SVM requires more computational power. Since we need to create two models, it is better if we can apply a model which can give reasonable results with lower computational power.

During the training of ME model, the dictionary used in the MM models consists of all words from the training data only. While during testing phase, the dictionary is added with the unknown words extracted from the unknown word processing phase. After the initial segmentation using FMM and BMM models, the output words of the MMs are converted into characters, where each character is assigned with a position tag. These tags show the character position in a word, as described in Table 3.1. The output of MMs will be used as features in ME models. For example, for the sentence "迎新春联谊会上" (At the New Year gathering party), FMM has the position tags as "BEBESBE" and BMM has "SBEBEBE". The feature templates are as the following. Output of FMM is represented by $f_{i-2}, f_{i-1}, f_i, f_{i+1}, f_{i+2}$ and output of BMM is represented by $b_{i-2}, b_{i-1}, b_i, b_{i+1}, b_{i+2}$. Each character will be tagged by the ME model based on these features.

1. Characters. Unigram ($c_{i-2}$, $c_{i-1}$, $c_i$, $c_{i+1}$, $c_{i+2}$). Bigram ($c_{i-2}c_{i-1}$, $c_{i-1}c_i$, $c_{i-1}c_{i+1}$, $c_ic_{i+1}$, $c_{i+1}c_{i+2}$).

2. Output of FMM and BMM. ($f_{i-2}b_{i-2}$, $f_{i-1}b_{i-1}$, $f_ib_i$, $f_{i+1}b_{i+1}$, $f_{i+2}b_{i+2}$).

3. Previously estimated tags. ($t_{i-2}$, $t_{i-1}$).

After the character tagging, the same rules for inconsistency tagging (Table 4.12) is applied, and finally the characters are converted back to words.

### Experiments and Results

The experiments are conducted using SIGHAN bakeoff data as described in Section 1.2. We will compare our results with closed testing.

### Evaluation on Unknown Word Extraction

The unknown words are extracted from the testing data using the unknown word model. Table 4.13 shows the accuracy of the unknown word extraction. Only the results on distinct words are shown.

$$
\begin{aligned}
\textit{Recall} &= \frac{\textit{no. of valid extracted unknown words}}{\textit{total no. of distinct unknown words in gold data}} \\
\textit{Precision} &= \frac{\textit{no. of valid extracted unknown words}}{\textit{total no. of distinct unknown words extracted}} \\
\textit{F-measure} &= \frac{2 \times \textit{Recall} \times \textit{Precision}}{\textit{Recall} + \textit{Precision}}
\end{aligned}
$$

| Corpus | Pruning | Recall | Precision | F-measure |
|---|---|---|---|---|
| PKU | Before | 72.40 | 53.59 | 61.59 |
| | After | 66.61 | 61.19 | 63.79 |
| CHTB | Before | 69.58 | 49.78 | 58.03 |
| | After | 68.05 | 58.94 | 63.17 |
| HK | Before | 74.58 | 54.20 | 62.78 |
| | After | 69.72 | 61.91 | 65.58 |
| AS | Before | 74.85 | 51.41 | 60.95 |
| | After | 68.42 | 58.21 | 62.90 |

Table 4.13. Accuracy of unknown word extraction (distinct words only)

We can see from this table that after the pruning, the recalls of unknown words drop, but the precisions increase. However, the balance F-measures have

increased after pruning. As we shall see in the next section, although the recalls of unknown words drop, the overall segmentation by this pruning step improves.

**Segmentation Result**

Figure 4.10 compares our results with the bakeoff results. Overall, we have outperformed almost all the participants except for CTB dataset. In addition, our method has the highest recall for unknown words compared with others.

Table 4.14 shows the detail results of our system[13]. We compare the performance on with or without unknown word detection, and with or without pruning. Apparently, we need unknown word detection to improve the overall segmentation. However, while the accuracy of unknown word increases (as in the row 'With unkword detection'), the accuracy of known words drops. In the next row, we have shown that re-segmentation using the disambiguation model improves the results, as those missing words (found in one context but not the other) can be corrected. Finally, by applying the pruning step, we have again improved on the overall segmentation accuracy because some of the invalid unknown words have been eliminated. However, if the unknown word rate is low, such as AS corpus, it would be better if all the detected words are used for re-segmentation because the pruning steps eliminate too many valid unknown words (5%) relatively.

We have also compared our results with some recent work. Compare with the previous method where a combination of maximum matching algorithm and the state-of-the-art classifier, Support Vector Machines, for segmentation, this method has done a lot better as we can cover better the problem of known words and unknown words. The most recent work on segmentation are reported by [29] and [33]. Nakagawa [29] used word-level and character-level information for segmentation which is similar to our method. He used a Markov model for word-level probability, and maximum entropy model for character-level probability. Then he built a lattice based on both probabilities and solved the problem by using Viterbi algorithm. Both word-level and character-level are used at the same time, and both known word and unknown word segmentation are conducted simultaneously. His method achieved better results than ours. The way that he applied the word-level (HMM) and character-level (ME) information in the lattice is much more efficient than our method. Peng et al. [33] used conditional random

[13]Note that we have converted some ascii characters (such as numbers and alphabets) to GB or Big5 code before processing. This step will automatically make some unknown words become known words.

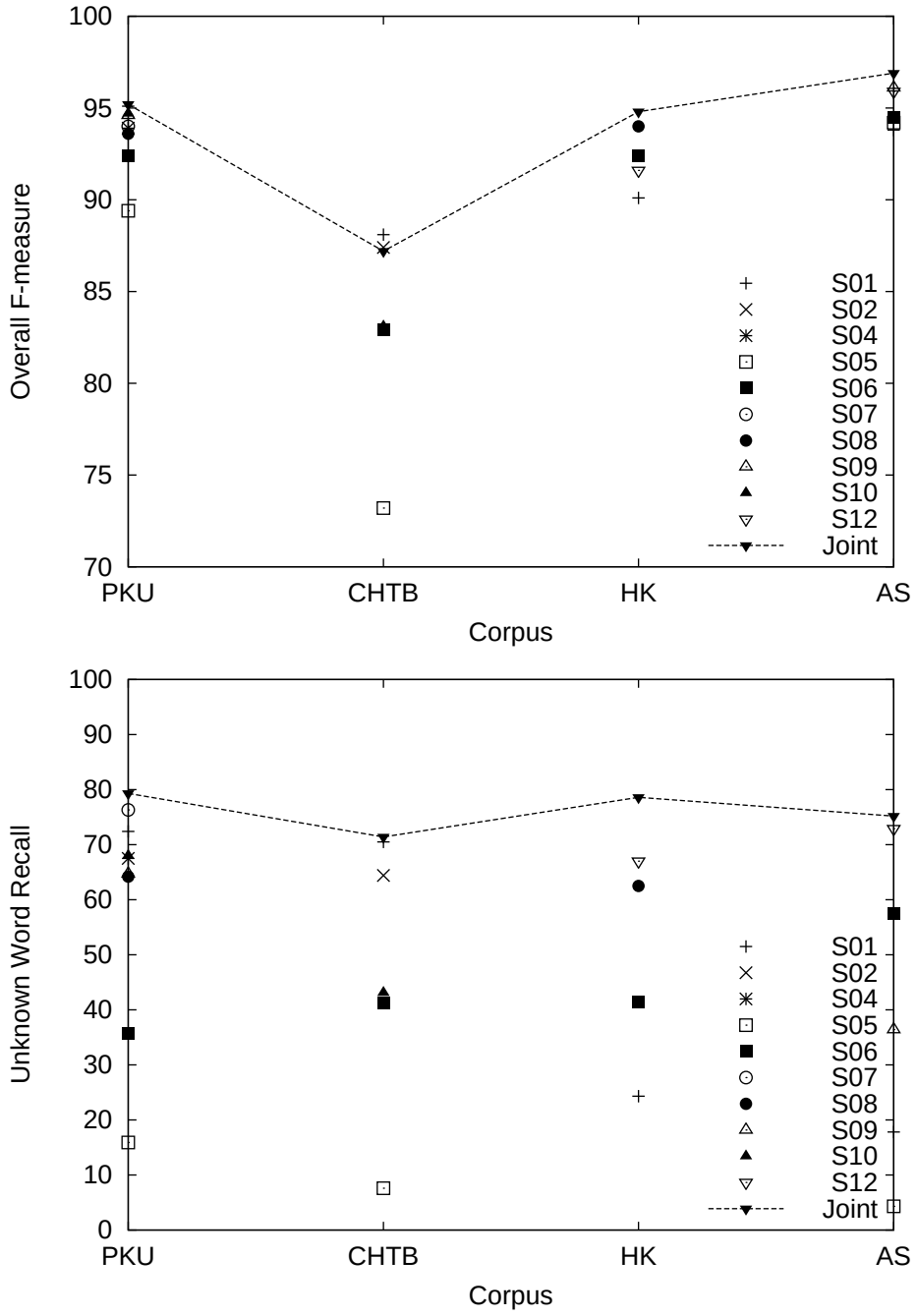



Figure 4.10. Comparison of bakeoff results (overall F-measure and unknown word recall)

| Corpus | Model | *Rec* | *Prec* | *F-mea* | $Rec_{unk}$ | $Rec_{known}$ |
|---|---|---|---|---|---|---|
| PKU | Disambiguition only | 94.7 | 89.7 | 92.1 | 40.0 | 98.7 |
| | With unkword detection | 94.4 | 94.7 | 94.5 | 82.2 | 95.3 |
| | Joint (without pruning) | 94.4 | 95.4 | 94.9 | 82.8 | 95.3 |
| | Joint (with pruning) | 95.0 | 95.4 | 95.2 | 79.3 | 96.2 |
| | Goh (MM+SVM) | 95.5 | 94.1 | 94.7 | 71.0 | 97.3 |
| | Nakagawa (HMM+ME) | 95.7 | 95.2 | 95.4 | 77.4 | 97.0 |
| | Peng (CRF) | 94.7 | 93.5 | 94.1 | 66.0 | n.a. |
| CTB | Disambiguition only | 82.2 | 67.4 | 74.1 | 10.5 | 98.0 |
| | With unkword detection | 84.5 | 85.4 | 85.0 | 70.0 | 87.7 |
| | Joint (without pruning) | 84.3 | 85.9 | 85.1 | 70.6 | 87.4 |
| | Joint (with pruning) | 86.7 | 87.7 | 87.2 | 71.4 | 90.1 |
| | Goh (MM+SVM) | 86.0 | 83.5 | 84.7 | 57.7 | 92.2 |
| | Peng (CRF) | 87.0 | 82.8 | 84.9 | 55.0 | n.a. |
| HK | Disambiguition only | 93.9 | 84.0 | 88.7 | 25.6 | 99.2 |
| | With unkword detection | 94.7 | 93.3 | 94.0 | 79.6 | 95.8 |
| | Joint (without pruning) | 94.7 | 94.2 | 94.4 | 80.6 | 95.8 |
| | Joint (with pruning) | 95.4 | 94.2 | 94.8 | 78.6 | 96.7 |
| | Goh (MM+SVM) | 95.4 | 92.1 | 93.7 | 65.5 | 97.7 |
| | Nakagawa (HMM+ME) | 95.1 | 94.8 | 95.0 | 71.5 | 96.9 |
| | Peng (CRF) | 94.0 | 91.7 | 92.8 | 53.1 | n.a. |
| AS | Disambiguition only | 97.2 | 94.3 | 95.7 | 23.3 | 98.8 |
| | With unkword detection | 97.1 | 96.5 | 96.8 | 79.8 | 97.5 |
| | Joint (without pruning) | 97.0 | 96.9 | 97.0 | 80.2 | 97.4 |
| | Joint (with pruning) | 97.2 | 96.7 | 96.9 | 75.2 | 97.6 |
| | Goh (MM+SVM) | 97.0 | 94.8 | 95.9 | 69.0 | 97.6 |
| | Nakagawa (HMM+ME) | 97.3 | 97.1 | 97.2 | 71.7 | 97.9 |
| | Peng (CRF) | 96.2 | 95.0 | 95.6 | 29.2 | n.a. |

Table 4.14. Segmentation results of joint method

fields (CRF) for word segmentation. Their method is also character-based and they only output 2 labels to show whether there is a word boundary or not. CRFs consider richer domain knowledge and are discriminatively-trained, which are often more accurate. However, in their experiment, the results shown did not out-perform our method. This could be because it is just a first trial on using

CRFs for word segmentation and further survey on the feature sets is probably needed.

## 4.6 Unknown Word POS Tag Guessing

There are not many work done for guessing the part-of-speech of unknown words. Most of the work are on POS tagging for all the words which is usually done after word segmentation processing. The main research is still focusing on word segmentation, because if one could not get a correct segmentation, it is still too early to talk about POS tagging. In [3], work is done for guessing the category of the unknown words by using affix-category association strength, of mutual information and dice. They also consider putting weights on the association strength, as some of the characters are strongly associated with one category, but some characters are loosely associated with a few categories. Therefore, besides internal components, context also play an important role in guessing the POS tags on unknown words.

In our approach, we propose to introduce some features that can be used in machine-learning based method such as Maximum Entropy Models as described in Section 2.3. Our features cover both contextual information and also internal component information.

### 4.6.1 Context Features

The context features are made up from the context, meaning words surrounding the unknown words, as some clues for POS tag guessing.

For example, if we have an unknown word $w_i$ with a context of:

$$h_i = \{t_{i-2}, t_{i-1}, t_{i+1}, t_{i+2}, w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2}\}$$

We can provide some features to it in order to guess it POS tag $t_i$. For example, in the sentence "田/nr 泳/nr 是/v 一个/m 文秀/unk 的/u 川/j 妹子/n" (Tian Yong is a lovely girl from Szechuan), "文秀" (lovely) is a detected unknown word. To determine the POS tag $t_i$ of the unknown word "文秀", the basic context features used are as below.

$$t_{i-2} = \text{v},\ t_{i-1} = \text{m},\ t_{i+1} = \text{u},\ t_{i+2} = \text{j},$$
$$w_{i-2} = 是,\ w_{i-1} = 一个,\ w_{i+1} = 的,\ w_{i+2} = 川$$

We call this as unigram feature. Furthermore, bigram feature may be also helpful in the tagging. For the same example, the bigram features are as below.

$$t_{i-2}t_{i-1} = \text{vm},\ t_{i-1}t_{i+1} = \text{mu},\ t_{i+1}t_{i+2} = \text{uj}$$
$$w_{i-2}w_{i-1} = \text{是一个},\ w_{i-1}w_{i+1} = \text{一个的},\ w_{i+1}w_{i+2} = \text{的川}$$

These will provide some context features to be used in the Maximum Entropy models.

### 4.6.2 Internal Component Features

Besides context features, the clues that are used to guess the POS tags are always the internal components of the words. For example, a word that begins with character "非" is normally a noun-modifier, and a word that ends with character "化" is normally a verb and etc. Therefore, the prefix and the suffix of a word are the important clues for telling the POS tags. In Chinese, there are more suffixes than prefixes. Although we do not analyze again the components of an unknown word whether it contains prefix or suffix, we just take the first character and the last character of the word as features.

The other feature is the length of the unknown words. Normally if a word has 4 characters, than it is probably a collocation or idiomatic phrase. A Chinese person name normally has one or two characters only. If a word has more than 4 characters, then it may be a proper noun, such as foreign name etc. Therefore, the unknown word length can play an important role too.

For the same example above, prefix($w_i$) = "文", suffix($w_i$) = "秀" and length($w_i$) = 2.

### 4.6.3 Experiments and Results

The ME model used for POS tag guessing was trained on the unknown words only. The setting is the same as in the experiments in Section 4.4.2. The unknown words are those words that exist only once in the dictionary. With this setting, it covers the unknown POS tags more evenly. There are 20,876 unknown words in the training data. During the testing, since not all unknown words were detected correctly, there was no point to guess the POS tags for wrongly detected unknown words. Therefore, only those unknown words that were correctly detected were tested. From the experiment output from Section 4.4.2, there are 2,751 correctly detected unknown words from forward chunking (indicated by *Forward* in Table

4.15), and 2,823 from backward chunking (indicated by *Backward* in Table 4.15). We also tested with all unknown words (4,845 words) in the test data (indicated by *All* in Table 4.15).

As shown in Section 4.4.2, we obtained only about 64% precision for unknown word detection. Therefore, we evaluated the POS tag guessing results in two ways. The first was evaluated based on the correctly detected unknown words, and the second was based on all detected unknown words (of course those wrongly detected words are treated as wrong POS tags as well). We evaluated the results with the following equations.

*POS accuracy of correctly detected unkwords*

$$= \frac{\#\ of\ correctly\ POS\text{-}tagged\ unknown\ words}{\#\ of\ correctly\ detected\ unknown\ words}$$

*POS accuracy of all detected unkwords*

$$= \frac{\#\ of\ correctly\ POS\text{-}tagged\ unknown\ words}{total\ \#\ of\ detected\ unknown\ words}$$

Table 4.15 shows the results of the POS tag guessing for unknown words. *Forward* shows the results using forward chunking is SVM as test data, *Backward* shows the results using backward chunking, and *All* shows the results using all unknown words in the test data. The rows marked with unigram shows the results where we use only the unigram context features. The rows marked with +bigram show the results of unigram plus bigram context features. The remaining rows (+others) are the results obtained if we also include the internal component features. We obtained about 67-78% accuracy if the unknown words were correctly detected and 41-50% for overall detection. The results also show that combining unigram and bigram features, with the internal components features gives the best result.

After assigning the POS tags to the unknown words, we evaluated the POS tagging performance. Table 4.16 shows the overall POS tagging results. We obtained an F-measure of 91.58, an increment of 1.85, compared with using only HMM model. We could not get a good result overall because even the known words were tagged wrongly with the baseline HMM model. Furthermore, mistakes

| Features | Test data | POS accuracy of correctly detected unkwords | POS accuracy of all detected unkwords |
|---|---|---|---|
| unigram | *Forward* | 67.21% | 44.40% |
| | *Backward* | 67.45% | 41.52% |
| | *All* | 59.48% | - |
| +bigram | *Forward* | 67.65% | 43.62% |
| | *Backward* | 67.84% | 41.75% |
| | *All* | 60.52% | - |
| +bigram+others | *Forward* | 77.72% | **50.12%** |
| | *Backward* | **78.00%** | 48.02% |
| | *All* | 71.27% | - |

Table 4.15. Results of POS guessing for unknown words

| | Recall | Precision | F-measure |
|---|---|---|---|
| Only using HMM | 91.06 | 88.43 | 89.73 |
| HMM+Character-based+SVM/F | 90.27 | 90.22 | 90.25 |
| HMM+Character-based+SVM/B | 90.13 | 90.25 | 90.19 |
| HMM+Character-based+SVM/F+POS/ME | 92.08 | 91.01 | 91.54 |
| HMM+Character-based+SVM/B+POS/ME | **92.11** | **91.07** | **91.58** |

Table 4.16. Results of overall POS tagging

made by unknown word detection have also caused some correctly segmented words to be wrong at final stage.

We used the ME model to guess the POS tags of unknown words only. For those known words that have been tagged by the HMM model, they remained unchanged. The problem is that if the left-right context of an unknown word is tagged wrongly by the HMM model, then the unknown word will probably be tagged wrongly as well. Our HMM model achieve only an F-measure of 89.73 for initial POS tagging, therefore it is very difficult to guess the POS tags of unknown words as the initial tagging was imperfect.

Tseng et. al. [41] proposed to use a rich feature set for unknown POS tag guessing using maximum entropy Markov models. Their features include lexical

feature, affixation, morpheme of prefix and suffix, external resources with Sinica corpus, verb affix, radicals, named entity morpheme and length of word. The experimental results on Penn Chinese Treebank show that each of these features help to improve the accuracy bit by bit. The best result is by using all of them. The purpose of the research is to show that the morphological features could help to POS tag unknown words across language varieties such as media resources from Mainland China, Hong Kong and Taiwan. The unknown word rate increases if the training data is taken from different sources. The average unknown word rate is 12%. The accuracy of the system is 91.97% overall and 79.86% for unknown words. If the training data also consists of texts collected from different sources, then the overall accuracy is 93.74% and 86.33% for unknown words. Although the accuracy of the POS tagging is high, the complexity of the system is with the preparation of the feature set. It needs a lot of extra resources to build the feature set.

## 4.7 Conclusion

To conclude the studies of this chapter, we would like to raise a few issues here. First, whether we need a proper dictionary in morphological analysis, or we can just build the dictionary from the training corpus? Second, should we combine word segmentation with unknown word detection, or they should be done in separate phases?

Following our experiments, we conclude that a proper dictionary would help a lot in real world morphological analysis. Although a dictionary can be built from a corpus, it will not cover all common words from real world text unless the corpus is huge enough and it is built from various genres, resources and domains. However, no one can find such a training corpus until now. A proper dictionary will reduce our effort to detect the unknown words of "common" words, which should not be a burden to the system. The system should focus more on compound words and morphological derived words, named entities and factoids. Therefore, in order to build a practical system, a proper dictionary is needed.

As discussed above, one can embed an unknown word detection model inside the word segmentation model. The merit of doing this is that we can detect the unknown words immediately and the overall segmentation accuracy can be improved. However, since the unknown word detection can never yield hundred percent precision, it also means that we will over-generate the words, and pro-

duce some false unknown words to the output. Of course, in another words, the accuracy of known words will be deteriorated as well. We believe that one would assume that all output words from word segmentation process should be correct ones. Even if we could not produce some correct unknown words, we would not want to have any false unknown words in the text. Therefore, it is better to separate the word segmentation and unknown word detection processes. The unknown word detection process should produce only real unknown words, which means high precision, with reasonable recall. The word segmentation process should focus only on solving ambiguity problems with the aid of a dictionary. In this case, there will be no false unknown words in the segmentation output. Our design of the morphological analyzer in Chapter 5 will follow this direction.

## 4.8 Summary

In this chapter, we have discussed the problem of unknown words and proposed some methods towards solving it. We have shown that character-based tagging could produce better accuracy for unknown word detection because it is easier to join characters to form new words. Besides, the result is better if we focus on the detection for certain type of unknown words. This is especially true for named entity detection such as person names where a separate feature set is provided. We also discussed the relation between known word segmentation and unknown word detection and concluded that they should be done in separate phases in order to get optimum results. Finally unknown word POS tag guessing is better by joining contextual features with internal component features because these two features have the most influences on deciding the POS tags.

# Chapter 5

# Chinese ChaSen - a Practical System

To date, a freely usable Chinese morphological analysis system is still not widely available. Furthermore, there is no single segmentation standard for all tagged corpora provided by different institutions. Some systems are available which are developed by different corpora providers, according to their segmentation standard. For example, Peking University (PKU) corpus has their own system and Sinica corpus also has their own system. The systems cannot be used interchangeability because the segmentation outputs and POS tagsets are different. Therefore, it is necessary to build a system (or with some modification) for each segmentation standard type. Penn Chinese Treebank (CTB) is another tagged corpus provided by Linguistic Data Consortium (LDC) [24]. The segmentation standard and POS tagset is again different from PKU [19, 51] and Sinica [18]. Since CTB corpus is a bracketed corpus, it can also be used for training of parsing. Therefore, it is widely used by a lot of researches in Chinese language understanding. As far as we know, there is still no system (freely) available for Penn Chinese Treebank standard. Since this treebank is widely used by a lot of researches that do parsing, probably it is a good idea to build a practical morphological analyzer for CTB standard.

## 5.1 Penn Chinese Treebank (CTB)

Penn Chinese Treebank is a segmented, part-of-speech tagged and fully bracketed corpus produced by the Linguistic Data Consortium. It is an ongoing project. The project aims to built a corpus annotated with morphological, syntactic, se-

mantic and discourse structures[1].The latest CTB version 5.0 has about 500,000 words. Even if we can extract all words from CTB to build a dictionary, the number is not enough for a real world working system. Therefore, we plan to enlarge the dictionary by using some unknown word extraction methods.

For evaluation purpose, CTB version 4.0 (about 437,000 words) is used as the training corpus. The exclusive part from version 5.0 (about 110,000 words) is used as the testing data. The basic dictionary is built from the training data only.

## 5.2 New Segmentation Unit

As we have mentioned before, no single segmentation standard is agreeable across different instituitions. In SIGHAN bakeoff [37], we could see that different instuitions have provided different segmentation standards. Most of the disagreements in the standards come from the segmentation of morphologically derived words [46] and named entities. For example, some would say that "孩子们/NN" (children) as one word and some would prefer to it as two words "孩子/NN" and "们/M". For named entities such as Chinese person names, whether a string of a surname and a given name should be one word or two words, is also under argument. It would be nice if we can build a system that suits everyone's needs but it sounds almost impossible. Wu [46] tried to define tree structures to morphologically derived words but that will need a lot of human efforts as they are all based on rules defined. Gao et al. [14] have tried to modify their current system to adapt for all segmentation standards in SIGHAN bakeoff using the transformation-based learning method [2]. Since we would like to build a system for CTB, we try to define our segmentation units as close as to the CTB standard, or at least to be able to modify back to the CTB standard easily.

There are a few changes that we have made on the CTB corpus to suit our purpose and to ease our processing. We refer to this new segmentation units as minimal segmentation units. The changes are made on proper names, foreign words and numeral type words only. Figure 5.1 shows the desired output of minimal unit segmentation and the transformation to the CTB standard. The original segmentation and POS tag guidelines can be found in [10] and [9]. With our new segmentation units, we have increased the number of POS tags from 33

[1]The semantic annotation is denoted as Chinese Proposition Bank and discourse structure is denoted as Chinese Discourse Bank.

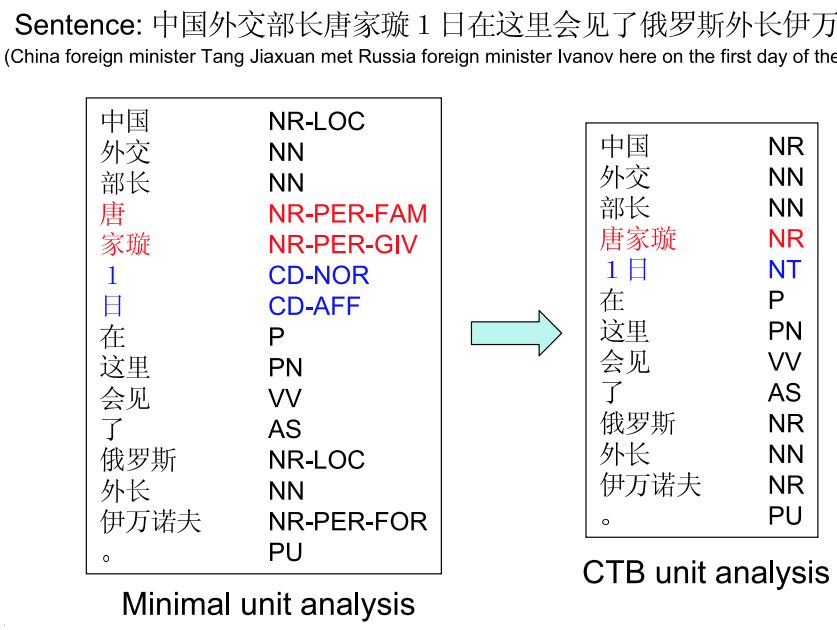



Figure 5.1. Transformation from minimal unit segmentation to CTB unit segmentation

to 42 tags. Appendix B shows the POS tagset of our definition.

There are two advantages of defining the minimal segmentation units. First, we can reduce the size of the dictionary by eliminating those productive numeral type words and foreign words. Second, splitting family names and given names for CJK names in the dictionary can make the combinations of two of them more freely in the text. Furthermore, this design can be applied to the analysis of compound words and morphological derived words in the future. These two groups of words are the main types of unknown words in the text. If we define the minimal units for compound words such as "策划/NN 室/NN" (planning room), then we can combine them into "策划室/NN" in the second layer. In this case, we can create a compound word dictionary which can show the internal structure of the compound words more precisely. Currently, our implementation does not include the analysis of compound words and morphological derived words. We now describe each modification in the following sections.

### 5.2.1 Proper Names

In CTB, all proper names are grouped under one single POS tag as NR. In our new segmentation units, we divide the proper names into 7 new groups. This

is because we think that the new groups are more informative and are useful for named entity extraction in the future. First, the person names are identified and 4 groups are introduced, family names (NR-PER-FAM), given names (NR-PER-GIV), foreign names (NR-PER-FOR), and other names (NR-PER-OTH). The family names and given names apply only to Chinese, Japanese and Korean (CJK) names. Originally, CTB does not split family names and given names but we split them into two units. Then, we also define place names (NR-LOC), organization names (NR-ORG) and other proper names (NR-OTH). Based on these definitions, we have manually made the changes to the corpus.

### 5.2.2 Foreign Words

Foreign words refer to those that consist of alphabets. As the combination of alphabets is arbitrarily, we do not want to register these words in the dictionary. In our dictionary, only ａ－ｚ and Ａ－ Ｚ are registered as POS tag FW. As a result, we must cut the foreign words into smaller units, meaning one-character units, in our training corpus. For example, the original word "Ｐ／Ｅ 值/NN" (P/E value) now becomes "Ｐ/FW ／/PU Ｅ/FW 值/NN". The weakness of this changes is that we will lose the information of the original POS tag, in this case NN. However, the merit is that we do not need to bother about the foreign word segmentation and POS tagging. Although currently we do not register any foreign words in our dictionary, it is possible to add these words in the dictionary if we think that they are frequently used words and it is necessary to register them.

### 5.2.3 Numbers

The last changes are on numeral type words. This group of words is very productive and it is impossible to register all possible combinations in the dictionary. Therefore, our dictionary contains only characters ０－９ and 零–九(zero-nine), 十(ten), 百(hundred), 千(thousand), 万(ten thousand) etc. We also cut the numeral words into one character unit. However, it is simple if we want to combine them in the later stage. There are three types of numeral type words in the corpus, time nouns (NT:三月, March), cardinal numbers (CD:十多, more than ten) and ordinal numbers (OD:第九, number nine). In these examples, there are some characters in the words which are not numbers. Therefore, we introduce 6 new POS tags to tag these numeral words. CD-NOR is used to tag all numbers

( 0 – 9 , 零–九, etc), OD-NOR and NT-NOR are used to tag those words that do not consist of numbers, such as "首" (first) and "半夜" (midnight). Then, we also introduce CD-AFF, OD-AFF and NT-AFF to cater for those characters (affixes) which are not numbers but exist in the numeral type words. Finally the new segmentation for the earlier examples become "三/CD-NOR 月/NT-AFF", "十/CD-NOR 多/CD-AFF" and "第/OD-AFF 九/CD-NOR". We have made the changes in the corpus based on these rules. We also extracted the affixes of these words and added them to the dictionary.

## 5.3 Preparation of System Dictionary

Our morphological analyzer can only segment and POS tag known words that exist in the system dictionary. It does not deal with unknown words in a straightforward manner. We will extract unknown words in another module. The extracted unknown words must be verified by human before adding into the system dictionary. There are two reasons why we choose to do it this way. First, omitting unknown word problem in the analysis process can reduce the complexity of the system. We just need to focus on solving the ambiguity problem: segmentation and POS tagging. Second, we do not want to introduce false unknown words in the output of the morphological analysis. Till date, there is no system that can produce unknown word detection with 100% precision. In other words, some of the extracted unknown words are incorrect. If we use all of them in the analysis, we will output these false unknown words. We prefer wrong segmentation with chunks of correct units, rather than a segment that never exist in Chinese text. To support this statement, consider the following examples.

(1) 回来/ 请/ 家教/ 老师/ 上课/
come back/ hire/ family education/ teacher/ give a class
(Come back home and hire a family teacher to give a class)
(2) 回来/ 请/ 家/ 教/ 老师/ 上课/
come back/ hire/ home/ teach/ teacher/ give a class
(3) 回来/ 请家/ 教/ 老师/ 上课/
come back/ ??/ teach/ teacher/ give a class

Suppose that "家教" is an unknown word. The first sentence shows the correct segmentation. The second and third sentences show some wrong segmentations.

The second sentence shows a segmentation without unknown word detection. The third sentence shows a segmentation with unknown word detection but the detected word is wrong. Although both segmentation outputs are wrong, we assume that most people will prefer the second sentence rather than the third sentence. There is no incorrect words in the second sentence but there is a false unknown word "请家" in the third sentence, which does not carry any meaning in Chinese.

Therefore, we want to build a system dictionary that will contain a large number of words using unknown word detection method. However, we must make sure that all the words registered in the dictionary are correct Chinese words.

### 5.3.1 Extraction from CTB

Based on the segmentation units that have been described above, we have made the changes to the corpus accordingly. After that we extracted all words from CTB 4.0 to build our initial dictionary. We leave the exclusive part in CTB 5.0 as testing data (for both evaluation on morphological analysis and unknown word extraction). We also removed some noise which we found not suitable to be used as the entries in the dictionary[2]. Finally, we built an initial dictionary that contains 33,438 entries (word/POS word pairs, a word can have more than one POS tag). There are 28,390 words if we consider the word tokens only. To build a practical system, this number is too small. Therefore, we must find some ways to increase the number of entries in the dictionary.

### 5.3.2 Collection of Proper Nouns from Web

We collected various proper nouns from the web. These include place names (5,365 place names in China), country and capital names (391), and Chinese family names (436). These names are quite common on the web and they can be used directly in our system.

[2]For example, the phrase （/PU 四/NR-LOC ）/PU 川/NR-LOC （/PU 西/NR-LOC ）/PU 藏/NR-LOC 公路/NN (the road between Sichuan and Tibet) has four location names which we think are not real abbreviations to be used normally.

### 5.3.3 Unknown Word Extraction from Chinese Gigaword

Chinese Gigaword (CGW) is a raw text corpus provided by LDC. The size is about 1,118,380K Chinese characters. We use this corpus to extract new words to add into our system dictionary.

#### General Unknown Word Extraction and POS Tag Guessing

The unknown word extraction method used is similar to the one described in Section 4.5.2. In this approach, we assign each character with a character type such as NUMber, ALPhabet, SYMbol or HANzi, and label each character with BIES tagset[3]. We use Maximum Entropy Models for the character-based tagging. We found that this method gives us the best recall of unknown words although the precision is a bit lower. In Section 4.5.2, we have also applied some pruning steps to delete some false unknown words. However, since this step deteriorates the recall, we do not do the pruning in this round as our purpose is to collect as many unknown words as possible.

For evaluation, we conducted the experiments with the test data in CTB 5.0. Using the initial dictionary, there are about 9.2% of unknown word/POS pairs in the test data. Out of this, 7.8% are unknown words, 1.3% are unknown POS (the words exist in the dictionary but are with different POS tags). Currently, our method can solve only the problem of unknown word but not the unknown POS. We leave the unknown POS problem for the future work.

The features that we use in the ME model for tagging are: 2 characters each from left and right contexts $(c_{i-2}, c_{i-1}, c_i, c_{i+1}, c_{i+2}, c_{i-2}c_{i-1}, c_{i-1}c_i, c_ic_{i+1}, c_{i+1}c_{i+2})$, character types $(y_{i-2}, y_{i-1}, y_i, y_{i+1}, y_{i+2}, y_{i-2}y_{i-1}, y_{i-1}y_i, y_iy_{i+1}, y_{i+1}y_{i+2})$, and 2 previously tagged labels $(p_{i-2}, p_{i-1})$. We get 72.2% recall for tokens, 72.1% recall for types, and 50.6% precision for types. In another words, although we can get quite high recalls but the precision is not so good. Only about half of the words are correctly extracted. However, since we want to increase the size of the dictionary, higher recall means that we get more words.

After unknown word extraction, we need to assign POS tags to them. We used the same method as described in Section 4.6. The training data are those words that exist only once in the corpus. This covers all major unknown POS tags. The top 11 POS tags for unknown words are: NN, VV, NR-PER-GIV, NR-LOC, JJ, NR-ORG, NR-OTH, NR-PER-FOR, VA, NR-PER-OTH, and AD (the

[3]B - first character, I - intermediate character, E - last character, S - single character word.

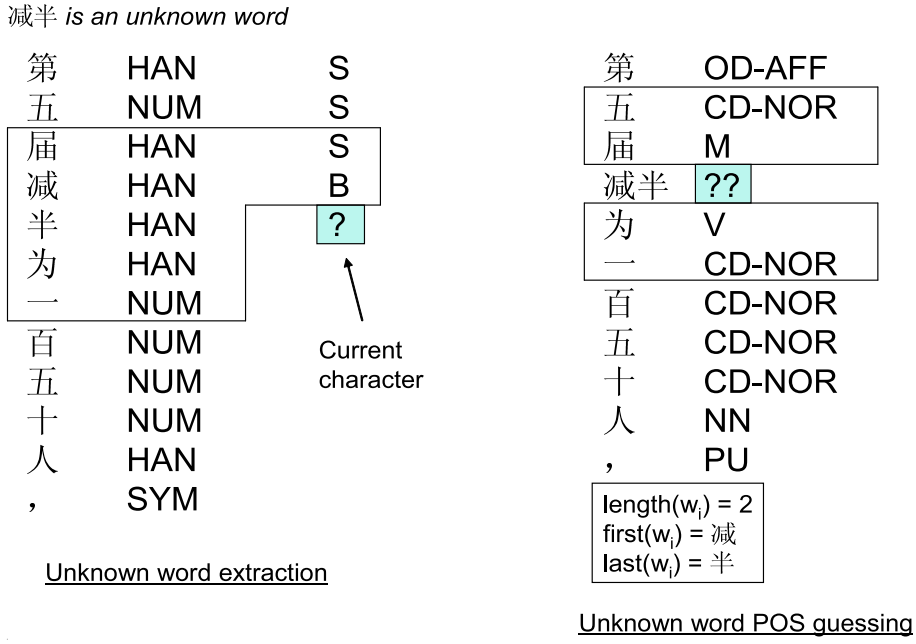



Figure 5.2. Feature sets used for unknown word processing

rest are really minor, only about 1.5%)[4]. The features used are the context words ($w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2}, w_{i-2}w_{i-1}, w_{i-1}w_{i+1}, w_{i+1}w_{i+2}$), POS tags ($t_{i-2}$, $t_{i-1}$, $t_{i+1}$, $t_{i+2}, t_{i-2}t_{i-1}, t_{i-1}t_{i+1}, t_{i+1}t_{i+2}$) and the internal component features (first character, last character and word length). The context features (known words and POS tags) are taken from the morphological analysis (will be described in Section 5.4.1). Figure 5.2 shows the feature sets used for unknown word extraction and unknown word POS tagging. Our experiment results show that the accuracy of tagging unknown words is 68.2% if only context features are used and 75.2% if all features are used.

Using this method, we tried on a small part of the CGW corpus for testing purpose. Using file xin200209, we extracted 2,258 new words for the first 625 sentences (45,836 characters). Manually checking on the output, we divide the results into three categories.

- Both word and POS are correct 40.7%(921/2258)
- Word is correct but POS is wrong 34.1%(772/2258)
- Both word and POS are wrong 24.6%(566/2258)

[4]The description of the POS tagset can be found in Appendix B.

From this result, we can say that 40.7% of the words can be used directly. 34.1% words can be used after correcting the POS tags. In short, we estimate that 74.9% of the words extracted from CGW corpus are usable in our dictionary.

In a real run of the method applied to CGW, we extracted 51,412 word/POS pairs from file xin200209. Then we hire 4 native Chinese to check on the words manually in one month time. 14,537 words are correct, 10,643 words have been corrected with their POS tags. Since it is done manually, we also ask the checkers to correct some of the word boundaries to obtain correct words (7,785 words). Finally, manual checking on the words gives us a total of 26,281 (51.12%) correct words[5]. Although the result is a bit lower than our estimation, we still manage to get quite a number of new words. Table 5.1 shows the distribution of the words for each POS tag.

| POS Tag | No. of words |
|---|---:|
| AD | 309 |
| JJ | 652 |
| LC | 62 |
| M | 71 |
| NN | 10,287 |
| NR-LOC | 2,575 |
| NR-ORG | 1,021 |
| NR-OTH | 485 |
| NR-PER-FAM | 152 |
| NR-PER-FOR | 2,180 |
| NR-PER-GIV | 1,701 |
| NR-PER-OTH | 86 |
| NT-NOR | 80 |
| VA | 734 |
| VV | 5,806 |
| Others | 80 |
| Total | 26,281 |

Table 5.1. Distribution of new words obtained through manual verification

[5]There are some overlapping among these groups, so the final total is not the same as the total of all groups.

### Person Name Extraction

In [15, 52], we have seen that one can get better results if we just focus on the extraction of a certain type of unknown words, such as personal names. This is because we can train the system to be more specific to the type by providing specific features to it. For example, a Chinese person name normally comprises of a family name and a given name. A family name is normally one character long (very few with two characters) and it is almost a closed set. If we can provide the information about the family names, then it will be easier to guess the given names. At our disposal, we also have a set of characters that are possible to be used in transliteration foreign names. These will provide some extra features for extraction.

The method that we use here is similar to the one described in Section 4.4.2. First, an HMM-based analyzer is used to segment and POS tag the text, then an SVM-based chunker is used to the extract the person names. The difference is in the way of preparing the dictionary. Since our target is the Chinese given names and foreign names, we create a dictionary which consists of none of the both. It will make the HMM analyzer to wrongly segment all the names. In the second step, names are extracted by chunking process using SVM. We also provide family names and transliteration characters as the features. We assign each character with one of these 4 tags, FAM (family name), FRN (transliteration character), BTH (can be used for both), OTH (not in use for both). Currently we have collected 482 family names and 581 transliteration characters. The context window is three characters at both left and right sides. Figure 5.3 shows a snapshot of the chunking process.

We have conducted an experiment using the CTB 5.0 test data. In CTB 4.0 there exist 4,190 given name and 926 foreign name instances. We use these data for training. In the test data, there are 1,157 given names and 194 foreign names. Table 5.2 shows the results of our method. Although we could get quite good accuracy with CJK given names, we could not get a good result with foreign names. This may be because the training data for the foreign names is not enough.

Using this method, we extracted 4,622 person names from CGW, file xin199101. After manual checking, we obtained 3,976 (86%) words which are usable to our system. Since it is done manually, we also asked the checkers to correct some of the wrong POS and reassign boundaries if necessary. Table 5.3 shows the distribution of the words for each POS tag. The accuracy for given names and

| Position | Char. | POS-position | Char. Type | Chunk |
|---|---|---|---|---|
| *i - 6* | 国 | NN-B | SUR | O |
| *i - 5* | 大 | NN-E | FRN | O |
| *i - 4* | 议 | NN-B | OTH | O |
| *i - 3* | 长 | NN-E | FRN | O |
| *i - 2* | 苏 | NR-LOC-B | BTH | O |
| *i - 1* | 南 | NR-LOC-I | BTH | NR-PER-GIV-B |
| *i* | 成 | VV-S | SUR | ? |
| *i + 1* | 主 | JJ-B | OTH | |
| *i + 2* | 导 | JJ-E | OTH | |
| *i + 3* | 议 | NN-B | OTH | |
| *i + 4* | 事 | NN-E | OTH | |

'The chairman of the National Assembly Su Nancheng leads the meeting', Char. - Chinese character, POS-position - POS tag plus position tag, Char. Type - character type, Chunk - label for unknown word

Figure 5.3. An illustration of the features used for chunking of person names

| | Recall | Precision | F-measure |
|---|---|---|---|
| CJK given name | 89.02 | 70.12 | 78.45 |
| Foreign names | 39.69 | 56.62 | 46.67 |
| Average | 81.94 | 68.97 | 74.90 |

Table 5.2. Results for Person Name Extraction

foreign names only is about 66%, which follows our estimation during the testing experiments.

### Checking with other Resources

From our past experience, we realize that manual checking on unknown words is a time consuming task. Therefore, we also look for other solutions to speed up the process. One way is to use other resources for double checking as described below.

Sinica Corpus

Sinica corpus [18] is the first tagged balanced corpus which contains about 5 millions words. Texts are collected from different areas and classified according

| POS Tag | No. of words |
|---|---|
| NN | 271 |
| NR-LOC | 413 |
| NR-ORG | 47 |
| NR-OTH | 40 |
| NR-PER-FOR | 1,096 |
| NR-PER-GIV | 1,947 |
| VV | 97 |
| Others | 65 |
| Total | 3,976 |

Table 5.3. Distribution of new words obtained through manual verification on person names

to five criteria: genre, style, mode, topic, and source. Therefore, this corpus is a representative sample of modern Chinese language. Moreover, the size is 10 times larger than CTB.

Sinica corpus uses a different POS tagset as CTB corpus. It has 46 simplified POS tags, as compared to 33 tags in CTB. Basically the segmentation standard between CTB and Sinica is very similar [10] but there are also some differences. For example, "学生们/NN" (students) is segmented as one word in CTB corpus but as two words "学生/Na 们/Na" in Sinica corpus. From Sinica corpus, we could get around 150,000 distinct words. Leaving out the copyright problem to use the resources from Sinica, we cannot use the list of words directly from Sinica in our system since the segmentation standard is different. Therefore, we choose to use it in another way. First, we extract the new words from CGW using our unknown word extraction model. Instead of checking manually by human, we double check the words with Sinica corpus entries. If the words are found, then we assume that these words are correct ones. Although using a corpus requires copyright clearance but using the words in the corpus should not violate any legal law. No one can say that a word belongs to them as it is publicly used everywhere. However, we have obtained the permission from the Academia Sinica verbally to use their corpus as a reference.

In order to do this, first we need to compare the POS tagsets to find out equivalent POS tags. Table 5.4 shows the equivalent POS tags that we use for comparison since these are the words with high productivity. However, there

are a few pairs which cannot be used. First is the proper names. Since we have divided the proper names into more detailed categories, Sinica remains the same as original CTB which has only one tag, we cannot differenciate between them. In Sinica, place names also contain words such as the room number, street number, common place noun etc, so it is also not suitable to be used as we need to differentiate them separately. Same to time nouns which contain numeral type words by which to our definition should not be in the system dictionary. At the end, we use only the seven types of POS tags (marked with *) as shown in Table 5.4. In Sinica corpus, if a verb is used as a noun in that certain context, it is tagged with an additional tag +nom. However, it is just a simple NN in CTB. Therefore, we also include the pairs verb-noun in the list for references.

| POS Tag | Sinica Tag | CTB Tag |
|---|---|---|
| *Adjective | A | JJ |
| *Adverb | D, Da, Dfa, Dfb, Dk | AD |
| *Common noun | Na | NN |
| Proper name | Nb | NR-* |
| Place name | Nc (incl. num) | NR-LOC |
| *Localizer | Ncd | LC |
| Time noun | Nd (incl. num) | NT |
| *Measure noun | Nf | M |
| *Verb | V?[?], (+nom) | VV, NN |
| *Stative verb | VH[?], (+nom) | VA, NN |

Table 5.4. Matching between Sinica and CTB POS tagset

Since Sinica corpus is written in traditional Chinese, we have to convert it to simplified Chinese before we can use it for comparison. We exclude those words that cannot be converted successfully[6]. As a result, we obtain a list of 105,030 word/POS pairs for comparison.

We applied the unknown word extraction model in Section 5.3.3 to the whole CGW corpus. Then we compared the extracted words with the Sinica dictionary. We manage to extract 33,286 new entries which we are sure to be correct ones since they also exist in Sinica corpus. Table 5.5 shows the distribution of the words for each POS tag. Most of them are common nouns, followed by verbs.

[6]Conversion between traditional and simplified Chinese is not a trivial task, please refer to [16] for details.

| POS Tag | No. of words |
|---|---|
| AD | 351 |
| JJ | 398 |
| M | 38 |
| NN | 20,947 |
| VA | 1,741 |
| VV | 9,811 |
| Total | 33,286 |

Table 5.5. Distribution of new words obtained through automatic verification

Chinese Given Names

We also manage to download a list of Chinese names from the web[7]. The names were taken from the Taiwan Joint College Entrance Examination (JCEE)[8]. The names are those high school graduates who passed the exams from year 1994–1997. They also provide a list of family names and a list of given names together with their frequencies. From a total of 217,913 uniq names, they give 619 distinct family names and 75,581 distinct given names[9]. We found out that there are quite a lot of noise in the files because the way they cut the unique names into family names and given names are not so reliable[10]. Therefore, we decided not to use the family name list since we already have quite a number of them. However, we also do not want to use the given name list directly because it might contain error names as well. Our approach is the same as using Sinica corpus as a reference. First we extract the given names from the CGW using the method as described in Section 5.3.3, then we double check with the provided given name list to see if the names are inside the list. If they are in the list, then we assume that they

[7]http://technology.chtsai.org/namefreq

[8]http://www.csie.nctu.edu.tw/service/jcee/

[9]Since the name list is from Taiwan, it is written in Big5 code. We need to convert it to GB code. After the conversion, it has 71175 given names because some of the characters in traditional Chinese cannot be converted into simplified Chinese.

[10]If a name contains two or three characters, then the first character is the family name and the rest is given name, else the first two characters are family name and the rest is given name. This is not always true as a name with three character can be 1F2G or 2F1G, although the later case is more rare.

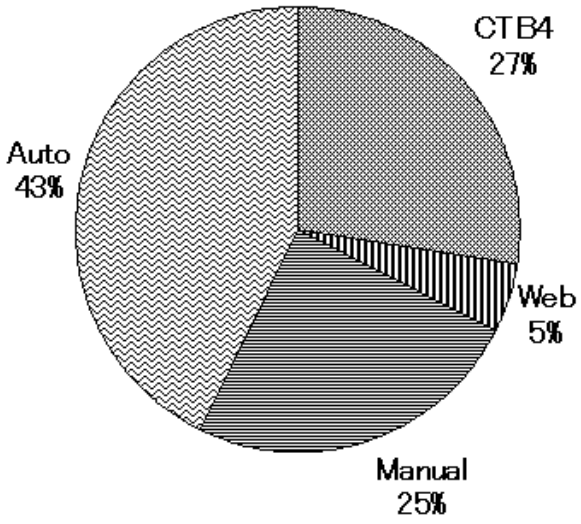



Figure 5.4. Composition of current dictionary

are correct given names. By this way, we manage to extract 18,818 given names from CGW automatically.

### 5.3.4 Composition of Current Dictionary

Figure 5.4 shows the composition of our current system dictionary. 27% (33,438) of the dictionary is extracted from CTB4.0, 5% (6,192) collection from the web, 25% (30,257) is extracted from manual checking and 43% (52,104) from auto checking. There are some overlapped entries between these groups. In total, we have collected 120,769 entries in our dictionary.

## 5.4 Two-layer Morphological Analysis

We propose a two-layer morphological analysis in our system. The first layer produces the segmentation and POS tags based on our definition, meaning the minimal segmentation units. The second layer transforms the output of the first layer to CTB original segmentation units. Figure 5.5 shows the overview of the system. The right hand side shows the process used for preparation of the training data and the system dictionary. The left hand side shows the process of two-layer analysis. The preparation of the training data and the system dictionary has already been described in the previous sections. We will describe the methods used in each layer of analysis in the following sections. Using this approach, we

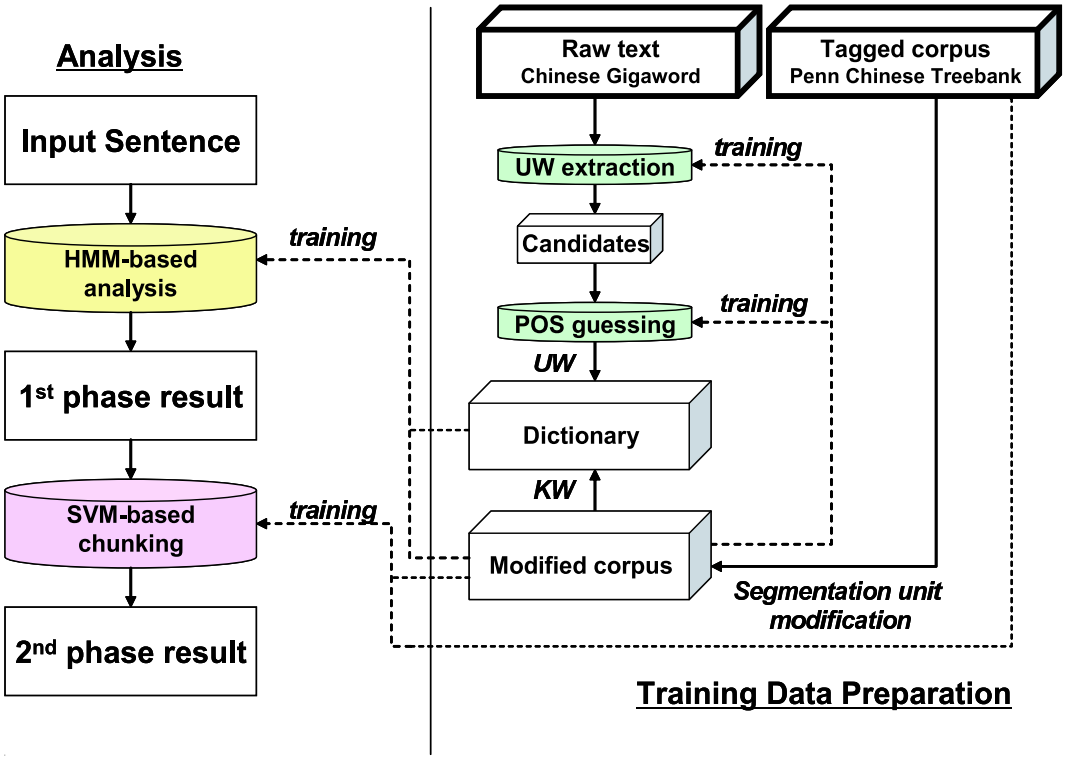



Figure 5.5. Overview of two-layer morphological analysis

will produce two sets of outputs which give different different size of segmentation units for certain types of words.

### 5.4.1 Minimal Unit Analysis - ChaSen

We use *ChaSen* [28] in our first layer analysis. Although ChaSen is originally built for Japanese language, it can be adopted easily to Chinese with slight modification[11]. In fact, it is easier to set up the system in Chinese as we do not need to define grammar in Chinese since it does not have morphological changes such as inflection. We just need a training corpus and a dictionary for the training process. The system is based on Hidden Markov Models (Section 2.1).

Table 5.6 shows the results of the first layer analysis. The results are calculated based on the minimal units as shown in equations below.

[11]The only modification done is with the tokenization module. In Japanese, there are one-byte characters for katakana, but in Chinese all words are two bytes. We just need to remove the checking of one-byte characters besides ASCII characters.

| | Segmentation | | | POS Tagging | | |
|---|---|---|---|---|---|---|
| | Rec | Prec | F-mea | Rec | Prec | F-mea |
| CTB4 Dic | 90.0 | 83.1 | 86.4 | 82.1 | 75.8 | 78.8 |
| + manual extraction | 91.3 | 86.3 | 88.8 | 83.3 | 78.7 | 80.9 |
| + auto extraction | 92.8 | 90.0 | 91.4 | 84.7 | 82.2 | 83.5 |
| no unknown | 97.1 | 97.8 | 97.4 | 90.1 | 90.7 | 90.4 |
| closed | 97.3 | 98.1 | 97.7 | 91.1 | 91.8 | 91.5 |

Table 5.6. Results of first layer analysis

| | Unknown POS | Unknown Word | Total |
|---|---|---|---|
| CTB4 Dic | 1.3% | 7.8% | 9.2% |
| + manual extraction | 1.7% | 5.7% | 7.4% |
| + auto extraction | 1.9% | 3.5% | 5.4% |

Table 5.7. Unknown word rate after dictionary expansion

$$
\begin{aligned}
Recall &= \frac{no.\ of\ correctly\ segmented/POS\text{-}tagged\ minimal\ unit\ words}{total\ no.\ of\ minimal\ unit\ words\ in\ gold\ data} \\
Precision &= \frac{no.\ of\ correctly\ segmented/POS\text{-}tagged\ minimal\ unit\ words}{total\ no.\ of\ minimal\ unit\ words\ segmented/POS\text{-}tagged} \\
F\text{-}measure &= \frac{2 \times Recall \times Precision}{Recall + Precision}
\end{aligned}
$$

Table 5.7 shows the unknown POS and unknown word rates after dictionary expansion. [CTB4 Dic] contains only the entries extracted from CTB 4.0, which is 33,438 entries. During the first phase of manual extraction from CGW and collection from the web, we manage to increase the dictionary to 68,626 entries [+ manual extraction]. At the second phase of extraction, with auto-checking with other resources, we further increase the dictionary to 120,769 entries [+ auto extraction]. We can see the improvement on the analysis results with the increment of size of dictionary. We realize a decreasing in the unknown word rates. However, we still have the problem with unknown POS when the words are actually in the dictionary but with different POS tags compared with the test data.

Since our morphological analysis system does not deal with unknown words

directly, we also evaluate the system by assuming that no unknown words exist in the test data. The row [no unknown] shows the results by retrieving all the entries from both training and testing data for building the dictionary. There are 39,896 entries in total, 6,458 entries more than [CTB4 Dic]. The training of HMM takes only the training data and the dictionary into account. The row [closed] shows the results where the training of HMM also includes the testing data. We can say that the [closed] is the perfect case of the system and [no unknown] is more to reality if we can expand the dictionary to a certain extent. From the results, we can see that our system is still far from perfect. Besides increasing the entries in the dictionary, we must also find a better way to improve the accuracy of POS tagging.

### 5.4.2 CTB Unit Analysis - YamCha

The second layer takes the output from the first layer and joins the words by chunking. In order to obtain the original segmentation and POS tags, our task is to join up family names and given names, numbers, numeral type time nouns, and foreign words. The only difference with the original POS tags is that we cannot get back the original POS tags for foreign words. We used *YamCha* as described in Section 2.2 for chunking as it is proved to be efficient for this task. The system is based on Support Vector Machines. The feature sets used are two words and POS tags at both left and right sides of the current word, plus the previous two output labels. The output labels are NR-PER-B, NR-PER-I, CD-B, CD-I, OD-B, OD-I, NT-B, NT-I, FW-B, FW-I and O, cater for CJK person names, cardinal numbers, ordinal numbers, time nouns and foreign words.

| | Segmentation | | | POS Tagging | | |
|---|---|---|---|---|---|---|
| | Rec | Prec | F-mea | Rec | Prec | F-mea |
| CTB4 Dic | 88.5 | 81.1 | 84.6 | 80.2 | 73.6 | 76.7 |
| + manual extraction | 89.8 | 84.8 | 87.2 | 81.4 | 76.8 | 79.1 |
| + auto extraction | 91.4 | 88.8 | 90.1 | 83.0 | 80.6 | 81.8 |

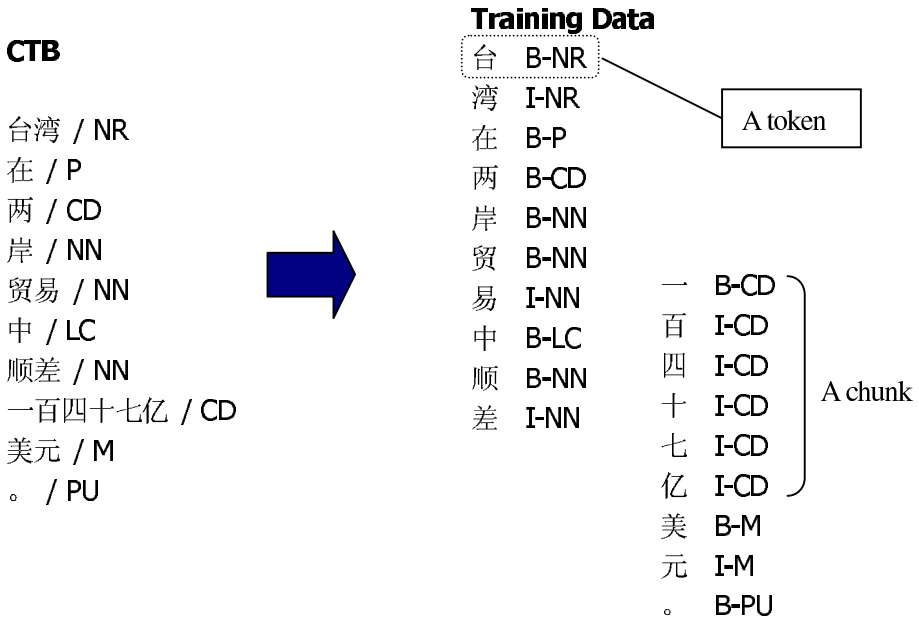
Table 5.8. Results of second layer analysis

Table 5.8 shows the results of the second layer analysis. While the results of first layer are based on minimal units, the results of second layer are based on CTB units.

$$
\begin{aligned}
Recall &= \frac{no.\ of\ correctly\ segmented/POS\text{-}tagged\ CTB\ unit\ words}{total\ no.\ of\ CTB\ unit\ words\ in\ gold\ data} \\
Precision &= \frac{no.\ of\ correctly\ segmented/POS\text{-}tagged\ CTB\ unit\ words}{total\ no.\ of\ CTB\ unit\ words\ segmented/POS\text{-}tagged} \\
F\text{-}measure &= \frac{2 \times Recall \times Precision}{Recall + Precision}
\end{aligned}
$$

Since the results are based on different segmentation units, we cannot do a direct comparison. However, for a rough comparison, the difference between the first and second layer analysis is quite small. This also means that the accuracy for chunking is high since the upper bound of the second layer depends on the accuracy of the first layer. In this way, we can easily convert the minimal unit segmentation back to CTB standard.

## 5.5 Comparison with Other Systems




'Taiwan has a surplus of 14.7 billion on the trade between Taiwan and mainland'

Figure 5.6. Conversion from word-based to character-based features by Yoshida

Chinese word segmentation and POS tagging have also been done in [50]. They used the same chunker, *YamCha*, and Chinese Penn Treebank (with 100,000

words) in their experiment. They also split the words into characters, and labeled the characters with the BI chunking tag set (B - first character, I - intermediate character), as shown in Figure 5.6 [12]. The context window size is two characters at both left and right sides, but only the characters and the previously tagged POS tags are used as features for chunking. Their method processes word segmentation and POS tagging simulteneously for solving both ambiguity problem and unknown word detection. They obtained about 88% accuracy for overall POS tagging and 40% for unknown word detection. The problem with this method is that the time used for training and analysis is long because it is based on the number of POS tags and the BI tags. Therefore, for Penn Chinese Treebank, with 33 POS tags and 2 BI tags, they need to classify the characters into 66 classes. If the number of POS tags increases, such as using Peking University corpus, with 39 POS tags, they will need 78 classes. They also conducted an experiment using the Peking University corpus and obtained an accuracy of 92% for overall POS tagging.

There exist some practical systems that have been developed by some institutions or companies, such as Tsinghua University, Peking University and Basis Technology. The system CSeg&Tag 1.1 [39] (60,133 word entries) developed by Tsinghua University reported that the segmentation precision is ranging from 98.0% to 99.3%, POS tagging precision from 91.0% to 97.1%, and the recall and precision for unknown words are from 95.0% to 99.0% and from 87.6% to 95.3%, respectively. The SLex 1.1 system, developed by Peking University (70K over word entries), reported an accuracy of 97.05% for segmentation and 96.42% for POS tagging. Basis Technology presented a commercial product, a Chinese Morphological Analyzer (CMA) [7, 8] which has 1.2 million entries in their dictionary (the accuracy is not known). The dictionaries that they used are much bigger than others. Therefore, the unknown word rate should be lower. Furthermore, all of them have combined statistics based and rule based methods in their approaches. They have used some rules that have been handcrafted by human over the past 10-20 years. Therefore, it is quite difficult for us to be as competitive as them because we do not have the expert to create those heuristic rules. These rules are very useful in handling some special situations such as duplication of words and segmentation inconsistencies.

There are also some systems which are downloadable from the web. ICT-

[12]NR - proper noun, P - preposition, CD - cardinal number, NN - common noun, LC - localizer, M - measure word, PU - punctuation. Note that tag "O" is not used for tagging.

CLAS (Institute of Computing Technology, Chinese Lexical Analysis System)[13] [53] is an integrated system that uses an approach based on multi-layer Hidden Markov Models. ICTCLAS provides word segmentation, POS tagging and unknown word recognition. The unknown words cover only person names, location names and organization names. Their experiment results show that ICTCLAS achieved 98.25% accuracy for word segmentation, 95.63% for POS tagging with 24 tags and 93.38% with 48 tags. The way they set up the experiments is not clear. We do not know how many words exist in their core dictionary. However, according to their report, the unknown word recall is over 90%, and the highest is with Chinese person names, 98%. Their system is trained on Peking University corpus.

Microsoft Research Asia (MSR) also provides a free segmenter for download (S-MSRSeg)[14]. S-MSRSeg is a simplified version of the MSRSeg system described in [14]. It does not provide the functionalities of new word identification, morphology analysis and adaptation to various standards. They apply a source-channel approach to word segmentation, and a class-based model and context model for new word identification. They have a big training corpus with 20M tokens and a dictionary of 158K entries. Their testing on a set of 226K tokens showed 95.5–96.2% recall and 95.0–95.6% precision for word segmentation and 60.4–78.4% recall and 46.2–68.1% precision for new word identification. However, if the test is done without new word identification, the recall and precision drop to 94.5–96.4% and 92.6–94.7%, respectively. MSR also define their own segmentation standard. However, they also proposed some methods to adapt their system to other standards.

## 5.6 Continuous Work on Unknown Word Extraction

Currently our dictionary has 120,769 entries, which is quite compatible with other system. However, we still like to add more to the dictionary as we believe that there are a lot more words in the real text that are not in our dictionary yet. The automatic extraction using other resources only cover part of the word types (POS tags). Therefore, our current dictionary can be said not "balanced" as some of the word types such as organization names, foreign names, time nouns

[13] http://www.ict.ac.cn/freeware/003_ictclas.asp

[14] http://131.107.65.76/research/downloads/default.aspx

| Char. | POS-position | Char. Type | Chunk |
|---|---|---|---|
| 刘 | NR-PER-FAM-S | SUR | O/0.99974 |
| 玉 | NN-S | FRN | NR-PER-GIV-B/0.999243 |
| 兰 | NR-PER-FAM-S | BTH | NR-PER-GIV-I/0.996695 |
| 难 | AD-S | OTH | O/0.853199 |
| 掩 | VV-S | OTH | O/0.97263 |
| 焦 | NN-B | SUR | O/0.999997 |
| 虑 | NN-E | OTH | O/0.999893 |
| 地 | DEV-S | FRN | O/0.999748 |
| 说 | VV-S | OTH | 0/0.999905 |

'Liu Yülan said anxiously', Char. - Chinese character, POS-position - POS tag plus position tag, Char. Type - character type, Chunk - label for unknown word

Figure 5.7. An example of the confidence measure

etc., may not be enough yet.

The process of human checking on extracted unknown words is very time consuming. Since we have collected quite a number of words in our system dictionary, perhaps it is time for us to change the direction from high "recall" to high "precision". In other words, if we can get higher precision in our unknown word extraction model, we will get better qualitative results rather than quantitative results. As a result, manual checking on the unknown words will require less effort.

### 5.6.1 Pruning using Confidence Measure

CRF++ as described in Section 2.4 provides us a measurement for improving the precision of the extracted unknown words. Each output is attached with a confidence measure (marginal probability), showing how confidence is the answer. Figure 5.7 shows an example of the output from CRF++. If we multiply the confidence measure of each character in an unknown word, then we get the total confidence measure for the word. In this case, the confidence measure for the unknown word "玉兰" (Yülan, a person name) is $0.999243 \times 0.996695 = 0.995941$.

If we set the threshold of the confidence measure to be higher, than we get higher precision, though the recall deteriorates. Our preliminary results show that CRF++ performs not only better than *YamCha* and ME, but also provides us a measurement to control the precision of the outputs. Table 5.9 shows the

results of using CRF++ with different thresholds (in brackets) of the confidence measure. We plan to use this method for manual checking in the future.

| | Model | Recall | Precision | F-measure |
|---|---|---|---|---|
| CJK given name | *YamCha* | 89.02 | 70.12 | 78.45 |
| | CRF++ | 88.76 | 72.27 | 79.67 |
| | CRF++ (0.70) | 82.63 | 82.41 | 82.52 |
| | CRF++ (0.95) | 65.51 | **93.23** | 76.95 |
| Foreign names | *YamCha* | 39.69 | 56.62 | 46.67 |
| | CRF++ | 57.73 | 55.45 | 56.57 |
| | CRF++ (0.70) | 31.96 | 73.81 | 44.60 |
| | CRF++ (0.95) | 11.34 | **88.00** | 20.09 |

| | Model | Recall (token) | Recall (type) | Precision (type) |
|---|---|---|---|---|
| Unknown word | ME | 72.2 | 72.1 | 50.6 |
| | CRF++ | 75.6 | 75.4 | 57.0 |
| | CRF++ (0.70) | | 55.6 | 76.3 |
| | CRF++ (0.90) | | 37.7 | **85.5** |
| | CRF++ plus pruning by contexts | | 73.7 | 62.9 |

Table 5.9. Results of unknown word extraction by applying pruning methods

### 5.6.2 Pruning by Checking the Contexts

In Section 4.5.2, we have introduced a pruning method to increase the precision of unknown word extraction. The method applied some rules which are based on the contexts to eliminate some false unknown words. The contexts are the adjacent words and internal components of the unknown words. Here, we extend the method to cater for larger window size. The concept is as below. Let $w_{i-2}$, $w_{i-1}$, $w_i$, $w_{i+1}$, $w_{i+2}$ be five continuous words in the text where $w_i$ is an unknown word candidate and $w_i = e_{i,1}e_{i,2}...e_{i,n}$ where $e_{i,j}$ is a character and $n$ is the length of the word. We assume that if the unknown word forms a known word with adjacent characters or words, then it is not a valid unknown word. Therefore, if any one of the following words exists in the dictionary, then the unknown word

is deleted from the list[15]:

1. $e_{i-1,n}e_{i,1}$ - the last character of previous word and the first character of unknown word
2. $w_{i-1}e_{i,1}$ - the previous word and the first character of unknown word
3. $e_{i-1,n}w_i$ - the last character of previous word and the unknown word
4. $w_{i-1}w_i$ - the previous word and the unknown word
5. $e_{i,n}e_{i+1,1}$ - the last character of unknown word and the first character of next word
6. $e_{i,n}w_{i+1}$ - the last character of unknown word and the next word
7. $w_ie_{i+1,1}$ - the unknown word and the first character of next word
8. $w_iw_{i+1}$ - the unknown word and the next word
9. $e_{i-1,n}w_ie_{i+1,1}$ - the last character of previous word, the unknown word and the first character of next word
10. $w_{i-1}w_ie_{i+1,1}$ - the previous word, the unknown word and the first character of next word
11. $e_{i-1,n}w_iw_{i+1}$ - the last character of previous word, the unknown word and the next word
12. $w_{i-1}w_iw_{i+1}$ - the previous word, the unknown word and the next word
13. $e_{i,n}w_{i+1}e_{i+2,1}$ - the last character of unknown word, the next word and the first character of second next word
14. $w_iw_{i+1}e_{i+2,1}$ - the unknown word, the next word and the first character of second next word
15. $e_{i,n}w_{i+1}w_{i+2}$ - the last character of unknown word, the next word and the second next word
16. $w_iw_{i+1}w_{i+2}$ - the unknown word, the next word and the second next word

[15]Items 1-8 are the same as in Section 4.5.2.

17. $e_{i-2,n}w_{i-1}e_{i,1}$ - the last character of second previous word, the previous word and the first character of unknown word

18. $w_{i-2}w_{i-1}e_{i,1}$ - the second previous word, the previous word and the first character of unknown word

19. $e_{i-2,n}w_{i-1}w_i$ - the last character of second previous word, the previous word and the unknown word

20. $w_{i-2}w_{i-1}w_i$ - the second previous word, the previous word and the unknown word

The reason why we enlarge the window size is that the current unknown word detection method has the tendency to extract shorter unit words rather than long words. Therefore, during manual checking, some shorter units are combined manually to give long words. For the future manual checking, we would like to eliminate these shorter unit words that are actually belong to some long words.

For those unknown words with length $n$ greater than 4 characters, it is possible that it includes a known word inside, especially an idiomatic phrase. Therefore, if either $e_1e_2e_3e_4$ (the first 4 characters), $e_2e_3e_4e_5$ (the second 4 characters), ... or $e_{n-3}e_{n-2}e_{n-1}e_n$ (the last four characters) exists in the dictionary (except those words that are numbers, alphabets or symbols), then the unknown word candidate is deleted from the list.

The last row in Table 5.9 shows the result using the CRF++ model plus pruning by contexts. Before pruning, CRF++ itself gave us 75.4% recall and 57% precision. After pruning, the recall has become 73.7%, a bit lower, but the precision has increased to 62.86%. This means that we have a better quality result although the quantity decreases.

Using the above two screening methods, we have run on the CGW for the texts in the whole year of 2002 (xin2002). Table 5.10 shows the numbers of unknown word candidates extracted by different levels of pruning. From the output, we know that the pruning has decreased a lot the number of candidates. In other words, it will reduce the effort of manual checking by 88% compared to without pruning, and with better quality candidates. Currently we are still working on the manual checking of these 33,643 candidates.

| Model | Word Type | Word/POS Type |
|---|---|---|
| CRF++ only | 239,798 | 288,312 |
| CRF++ (0.90) | 46,228 | 50,845 |
| CRF++ (0.90) plus pruning by contexts | 30,805 | **33,643** |

Table 5.10. Results of unknown word extraction on CGW by applying pruning methods

## 5.7 Summary

In this chapter, we presented a two-layer morphological analyzer for Chinese text. The first layer produces minimal unit segmentation with detailed POS tags and the second layer transforms the minimal units into CTB standard. The design enables us to reduce the size of the dictionary by splitting some high productive words into smaller units. In order to attain a practical system, our initial system dictionary was too small, only contains 33,438 entries. Therefore, we looked for some ways to enlarge our system dictionary using unknown word detection methods. Currently, our dictionary contains 120,769 entries which is quite compatible with other systems. Our results showed that by increasing the number of entries in the dictionary, the accuracy of word segmentation and POS tagging is also improved.

# Chapter 6

# Conclusion and Future Work

## 6.1 Conclusion

In this thesis, we discussed the problems in Chinese morphological analysis: Word segmentation and POS tagging. During word segmentation, two major problems encountered: Segmentation ambiguity and unknown word occurrences. Segmentation ambiguities are dealt with known words only, and the correct answers highly depend on the contexts. Unknown words are those words not found in the system dictionary, and must be detected from the text. The word formation powers of characters play an important role in the detection. Some characters have high tendency towards formation of new words while some are not. These tendencies of characters can be learned using machine-learning methods. Therefore, a character-based method is a better solution for unknown word detection. There are also ambiguities with the POS tagging because a word could hold different POS tags based on the context. Furthermore, if a word is unknown, then we need to guess the POS tag of the word based on the context and the internal components of the word.

In this research, we studied the problems and proposed some methods for solving them. Our approaches are mainly machine-learning based models. Our task is to find the best suited feature set for improving the results. Our main target is unknown word detection and POS tag guessing of unknown words. At the end of the research, we collected a set of new words for our system dictionary. Using this dictionary, we successfully built a morphological analyzer for Chinese text. Although the accuracy of the morphological analyzer is not yet satisfactory, it provides a room for improvement in the future.

## 6.2 Future Work

### Unknown POS

During the morphological analysis, we also face the problem of unknown POS. In other words, there exist some words in the text, though these words exist in the system dictionary, they are with different POS tags. This problem occurs because a word can carry more than one POS tag, depending on the context where it appears. Therefore, if the word appears in a place where it has never been seen in the training, then we have no way to assign it with the correct POS tag currently. Therefore, we also need to check on the possibility of a word being assigned with a tag different from the one in the dictionary.

### Morphological Analysis on Compound and Derived Words

As discussed in Introduction, morphological analysis should provide more details analysis on the language instead of just word segmentation and POS tagging. In the future, the system should include the function of analysis on compounds words and morphological derived words. In this case, new compound words and morphological derived words can be easily detected from the text directly even they are not listed in the dictionary. This will help in semantic analysis in the future as usually the whole meaning of a word in Chinese has high relation with the meaning of each component of the word.

### Adaptation to Various Standard

Gao et. al. [14] proposed a method to adapt a current segmentation system to various types of standard (as in SIGHAN bakeoff). It is clear that no single system that can produce the same results as defined by difference resources. Their method makes use of a transformation-based learning method [2], which will transform the initial segmentation into target segmentation. By doing this, their system can be customized to various types of standard. Our system can follow the same direction and try to customize the system for various standards in the future.

# Acknowledgements

First of all, I would like to thank the Japanese Ministry of Education for offering me the Monbukagakusho scholarship. Being able to study in overseas, and especially in Japan, was for sure an invaluable opportunity. The life in Japan will be an unforgettable experience for me.

Secondly, being able to enroll for the Master and Doctorate courses at the Nara Institute of Science and Technology was an honor to me. The school provides a perfect environment, with more than enough facilities, for students to carry out their research. They also provide an open environment to access to new technologies, which is of course a very important point in the information science domain, where things advance rapidly.

In order to do a good research, a good supervisor is indeed necessary. I am lucky to have Professor Matsumoto as my supervisor. He has always given me insightful advice throughout my study, and guided me all the way to carry out my research. Without his patient guidance, I would not hope to graduate today. I would also like to thank Associate Professor Inui and Dr. Asahara for their continuous advice and illuminating discussions. And to all my colleagues, I appreciate their kindness in helping me to solve many problems encountered during my research.

I would also like to thank Professor Matsumoto, Professor Shikano, Professor Uemura, Associate Professor Inui and Dr. Asahara who have spent their time and effort reading this thesis and given me valuable comments on it. I have tried my best to revise this thesis according to their suggestions. Nothing is perfect in the world. Therefore, the remaining errors in the thesis are, of course, mine.

To my family, I thank my parents for teaching me the value of education at a young age. To my husband, Yves, I appreciate his encouragement and support, so that I can complete my study. I want to thank my daughter, Mireille too, for her wonderful smile everyday which has given me the strength to go through all these hard going works.

# References


[1] Masayuki Asahara and Yuji Matsumoto. Unknown Word Indentification in Japanese Text Based on Morphological Analysis and Chunking (in Japanese). In *IPSJ SIG Notes Natural Language, Information Proceesing Society of Japan, 2003-NL-154*, pages 47–54, 2003.

[2] Eric Brill. Some Advances in Transformation-based Part of Speech Tagging. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 1994.

[3] Chao-Jan Chen, Ming-Hong Bai, and Keh-Jiann Chen. Category Guessing for Chinese Unknown Words. In *Proceedings of NLPRS*, pages 35–40, 1997.

[4] Keh-Jiann Chen and Ming-Hong Bai. Unknown Word Detection for Chinese By a Corpus-based Learning Method. In *Proceedings of ROCLING X, International Conference Research on Computational Linguistics*, pages 159–174, 1997.

[5] Keh-Jiann Chen and Wei-Yun Ma. Unknown Word Extraction for Chinese Documents. In *Proceedings of COLING*, pages 169–175, 2002.

[6] Tung-Hui Chiang, Jing-Shin Chang, Ming-Yu Lin, and Keh-Yih Su. Statistical Models for Word Segmentation and Unknown Word Resolution. In *Proceedings of ROCLING V*, pages 123–146, 1992.

[7] Thomas Emerson. Segmenting Chinese in Unicode. In *16th International Unicode Conference*, 2000.

[8] Thomas Emerson. Segmenting Chinese Text. *MultiLingual Computing & Technology*, 12(2), 2001.

[9] Fei Xia. The Part-of-Speech Tagging Guidelines for the Penn Chinese Treebank (3.0). 2000.

[10] Fei Xia. The Segmentation Guidelines for the Penn Chinese Treebank (3.0). 2000.

[11] Guohong Fu and K.K. Luke. An Integrated Approach for Chinese Word Segmentation. In *Proceedings of PACLIC 17*, 2003.

[12] Guohong Fu and K.K. Luke. Chinese Unknown Word Identification Using Class-based LM. In *Proceedings of IJCNLP*, pages 262–269, 2004.

[13] Guohong Fu and Xiaolong Wang. Unsupervised Chinese Word Segmentation and Unknown Word Identification. In *Proceedings of NLPRS*, 1999.

[14] Jianfeng Gao, Andi Wu, Mu Li, Chang-Ning Huang, Hongqiao Li, Xinsong Xia, and Haowei Qin. Adaptive Chinese Word Segmentation. In *Proceedings of ACL*, pages 463–470, 2004.

[15] Chooi-Ling Goh, Masayuki Asahara, and Yuji Matsumoto. Chinese Unknown Word Identification Using Character-based Tagging and Chunking. In *Companion Volume to the Proceedings of ACL, Interactive Poster/Demo Sessions*, pages 197–200, 2003.

[16] Jack Halpern. Lexicon-based Orthographic Disambiguation in CJK Intelligent Information Retrieval. In *Proceedings of Asian Language Resources Workshop*, pages 17–23, 2002.

[17] Changning Huang. Segmentation Problem in Chinese Processing (in Chinese). *Applied Linguistics*, 1:72–78, 1997.

[18] Institue of Information Science and CKIP, Academia Sinica. Sinica Corpus. http://www.sinica.edu.tw/SinicaCorpus.

[19] Institute of Computational Linguistics, Peking University. Beijing University Corpus. http://www.icl.pku.edu.cn/Introduction/corpustagging.htm.

[20] Institute of Computational Linguistics, Peking University. Chinese Text Segmentation and POS Tagging. http://www.icl.pku.edu.cn/nlp-tools/segtagtest.htm.

[21] Taku Kudo and Yuji Matsumoto. Chunking with Support Vector Machines. In *Proceedings of NAACL*, pages 192–199, 2001.

[22] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 282–289, 2001.

[23] Mu Li, Jianfeng Gao, Changning Huang, and Jianfeng Li. Unsupervised Training for Overlapping Ambiguity Resolution in Chinese Word Segmentation. In *Proceedings of Second SIGHAN Workshop on Chinese Language Processing*, pages 1–7, 2003.

[24] Linguistic Data Consortium. The Penn Chinese Treebank Project. http://www.cis.upenn.edu/~chinese/.

[25] Jin-Kiat Low, Hwee-Tou Ng, and Wenyuan Guo. A Maximum Entropy Approach to Chinese Word Segmentation. In *Proceedings of Fourth SIGHAN Workshop*, pages 161–164, 2005.

[26] Xiao Luo, Maosong Sun, and Benjamin K. Tsou. Covering Ambiguity Resolution in Chinese Word Segmentation Based on Contextual Information. In *Proceedings of COLING*, pages 598–604, 2002.

[27] Wei-Yun Ma and Keh-Jiann Chen. A Bottom-up Merging Algorithm for Chinese Unknown Word Extraction. In *Proceedings of Second SIGHAN Workshop on Chinese Language Processing*, pages 31–38, 2003.

[28] Yuji Matsumoto, Akira Kitauchi, Tatsuo Yamashita, Yoshitaka Hirano, Hiroshi Matsuda, Kazuma Takaoka, and Masayuki Asahara. Morphological Analysis System ChaSen version 2.2.9 Manual, 2002. http://chasen.naist.jp/.

[29] Tetsuji Nakagawa. Chinese and Japanese Word Segmentation Using Word-Level and Character-Level Information. In *Proceedings of COLING*, pages 466–472, 2004.

[30] Hwee-Tou Ng and Jin-Kiat Low. Chinese Part-of-Speech Tagging: One-at-a-Time or All-at-once? Word-based or Character-based? In *Proceedings of EMNLP*, pages 277–284, 2004.

[31] Jian-Yun Nie, Marie-Louise Hannan, and Wanying Jin. Unknown Word Detection and Segmentation of Chinese Using Statistical and Heuristic Knowledge. *Communications of COLIPS*, Vol.5:47–57, 1995.

[32] Jorge Nocedal and Stephan J. Wright. *Numerical Optimization (Chapter 9)*. Springer, New York, 1999.

[33] Fuchun Peng, Fangfang Feng, and Andrew McCallum. Chinese Segmentation and New Word Detection using Conditional Random Felds. In *Proceedings of COLING*, pages 562–568, 2004.

[34] Adwait Ratnaparkhi. A Maximum Entropy Part-of-speech Tagger. In *Proceedings of EMNLP*, 1996.

[35] Erik F. Tjong Kim Sang and John Veenstra. Representing Text Chunks. In *Proceedings of EACL*, pages 173–179, 1999.

[36] Dayang Shen, Maosong Sun, and Changning Huang. The Application & Implementation of Local Statistics in Chinese Unknown Word Identification (in Chinese). *Communications of COLIPS*, Vol.8, 1998.

[37] Richard Sproat and Thomas Emerson. The First International Chinese Word Segmentation Bakeoff. In *Proceedings of Second SIGHAN Workshop on Chinese Language Processing*, pages 133–143, 2003.

[38] Richard Sproat, William Gale, Chilin Shin, and Nancy Chang. A stochastic Finite-State Word-Segmentation Algorithm for Chinese. *Computational Linguistics*, pages 377–404, 1996.

[39] Maosong Sun, Dayang Shen, and Changning Huang. CSeg&Tag1.0: A Practical Word Segmentation and POS Tagger for Chinese Texts. In *fifth Conference on Applied Natural Language Processing*, pages 119–126, 1997.

[40] Huihsin Tseng and Keh-Jiann Chen. Design of Chinese Morphological Analyzer. In *Proceedings of First SIGHAN Workshop*, 2002.

[41] Huihsin Tseng, Daniel Jurafsky, and Christopher Manning. Morphological Features Help POS Tagging of Unknown Words Across Language Varieties. In *Proceedings of Fourth SIGHAN Workshop*, pages 32–39, 2005.

[42] Kiyotaka Uchimoto, Qing Ma, Masaki Murata, Hiromi Ozaku, and Hitoshi Isahara. Named Entity Extraction Based on A Maximum Entropy Model and Transformational Rules. In *Proceedings of ACL*, 2000.

[43] Syunsuke Uemura. Automatic Compilation and Retrieval of Modern Japanese Concordance. *Journal of Information Processing*, 1(4):172–179, March 1979.

[44] Syunsuke Uemura, Yasuo Sugawara, Mantaro J. Hashimoto, and Akihiro Furuya. Automatic Compilation of Modern Chinese Concordances. In *Proceedings of COLING*, pages 323–329, 1980.

[45] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

[46] Andi Wu. Customizable Segmentation of Morphologically Derived Words in Chinese. *International Journal of Computational Linguistics and Chinese Language Processing*, 8(1):1–28, 2003.

[47] Zimin Wu and Gwyneth Tseng. Chinese Text Segmentation for Text Retrieval: Achievements and problems. *Journal of the American Society for Information Science*, 44(9):532–542, 1993.

[48] Nianwen Xue and Susan P. Converse. Combining Classifiers for Chinese Word Segmentation. In *Proceedings of First SIGHAN Workshop on Chinese Language Processing*, 2002.

[49] Nianwen Xue and Libin Shen. Chinese Word Segmentation as LMR Tagging. In *Proceedings of Second SIGHAN Workshop on Chinese Language Processing*, pages 176–179, 2003.

[50] Tatsumi Yoshida, Kiyonori Ohtake, and Kazuhide Yamamoto. Performance Evaluation of Chinese Analyzers with Support Vector Machines. *Journal of Natural Language Processing*, 10(1):109–131, 2003.

[51] Shiwen Yu, Huiming Duan, Xuefeng Zhu, Bin Swen, and Baibao Chang. Specification for Corpus Processing at Peking University: Word Segmentation, POS Tagging and Phoenetic Notation (in Chinese). *Journal of Chinese Language and Computing*, 13(2):121–158, 2003.

[52] Hua-Ping Zhang, Qun Liu, Hao Zhang, and Xue-Qi Cheng. Automatic Recognition of Chinese Unknown Words Based on Roles Tagging. In *Proceedings of First SIGHAN Workshop*, 2002.

[53] Hua-Ping Zhang, Hong-Kui Yu, De-Yi Xiong, and Qun Liu. HHMM-based Chinese Lexical Analyzer ICTCLAS. In *Proceedings of Second SIGHAN Workshop on Chinese Language Processing*, pages 184–187, 2003.

[54] Guo-Dong Zhou and Kim-Teng Lua. Detection of Unknown Chinese Words Using a Hybrid Approach. *Computer Processing of Oriental Language*, 11(1):63–75, 1997.

[55] GuoDong Zhou. A Chunking Strategy Towards Unknown Word Detection in Chinese Word Segmentation. In *Proceedings of Second IJCNLP*, pages 530–541, 2005.

## Abbreviations of Conference Names

| | |
|---|---|
| ACL | Annual Meeting of the Association for Computational Linguistics |
| COLING | International Conference on Computational Linguistics |
| EACL | European Chapter of the Association for Computational Linguistics |
| EMNLP | Conference on Empirical Methods in Natural Language Processing |
| IJCNLP | International Joint Conference on Natural Language Processing |
| NAACL | Meeting of the North American Chapter of the Association for Computational Linguistics |
| NLPRS | Natural Language Processing Pacific Rim Symposium |
| PACLIC | Pacific Asia Conference on Language, Information and Computation |
| ROCLING | R.O.C. Computational Linguistics Conference |

# Appendix

## A Examples of Acceptable Detected Unknown Words

1. 对近代京剧史、论研究与批评，我们所知还不多
2. 现代人，何处是家乡？——流动中的乡情和亲情
3. 流露出对深圳的感情，处事方式也非常深圳化，
4. 田泳的四川话已经讲得很结巴，甚至给家里人
5. 中国人是恋乡的民族。我们的祖先世世代代
6. 这是“文化视角”的第二篇，上篇话题是“
7. 沈阳杂技团的《双爬竿———少林晨练》获
8. 杂技团在北京木樨地搭起一座临时马戏棚演出，
9. 并题词：“世庆贤台雅赏”（见图）。
10. 我疼得勾着腰在地上蹦跳，嘴里啊啊地叫着，
11. 所谓练字就是练神，练神就是练心，
12. 古人讲求心斋，曾说一位工匠在雕刻时
13. 老将军配有专车，但他很少乘坐。
14. 由人民出版社、新华文摘社、青岛双星集团
15. “四荒”资源使水土保持由防护型转向开发型
16. 大力发展特色农业『龙型』经济带动百万农民
17. 如《望》周刊社开展了以坚持“三讲”是办刊
18. 亚龙湾国家级旅游开发区、通什民族文化村、
19. 湖北万名税官竞争上岗
20. 还趁着圩日在集市摆摊咨询，踏进农户家现场
21. 幼儿园小朋友们举行了一年一度的『鸟婚』仪式。
22. 总是让先来者登车，极少发生争抢、推挤的现象。
23. 啤酒店里充满欢声笑语，弥漫着酒气芳香。
24. 倘若对别人的恭维照收不误，对自己的一功之德
25. 每家都要把常年贴在灶头上的灶公公像揭下来
26. 有的一副『大肚佛』形象，世事不评说，
27. 京韵大鼓联唱《百鸟朝凤》、
28. 目前各华埠的舞龙舞狮队正在抓紧彩排；
29. 黑龙江呼兰河域，万历四十七年被努尔哈赤所并，
30. 其中有挪威国王和王后，也有美国影星史泰龙。

# B Chinese ChaSen POS Tagset

The POS tagset is based on CTB POS tagset plus the newly defined POS tags. There are a total of 42 POS tags. Tags marked with * are newly defined POS tags.

| POS Tag | Description | Examples |
|---|---|---|
| AD | adverb | 还 |
| AS | aspect marker | 着 |
| BA | 把in ba-construction | 把, 将 |
| CC | coordinating conjuction | 和 |
| CD-NOR* | cardinal number | 一, 百 |
| CD-AFF* | affix used in cardinal number | 点, 多 |
| CS | subordinating conjunction | 虽然 |
| DEC | 的in a relative-clause | 的 |
| DEG | associative 的 | 的 |
| DER | 得in V-de construction and V-de-R | 得 |
| DEV | 地before VP | 地 |
| DT | determiner | 这 |
| ETC | for words 等, 等等 | 等, 等等 |
| FW | foreign words | ａ, ｚ, Ａ, Ｚ |
| IJ | interjection | 啊 |
| JJ | other noun-modifier | 男, 共同 |
| LB | 被in long bei-construction | 被, 给 |
| SB | 被in short bei-construction | 被, 给 |
| LC | localizer | 里 |
| M | measure word | 个 |
| MSP | other particle | 所 |
| NN | common noun | 书 |
| NR-PER-FAM* | CJK family name | 吴, 松本 |
| NR-PER-GIV* | CJK given name | 翠玲, 樱子 |
| NR-PER-FOR* | transliteration (foreign) person name | 阿里巴巴 |
| NR-PER-OTH* | other person name | 关公, 鲁迅 |
| NR-LOC* | place name | 中国, 富士山 |
| NR-ORG* | organization name | 富士通, 民主党 |
| NR-OTH* | other proper name | 木星, 秦朝 |
| NT-NOR* | temporal noun | 今天, 冬季 |
| NT-AFF* | affix used in temporal noun | 年, 月 |
| OD-NOR* | ordinal number | 首, 初 |
| OD-AFF* | affix used in ordinal number | 第 |
| ON | onomatopoeia | 哈哈, 哗哗 |
| P | preposition excluding 被and 把 | 从, 对于 |
| PN | pronoun | 他, 大家 |
| PU | punctuation | ？。、 |
| SP | sentence-final particle | 吗, 呢 |
| VA | predicative adjective | 红, 雪白 |
| VC | 是 | 是 |
| VE | 有as the main verb | 有 |
| VV | other verb | 走 |

# List of Publications

## Major Publications

### Journal Papers

(1) C.L. Goh, M. Asahara, Y. Matsumoto, "Machine Learning-based Methods to Chinese Unknown Word Detection and POS Tag Guessing", Journal of Chinese Language and Computing, Vol.16, No.4, pp.185-206, 2006.

(2) C.L. Goh, M. Asahara, Y. Matsumoto, "Training Multi-Classifiers for Chinese Unknown Word Detection", Journal of Chinese Language and Computing, Vol.15, No.1, pp.1-12, 2005.

(3) C.L. Goh, M. Asahara, Y. Matsumoto, "Chinese Word Segmentation by Classification of Characters" International Journal of Computational Linguistics and Chinese Language Processing, Vol.10 No.3., pp.381-396, September 2005.

### International Conferences/Workshops (Reviewed)

(4) C.L. Goh, J. Lü, Y.C. Cheng, M. Asahara, and Y. Matsumoto, "The Construction of a Dictionary for a Two-layer Chinese Morphological Analyzer", In Proc. of The 20th Pacific Asia Conference on Language, Information and Computation (PACLIC 20), pp.??–??, November 2006.

(5) C.L. Goh, M. Asahara, and Y. Matsumoto, "Building a Japanese-Chinese Dictionary Using Kanji/Hanzi Conversion", In Proc. of Natural Language Processing – IJCNLP 2005, Second International Joint Conference, Lecture Notes in Artificial Intelligence 3651, Robert Dale, Kam-Fai Wong, Jian Su, Oi Yee Kwong (eds.), pp.670–681, October 2005.

(6) C.L. Goh, M. Asahara, and Y. Matsumoto, "Training Multi-Classifiers for Chinese Unknown Word Detection", In Proc. of International Conference of Chinese Computing (ICCC), pp.1–8, March 2005.

(7) C.L. Goh, M. Asahara, and Y. Matsumoto, "Pruning False Unknown Words to Improve Chinese Word Segmentation", In Proc. of The 18th Pacific Asia Conference on Language, Information and Computation (PACLIC 18), pp.139–149, December 2004.

(8) C.L. Goh, M. Asahara, and Y. Matsumoto, "Chinese Word Segmentation by Classification of Characters", Proc. of Third SIGHAN Workshop on Chinese Language Processing, pp.57–64, July 2004.

(9) C.L. Goh, M. Asahara, and Y. Matsumoto, "Chinese Unknown Word Identification Using Position Tagging and Chunking", ACL 2003: 41st Annual Meeting of the Association for Computational Linguistics, Interactive Poster/Demo Sessions, Companion volume of the Proceedings, pp.197–200, July 2003.

**Local Workshops (Domestic)**

(10) C.L. Goh, Y.C. Cheng, J. Lü, M. Asahara, and Y. Matsumoto, "A Practical Morphological Analyzer Based on Penn Chinese Treebank Standard", In Proc. of the 12th Annual Meeting of the Association for Natural Language Processing (Gengo-shori-gakkai Nenzi Taikai Ronbun-shu), pp.540–543, March 2006.

(11) C.L. Goh, Y.C. Cheng, M. Asahara, and Y. Matsumoto, "The Development of Chinese ChaSen" (in Japanese), In Proc. of the 11th Annual Meeting of the Association for Natural Language Processing (Gengo-shori-gakkai Nenzi Taikai Ronbun-shu), pp.245–248, March 2005.

(12) C.L. Goh, M. Asahara, and Y. Matsumoto, "Solving Segmentation Ambiguities in Chinese", In Proc. of the 10th Annual Meeting of the Association for Natural Language Processing (Gengo-shori-gakkai Nenzi Taikai Ronbun-shu), pp.701–704, March 2004.

(13) C.L. Goh, M. Asahara, and Y. Matsumoto, "Multi-Classifier for Chinese Unknown Word Detection", In Proc. of the 10th Annual Meeting of the Association for Natural Language Processing (Gengo-shori-gakkai Nenzi Taikai Ronbun-shu), pp.705–708, March 2004.

(14) C.L. Goh, M. Asahara, and Y. Matsumoto, "Chinese Unknown Word Identification based on Morphological Analysis and Chunking", IPSJ SIG Notes, Information Processing Society of Japan, 2003-NL-155, pp.7–12, May 2003.

## Other Publications

### International Workshops

(15) M. Asahara, K. Fukuoka, A. Azuma, C.L. Goh, Y. Watanabe, Y. Matsumoto, and T. Tsuzuki, "Combination of Machine Learning Methods for Optimum Chinese Word Segmentation", In Proc. of Fourth SIGHAN Workshop on Chinese Language (Bakeoff paper), pp.134–137, October 2005.

(16) M. Asahara, C.L. Goh, X. Wang, and Y. Matsumoto, "Combining Segmenter and Chunker for Chinese Word Segmentation", In Proc. of Second SIGHAN Workshop on Chinese Language (Bakeoff paper), pp.144–147, July 2003.