Chinese Word Segmentation Using Minimal Linguistic Knowledge

Aitao Chen
School of Information Management and Systems
University of California at Berkeley
Berkeley, CA 94720, USA
aitao@sims.berkeley.edu

Abstract

This paper presents a primarily data-driven Chinese word segmentation system and its performances on the closed track using two corpora at the first international Chinese word segmentation bakeoff. The system consists of a new words recognizer, a base segmentation algorithm, and procedures for combining single characters, suffixes, and checking segmentation consistencies.

1 Introduction

At the first Chinese word segmentation bakeoff, we participated in the closed track using the Academia Sinica corpus (AS for short) and the Beijing University corpus (PK for short). We will refer to the segmented texts in the training corpus as the *training data*, and to both the unsegmented testing texts and the segmented texts (the reference texts) as the *testing data*. For details on the word segmentation bakeoff, see (Sproat and Emerson, 2003).

2 Word segmentation

New texts are segmented in four steps which are described in this section. New words are automatically extracted from the unsegmented testing texts and added to the base dictionary consisting of words from the training data before the testing texts are segmented, line by line.

2.1 Base segmentation algorithm

Given a dictionary and a sentence, our base segmentation algorithm finds all possible segmentations of the sentence with respect to the dictionary, computes the probability of each segmentation, and chooses the segmentation with the highest probability. If a sentence of n characters, $S = c_1c_2 \ldots c_n$, has a segmentation of m words, $S = w_1w_2 \ldots w_m$, then the probability of the segmentation is estimated as $p(S,T) = p(w_1w_2 \ldots w_m) \approx \prod_{i=1}^{m} p(w_i)$, where T denotes a segmentation of a sentence. The probability of a word is estimated from the training corpus as $p(w) \approx \frac{N(w)}{N}$, where $N(w)$ is the number of times that the word w occurs in the training corpus, and N is the number of words in the training corpus. When a word is not in the dictionary, a frequency of 0.5 is assigned to the new word. The dynamic programming technique is applied to find the segmentation of the highest probability of a sentence without first enumerating all possible segmentations of the sentence with respect to the dictionary. Consider the text fragment 牡丹花木, with respect to a dictionary containing the words 牡丹, 牡丹花, 花木, 花 and 木, it has three segmentations: (1) 牡丹 / 花木; (2) 牡丹花 / 木; and (3) 牡丹 / 花 / 木. The probabilities of the three segmentations are computed as: (1) p(牡丹)*p(花木); (2) p(牡丹花)*p(木); (3) p(牡丹)*p(花)*p(木). The probability of a word is estimated by its relative frequency in the training data. Assume the first segmentation has the highest probability, then the text fragment will be segmented into 牡丹 / 花木.

2.2 Combining single characters

New words are usually two or more characters long and are often segmented into single characters. For example, the word 空阔 is segmented into 空 / 阔 when it is not in the dictionary. After a sentence is segmented using the base algorithm, the consecutive single Hanzi characters are combined into a word if the in-word probabilities of the single characters are over a threshold which is empirically determined from the training data. The *in-word* probability of a character is the probability that the character occurs in a word of two or more characters.

Some Hanzi characters, such as 的 and 了, occur as words on their own in segmented texts much more frequently than in words of two or more characters. For example, in the PK training corpus, the character 了 occurs as a word on its own 11,559 times, but in a word only 875 times. On the other hand, some Hanzi characters usually do not occur alone as words, instead they occur as part of a word. As an example, the character 国 occurs in a word 17,108 times, but as a word alone only 794 times in the PK training data. For each character in the training data, we compute its in-word probability as follow: $p(C_{inword}) = \frac{N(C_{inword})}{N(C)}$, where $N(C)$ is the number of times that character C occurs in the training data, and $N(C_{inword})$ is the number of times that character C is in a word of two or more characters.

We do not want to combine the single characters that occur as words alone more often than not. For both the PK training data and the AS training data, we divided the training data into two parts, two thirds for training, and one third for system development. We found that setting the threshold of the in-word probability to 0.85 or around works best on the development data. After the initial segmentation of a sentence, the consecutive single-characters are combined into one word if their in-word probabilities are over the threshold of 0.85. The text fragment 人人身着迷彩服 contains a new word 迷彩服 which is not in the PK training data. After the initial segmentation, the text is segmented into 人人 / 身着 / 迷 / 彩 / 服 /, which is subsequently changed into 人人 / 身着 / 迷彩服 after combining the three consecutive characters. The in-word probabilities for the three characters 迷，彩， and 服 are 0.94, 0.98, and 0.99, respectively.

2.3 Combining suffixes

A small set of characters , such as 者，性 and 化， frequently occur as the last character in words. We selected 145 such characters from the PK training corpus, and 113 from the AS corpus. After combining single characters, we combine a suffix character with the word preceding it if the preceding word is at least two-character long.

2.4 Consistency check

The last step is to perform consistency checks. A segmented sentence, after combining single characters and suffixes, is checked against the training data to make sure that a text fragment in a testing sentence is segmented in the same way as in the training data if it also occurs in the training data. From the PK training corpus, we created a *phrase segmentation table* consisting of word quadgrams, trigrams, bigrams, and unigrams, together with their segmentations and frequencies. Our phrase table created from the AS corpus does not include word quad-grams to reduce the size of the phrase table. For example, from the training text 明天 / 就 / 是 / 元旦， we create the following entries (only some are listed to save space):

text fragment	freq	segmentation
明天就是元旦	1	明天 / 就 / 是 / 元旦
明天就是	1	明天 / 就 / 是
就是元旦	1	就 / 是 / 元旦
就是	1	就 / 是
元旦	1	元旦

After a new sentence is processed by the first three steps, we look up every word quad-grams of the segmented sentence in the phrase segmentation table. When a word quadgram is found in the phrase segmentation table with a different segmentation, we replace the segmentation of the word quad-gram in the segmented sentence by its segmentation found in the phrase table. This process is continued to word trigrams, word bigrams, and word unigrams. The idea is that if a text fragment in a new sentence is found in the training data, then it should be segmented in the same way as in the training data. As an example, in the PK testing data, the sentence 明天就是农历羊年的大年初一。 is segmented into 明天 / 就是 / 农历 / 羊 / 年 / 的 / 大年初一 / 。 after the first three steps (the two characters 羊 and 年 are not, but should be, combined because the in-word probability of character 羊， which is 0.71, is below the pre-defined threshold of 0.85). The word bigram 明天就是 is found in the phrase segmentation table with a different segmentation, 明天 / 就 / 是. So the segmentation 明天 / 就是 is changed to the segmentation 明天 / 就 / 是 in the final segmented sentence. In essence, when a text fragment has two or more segmentations, its surrounding context, which can be the preceding word, the following word, or both, is utilized to choose the most appropriate segmentation. When a text fragment in a testing sentence never occurred in the same context in the training data, then the most frequent segmentation found in the training data is chosen. Consider the text 就是 again, in the testing data, ，就是事先 is segmented into ，/ 就是 / 事先 by our base algorithm. In this case, 就是 never occurred in the context of ，就是事先，，就是 or 就是事先. The consistency check step changes ，/ 就是 / 事先 into ，/ 就 / 是 / 事先 since 就是 is segmented into 就 / 是 515 times, but is treated as one word 就是 105 times in the training data.

3 New words recognition

We developed a few procedures to identify new words in the testing data. Our first procedure is designed to recognize numbers, dates, percent, time, foreign words, etc. We defined a set of characters consisting of characters such as the digits '0' to '9' (in ASCII and GB), the letters 'a' to 'z', 'A' to 'Z' (in ASCII and GB), '零一二三四五六七八九十百千万亿．点分之第年％, and the like. Any consecutive sequence of the characters that are in this predefined set of characters is extracted and post-processed. A set of rules is implemented in the post-processor. One such rule is that if an extracted text fragments ends with the character 年 and contains any character in 十百千万亿第， then remove the ending character 年 and keep the remaining fragment as a word. For example, our recognizer will extract the text fragment 四百八十年 and 第九年 since all the characters are in the pre-defined set of characters. The post-processor will strip off the trailing character 年， and return 四百八十 and 第九 as words. For personal names, we developed a program to extract the names preceding texts such as （高山族） and （女）， a program to detect and extract names in a sequence of names separated by the Chinese punctuation "、", such as 这些作品是吕厚民、侯波、吕相友、， a program to extract

	steps	dict	R	P	F	R_{oov}	R_{iv}
1	1	pkd1	0.919	0.838	0.877	0.050	0.984
2	1	pkd2	0.940	0.892	0.915	0.347	0.984
3	1	pkd3	0.949	0.920	0.934	0.507	0.982
4	1-2	pkd3	0.950	0.935	0.942	0.610	0.975
5	1-3	pkd3	0.951	0.940	0.945	0.655	0.972
6	1-4	pkd3	**0.955**	**0.938**	**0.946**	**0.647**	**0.977**

Table 1: Results for the closed track using the PK corpus.

	steps	dict	R	P	F	R_{oov}	R_{iv}
1	1	asd1	0.950	0.936	0.943	0.000	0.970
2	1	asd2	0.950	0.943	0.947	0.132	0.968
3	1-2	asd2	0.951	0.952	0.951	0.337	0.964
4	1-3	asd2	0.949	0.952	0.951	0.372	0.961
5	1-4	asd2	**0.966**	**0.956**	**0.961**	**0.364**	**0.980**

Table 2: Results for the closed track using the AS corpus.

corpus	dict	R	P	F	R_{oov}	R_{iv}
AS	asd1	0.917	0.912	0.915	0.000	0.938
PK	pkd1	0.909	0.829	0.867	0.050	0.972

Table 3: Performances of the maximum matching (forward) using words from the training data.

personal names (Chinese or foreign) following title or profession names, such as 刘诗昆 in the text 著名钢琴家刘诗昆, and a program to extract Chinese personal names based on the preceding word and the following word. For example, the string 孙晓明 in 工作的孙晓明说 is most likely a personal name (in this case, it is) since 孙 is a Chinese family name, the string is three-character long (a typical Chinese personal name is either three or two-character long). Furthermore, the preceding word 的 and the following word 说 are highly unlikely to appear in a Chinese personal name. For the personal names extracted from the PK testing data, if the name is two or three-character long, and if the first character or two is a Chinese family name, then the family name is separated from the given name. The family names are not separated from the given names for the personal names extracted from the AS testing data. In some cases, we find it difficult to decide whether or not the first character should be removed from a personal name. Consider the personal name 叶利钦 which looks like a Chinese personal name since the first character is a Chinese family name, and the name is three-character long. If it is a translated foreign name (in this case, it is), then the name should not be split into family name and given name. But if it is the name of a Chinese personal name, then the family name 叶 should be separated from the given name. For place names, we developed a simple program to extract names of cities, counties, towns, villages, streets, etc, by extracting the strings of up to three characters appearing between two place name designators. For example, from the text 江西临川市洋洲镇洋洲村, our program will extract 洋洲镇 and 洋洲村.

4 Results

The last row (in boldface) in Table 1 gives our official results for the PK closed track. Other rows in the table present the results under different experimental conditions. The column labeled *steps* refers to the executed steps of our Chinese word segmentation algorithm. Step 1 segments a text using the base segmentation algorithm, step 2 combines single characters, step 3 attaches suffixes to the preceding words, and step 4 performs consistency checks. The four steps are described in details in section 2. The column labeled *dict* gives the dictionary used in each experiment. The *pkd1* consists of only the words from the PK training corpus, *pkd2* consists of the words in pkd1 and the words converted from pkd1 by changing the GB encoding to ASCII encoding for the numeric digits and the English letters, and *pkd3* consists of the words in pkd2 and the words automatically extracted from the PK testing texts using the procedures described in section 3. The columns labeled *R*, *P* and *F* give the recall, precision, and F score, respectively. The columns labeled R_{oov} and R_{iv} show the recall on out-of-vocabulary words and the recall on in-vocabulary words, respectively. All evaluation scores reported in this paper are computed using the score program written by Richard Sproat. We refer readers to (Sproat and Emerson, 2003) for details on the evaluation measures. For example, row 4 in table 1 gives the results using pkd3 dictionary when a sentence is segmented by the base algorithm, and then the single characters in the initial segmentation are combined, but suffixes are not attached and consistency check is not performed. The last row in table 2 presents our official results for the closed track using the AS corpus. The *asd1* dictionary contains only the words from the AS training corpus, while the *asd2* consists of the words in *asd1* and the new words automatically extracted from the AS testing texts using the new words recognition described in section 3. The results show that new words recognition and joining single characters contributed the most to the increase in precision, while the consistency check contributed the most to the increase in recall. Table 3 gives the results of the maximum matching using only the words in the training data. While the difference between the F-scores of the maximum matching and the base algorithm is small for the PK corpus, the F-score difference for the AS corpus is much larger. Our base algorithm performed substantially better than the maximum matching for the AS corpus. The performances of our base algorithm on the testing data using the words from the training data are presented in row 1 in table 1 for the PK corpus, and row 1 in table 2 for the AS corpus.

5 Discussions

In this section we will examine in some details the problem of segmentation inconsistencies within the training data, within the testing data, and between training data and testing data. Due to space limit, we will only report our findings in the PK corpus though the same kinds of inconsistencies also occur in the AS corpus. We understand that it is difficult, or even impossible, to completely eliminate segmentation inconsistencies. However, perhaps we could learn more about the impact of segmentation inconsistencies on a system's performance by taking a close look at the problem.

We wrote a program that takes as input a segmented corpus and prints out the shortest text fragments in the corpus that have two or more segmentations. For each text fragment, the program also prints out how the text fragment is segmented, and how many times it is segmented in a particular way. While some of the text fragments, such as 等同 and 才能, truly have two different segmentations, depending on the contexts in which they occur or the meanings of the text fragments, others are segmented inconsistently. We ran this program on the PK testing data and found 21 unique shortest text fragments, which occur 87 times in total, that have two different segmentations. Some of the text fragments, such as 黄金周, are inconsistently segmented. The fragment 黄金周 occurs twice in the testing data and is segmented into 黄金 / 周 in one case, but treated as one word in the other case. We found 1,500 unique shortest text fragments in the PK training data that have two or more segmentations, and 97 unique shortest text fragments that are segmented differently in the training data and in the testing data. For example, the text 冰清玉洁 is treated as one word in the training data, but is segmented into 冰 / 清 / 玉 / 洁 in the testing data. We found 11,136 unique shortest text fragments that have two or more segmentations in the AS training data, 21 unique shortest text fragments that have two or more segmentations in the AS testing data, and 38 unique shortest text fragments that have different segmentations in the AS training data and in the AS testing data.

Segmentation inconsistencies not only exists within training and testing data, but also between training and testing data. For example, the text fragment 极大地 occurs 35 times in the PK training data and is consistently segmented into " 极大 / 地, but the same text fragment, occurring twice in the testing data, is segmented into 极 / 大 / 地 in both cases. The text 最新 occurs 67 times in the training data and is treated as one word 最新 in all 67 cases, but the same text, occurring 4 times in the testing data, is segmented into 最 / 新 in all 4 cases. The text 除夕夜 occurs 16 times in the training data, and is treated as one word in all cases, but in the testing data, it is treated as one word in three cases and segmented into 除夕 / 夜 in one case. The text 抱有 is segmented into 抱 / 有 in 8 cases, but treated as one word in one case in the training data. A couple of text fragments seem to be incorrectly segmented. The text 塞尔维亚族人 in the testing data is segmented into 塞尔维亚 / 族人, and the text 共庆春节 segmented into 共 / 庆春节.

Our segmented texts of the *PK* testing data differ from the reference segmented texts for 580 text fragments (427 unique). Out of these 580 text fragments, 126 text fragments are among the shortest text fragments that have one segmentation in the training data, but another in the testing data. This implies that up to 21.7% of the mistakes committed by our system may have been impacted by the segmentation inconsistencies between the PK training data and the PK testing data. Since there are only 38 unique shortest text fragments found in the AS corpus that are segmented differently in the training data and the testing data , the inconsistency problem probably had less impact on our AS results. Out of the same 580 text fragments, 359 text fragments (62%) are new words in the PK testing data. For example, the proper name 巴拉迪, which is a new word, is incorrectly segmented into 巴 / 拉迪 by our system. Another example is the new word 时好时坏 which is treated as one word in the testing data, but is segmented into 时 / 好 / 时 / 坏 by our system. Some of the longer text fragments that are incorrectly segmented may also involve new words, so at least 62%, but under 80%, of the incorrectly segmented text fragments are either new words or involve new words.

6 Conclusion

We have presented our word segmentation system and the results for the closed track using the AS corpus and the PK corpus. The new words recognition, combining single characters, and checking consistencies contributed the most to the increase in precision and recall over the performance of the base segmentation algorithm, which works better than maximum matching. For the closed track experiment using the PK corpus, we found that 62% of the text fragments that are incorrectly segmented by our system are actually new words, which clearly shows that to further improve the performance of our system, a better new words recognition algorithm is necessary. Our failure analysis also indicates that up to 21.7% of the mistakes made by our system for the PK closed track may have been impacted by the segmentation inconsistencies between the training and testing data.

References

Richard Sproat and Tom Emerson. 2003. *The First International Chinese Word Segmentation Bakeoff*. In proceedings of the Second SIGHAN Workshop on Chinese Language Processing, July 11-12, 2003, Sapporo, Japan.