Computational Linguistics and Chinese Language Processing
Vol. 10, No. 3, September 2005, pp. 381-396
© The Association for Computational Linguistics and Chinese Language Processing

Chinese Word Segmentation by Classification of Characters

Chooi-Ling Goh*, Masayuki Asahara* and Yuji Matsumoto*

Abstract

During the process of Chinese word segmentation, two main problems occur: segmentation ambiguities and unknown word occurrences. This paper describes a method to solve the segmentation problem. First, we use a dictionary-based approach to segment the text. We apply the Maximum Matching algorithm to segment the text forwards (FMM) and backwards (BMM). Based on the difference between FMM and BMM, and the context, we apply a classification method based on Support Vector Machines to re-assign the word boundaries. In so doing, we use the output of a dictionary-based approach, and then apply a machine-learning-based approach to solve the segmentation problem. Experimental results show that our model can achieve an F-measure of 99.0 for overall segmentation, given the condition that there are no unknown words in the text, and an F-measure of 95.1 if unknown words exist.

Keywords: Chinese, word segmentation, segmentation ambiguity, unknown word, maximum matching algorithm, support vector machines

1. Introduction

The first step in Chinese information processing is word segmentation. This is because in written Chinese, all characters are joined together, and there are no separators to mark word boundaries. A similar problem also occurs with languages like Japanese, but at least with Japanese, there are three types of characters (hiragana, katakana and kanji). This helps provide clues for finding word boundaries. In the case of Chinese, as there is only one type of character (hanzi), more segmentation ambiguities may occur in a text. During the process of segmentation, two main problems are encountered: segmentation ambiguities and unknown word occurrences. This paper focuses on solving the segmentation ambiguity problem and proposes a sub-model to solve the unknown word problem. There are basically two types of segmentation ambiguity: covering ambiguity and overlapping ambiguity. The definitions are

* Graduate School of Information Science, Nara Institute of Science and Technology, Japan
E-mail: {ling-g, masayu-a, matsu}@is.naist.jp

[Received August 31, 2004; Revised February 21, 2005; Accepted February 21, 2005]

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)

We intend to solve the ambiguity problems by combining a dictionary-based approach with a statistical model. In so doing, we make use of the information in a dictionary in a statistical approach. The Maximum Matching (MM) algorithm, a very early and simple dictionary-based approach, is used to initially segment the text by referring to a dictionary. It tries to match the longest possible words found in the dictionary. We can parse a sentence either forwards or backwards. Normally, the differences between the results of forward and backward parsing will indicate the locations where overlapping ambiguities occur. Then, we use a Support Vector Machine-based (SVM) classifier to decide which output should be the correct answer. As for covering ambiguities, in most cases, forward and backward MM will give the same output. In this case, we just make use of the contexts to decide whether or not to split a word into two or more words. Our experimental results show that the proposed method can solve 92% of overlapping ambiguities and 52% of covering ambiguities.

2. Previous Works

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*. [2003] 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 [Luo *et al.* 2002]. They focused on 90 ambiguous words and achieved an accuracy of 96.58%.

Most of the previous methods reported on the accuracy of overall segmentation. Recently, many researches have adopted multiple models. Furthermore, most researchers have realized that character-based approaches are more effective than word-based approaches to Chinese word segmentation. In [Xue and Converse 2002], two classifiers were combined to perform Chinese word segmentation. First, a Maximum Entropy model was used to segment the text, and then an error driven transformation model was used to correct the word boundaries. Their method also used character-based tagging to assign the positions of characters in words. They achieved an F-measure of 95.17 using the Penn Chinese Treebank. Another recent study was that of Fu and Luke [2003], who proposed hybrid models for integrated segmentation. Modified word juncture models and word-formation patterns were used to find word boundaries and at the same time to identify unknown words. They achieved and F-measure of 96.1 using the Peking University Corpus. As the above studies used different corpora in their experiments, it is difficult to tell which method performed better.

Solving the unknown word problem is also an important step in word segmentation. An unknown word is a word not found in a dictionary. Therefore, it cannot be segmented correctly by simply referring to the dictionary. Many approaches for unknown word detection

have been proposed [Chen and Bai 1997; Chen and Ma 2002; Fu and Wang 1999; Lai and Wu 1999; Ma and Chen 2003; Nie *et al*. 1995; Shen *et al.* 1998; Zhang *et al*. 2002; Zhou and Lua 1997]. These include rule-based, statistics-based, and hybrid models. We cannot ignore the unknown word problem since there are always some unknown words (such as person names, numbers etc.) in a text even when we use a very large dictionary. The creation of new words in Chinese is a continuous process. For example, names for new diseases, technical terms, and new expressions are always being created. The accuracy is better if one focuses only on certain types of unknown words such as person names, place names, or transliteration names, when accuracy of over 80% can be achieved. However, for general unknown words, such as common nouns, verbs etc., the accuracy ranges from only 50% to 70%.

3. Proposed Method

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 a machine learning based model, which in our case is the Support Vector Machines model [Vapnik 1995]. Our method employs a model to solve the ambiguity problem and, at the same time, embeds a model to detect unknown words. We will next describe the method in more detail in the following section.

3.1 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

[1997], 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.

3.2 Support Vector Machines

Support Vector Machines (SVM) [Vapnik 1995] are binary classifiers that search for a hyperplane with the largest possible margin between positive and negative samples (see Figure 1). Suppose we have a set of training data for a binary class problem: $(\mathbf{x}_1, y_1),\ldots, (\mathbf{x}_N, y_N)$, where $\mathbf{x}_i \in R^n$ is the feature vector of the ith sample in the training data and $y_i \in \{+1, -1\}$ is its label. The goal is to find a decision function which accurately predicts the label 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 $\mathbf{x}$ is a negative member. The vectors $\mathbf{z}_i$ are called support vectors, and they are assigned a non-zero weight α_i. Support vectors and the parameters are determined by solving a quadratic programming problem. $K(\mathbf{x}, \mathbf{z})$ is a kernel function which computes an extended inner product of input vectors. We use a polynomial kernel function of degree 2, that is, $K(\mathbf{x}, \mathbf{z}) = (1 + \mathbf{x} \cdot \mathbf{z})^2$.

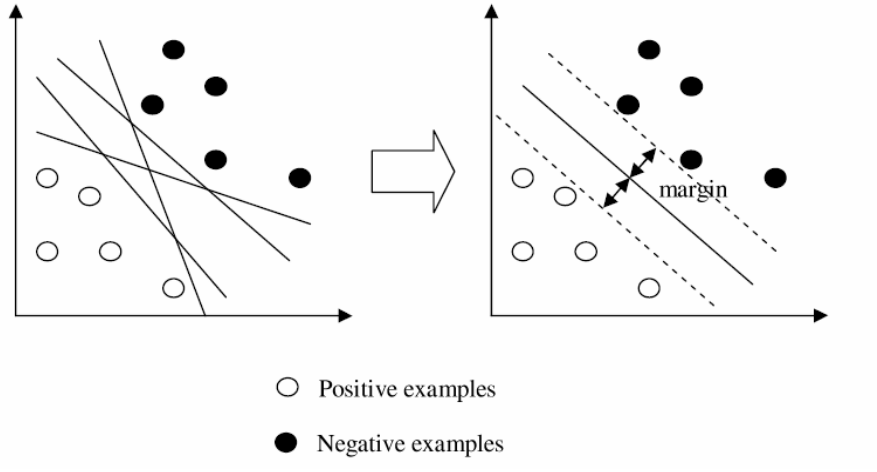

Figure 1. Maximizing the margin

We use *YamCha* [Kudo and Matsumoto 2001] to train our SVM models. *YamCha* is an SVM-based multi-purpose chunker. It extends binary classification to *n*-class classification for natural language processing purposes, where we would normally want to classify the words into several classes, as in the case of POS tagging or base phrase chunking. Two straightforward methods are mainly used for this extension, the "one-vs-rest" method and the "pairwise" method. In the "one-vs-rest" method, *n* binary classifiers are used to compare one class with the rest of the classes. In the "pairwise" method, $\binom{n}{2}$ binary classifiers are used to compare between all pairs of classes. We need to classify the characters into 4 categories (B, I, E or S, as shown in Table 1) in our method. We used the "pairwise" classification method in our experiments because it is more efficient during the training phase. Details of the system can be found in [Kudo and Matsumoto 2001].

Table 1. Position tags in a word (BIES tags)

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

3.3 Classification of Characters

We intend to classify the characters using the SVM-based chunker [Kudo and Matsumoto 2001] as described in Section 3.2. [Xue and Converse 2002] 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 postion 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 by SVM is guided by a dictionary for known word segmentation. The similarities and differences between FMM and BMM are used to train the SVM to solve 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 1. The BIES tags are as described in [Uchimoto *et al*. 2000] and [Sang and Veenstra 1999] 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 2 shows the features used to determine the tag of the character "春" at location *i*. 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 segmentation "迎/新春/联谊会/上/" (welcome/ new year/ get-together party/ at/).

Position	Char.	FMM	BMM	Output
i-2	迎	B	S	S
i-1	新	E	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 2. An illustration of classification process applied to "At the New Year gathering party"

4. Experiments and Results

We run our experiments with two datasets, the PKU Corpus and the SIGHAN Bakeoff data. The evaluation was conducted using the tool provided in SIGHAN Bakeoff [Sproat and Emerson 2003].

4.1 Experiment with the PKU Corpus

4.1.1 Accuracy on Solving Ambiguity Problem

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, for training and testing, respectively. Since our purpose in this experiment was only to solve the ambiguity problem, not the unknown word

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

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 62,030 entries (referred to as Experiment 1). 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 2, "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 ambiguity 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.

Table 2. Disambiguation results obtained with the PKU Corpus

Category	Conditions	No. of Char.	Percentage
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* ≠ *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* ≠ *Out*	1738	0.51%
Allwrong	$O_f = O_b$ = *Ans* and *Ans* ≠ *Out*	571	0.17%
Total		342726	100.00%

Table 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.

Table 3. Segmentation results obtained with the PKU Corpus

	FMM	BMM	SVM (char. only)	FMM +SVM	BMM +SVM	FMM+BMM+SVM (=Experiment 1)
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 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.

4.1.2 Accuracy in Solving the Unknown Word Problem

The corpus used in this experiment was the same as that described in Section 4.1.1, but the setting is different. In this round, we divided the corpus into three sets, referred to as Set 1, Set 2, and Set 3. Set 1 plus Set 2 (80%) was used for training, and Set 3 (20%) was used for testing, just as in the previous experiment. The difference was in the preparation of the dictionary. It was prepared in two ways. In the first case, all the words from Set 1 and Set 2 were used to create the dictionary. There were 49,433 entries in the dictionary and 8,346 (4.0%) unknown words in the testing data (referred to as Experiment 2). This experiment was conducted to investigate the performance of the method when unknown words exist. In the second case, only the words from Set 1 were used to create the dictionary, resulting in a situation where unknown words existed in the training data (referred to as Experiment 3). The top part of Table 4 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 served as a learning model for unknown word detection[2]. When we segmented Set 2 using FMM and BMM, most of the unknown words were segmented into single characters (namely tag 'S'). Based on these tags and contexts, the SVM-based chunker was trained to change the

[2] 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.

tags into the correct answers. The last experiment (referred to as Experiment 4) was the opposite of Experiment 2; nothing was used to create the dictionary. All the words were considered to be unknown words. Only the characters were used as features during the classification phase, meaning that no information from FMM and BMM was available.

Table 4. Different settings and segmentation results with unknown words (PKU Corpus)

	Experiment 1	Experiment 2	Experiment 3			Experiment 4
Set 1(%)/ Set 2(%)		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

The bottom part of Table 4 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 in 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 3 shows the F-measure results for segmentation and recall results for unknown words and known words, when different proportions of the training corpus were used to create the dictionary.

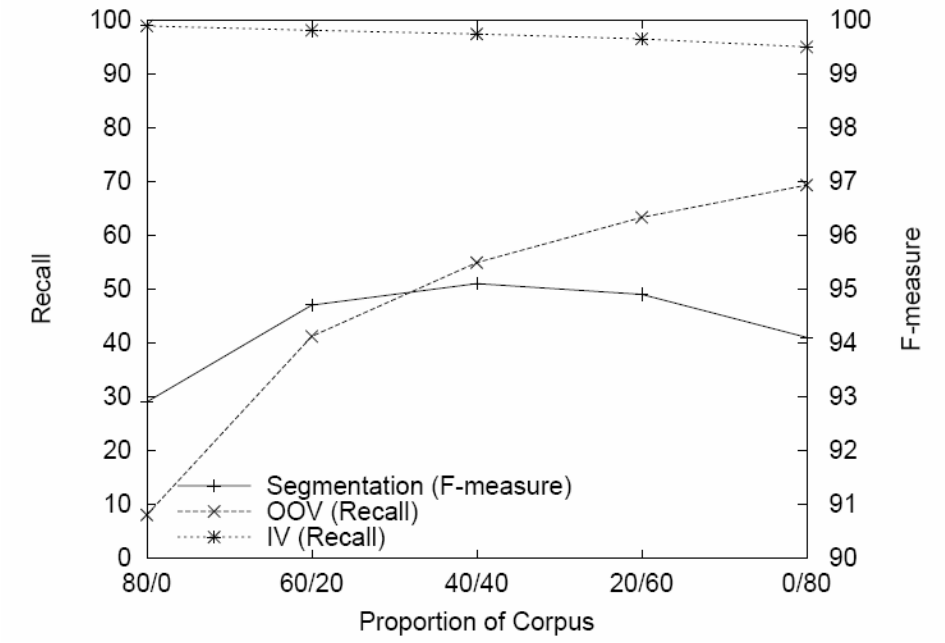

Figure 3. Accuracy of segmentation (F-measure), OOV (Recall) and IV (Recall)

4.2 Experiment with SIGHAN Bakeoff Data

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 segmentation. Therefore, it is always difficult to compare the results obtained with different methods as the data used is different. The First International Chinese Word Segmentation Bakeoff [Sproat and Emerson 2003] intended to evaluate the accuracy of different segmenters by standardizing the training and testing data. In their closed test, only the training data were 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 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 obtained distribution of the data is shown in Table 5. 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.

Table 5. Bakeoff data

Corpus	# of train words	# of test words	Unknown word rate	Size of original dictionary	Size of dictionary used
PKU	1.1M	17,194	6.9%	55,226	36,830
CHTB	250K	39,922	18.1%	19,730	12,274
AS	5.8M	11,985	2.2%	146,226	100,161
HK	240K	34,955	7.1%	23,747	17,207

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[3]. As a result, the new dictionaries contained fewer entries than the original dictionaries. Table 5 shows the details for the setting.

Table 6. Segmentation results obtained with bakeoff data

Corpus	*Recall*	*Precision*	*F-measure*	$Recall_{unknown}$	$Recall_{known}$
PKU	95.5	94.1	94.7	71.0	97.3
CHTB	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

As observed in [Sproat and Emerson 2003], 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 6. 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. During the bakeoff, only two participants took part in all four tracks in the closed test. We obtained better results than one of them [Asahara *et al.* 2003], 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 algorithm, achieved better results than theirs when complicated statistics-based models were involved. On the other hand, compare to the results obtained by [Zhang *et al*. 2003], we only obtained better results for two

[3] 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.

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 [Zhang *et al.* 2002], which gave better results for unknown word recognition. The bottom of Figure 4 shows the results of unknown word detection. Again, our method performed comparatively well in detecting unknown words.

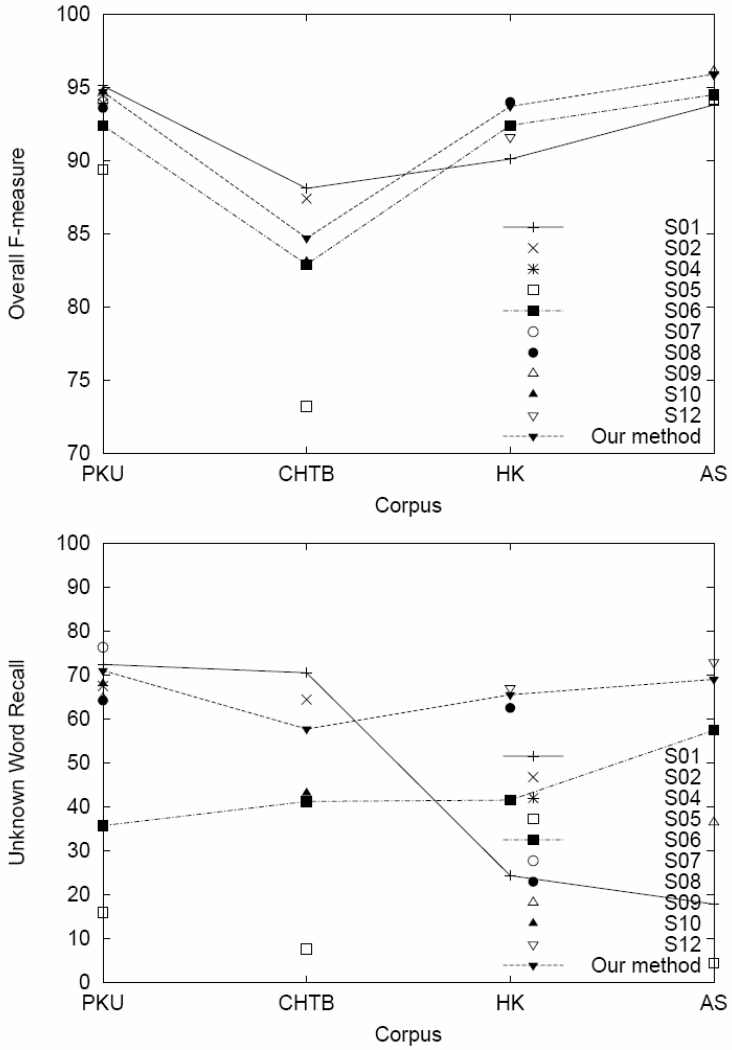

Figure 4. Comparision of bakeoff results (overall F-measure and unknown word recall)

Regarding Chinese word segmentation problem as character tagging problem has previously been seen in [Xue and Converse 2002]. 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 [2002] 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 [Xue and Shen 2003]. 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 CHTB 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 words, as in Experiment 4, can be obtained.

5. Conclusion

Our proposed method generated better results than the baseline models, namely, FMM and BMM. We achieved nearly 99% recall when unknown words did not exist. However, in the real world, unknown words always exist in texts, even if we use a very large dictionary. Therefore, we also embed a model to detect unknown words. Unfortunately, while the accuracy achieved in unknown word detection increases, the performance in solving the known word ambiguity problem declines. As shown by the experiments on the bakeoff data, our model works well only when the training corpus is large. In conclusion, while our model is suitable for solving the segmentation ambiguity problem, it can also perform unknown word detection at the same time. However we still need to find a balance that will enable us to solve these two problems optimally. We also need to research the relationship between the training corpus size and the best proportion of the corpus used to create the dictionary for training to solve the ambiguity problem and perform unknown word detection.

Acknowledgements

Thanks go to Mr. Kudo for his Support Vector Machine-based chunker tool, *Yamcha*. We also thank Peking University and SIGHAN for providing the corpora used in our experiments. Finally, we thank the reviewers for their invaluable and insightful comments.

Reference

Asahara, M., C.L. Goh, X.J. Wang and Y. Matsumoto, "Combining Segmenter and Chunker for Chinese Word Segmentation," In *Proceedings of Second SIGHAN Workshop on Chinese Language Processing*, 2003, pp. 144–147.

Chen, K.J. and M.H. Bai, "Unknown Word Detection for Chinese By a Corpus-based Learning Method," In *Proceedings of ROCLING X*, 1997, pp. 159–174.

Chen, K.J. and W.Y. Ma, "Unknown Word Extraction for Chinese Documents," In *Proceedings of COLING 2002*, 2002, pp. 169–175.

Fu, G.H. and K.K. Luke, "An Integrated Approach for Chinese Word Segmentation," In *Proceedings of PACLIC 17*, 2003, pp. 80–87.

Fu, G.H. and X.L. Wang, "Unsupervised Chinese Word Segmentation and Unknown Word Identification," In *Proceedings of NLPRS*, 1999, pp. 32–37.

Huang, C.N., "Segmentation Problem in Chinese Processing," *Applied Linguistics*, 1, 1997, pp. 72–78.

Kudo, T. and Y. Matsumoto, "Chunking with Support Vector Machines," In *Proceedings of NAACL*, 2001, pp. 192–199.

Lai, Y.S. and C.H. Wu, "Unknown Word and Phrase Extraction Using a Phrase-Like-Unit-Based Likelihood Ratio," In *Proceeding of ICCPOL '99*, 1999, pp. 5–9.

Li, M., J.F. Gao, C.N. Huang and J.F. Li, "Unsupervised Training for Overlapping Ambiguity Resolution in Chinese Word Segmentation," In *Proceedings of Second SIGHAN Workshop on Chinese Language Processing*, 2003, pp. 1–7.

Luo, X., M.S. Sun and B. K. Tsou, "Covering Ambiguity Resolution in Chinese Word Segmentation Based on Contextual Information," In *Proceedings of COLING 2002*, 2002, pp. 598–604.

Ma, W.Y. and K.J. Chen, "A Bottom-up Merging Algorithm for Chinese Unknown Word Extraction," In *Proceedings of Second SIGHAN Workshop on Chinese Language Processing*, 2003, pages 31–38.

Nie, J.Y., M.-L. Hannan and W.Y. Jin, "Unknown Word Detection and Segmentation of Chinese Using Statistical and Heuristic Knowledge," *Communications of COLIPS*, 5, 1995, pp. 47–57.

Sang, E. F.-T.K. and J. Veenstra, "Representing Text Chunks," In *Proceedings of EACL '99*, 1999, pp. 173–179.

Shen, D.Y., M.S. Sun, and C.N. Huang, "The application & implementation of local statistics in Chinese unknown word identification," *Communications of COLIPS*, 8(1), 1998, pp. 119–128.

Sproat, R. and T. Emerson, "The First International Chinese Word Segmentation Bakeoff," In *Proceedings of Second SIGHAN Workshop on Chinese Language Processing*, 2003, pp. 133–143.

Uchimoto, K., Q. Ma, M. Murata, H. Ozaku and H. Isahara, "Named Entity Extraction Based on A Maximum Entropy Model and Transformational Rules," In *Processing of the ACL 2000*, 2000, pp. 326–335.

Vapnik, V. N., *The Nature of Statistical Learning Theory*, Springer, 1995.

Xue, N.W. and S. P. Converse, "Combining Classifiers for Chinese Word Segmentation," In *Proceedings of First SIGHAN Workshop on Chinese Language Processing*, 2002, pp. 57–63.

Xue, N.W. and L.B. Shen, "Chinese Word Segmentation as LMR Tagging," In *Proceedings of Second SIGHAN Workshop on Chinese Language Processing*, 2003, pp. 176–179.

Zhang, H.P., Q. Liu, H. Zhang and X.Q. Cheng, "Automatic Recognition of Chinese Unknown Words Based on Roles Tagging," In *Proceedings of First SIGHAN Workshop on Chinese Language Processing*, 2002, pp. 71-77.

Zhang, H.P., H.K. Yu, D.Y. Xiong and Q. Liu, "HHMM-based Chinese Lexical Analyzer ICTCLAS," In *Proceedings of Second SIGHAN Workshop on Chinese Language Processing*, 2003, pp. 184–187.

Zhou, G.D. and K.T. Lua, "Detection of Unknown Chinese Words Using a Hybrid Approach," *Computer Processing of Oriental Language*, 11(1), 1997, pp. 63–75.