Towards Building a Corpus-based Dictionary for Non-word-boundary Languages

Tanapong Potipiti, Virach Sornlertlamvanich, and Thatsanee Charoenporn

National Electronics and Computer Technology Center, Thailand

Abstract

Corpus-based lexicography is an effective task for building a dictionary for languages, which exhibit explicit word boundaries. However, for non-word-boundary languages such as Japanese, Chinese and Thai, it is an arduous job. Because in these languages, there are no clear criteria what words are, the most difficult task for building a corpus-based dictionary for these languages is the process of selecting word list or lexicon entries. We propose a practical solution for this task by applying the c4.5 learning algorithm for building the lexicon list. Applying our algorithm with Thai corpora, the experiment yields promising results about 85% in both training and test corpus.

1 Introduction

For all classical dictionary compiling, the formidable task is concerning with a notion of "word". For all inflectional languages, how to determine that inflectional and derivational variants are one unit or several is problem. The isolating language, on the other hand, one faces other different problem. How to define word for the isolating language such as Thai, which is non-word-breaking language, is the dominant problem. Since there is, at present, no clear principle to define "word". Compound words are always set to discuss.

Presently, in the age of information, a multitude of data and knowledge is created everyday. Inevitably, new terms are employed to convey new concepts. How to collect all new concepts or word lists is then the subsequent problem. The traditional dictionary can not certainly cover all new words. The corpus-based dictionary is proposed as a promising method. It is quite an effective method to build a corpus-based dictionary for the languages that exhibit word boundaries such as English. However, for the non-word-breaking languages, it is quite a demanding task to create a corpus-based dictionary.

Basically, picking up a lexicon list is the first step. The fist step of creating a dictionary is to pick a lexicon list. Selecting a word list is an easy and automatic task for the languages that have explicit word boundary. However, for non-word-boundary languages, this may be the most difficult job. We propose a new module for compiling "lexicon" for Thai dictionary, aiming to extract "lexicon" from corpus by stochastic weigh. The lexicon list extracted from the corpus, then, will be pass through the word segmentation module to retrieve only the correct "word". The next step of the automatically word list producing is to define word definition. By using a very large corpus, expected word list with their context in sentences is retrieved. And by these plenty of sentences, words in the same conceptual meaning will be manually grouped to be each definition.

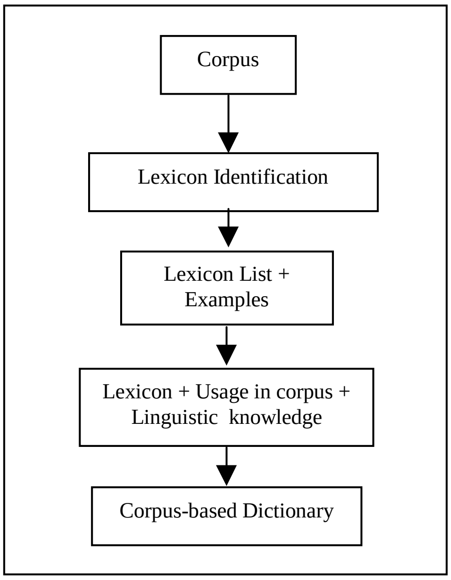

Fig. 1: How to Build a Corpus-based Dictionary

In this paper, we introduce an automatic algorithm for corpus-based lexicon extraction for the non-word-boundary languages. Applying our method for Thai corpora, we have got an impressive result with a high accuracy.

2 Related Works

Related literatures on lexicon extraction for non-word-boundary languages can be found in researches on Japanese and Thai languages. Nagao et al. (1994) has provided an effective method to construct a sorted file that facilitates the calculation of n-gram data for lexicon extraction. But their algorithm did not yield satisfactory accuracy; there were many invalid word extracted. The following work (Ikehara et al., 1995) improved the sorted file to avoid repeating in counting strings. The extraction result was better, but the determination of the longest strings is always made consecutively from left to right. If an erroneous string is extracted, its errors will propagate through the rest of the input strings. Sornlertlamvanich and Tanaka (1996) employed the frequency of the sorted character n-grams to extract Thai open compounds; the strings that experienced a significant change of occurrences when their lengths are extended. This algorithm reports about 90% accuracy of Thai open compound extraction. However, the algorithm emphasizes on open compound extraction and has to limit the range of n-gram to 4-20 grams for the computational reason. This causes limitation in the size of corpora and efficiency in the extraction.

3 Our Approach

3.1 The C4.5 Learning Algorithm

Decision tree induction algorithms have been successfully applied for NLP problems such as sentence boundary disambiguation (Palmer et al. 1997), parsing (Magerman 1995) and word segmentation (Meknavin et al. 1997). We employ the c4.5 (Quinlan 1993) decision tree induction program as the learning algorithm for lexicon extraction.

3.2 Attributes

We treat the lexicon extraction problem for non-word-boundary languages as the problem of word/non-word string disambiguation. The next step is to identify the attributes that are able to disambiguate word strings from non-word strings. The attributes used for the learning algorithm are as follows.

3.2.1 *Left Mutual Information and Right Mutual Information*

Mutual information (Church et al. 1991) of random variable *a* and *b* is the ratio of probability that *a* and *b* co-occur, to the independent probability that *a* and *b* co-occur. High mutual information indicates that *a* and *b* co-occur more than expected by chance. Our algorithm employs left and right mutual information as attributes in word extraction procedure. The left mutual information (*Lm*), and right mutual information (*Rm*) of string $x\boldsymbol{y}z$ are defined as:

$$Lm(x\boldsymbol{y}z) = \frac{p(x\boldsymbol{y}z)}{p(x)\,p(\boldsymbol{y}z)},$$

$$Rm(x\boldsymbol{y}z) = \frac{p(x\boldsymbol{y}z)}{p(x\boldsymbol{y})\,p(z)},$$

where

x is the rightmost character of $x\boldsymbol{y}z$

$\boldsymbol{y}$is the middle substring of $x\boldsymbol{y}z$

z is the leftmost character of $x\boldsymbol{y}z$

$p(\ )$ is the probability function.

If $x\boldsymbol{y}z$ is a word, both $Lm(x\boldsymbol{y}z)$ and $Rm(x\boldsymbol{y}z)$ should be high. On the contrary, if $x\boldsymbol{y}z$ is a non-word string but consists of words and characters, either of its left or right mutual information, or both must be low. For example, Thai string “กปรากฏ” which consists of a Thai character ‘ก’ and a Thai word “ปรากฏ” must have low left mutual information.

3.2.2 *Left Entropy and Right Entropy*

Entropy (Shannon 1948) is the information measuring disorder of variables. The left and right entropy is exploited as another two attributes in our word extraction. Left entropy (*Le*), and right entropy (*Re*) of string $\boldsymbol{y}$ are defined as:

$$Le(\boldsymbol{y}) = -\sum_{\forall x \in \boldsymbol{A}} p(x\boldsymbol{y} \mid \boldsymbol{y}) \cdot log_2\, p(x\boldsymbol{y} \mid \boldsymbol{y})$$

$$Re(\boldsymbol{y}) = -\sum_{\forall z \in \boldsymbol{A}} p(\boldsymbol{y}z \mid \boldsymbol{y}) \cdot log_2\, p(\boldsymbol{y}z \mid \boldsymbol{y})$$

where

$\boldsymbol{y}$ is the considered string,

$\boldsymbol{A}$ is the set of all alphabets

x, z is any alphabets in $\boldsymbol{A}$.

If $\boldsymbol{y}$ is a word, the alphabets that come before and after $\boldsymbol{y}$ should have varieties or high entropy. If $\boldsymbol{y}$ is not a complete word, either of its left or right entropy, or both must be low. For example, Thai string “ปราก” is not a word but a substring of word “ปรากฏ”. Thus the choices of the right adjacent alphabets to “ปราก” must be few and the right entropy of “ปราก”, when the right adjacent alphabet is “ฏ”, must be low.

3.2.3 Frequency

It is obvious that the iterative occurrences of words must be higher than those of non-word strings.

String frequency is also useful information for our task. Because the string frequency depends on the size of corpus, we normalize the count of occurrences by dividing by the size of corpus and multiplying by the average value of word length:

$$F(\boldsymbol{s}) = \frac{N(\boldsymbol{s})}{Sc}.Avl$$

where

$\boldsymbol{s}$ is the considered string
$N(\boldsymbol{s})$ is the number of the occurrences of $\boldsymbol{s}$ in corpus
Sc is the size of corpus
Avl is the average word length.

We employed the frequency value as another attribute for the c4.5 learning algorithm.

3.2.4 Length

Short strings are more likely to happen by chance than long strings. Then, short and long strings should be treated differently in the disambiguation process. Therefore, string length is also used as an attribute for this task.

3.2.5 Functional Words

Functional words are frequently used in texts of non-word-boundary language. These functional words are used often enough to mislead the occurrences of string patterns. To filter out these noisy patterns from word extraction process, discrete attribute *Func*(***s***):

Func(***s***) = 1 if string ***s*** contains
functional words,
= 0 if otherwise,

is applied.

3.2.6 First Two and Last Two Characters

A very useful process for our disambiguation is to check whether the considered string complies with the spelling rules of that language or not. We employ the words in the dictionary as spelling examples for the first and last two characters. Then we define attributes $Fc(\boldsymbol{s})$ and $Lc(\boldsymbol{s})$ for this task as follows.

$$Fc(\boldsymbol{s}) = \frac{N(s_1 s_2 *)}{ND}$$

$$Lc(\boldsymbol{s}) = \frac{N(*s_{n-1} s_n)}{ND}$$

where $\boldsymbol{s}$ is the considered string and $\boldsymbol{s} = s_1 s_2 \ldots s_{n-1} s_n$
$N(s_1 s_2 *)$ is the number of words in the dictionary that begin with $s_1 s_2$
$N(*s_{n-1} s_n)$ is the number of words in the dictionary that end with $s_{n-1} s_n$
ND is the number of words in the dictionary.

3.3 Applying C4.5 to Thai Lexicon Extraction

We have applied our approach explained above to Thai corpora. The process of applying c4.5 to our lexicon extraction problem is shown in Figure 1. Firstly, we construct a training set for the c4.5 learning algorithm. We apply Yamamoto et al. (1998)'s algorithm to extract all strings from a 1-MB plain Thai corpus. For practical and reasonable purpose, we select only the 2-to-30-character strings that occur more than 2 times, have positive right and left entropy, and conform to the simple Thai spelling rules. To this step, we get about 30,000 strings. These strings are manually tagged as words or non-word strings. The string attributes explained above are calculated for each string. Then the string attributes and tags are used as the training example for the learning algorithm. The decision tree is then constructed from the training data.

The decision tree we have got is applied for word extraction from the other plain 1-MB corpus (the test corpus). The experimental results will be discussed in the next section.

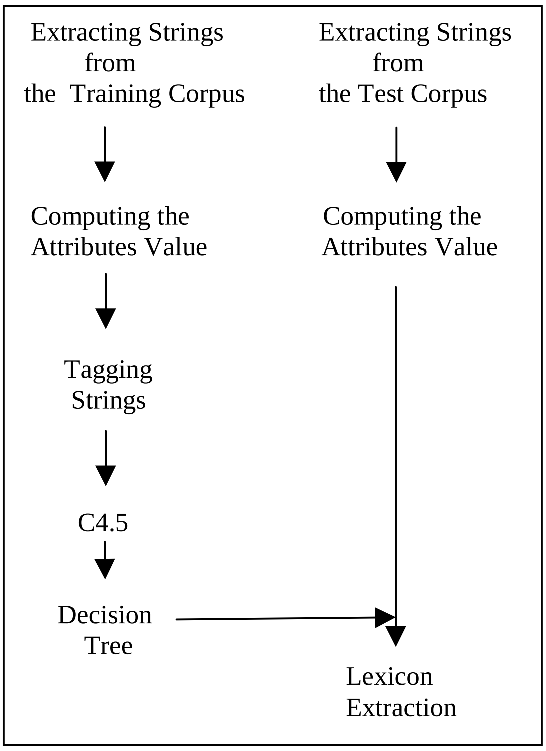
Extracting Strings from the Training Corpus
Extracting Strings from the Test Corpus
Computing the Attributes Value
Computing the Attributes Value
Tagging Strings
C4.5
Decision Tree
Lexicon Extraction

Figure 1: Overview of the Process

4 Experimental Results

4.1 The Results

To measure the accuracy of the algorithm, we consider two statistical values: precision and recall. The precision of our algorithm is 87.3% for the training set and 84.1% for the test set. The recall of extraction is 56% in both training and test sets.

Table 1: The precision of lexicon extraction

	No. of strings extracted by the decision tree	No. of words extracted	No. of non-word strings extracted
Training Set	1882 (100%)	1643 (87.3%)	239 (12.7%)
Test Set	1815 (100%)	1526 (84.1%)	289 (15.9%)

Table 2: The recall of lexicon extraction

	No. of words that has more than 2 occurrences in corpus	No. of words extracted by the decision tree	No. of words in corpus that are found RID
Training Set	2933 (100%)	1643 (56.0%)	1833 (62.5%)
Test Set	2720 (100%)	1526 (56.1%)	1580 (58.1%)

5 Conclusion

In this paper, we have applied the c4.5 learning algorithm for the automatic task of lexicon extraction for non-word-boundary languages. C4.5 can construct a good decision tree for word/non-word disambiguation. The learned attributes, which are mutual information, entropy, word frequency, word length, functional words, first two and last two characters, can capture useful information for lexicon extraction. Applying to Thai corpora, our approach yields about 85% and 56% in precision and recall measures respectively. Our future work is to apply this algorithm with larger corpora to build a corpus-based Thai dictionary.

References

Church, K.W., Robert L. and Mark L.Y. (1991) A Status Report on ACL/DCL. *Proceedings of 7th Annual Conference of the UW Centre New OED and Text Research: Using Corpora*, pp. 84-91

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

Magerman, D.M. (1995) Statistical decision-tree models for parsing. *Proceeding of 33rd Annual Meeting of Association for Computational Linguistics*

Meknavin, S., Charoenpornsawat, P. and Kijsirikul, B. (1997) Feature-based Thai Word Segmentation. *Proceeding of the Natural Language Processing Pacific Rim Symposium 1997*, pp. 35-46

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. *Proceeding of COLING 94*, Vol. 1, pp. 611-15

Palmer, D.D. and Hearst M.A. (1997) Adaptive Multilingual Sentence Boundary Disambiguation. *Computational Linguistics Vol*. 27, pp. 241-267

Quinlan, J.R. (1993) *C4.5 Programs for Machine Learning.* Morgan Publishers San Mated, California, 302 p.

Shannon, C.E. (1948) A Mathematical Theory of Communication. *Bell System Technical Journal* 27, pp. 379-423

Sornlertlamvanich, V. and Tanaka, H. (1996) The Automatic Extraction of Open Compounds from Text. *Proceeding of COLING 96 Vol. 2,* pp. 1143-1146

Yamamoto, M. and Church, K.W. (1998) Using Suffix Arrays to Compare Term Frequency and Document Frequency for All Substrings in Corpus. *Proceeding of Sixth Workshop on Very Large Corpora* pp. 27-37