The Automatic Extraction of Open Compounds from Text Corpora

Virach Sornlertlamvanich and Hozumi Tanaka
Department of Computer Science, Tokyo Institute of Technology
2-12-1, Ōokayama, Meguro-ku, Tokyo, Japan 152
{virach,tanaka}@cs.titech.ac.jp

Abstract

This paper describes a new method for extracting open compounds (uninterrupted sequences of words) from text corpora of languages, such as Thai, Japanese and Korea that exhibit unexplicit word segmentation. Without applying word segmentation techniques to the inputted plain text, we generate n-gram data from it. We then count the occurrence of each string and sort them in alphabetical order. It is significant that the frequency of occurrence of strings decreases when the window size of observation is extended. From the statistical point of view, a word is a string with a fixed pattern that is used repeatedly, meaning that it should occur with a higher frequency than a string that is not a word. We observe the variation of frequency of the sorted n-gram data and extract the strings that experience a significant change in frequency of occurrence when their length is extended. We apply this occurrence test to both the right and left hand sides of all strings to ensure the accurate detection of both boundaries of the string. The method returned satisfying results regardless of the size of the input file.

1 Introduction

This paper discusses a method automatic extraction of candidates for open compound registration. An open compound refers to an uninterrupted sequence of words, generally functioning as a single constituent (Smadja and McKeown , 1990). We propose a new method of extraction for languages which have no specific use of punctuation to signify word boundaries. Our method is applied to n-gram text data using statistical observation of the change of frequency of occurrence when the window size of string observation is extended (character) cluster-wise. We generate both rightward and the leftward sorted n-gram data, then determine the left and right boundaries of a string using the methods of *competitive selection* and *unified selection.* In this paper, we examine the result of applying our method to Thai text corpora and also introduce conventional Thai spelling rules to avoid extracting invalid strings.

Previous work (Nagao et al., 1994) has shown an effective way of constructing a sorted file for the efficient calculation of n-gram data. However a surprisely large number of invalid strings were also extracted. Subsequent work, (Ikehara et al., 1995) has extended the sorted file to avoid repeating the counting of substrings contained in strings already counted. This meant the extraction of only the longest strings in the order of frequency of occurrence. The result of extraction was improved as a result, but the determination of longest strings is always made consecutively from left to right. If an erroneous string is extracted, its error directly propagates through the rest of input. It is possible that a string with an invalid starting pattern will be extracted because a string too long in character length has been extracted previously.

In the following sections, we firstly describe the necessity for making this statistical observation for extracting open compounds from Thai text corpora. Then, the methodology of data preparation and open compound extraction is explained. Finally, we discuss the result of an experiment on both large and small test corpora to investigate the effectiveness of our method.

2 Problem Description

It is a non-trivial task to identify a word in the text of a language which has no specific punctuation to mark word boundaries. Up to the present, lexicographers' efforts have been inhibited by insufficient corpora and limited computational facilities. Almost all lexicon knowledge bases have been created with reliance on human intuition. In recent years, a large amount of text corpora have become available, and it is now becoming possible to conduct more rigorous experiments on text corpora. We address the following problems in such a way that they are able to be solved by the way of statistical methods.

1. There is no good evidence to support the itemization of a word in a dictionary. In traditional dictionary making, lexicographers have had to rely on citations collected by human readers from limited text corpora. More rare words rather than common words are found even in standard dictionaries (Church and Hanks , 1990). This is the problem in making a lexical entry list in dictionary construction.
2. It is hard to decide where to segment a string into its component words. It is also hard to enumerate words from a text, though it is reported that the accuracy of recent word segmentation methods using a dictionary and heuristic methods is higher than 95% in case of Thai (Virach , 1993). The accuracy depends mostly on word entries in the dictionary, and the priority for selecting between candidate words when there is more than one solution for word segmentation. This is the problem in assigning priority information for selection.

3 Word Extraction from Text Corpora

We used a window size of 4 to 32 for n-gram data accumulation. The value is arbitrary but this range has proven sufficient to avoid collecting illegible strings.

3.1 Algorithm

Define that,

|a| is the number of clusters[1] in the string 'a',

n(a) is the number of occurrences of the string 'a', and

n(a+1) is the number of occurrences of the string 'a' with one additional cluster added.

As the length of a string increases the number of occurrences of that string will decrease. Therefore,

$$n(a+1) \leq n(a). \tag{1}$$

For the string 'a', n(a+1) decreases significantly from n(a) when 'a' is a frequently used string in contrast to 'a+1'. From this, it can be seen that 'a' is a rigid expression of an open compound when it satisfies the condition

$$n(a+1) \ll n(a). \tag{2}$$

In such a case, 'a' is considered a rigid expression that is used frequently in the text, and 'a+1' is just a string that occurs in limited contexts.

[1]The smallest stand-alone character unit as by the spelling rules.

Since we count the occurrence of strings generated from an arbitrary position in the text, with only the above observation, only the right end position of a string can be assumed to determined a rigid expression. To identify the correct starting position of a string, we apply the same observation to the leftward extension of a string. Therefore, we have to include the direction to the string observation.

Further define that,

+a is the right observation of the string 'a', and

-a is the left observation of the string 'a'. Then,

n(+a+1) is the number of occurrences of the string 'a' with one cluster added to its right, and

n(-a+1) is the number of occurrences of the string 'a' with one cluster added to its left.

Following the same reasoning as above, we will obtain,

$$n(+a+1) \leq n(a), and \tag{3}$$

$$n(-a+1) \leq n(a). \tag{4}$$

A string 'a' is a rigid expression if it satisfies the following conditions,

$$n(+a+1) \ll n(a), and \tag{5}$$

$$n(-a+1) \ll n(a). \tag{6}$$

3.2 Data preparation

Following are the steps for creating n-gram text data according to the fundamental features of Thai text corpora. The results are shown in Table 1 and Table 2. In each table, "n" is the number of occurrences and "d" is the difference in occurrence with the next string.

1. Tokenize the text at locations of spaces, tabs and newline characters.
2. Produce n-gram strings following Thai spelling rules. Only strings that have possible boundaries are generated, and their occurrence counted. For example, shifting a string from 'a+6' to 'a+7' in the Table 1, the string at 'a+7' is 'กระทรวงการคลัง' and not 'กระทรวงการคล' despite the first character after 'a+6' being 'ล'. According to the Thai spelling rules, the character 'ั' can never stand by itself. It needs both of an initial consonant and a final consonant. We call this three character unit a cluster.
3. Create both rightward (Table 1) and leftward (Table 2) sorted strings. The frequency of each string is the same but the strings are lexically reversed and ordered based on this reversed state.

4. Calculate the difference between the occurrence of adjoining strings in the sorted lists. Let d(a) be the difference value of the string ‘a’, then

$$d(a) = n(a) - n(a+1). \qquad (7)$$

   The difference value (d) is generated separately for the rightward and leftward sorted string tables.

The occurrences (n) in both Table 1 and Table 2 apparently support the conditions (3) and (4).

String	Rightward sorted string	n	d
a	กระท	513	68
a+1	กระทร	445	0
a+2	กระทรว	445	0
a+3	กระทรวง	445	142
a+4	กระทรวงกา	303	0
a+5	กระทรวงการ	303	22
a+6	กระทรวงการค	281	0
a+7	กระทรวงการคลัง	281	274
a+8	กระทรวงการคลังกำ	7	0

Table 1: Example of a Rightward Sorted String Table

String	Leftward sorted string	n	d
-b	การกระทรวง	172	0
-b+1	ว่าการกระทรวง	172	0
-b+2	รีว่าการกระทรวง	172	42
-b+3	ตรีว่าการกระทรวง	130	9
-b+4	นตรีว่าการกระทรวง	121	0
-b+5	มนตรีว่าการกระทรวง	121	7
-b+6	รัฐมนตรีว่าการกระทรวง	114	107
-b+7	งรัฐมนตรีว่าการกระทรวง	7	0

Table 2: Example of a Leftward Sorted String Table

3.3 Extraction

3.3.1 Competitive selection

According to condition (5) the string ‘a’ (กระท) in Table 3 is considered an open compound because the difference of between n(a) and n(a+1) is as high as 450. However, 'กระท' is an illegible string and cannot be used on as individual basis in general text. Observing the same string ‘a’ in Table 1, the difference between n(a) and n(a+1) is only 68. It is not comparably high enough to be selected. Therefore, we have to determine the minimum value of the difference when there is more than one branch extended from a string.

String	Rightward sorted string	n	d
a	กระท	513	450
a+1	กระทบ	63	22
a+2	กระทบก	41	0
a+3	กระทบกระ	41	0
a+4	กระทบกระเทือ	41	0
a+5	กระทบกระเทือน	41	25
a+6	กระทบกระเทือนต่อ	16	11
a+7	กระทบกระเทือนต่อกา	5	0

Table 3: A Further Example of the Count of a Rightward Sorted String Table

3.3.2 Unified selection

In Figure 1, we obtain the string 'รกระทรวงการคลัง' by observing the significant change in d just before the next string 'รกระทรวงการคลังรัก'. The string could be wrongly selected if we do not observe its behaviour in the leftward sorted string table, to determine the correct left boundary. Thus, we observe the count of string 'รกระทรวงการคลัง' when it is extended leftward, as shown in Figure 2.

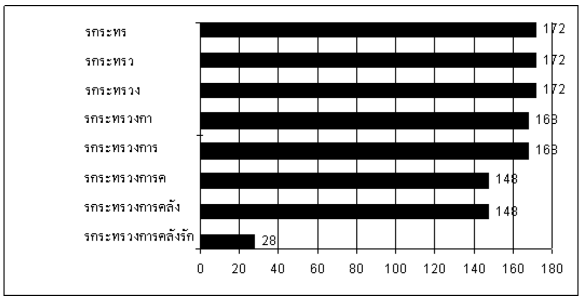

Figure 1: Rightward Sorted Strings Starting from an Arbitrary String

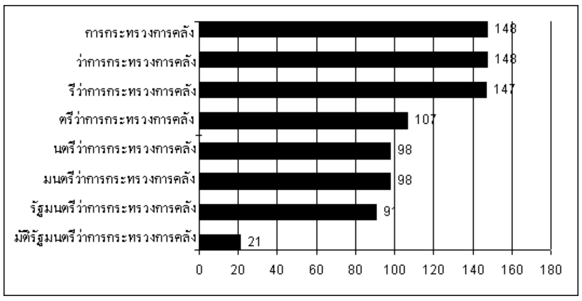

Figure 2: Leftward Sorted Strings Starting from an Arbitrary String

By unifying the results of both methods of the observation, we finally obtain the word

'รัฐมนตรีว่าการกระทรวงการคลัง'.

4 Experimental Results

We have applied our method to an actual Thai text corpora without word segmentation preprocessing.

4.1 Natural language data

We selected 'Thai Revenue Code (1995)', as large as 705,513 bytes, and 'Convention for Avoidance of Double Taxation between Thailand and Japan', which has a smaller size of 40,401 bytes. The purpose is to show that our method is effective regardless of the size of the data file.

4.2 Results of extraction

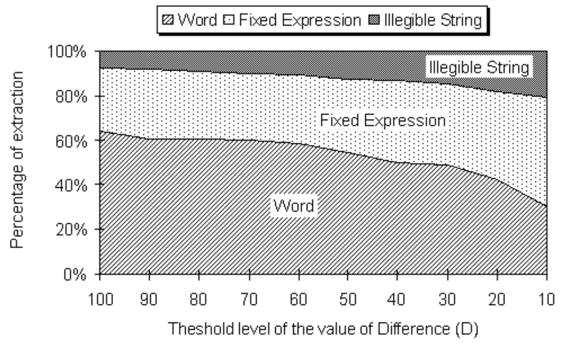

Figure 3: Result of Extraction of 'Thai Revenue Code (705,513 bytes)'

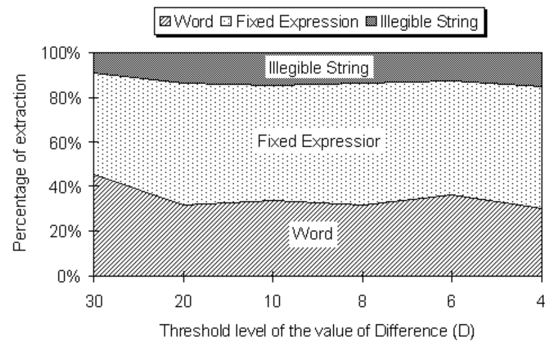

Figure 4: Result of Extraction of 'Convention for Thailand-Japan (40,401 bytes)'

The results of extraction examined in both large and small file sizes are very satisfactory. Very few illegible strings are extracted though the threshold of the difference value is set to be as low as 10 in Figure 3, and 4 in Figure 4. The suitable value of the threshold of difference varies with the size of text corpus file. To obtain more meaningful strings from a large file, we have to set a relatively high threshold of extraction. One of the advantages of our method is that there is an inherent trade-off between the quantity and the quality of the extracted strings. In the case of Figure 3, to limit the amount of illegible strings to not exceed 15% of the total extracted strings, we set the threshold to 30. As a result, we obtained 154 words, 114 fixed expressions and only 46 illegible strings. Furthermore, we found that of the 154 words appearing in the text, there were 84 words that were not found in a standard Thai dictionary.

5 Conclusion

This paper has shown an algorithm for data preparation and open compound extraction. The *competitive selection* and *unified selection* of rightward and leftward sorted strings play an important role in improving accuracy of the extraction. In the experiment, we applied Thai spelling rules to restrict the search path for string counts. Some types of spelling irregularities can be excluded by this process. By adjusting the value of threshold, we can extract suitable entries for open compound registration regardless of the size of the input file. Furthermore, our method has ensured the extraction of new words from the text file of the language that has no explicit word boundary, such as Thai.

References

Church, K. W. and Hanks, P. 1990. Word Association Norms, Mutual Information, and Lexicography. *Computational Linguistics*, Vol.16, No.1, pages 22-29.

Ikehara, S., Shirai S. and Kawaoka, T. 1995. Automatic Extraction of Uninterrupted Collocations by n-gram Statistics. *Proceedings of The first Annual Meeting of The Association for Natural Language Processing*, pages 313-316 (in Japanese).

Nagao, M. and Mori, S. 1994. A New Method of N-gram Statistics for Large Number of n and Automatic Extraction of Words and Phrases from Large Text Data of Japanese. *Proceedings of COLING 94*, Vol.1, pages 611-615.

Smadja, F. A. and McKeown, K. R. 1990. Automatically Extracting and Representing Collocations for Language Generation. *Proceedings of ACL-90*, pages 252-259.

Sornlertlamvanich, Virach. 1993. Word Segmentation for Thai in Machine Translation System. *Machine Translation*, National Electronics and Computer Technology Center, (in Thai).