แบบจำลองผสมแบบแยกแยะสำหรับการแบ่งคำไทย

ฆนาศัย กรึงไกร[1,2] ชุลีรัตน์ จรัสกุลชัย[3] Jun'ichi Kazama[2]

[1]Graduate School of Engineering, Kobe University
1-1 Rokkodai-cho, Nada-ku, Kobe 657-8501 Japan
[2]National Institute of Information and Communications Technology
3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0289 Japan
[3]Depatment of Computer Science, Faculty of Sciences
Kasetsart University, Bangkok, Thailand
canasai@nict.go.jp, fscichj@ku.ac.th, kazama@nict.go.jp

บทคัดย่อ

การแบ่งคำ เป็นงานพื้นฐานในการประมวลผลภาษาไทยด้วยคอมพิวเตอร์ ความกำกวมของขอบเขตคำและคำที่ระบบไม่รู้จัก (ไม่ปรากฏในพจนานุกรมของระบบ) เป็นสาเหตุหลักสำคัญที่ทำให้การแบ่งคำไทยยาก ในงานวิจัยนี้ เรานำเสนอแบบจำลองผสมแบบแยกแยะ ซึ่งแทน “เสิร์จเสปซ” ด้วย “แลททิส” ที่ประกอบไปด้วย “โหนด” ในระดับคำและระดับกลุ่มอักขระ และสามารถใช้ประโยชน์ของข้อมูลบน “โหนด” เหล่านี้ เพื่อจัดการคำที่ระบบรู้จักและไม่รู้จักตามลำดับ วิธีการของเราอยู่บนพื้นฐานของการเรียนรู้แบบ “ออนไลน์ลาร์จมาร์จิน” ที่เรียกว่า MIRA (Margin Infused Relaxed Algorithm) เราได้ทำการทดลองบนคลังข้อความของ BEST (ชุดที่ 1-5) เพื่อแสดงความเป็นไปได้และประสิทธิผลของวิธีที่นำเสนอ

คำสำคัญ: Thai Word Segmentation, Hybrid Model, Discriminative Online Learning, Margin Infused Relaxed Algorithm

A Discriminative Hybrid Model for Thai Word Segmentation

Canasai Kruengkrai[1,2] Chuleerat Jaruskulchai[3] Jun'ichi Kazama[2]

[1]Graduate School of Engineering, Kobe University
1-1 Rokkodai-cho, Nada-ku, Kobe 657-8501 Japan
[2]National Institute of Information and Communications Technology
3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0289 Japan
[3]Depatment of Computer Science, Faculty of Sciences
Kasetsart University, Bangkok, Thailand
canasai@nict.go.jp, fscichj@ku.ac.th, kazama@nict.go.jp

Abstract

Word segmentation is a fundamental task in natural language processing (NLP) for Thai. Word boundary ambiguities and unknown words are two sources of problems that make word segmentation difficult. In this paper, we propose a discriminative hybrid model that represents the search space with a lattice containing word-level and character-cluster-level nodes and exploits information on these nodes to handle known and unknown words, respectively. Our approach is essentially based on online large-margin learning called MIRA (Margin Infused Relaxed Algorithm). We conducted experiments on BEST corpus (based on data sets no. 1-5) to show the feasibility and effectiveness of the proposed approach.

Key words: Thai Word Segmentation, Hybrid Model, Discriminative Online Learning, Margin Infused Relaxed Algorithm

1 Introduction

Word segmentation is a fundamental task in natural language processing (NLP) for Thai. Other NLP tasks ranging from word alignment to machine translation rely on results from word segmentation. Research in automatic Thai word segmentation can be dated back to 1980's when the problem of word segmentation was considered as a simple process of syllable separation [28, 2, 25]. However, the higher-level NLP tasks often requires more coarse-grained components, *words*, as primitive units.

More recently, machine learning-based approaches have been applied to the Thai word segmentation problem, e.g., Markov models [12]; RIPPER and Winnow [20]; Decision Trees [26, 29]; naive Bayes, support vector machines, and conditional random fields [9]; and other unsupervised learning techniques [10, 1]. Another study that focuses on full morphological analysis, which consists of joint word segmentation and part-of-speech tagging, can be found in [15, 14]. Despite a long research path, most of previous studies in Thai word segmentation are difficult to fairly compare due to the lack of the standard test corpus and the agreement in the definition of Thai word formation.

To solve this problem, NECTEC (National Electronics and Computer Technology Center) has defined an official guideline of Thai word segmentation and launched a set of shared tasks called BEST (Benchmark for Enhancing the Standard of Thai language processing) [7]. For the BEST 2009 shared task, NECTEC has released a very large corpus containing around 4.4 million words of Thai written texts. This motivates us to design a learned model that can analyze the nature of Thai word formation.

1.1 Challenges

In order to make an accurate analyzer, there are several issues that have to be considered. Here, we characterize them as the following problems.

- Word Boundary Ambiguity: Given an input, unsegmented sentence, there are many possible ways to segment it, depending on the definition of word formation and the entire sentence meaning. By considering word segmentation as a search task, the goal is to search the most likely path out of all candidate paths in the search (or output) space. The obvious questions are: how can we represent such search space efficiently and what are the learning and decoding algorithms that can guarantee reasonable performance?

- Unknown Word Problem: Handling of unknown words is an important problem because they are difficult to identify and often decrease the system performance. Unknown words are defined as words that do not occur in the system's dictionary. In Thai, the problem becomes more complicated since unknown words can exists in three different forms: *explicit*, *mixed*, and *hidden* [13]. Previous studies typically performed unknown identification in a pipelined manner [3, 8]. However, if the search space did not contain the target unknown word at the beginning step or the system failed to include it in a set of candidate unknown words, it is unlikely to identify that unknown word in the later step.

1.2 Contributions

Our goal is to solve the above challenges based on a unified framework.

- We present a discriminative hybrid model for solving Thai word segmentation (Section 2). Our model represents the search space with a lattice containing word-level and character-cluster-level nodes and exploits information on these nodes to handle known and unknown words, respectively. In order to model the process of word segmentation, we introduce an online learning algorithm that combines MIRA (Margin Infused Relaxed Algorithm) [5] with an efficient dynamic programming search [21].

- We develop a practical system that can give satisfactory performance without much effort in parameter tuning.

- We provide empirical results to support our claims using the BEST 2009 shared task corpus (Section 3).

2 Approach

In this section, we briefly describe the theoretical background in our approach.

2.1 Problem formulation

In the process of word segmentation, the task is to predict a path of word hypotheses, $\boldsymbol{y} = (y_1, \ldots, y_{\#\boldsymbol{y}}) = (\langle w_1, t_1 \rangle, \ldots, \langle w_{\#\boldsymbol{y}}, t_{\#\boldsymbol{y}} \rangle)$, for a given character sequence $\boldsymbol{x} = (c_1, \ldots, c_{\#\boldsymbol{x}})$, where w is a word (or character), t is its corresponding tag, and a "#" symbol denotes the number of elements in each variable.

In order to scope the problem of Thai text processing to the morphological level and simplify modeling, we assume that $\boldsymbol{x}$ is an ill-formed sentence since the Thai writing system has no explicit sentence boundaries. We also note that the number of word hypotheses can vary according to a considered path. This is different from a typical sequence labeling task (e.g., part-of-speech (POS) tagging in English) in which the number of tokens is the same for all candidate paths since $\boldsymbol{x}$ is already segmented.

The goal of a learning algorithm is to learn a mapping from inputs (or unsegmented sentences) $\boldsymbol{x} \in \mathcal{X}$ to outputs (or segmented paths) $\boldsymbol{y} \in \mathcal{Y}$ based on training samples of input-output pairs, $\mathcal{S} = \{\{(\boldsymbol{x}_t, \boldsymbol{y}_t)\}_{t=1}^{T} : (\boldsymbol{x}_t, \boldsymbol{y}_t) \in \mathcal{X} \times \mathcal{Y}, T \in \mathbb{N}\}$.

2.2 Search space representation

We model the search space with a lattice which is an ordered graph, efficiently representing rich information of word hypothesis sequences. We denote by $\mathcal{Y}_t = \{\boldsymbol{y}_t^1, \ldots, \boldsymbol{y}_t^K\}$ a lattice consisting of candidate paths for a given sentence $\boldsymbol{x}_t$. Each node in a path $\boldsymbol{y}_t$ corresponds to a word hypothesis that can contain any morphological information.

2.2.1 The original hybrid model

The choice of search space representation, which is related to the lattice shape, greatly affects how unknown words are processed. There are three basic approaches: word-based [17, 32], character-based [30, 24], and hybrid [22, 23]. While the word-based model has difficulties in handling unknown words, the character-based model gives high accuracy for unknown words but lower accuracy for known words. The hybrid model integrates these two approaches to compensate for each other's weaknesses. Experiments on morphological analysis have proven the effectiveness of the hybrid model [23]. Therefore, we apply the hybrid model to our framework.

In the original hybrid model [22], given an input sentence, word-level nodes are generated by using a system's dictionary, and then character-level nodes which corresponds to all characters in the sentence are generated. The system's dictionary is constructed from words that only occur in the training set. Since the BEST corpus does not provide POS tags, all word-level nodes are assigned with one tag, `W`.

Character-level nodes have spacial tags called position-of-character (POC) tags which indicate word-internal positions of the characters. POC tags include {`B`, `I`, `E`, `S`}, indicating the beginning of a word, the middle of a word, the end of a word, and a single-character word, respectively. For Chinese and Japanese, one can split the sentence into characters according to their encoding schemes. For Thai, this task becomes more complicated because characters are divided into four main categories: consonants, vowels, tonal marks, and special symbols.

2.2.2 Applying the hybrid model for Thai

Unlike Chinese and Japanese where a character can have its own meaning, a Thai character is meaningless. The Thai writing system is alphabetic in which each word is composed of

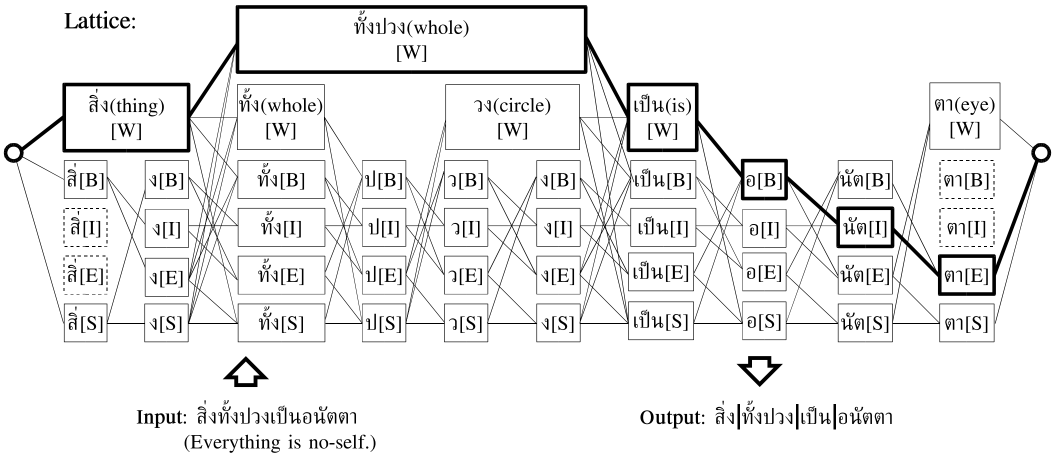

Figure 1: Example of a lattice in the hybrid model.

several characters. For example, the word 'เป็น' (is) consists of four characters: the vowel 'เ', the consonant 'ป', the diacritical mark '็', and the final consonant 'น'.

Representing a Thai character with a set of character-level nodes causes a very large search space which makes decoding inefficient. To alleviate this problem, we chunk Thai characters into a more coarse-grained unit called a *character cluster* by using a set of Thai writing rules [11]. In the above example, the word 'เป็น' can be grouped into a character cluster according to the writing rules. We can process each character cluster as a single character in Chinese and Japanese.

Figure 1 shows an example of a lattice for a Thai sentence 'สิ่งทั้งปวงเป็นอนัตตา' where the word 'อนัตตา' (no-self) is an unknown word. In the lattice, there can be several correct paths for an input sentence. The most probable path is selected according to a learned model, which is estimated from the training set. Here, the path indicated by the bold line obtains the highest score and is chosen as the best candidate path. We note that some nodes and state transitions are not allowed. For example, I and E nodes cannot occur at the beginning of the lattice (denoted by dashed boxes), or the transitions from I to B nodes are forbidden. These nodes and transitions are ignored during lattice processing.

With this lattice representation, we can consistently handle unknown words with character-cluster-level nodes. In other words, we use word-level nodes to identify known words and character-cluster-level nodes to identify unknown words. This is in contrast to work by Kruengkrai and Isahara [14] where candidate unknown word nodes are generated from all possible substrings in uncertainty ranges.

Algorithm 1 Generic Online Learning Algorithm

```
Input: Training set $\mathcal{S} = \{(\boldsymbol{x}_t, \boldsymbol{y}_t)\}_{t=1}^{T}$
Output: Weight vector $\mathbf{w}$
$\mathbf{w}^{(0)} = \mathbf{0}; \mathbf{v} = \mathbf{0}; i = 0$
for $iter = 1$ to $N$ do
    for $t = 1$ to $T$ do
        $\mathbf{w}^{(i+1)} =$ update $\mathbf{w}^{(i)}$ according to $(\boldsymbol{x}_t, \boldsymbol{y}_t)$
        $\mathbf{v} = \mathbf{v} + \mathbf{w}^{(i+1)}$
        $i = i + 1$
    end for
end for
$\mathbf{w} = \mathbf{v}/(N \times T)$
```

2.3 Discriminative online learning

2.3.1 Margin Infused Relaxed Algorithm

In the hybrid model, a lattice $\mathcal{Y}_t$ can contain more than one thousand nodes, depending on the length of a given sentence $\boldsymbol{x}_t$ and the number of POS tags in the corpus. Therefore, we require a learning algorithm that can handle large and complex lattice structures efficiently. Online learning is conceptually attractive for the hybrid model since it quickly converges within a few iterations [18]. Algorithm 1 outlines a generic online learning algorithm used in our framework.

In this paper, we focus on an online learning algorithm called MIRA (Margin Infused Relaxed Algorithm) [5], which has desired properties in terms of accuracy and scalability. In particular, we use a generalized version of MIRA [6, 18] that can incorporate k-best decoding in the update procedure. MIRA combines the advantages of margin-based learning and perceptron-style learning with an optimization scheme. To understand the concept of k-best MIRA, we begin with a linear score function:

$$s(\boldsymbol{x}, \boldsymbol{y}; \mathbf{w}) = \langle \mathbf{w}, \mathbf{f}(\boldsymbol{x}, \boldsymbol{y}) \rangle \ , \tag{1}$$

where $\mathbf{w}$ is a weight vector, and $\mathbf{f}$ is a feature representation of an input $\boldsymbol{x}$ and an output $\boldsymbol{y}$. In this paper, we use a simple feature set as described in [16].

Learning a mapping between an input-output pair corresponds to finding a weight vector $\mathbf{w}$ such that the best scoring path of a given sentence is the same as (or close to) the correct path. Given a training example $(\boldsymbol{x}_t, \boldsymbol{y}_t)$, MIRA tries to establish a margin between the score of the correct path $s(\boldsymbol{x}_t, \boldsymbol{y}_t; \mathbf{w})$ and the score of the best candidate path $s(\boldsymbol{x}_t, \hat{\boldsymbol{y}}; \mathbf{w})$ according to the current weight vector $\mathbf{w}$ that is proportional to a loss function $L(\boldsymbol{y}_t, \hat{\boldsymbol{y}})$. In each iteration, MIRA updates the weight vector $\mathbf{w}$ by keeping the norm of the change in the weight vector as small as possible. With this framework, we can formulate the optimization problem as follows [18]:

$$\mathbf{w}^{(i+1)} = \mathrm{argmin}_{\mathbf{w}} \|\mathbf{w} - \mathbf{w}^{(i)}\| \tag{2}$$
$$\text{s.t. } \forall \hat{\boldsymbol{y}} \in \mathrm{best}_k(\boldsymbol{x}_t; \mathbf{w}^{(i)}) : s(\boldsymbol{x}_t, \boldsymbol{y}_t; \mathbf{w}) - s(\boldsymbol{x}_t, \hat{\boldsymbol{y}}; \mathbf{w}) \geq L(\boldsymbol{y}_t, \hat{\boldsymbol{y}}) \ ,$$

where $\mathrm{best}_k(\boldsymbol{x}_t; \mathbf{w}^{(i)}) \in \mathcal{Y}_t$ represents a set of top k best paths given the weight vector $\mathbf{w}^{(i)}$. The above quadratic programming (QP) problem can be solved by using Hildreth's algorithm [31]. Replacing Equation (2) into line 4 of Algorithm 1, we obtain the k-best MIRA.

2.3.2 Loss function

In a typical sequence labeling problem, one can use a Hamming loss to measure errors of a predicted path with respect to the correct path. Since, in our case, the number of tokens can vary according to a path, we instead estimate the loss function through false positives (FP) and false negatives (FN). Here, FP means the number of output nodes that are not in the correct path, and FN means the number of nodes in the correct path that cannot be recognized by the system. We define the loss function by:

$$L(\boldsymbol{y}_t, \hat{\boldsymbol{y}}) = FP + FN \ . \tag{3}$$

2.3.3 Decoding algorithm

The next question is how to efficiently generate $\mathrm{best}_k(\boldsymbol{x}_t; \mathbf{w}^{(i)})$. In this paper, we apply an algorithm called the Forward-DP Backward-A^* N-best search algorithm [21] to k-best MIRA. The algorithm consists of a forward search and a backward search. For the forward search, we use Viterbi-style decoding to find the best partial path up to each node in the lattice. For the backward search, we use A^* decoding to extend the partial paths and rank them using their full path scores.

In summary, we use k-best MIRA to iteratively update $\mathbf{w}^{(i)}$. The final weight vector $\mathbf{w}$ is the average of the weight vectors after each iteration. As reported in [4, 19], parameter averaging can avoid overfitting. In testing, we can use Viterbi-style decoding to search the most likely path $\hat{\boldsymbol{y}}$ for an input sentence $\boldsymbol{x}$ where:

$$\hat{\boldsymbol{y}} = \underset{\boldsymbol{y} \in \mathcal{Y}}{\mathrm{argmax}} \ s(\boldsymbol{x}, \boldsymbol{y}; \mathbf{w}) \ . \tag{4}$$

3 Experiments

In this section, we describe our experiments to examine the feasibility of the proposed approach.

3.1 Experimental setup

3.1.1 Corpus

We performed experiments on the BEST 2009 shared task corpus using data sets no. 1–5 [7]. The data sets no. 1–5 consist of texts in three different domains: encyclopedia, novel, and news. We used the latest sets that have been released so far, where the data set no. 1 was taken from the third edition, and the data sets no. 2–5 were taken from the second edition. We conducted our experiments on each domain separately to observe the difficulty of that domain.

Table 1 summarizes statistics of the data sets. We split the data sets into training and test sets by using no. 1–4 for training and no. 5 for testing. We used the same splitting method for every domain. The *out-of-vocabulary* (OOV) is defined as words in the test set that do not occur in the training set.

We should note that we removed some erroneous sentences from the corpus. Thus, the numbers of sentences and words in Table 1 are slightly different from the actual numbers. We used simple writing rules to detect incorrectly segmented words. For example, a Thai word cannot start with a tonal mark (-่, -้, -๊, or -๋), or two English words should not be separated without a space between them. We considered the sentences that contain these errors as inconsistency in corpus annotation.

3.1.2 Parameter setting

In our model, there are three tunable parameters in training, including the number of training iterations N, the number of top k best paths, and the frequency threshold r of rare words. We set the first two parameters to their conventional values by $N = 10$ and $k = 5$ for all experiments. We considered words that occur only once in the training set as rare words, so $r = 1$. These rare words are decomposed into character clusters for generating character-cluster-level nodes so that the algorithm can learn statistics of unknown words in the training phase.

3.1.3 Evaluation measures

We evaluated the system performance by the recall (R), precision (P), and F_1. We also calculated the recall on unknown and known words to observe whether the system can handle unknown words effectively. These measures can be computed as follows.

$$Recall\ (R) = \frac{\#\text{ of correct words}}{\#\text{ of words in the test set}}$$

$$Precision\ (P) = \frac{\#\text{ of correct words}}{\#\text{ of words in the system output}}$$

$$F_1 = \frac{2 \cdot R \cdot P}{R + P}$$

$$R_{unknown} = \frac{\#\text{ of correct unknown words}}{\#\text{ of unknown words in the test set}}$$

$$R_{known} = \frac{\#\text{ of correct known words}}{\#\text{ of known words in the test set}}$$

3.2 Results

3.2.1 Baseline and topline experiments

Following [27], we conducted the *baseline* and *topline* experiments. The objective is to make the lower and upper bounds for performance measures. Here, we used the maximum matching algorithm that can perform simple word segmentation by using a given dictionary. For the baseline experiment, the dictionary was constructed from words occurring in the training set. For the topline experiment, we instead used words occurring in the test set. Tables 2 and 3 show results of the baseline and topline experiments, respectively. We can see that the sentences in the novel domain seems to be the most difficult case for analysis.

3.2.2 Results of the proposed approach

Table 4 shows results of the proposed approach. Focusing on the F_1 scores, the proposed approach performs significantly better than the baseline and approaches the topline. Compared with the topline, we can see that the proposed approach yields the better recall scores on the encyclopedia and novel domains, but the lower precision scores on every domain which is not surprising since the topline uses words in the test set for making the system's dictionary.

Table 5 shows the training and testing times of the proposed approach. All experiments were conducted on an Intel® Xeon$^{\text{TM}}$ CPU 3.80GHz with 8 GB RAM. We can see that the proposed approach requires a reasonable training time while it is very efficient for testing. For example, the algorithm took around 2 hours to train the news domain containing $\approx$ 1.1 million words, and took less than 1 minute to analyze 7,098 sentences (479,702 words).

	Encyclopedia	Novel	News
# of training sentence	37,141 (sets no. 1–4)	35,240 (sets no. 1–4)	23,260 (sets no. 1–4)
# of training words	842,055	1,160,956	1,115,976
# of test sentences	13,350 (set no. 5)	14,855 (set no. 5)	7,098 (set no. 5)
# of test words	315,596	495,357	479,702
OOV rate	0.0234 (7,392/315,596)	0.0278 (13,756/495,357)	0.0244 (11,725/479,702)

Table 1: Statistics of the data sets in our experiments.

Corpus	R	P	F_1	$R_{unknown}$	R_{known}
Encyclopedia	$0.9065\ (\frac{286{,}086}{315{,}596})$	$0.8960\ (\frac{286{,}086}{319{,}275})$	0.9012	$0.2212\ (\frac{1{,}635}{7{,}392})$	$0.9229\ (\frac{284{,}451}{308{,}204})$
Novel	$0.8630\ (\frac{427{,}485}{495{,}357})$	$0.8736\ (\frac{427{,}485}{489{,}354})$	0.8682	$0.0491\ (\frac{676}{13{,}756})$	$0.8862\ (\frac{426{,}809}{481{,}601})$
News	$0.9077\ (\frac{435{,}428}{479{,}702})$	$0.8721\ (\frac{435{,}428}{499{,}312})$	0.8895	$0.0559\ (\frac{655}{11{,}725})$	$0.9290\ (\frac{434{,}773}{467{,}977})$

Table 2: Results of the *baseline* experiment using the maximum matching algorithm with the system's dictionary constructed from words occurring in the training set.

Corpus	R	P	F_1	$R_{unknown}$	R_{known}
Encyclopedia	$0.9721\ (\frac{306{,}779}{315{,}596})$	$0.9783\ (\frac{306{,}779}{313{,}568})$	0.9752	$0.9862\ (\frac{7{,}290}{7{,}392})$	$0.9717\ (\frac{299{,}489}{308{,}204})$
Novel	$0.9492\ (\frac{470{,}210}{495{,}357})$	$0.9732\ (\frac{470{,}210}{483{,}160})$	0.9611	$0.9903\ (\frac{13{,}623}{13{,}756})$	$0.9481\ (\frac{456{,}587}{481{,}601})$
News	$0.9828\ (\frac{471{,}430}{479{,}702})$	$0.9904\ (\frac{471{,}430}{476{,}019})$	0.9865	$0.9918\ (\frac{11{,}629}{11{,}725})$	$0.9825\ (\frac{459{,}801}{467{,}977})$

Table 3: Results of the *topline* experiment using the maximum matching algorithm with the system's dictionary constructed from words occurring in the test set.

Corpus	R	P	F_1	$R_{unknown}$	R_{known}
Encyclopedia	$0.9732\ (\frac{307{,}148}{315{,}596})$	$0.9530\ (\frac{307{,}148}{322{,}290})$	0.9630	$0.4594\ (\frac{3{,}396}{7{,}392})$	$0.9856\ (\frac{303{,}752}{308{,}204})$
Novel	$0.9678\ (\frac{479{,}431}{495{,}357})$	$0.9494\ (\frac{479{,}431}{504{,}998})$	0.9585	$0.3939\ (\frac{5{,}419}{13{,}756})$	$0.9842\ (\frac{474{,}012}{481{,}601})$
News	$0.9735\ (\frac{466{,}968}{479{,}702})$	$0.9592\ (\frac{466{,}968}{486{,}841})$	0.9663	$0.5255\ (\frac{6{,}162}{11{,}725})$	$0.9847\ (\frac{460{,}806}{467{,}977})$

Table 4: Results of the proposed approach.

Corpus	Training	Testing
Encyclopedia	66.183 min	38.709 sec
Novel	112.067 min	56.762 sec
News	132.183 min	56.817 sec

Table 5: Training and testing times of the proposed approach. All experiments were conducted on an Intel® Xeon™ CPU 3.80GHz with 8 GB RAM.

4 Conclusion

In this paper, we have described a discriminative hybrid model for Thai word segmentation. Our approach has two important points. The first one is robust search space representation based on the hybrid model in which word-level and character-cluster-level nodes are used to identify known and unknown words, respectively. The second one is online learning that combines MIRA with efficient k-best decoding. Our algorithm has a few parameters to be tuned to provide satisfactory performance, and it is scalable to large datasets due to the property of online learning.

References

[1] Wirote Aroonmanakun. Collocation and thai word segmentation. In *Proceedings of the 5th SNLP & 5th Oriental COCOSDA Workshop*, pages 68–75, 2002.

[2] Surin Charnyapornpong. *A Thai syllable separation algorithm.* Asian Institute of Technology, Master Thesis, 1983.

[3] Paisarn Charoenpornsawat, Boonserm Kijsirikul, and Surapant Meknavin. Feature-based thai unknown word boundary identification using winnow. In *Proceedings of the IEEE Asia-Pacific Conference on Circuits and Systems*, 1998.

[4] Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of EMNLP*, pages 1–8, 2002.

[5] Koby Crammer. *Online Learning of Complex Categorial Problems.* Hebrew Univeristy of Jerusalem, PhD Thesis, 2004.

[6] Koby Crammer, Ryan McDonald, and Fernando Pereira. Scalable large-margin online learning for structured classification. In *NIPS Workshop on Learning With Structured Outputs*, 2005.

[7] Benchmark for Enhancing the Standard of Thai language processing (BEST), 2009. `http://www.hlt.nectec.or.th/best/`.

[8] Choochart Haruechaiyasak, Sarawoot Kongyoung, and Matthew N. Dailey. A collaborative framework for collecting thai unknown words from the web. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 345–352, 2006.

[9] Choochart Haruechaiyasak, Sarawoot Kongyoung, and Matthew N. Dailey. A comparative study on thai word segmentation approaches. In *Proceedings of ECTI-CON*, 2008.

[10] Chuleerat Jaruskulchai. An automatic thai lexical acquisition from text. In *Proceedings of PRICAI*, 1998.

[11] Chuleerat Jaruskulchai. *An Automatic Indexing for Thai Text Retrieval.* George Washington University, PhD Thesis, 2004.

[12] Asanee Kawtrakul and Chalatip Thumkanon. A statistical approach to thai morphological analyzer. In *Proceedings of of the 5th Workshop on Very Large Corpora*, 1997.

[13] Asanee Kawtrakul, Chalatip Thumkanon, Yuen Poovorawan, Patcharee Varasrai, and Mukda Suktarachan. Automatic thai unknown word recognition. In *Proceedings of the Natural Language Processing Pacific Rim Symposium*, pages 341–348, 1997.

[14] Canasai Kruengkrai and Hitoshi Isahara. A two-pass search algorithm for thai morphological analysis. *Advances in Natural Language Processing and Applications, Research in Computing Science*, 33:81–92, 2008.

[15] Canasai Kruengkrai, Virach Sornlertlamvanich, and Hitoshi Isahara. A conditional random field framework for thai morphological analysis. In *Proceedings of LREC*, 2006.

[16] Canasai Kruengkrai, Kiyotaka Uchimoto, Jun'ichi Kazama, Kentaro Torisawa, and Hitoshi Isahara. Accurate language-independent morphological analysis using a discriminative hybrid model. *To submit to JNLP*, 2008.

[17] Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. Applying conditional random fields to japanese morphological analysis. In *Proceedings of EMNLP*, pages 230–237, 2004.

[18] Ryan McDonald. *Discriminative Training and Spanning Tree Algorithms for Dependency Parsing*. University of Pennsylvania, PhD Thesis, 2006.

[19] Ryan McDonald, Femando Pereira, Kiril Ribarow, and Jan Hajic. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of HLT/EMNLP 2005*, pages 523–530, 2005.

[20] Surapant Meknavin, Paisarn Charoenpornsawat, and Boonserm Kijsirikul. Feature-based thai word segmentation. In *Proceedings of of NLPRS*, 1997.

[21] Masaki Nagata. A stochastic japanese morphological analyzer using a forward-DP backward-A* n-best search algorithm. In *Proceedings of the 15th International Conference on Computational Linguistics*, pages 201–207, 1994.

[22] Tetsuji Nakagawa. Chinese and japanese word segmentation using word-level and character-level information. In *Proceedings of COLING*, pages 466–472, 2004.

[23] Tetsuji Nakagawa and Kiyotaka Uchimoto. A hybrid approach to word segmentation and pos tagging. In *Proceedings of ACL Demo and Poster Sessions*, 2007.

[24] Fuchun Peng, Fangfang Feng, and Andrew McCallum. Chinese segmentation and new word detection using conditional random fields. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 23–27, 2004.

[25] Yuen Poowarawan. Dictionary-based thai syllable separation. In *Proceedings of the Ninth Electronics Engineering Conference*, 1986.

[26] Virach Sornlertlamvanich, Tanapong Potipiti, and Thatsanee Charoenporn. Automatic corpus-based thai word extraction with the c4.5 learning algorithm. In *Proceedings of COLING*, pages 802–807, 2000.

[27] Richard Sproat and Thomas Emerson. The first international chinese word segmentation bakeoff. In *Proceedings of the 2nd SIGHAN Workshop on Chinese Language Processing*, pages 133–143, 2003.

[28] Yupin Thairatananond. *Towards the Design of a Thai Text Syllable Analyzer*. Asian Institute of Technology, Master Thesis, 1981.

[29] Thanaruk Theeramunkong and Sasiporn Usanavasin. Non-dictionary-based thai word segmentation using decision trees. In *Proceedings of the First International Conference on Human Language Technology Research*, pages 251–256, 2001.

[30] Nianwen Xue. Chinese word segmentation as character tagging. *International Journal of Computational Linguistics and Chinese*, 8(1):29–48, 2003.

[31] Stavros A. Zenios Yair Censor. *Parallel Optimization: Theory, Algorithms, and Applications.* Oxford University Press, 1997.

[32] Yue Zhang and Stephen Clark. Chinese segmentation with a word-based perceptron algorithm. In *Proceedings of ACL*, 2007.