Automatic Corpus-Based Thai Word Extraction with the C4.5 Learning Algorithm

VIRACH SORNLERTLAMVANICH, TANAPONG POTIPITI AND THATSANEE CHAROENPORN

National Electronics and Computer Technology Center,
National Science and Technology Development Agency,
Ministry of Science and Technology Environment,
22nd Floor Gypsum Metropolitan Tower 539/2 Sriayudhya Rd. Rajthevi Bangkok 10400 Thailand
Email: virach@nectec.or.th, tanapong@nectec.or.th, thatsanee@nectec.or.th

Abstract

"Word" is difficult to define in the languages that do not exhibit explicit word boundary, such as Thai. Traditional methods on defining words for this kind of languages have to depend on human judgement which bases on unclear criteria or procedures, and have several limitations. This paper proposes an algorithm for word extraction from Thai texts without borrowing a hand from word segmentation. We employ the c4.5 learning algorithm for this task. Several attributes such as string length, frequency, mutual information and entropy are chosen for word/non-word determination. Our experiment yields high precision results about 85% in both training and test corpus.

1 Introduction

In the Thai language, there is no explicit word boundary; this causes a lot of problems in Thai language processing including word segmentation, information retrieval, machine translation, and so on. Unless there is regularity in defining word entries, Thai language processing will never be effectively done. The existing Thai language processing tasks mostly rely on the hand-coded dictionaries to acquire the information about words. These manually created dictionaries have a lot of drawbacks. First, it cannot deal with words that are not registered in the dictionaries. Second, because these dictionaries are manually created, they will never cover all words that occur in real corpora. This paper, therefore, proposes an automatic word-extraction algorithm, which hopefully can overcome this Thai language-processing barrier.

An essential and non-trivial task for the languages that exhibit inexplicit word boundary such as Thai, Japanese, and many other Asian languages undoubtedly is the task in identifying word boundary. "Word", generally, means a unit of expression which has universal intuitive recognition by native speakers. Linguistically, word can be considered as the most stable unit which has little potential to rearrangement and is uninterrupted as well. "Uninterrupted" here attracts our lexical knowledge bases so much. There are a lot of uninterrupted sequences of words functioning as a single constituent of a sentence. These uninterrupted strings, of course are not the lexical entries in a dictionary, but each occurs in a very high frequency. The way to point out whether they are words or not is not distinguishable even by native speakers. Actually, it depends on individual judgement. For example, a Thai may consider 'ออกกำลังกาย' (exercise) a whole word, but another may consider 'ออกกำลังกาย' as a compound: 'ออก' (take) + 'กำลัง' (power) + 'กาย' (body). Computationally, it is also difficult to decide where to separate a string into words. Even though it is reported that the accuracy of recent word segmentation using a dictionary and some heuristic methods is in a high level. Currently, lexicographers can make use of large corpora and show the convincing results from the experiments over corpora. We, therefore, introduce here a new efficient method for consistently extracting and identifying a list of acceptable Thai words.

2 Previous Works

Reviewing the previous works on Thai word extraction, we found only the work of Sornlertlamvanich and Tanaka (1996). They 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.

The other works can be found in the research on the Japanese language. Nagao et al. (1994) has provided an effective method to construct a sorted file that facilitates the calculation of n-gram data. But their algorithm did not yield satisfactory accuracy; there were many invalid substrings 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.

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 word extraction.

The induction algorithm proceeds by evaluating content of a series of attributes and iteratively building a tree from the attribute values with the leaves of the decision tree being the value of the goal attribute. At each step of learning procedure, the evolving tree is branched on the attribute that partitions the data items with the highest information gain. Branches will be added until all items in the training set are classified. To reduce the effect of overfitting, c4.5 prunes the entire decision tree constructed. It recursively examines each subtree to determine whether replacing it with a leaf or branch would reduce expected error rate. This pruning makes the decision tree better in dealing with the data different from the training data.

3.2 Attributes

We treat the word extraction problem 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**y**z* are defined as:

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

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

where

x is the leftmost character of *x**y**z*
y is the middle substring of *x**y**z*
z is the rightmost character of *x**y**z*
p( ) is the probability function.

If *x**y**z* is a word, both *Lm*(*x**y**z*) and *Rm*(*x**y**z*) should be high. On the contrary, if *x**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, 'กปรากฏ' ('ก'(a Thai alphabet) + 'ปรากฏ'(The word means appear in Thai.)) 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 ***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

y is the considered string,
A is the set of all alphabets
x, z is any alphabets in ***A***.

If ***y*** is a word, the alphabets that come before and after ***y*** should have varieties or high entropy. If ***y*** is not a complete word, either of its left or right entropy, or both must be low. For example, 'ปราก' is not a word but a substring of word 'ปรากฏ' (appear). 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 Thai word length:

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

where

s is the considered string
N(***s***) is the number of the occurrences of ***s*** in corpus
Sc is the size of corpus
Avl is the average Thai 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 such as 'จะ' (will) and 'ก็' (then) are frequently used in Thai texts. 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 Thai spelling rules or not. We employ the words in the Thai Royal Institute 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 ***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 Word Extraction

The process of applying c4.5 to our word 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 plain and unlabelled 1-MB corpus which consists of 75 articles from various fields. For practical and reasonable purpose, we select only the 2-to-30-character strings that occur more than 2 times,

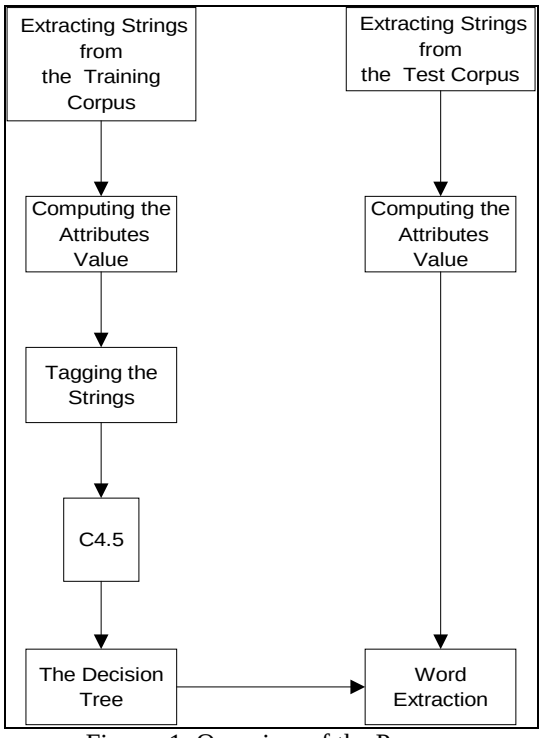

Figure. 1: Overview of the Process

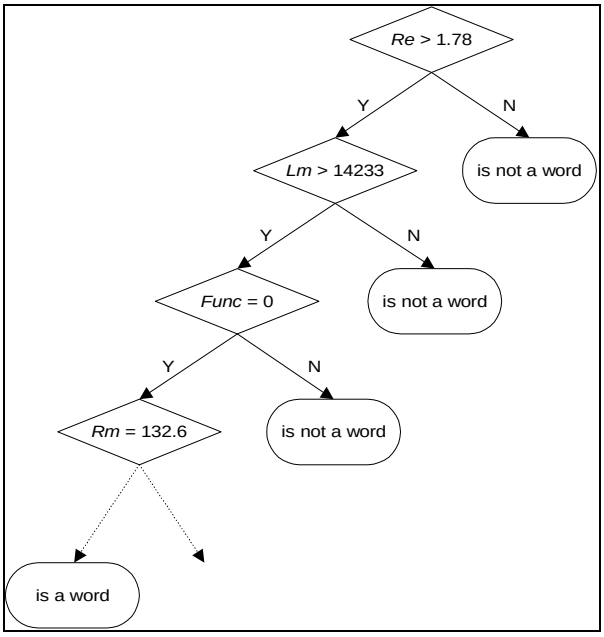

Figure 2: Example of the Decision tree

have positive right and left entropy, and conform to 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 strings' statistics explained above are calculated for each string. Then the strings' attributes and tags are used as the training example for the learning algorithm. The decision tree is then constructed from the training data.

In order to test the decision tree, another plain 1-MB corpus (the test corpus), which consists of 72 articles from various fields, is employed. All strings in the test corpus are extracted and filtered out by the same process as used in the training set. After the filtering process, we get about 30,000 strings to be tested. These 30,000 strings are manually tagged in order that the precision and recall of the decision tree can be evaluated. The experimental results will be discussed in the next section.

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. We compare the recall of our word extraction with the recall from using the Thai Royal Institute dictionary (RID). The recall from our approach and from using RID are comparable and our approach should outperform the existing dictionary for larger corpora. Both precision and recall from training and test sets are quite close. This indicates that the created decision tree is robust for unseen data. Table 3 also shows that more than 30% of the extracted words are not found in RID. These would be the new entries for the dictionary.

Table 1: The precision of word 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 word extraction

	No. of words that in 30,000 strings extracted	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%)

Table 3: Words extracted by the decision tree and RID

	No. of words extracted by the decision tree	No. of words extracted by the decision tree which is in RID	No. of words extracted by the decision tree which is not in RID
Training Set	1643 (100.0%)	1082 (65.9%)	561 (34.1%)
Test Set	1526 (100.1%)	1046 (68.5%)	480 (31.5%)

4.2 The Relationship of Accuracy, Occurrence and Length

In this section, we consider the relationship of the extraction accuracy to the string lengths and occurrences. Figure 2 and 3 depict that both precision and recall have tendency to increase as string occurrences are getting higher. This implies that the accuracy should be higher for larger corpora. Similarly, in Figure 4 and 5, the accuracy tends to be higher in longer strings. The new created words or loan words have tendency to be long. Our extraction, then, give a high accuracy and very useful for extracting these new created words.

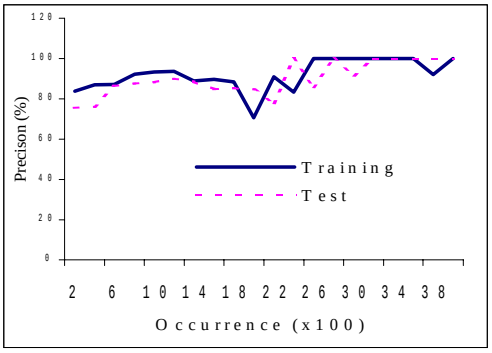

Figure 3: Precision-Occurrence Relationship

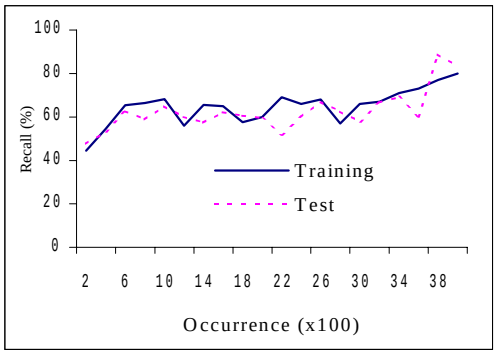

Figure 4: Recall-Occurrence Relationship

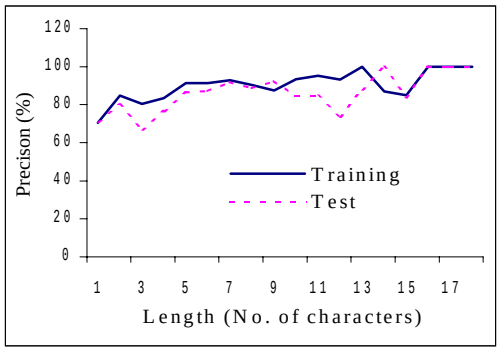

Figure 5: Precision-Length Relationship

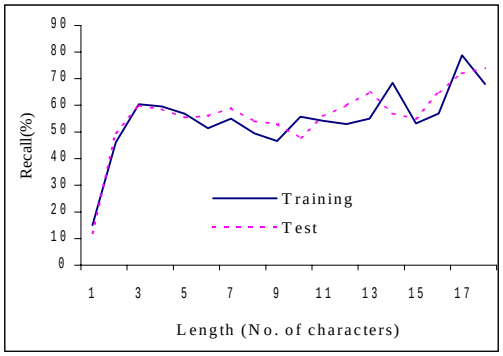

Figure 6: Precision-Length Relationship

5 Conclusion

In this paper, we have applied the c4.5 learning algorithm for the task of Thai word extraction. 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 word extraction. Our approach yields about 85% and 56% in precision and recall measures respectively, which is comparable to employing an existing dictionary. The accuracy should be higher in larger corpora. Our future work is to apply this algorithm with larger corpora to build a corpus-based Thai dictionary. And hopefully, our approach should be successful for other non-word-boundary languages.

Acknowledgement

Special thanks to Assistant Professor Mikio Yamamoto for providing the useful program to extract all substrings from the corpora in linear time.

References

Church, K.W., Robert L. and Mark L.Y. (1991) A Status Report on ACL/DCL. *Proceedings of the 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. *Proceedings 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., *Proceedings of the 33rd Annual Meeting of Association for Computational Linguistics.*

Meknavin, S., Charoenpornsawat, P. and Kijsirikul, B. (1997) Feature-based Thai Word Segmentation. *Proceedings 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. *Proceedings 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. *Proceedings 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. *Proceedings of the Sixth Workshop on Very Large Corpora* pp. 27-37.