Improving Thai Word and Sentence Segmentation Using Linguistic Knowledge

RUNSIMAN NARARATWONG††††, NATHAWUT KERTKEIDKACHORN††††, NAGUL COOHAROJANANONE††††, NONMEMBERS, AND HITOSHI OKADA††, MEMBER

SUMMARY Word boundary ambiguity in word segmentation has long been a fundamental challenge within Thai language processing. The Conditional Random Fields (CRF) model is among the best-known methods to achieve remarkably accurate segmentation. Nevertheless, current advancements appear to have left the problem of compound words unaccounted for. Compound words lose their meaning or context once segmented. Hence, we introduce a dictionary-based word-merging algorithm, which merges all kinds of compound words. Our evaluation shows that the algorithm can accomplish a high-accuracy of word segmentation, with compound words being preserved. Moreover, it can also restore some incorrectly segmented words. Another problem involving a different word-chunking approach is sentence boundary ambiguity. In tackling the problem, utilizing the part of speech (POS) of a segmented word has been found previously to help boost the accuracy of CRF-based sentence segmentation. However, not all segmented words can be tagged. Thus, we propose a POS-based word-splitting algorithm, which splits words in order to increase POS tags. We found that with more identifiable POS tags, the CRF model performs better in segmenting sentences. To demonstrate the contributions of both methods, we experimented with three of their applications. With the word merging algorithm, we found that intact compound words in the product of topic extraction can help to preserve their intended meanings, offering more precise information for human interpretation. The algorithm, together with the POS-based word-splitting algorithm, can also be used to amend word-level Thai-English translations. In addition, the word-splitting algorithm improves sentence segmentation, thus enhancing text summarization.

key words: Thai, word segmentation, topic extraction, translation, summarization

1. Introduction

Processing languages without word boundary delimiters, such as Thai, require word tokenization. Failing to chunk precisely may result in misleading interpretations. The ambiguity of a compound word, also called a compositional word, contributes to the problem. In word formation, a sequential combination of words forms a new word with a different meaning, grammatical property or communication role; the reverse process, word segmentation, may or may not produce a meaning that is similar to the meaning of the initially whole word, making the outcome ambiguous. For example, ‘นี่ตา’ [nì tā] (eye), when chunked, becomes ‘นิ่น’ [nīn] (much) and ‘ตา’ [tā] (cry).

Early word chunking methods evolved via the use of a dictionary. Longest matching [1], a simple greedy algorithm, finds and separates the longest recognizable word from the beginning of a sentence, then moves to the next one. The more sophisticated maximum matching [2] chooses one that contains the fewest words from possible segmentation choices. Later, with the help of grammatical information, statistics came into play [3], [4]. Despite these developments, the evolution of language means that no dictionary can always capture every word.

To progress beyond this limitation, a machine must learn. Machine-learning-based approaches involve the context in which words appear [5]. Decision tree and character clustering methods have outperformed the dictionary-based method [6], [7]; above all, we have seen the outstanding performance of a CRF model [8] across diverse contents. That being said, the compound word issue still exists, and building another training corpus to address this problem would be economically inefficient.

We built two post-processing methods on top of CRF-based word segmentation. The former, designed to enhance semantic interpretation, merges segmented compound words in a text chunked by the machine-learning approach, restoring the words’ original meaning and context. The latter splits words to boost POS tagging, resulting in better CRF-based sentence segmentation.

In the next section, we begin with relevant Thai linguistic fundamentals before exploring the evolution of word segmentation and presenting our new methods. We then explain the evaluation settings and experiment results, followed by the demonstration of how our methods as they improve three different text-processing tasks, and the conclusion.

2. Linguistic Fundamentals

Perhaps the word ‘รูปตา’ best represents the problematic word boundary ambiguity. One may choose to visualize a perfect round eye – ‘รูป’ [ruːp] for an eye and ‘ตา’ [tā] for the noun – or a restful vacation.
basking ("son" [tak]) in the wind ("mau" [kom]). "son" alone can never tell us which message it is meant to convey in the absence of context. We adapted segmentation rules from Benchmark for Enhancing the Standard of Thai Language Processing (BEST2009) [9] with a different approach to preserve compound words, that is, word-merging conditions, highlighted as follows:

2.1 Word Segmentation Based on Word Formation

Thai linguists have categorized compound words according to two types: ones with entirely unrelated roots, and ones with roots that are semantically comparable. Once merged, they become a new word, usually a noun or a verb. Dividing the former type would leave us with entirely irrelevant roots. For example, segmenting "awerapi" [maew mong] (scout) would create "awr" [maew] (cat) and "erapi" [mong] (look). Chunking the latter type may as well lead to ambiguity. Chunking "awrtes" [nag rong] (singer), "tn" [nag] may either mean much or be used as a modifier; "tes" [rong] means to utter or to cry.

2.2 Word Segmentation Based on Grammatical Rules

Compound conjunctions have multiple roots. We support our proposition that, even though their roots may resemble their meaning somewhat, they should also not be chunked – for example, "kireu" [dung naan] means therefore; "s" [dung] means similar, and "na" [naan] refers to a noun or a sentence that has already been said – the same applies to those with non-comparable roots and those containing a determiner. Nonetheless, chunking becomes a sensible approach for compound conjunctions consisting of multiple types – for example, "wo" [tae] means but; "n" [ta] means if, and "nhr" [ta] means but if.

2.3 Word Segmentation Based on Communication Roles

As with compound conjunctions, interrogative words are not all indivisible; again, chunking them would merely complicate the overall interpretation of a sentence (for example, "o Nhi" [tao dai] as an interrogative word means how much, "h" [tao] means equal, and "s" [dai] means which. However, a conjunction sometimes appears next to a classifier ("kon" is a unit classifier for a human; "maew" [nai] means which); as they are treated as different words, they should be chunked.

3. Literature Review

The longest matching and maximum matching algorithms are plagued with problems related to dictionary size and unknown words. Researchers have attempted to cope with the unknown – for example, Haruechaiyasak and his colleagues have proposed the unknown word collecting framework [10]. Theirs is an integrative framework that extracts unrecognizable words, which are then reviewed and corrected by humans before being added to a dictionary. Another project, LEXiTRON [11], is an expertise-based vocabulary-suggestion system. The community-driven dictionary currently holds over a hundred-thousand words (of which community members have added over twenty thousand) [11]. An alternative to expanding dictionaries is the statistical approach. Using a hand-tagged corpus, Asanee and Chalathip’s work [12] proves that statistical data information can be of great help in reducing not only word boundary ambiguity but als POS-tagging ambiguity and implicit spelling errors.

Researchers have made giant leaps since the development of machine learning. Thai character clusters (TCCs), which operate via a decision tree C4.5 model [6], [13], outperform traditional dictionary-based techniques. Each of TCCs contains contiguous characters – ones that are, generally speaking, inseparable. The clusters are as well a vital part of Limcharoen et al.’s work [14], which applies Generalized Left-to-Right (GLR) parsing and a statistical language model to word segmentation, and also Kruengkrai et al.’s word- and character-cluster hybrid model [15], which is useful for handling unknown words. Other approaches include statistical machine translation [16] and advanced word-boundary identification. Such algorithms use segmented sentences, selecting the one that maximizes the probabilities of the translation model and language model of a given unsegmented sentence.

CRF is behind the success of highly accurate word-segmentation approaches across multiple languages [17]. The statistical modeling method has been the preferred choice for various text-processing tasks, such as POS tagging, named-entity recognition, and sentence segmentation [18], [19]. In Thai word segmentation, Susapanit et al. [20], as well as Haruechaiyasak and Kongyong [8], have demonstrated how CRF can be implemented. Both sets of researchers found that, after several adjustments, they could leverage the characteristics of the language and achieve even greater accuracy. Despite these successes, however, the problems of compound-word ambiguity and the lack of POS tags in CRF-based sentence segmentation are yet to be solved. These are the areas to which our algorithms contribute.

4. Methodology

The entire process consists of two steps. First, we refine the CRF model for word segmentation, then perform the post-processing. The word-merging algorithm addresses the compound word problem, while the word-splitting algorithm targets the sentence segmentation issue.

4.1 CRF-Based Word Segmentation

CRF offers flexibility in customizing feature sets and templates. We apply a feature set that was introduced and tested

1Available at http://lexitron.nectec.or.th
Table 1 Character-type feature set [8]

| Tag | Type of Characters | Items |
|-----|-------------------|-------|
| c   | Consonant which can be a word ending character | ก,ฃ,ศ,ษ,ส,ง |
| n   | Consonant which cannot be a word ending character | บ,ป,ผ,ฝ,ฟ,ธ,น,ย |
| w   | Vowel that can be at the beginning of a word | ฮ,อ,อ(พ),เอ,ย,ย(ค),อิ,อุ |
| v   | Vowel that cannot be at the beginning of a word | ฮ,อ,อ(พ),เอ,ย,ย(ค),อิ,อุ |
| t   | Tonal             | ค,ฅ,ฉ,չ |
| s   | Symbol            | ง,จ,ฉ |
| d   | Digit             | 0-9 |
| q   | Quote             | ‘”’ |
| p   | Space within a word | - |
| o   | Other             | ส-Z |

Table 2 Feature set templates

| Type          | Feature |
|---------------|---------|
| Character     | Single [8] | Ct |
|               | Combine-1 | Ct, Cs, CtCs, CsCt, CcGt, CgCc, CgCcCt |
|               | Combine-2 | Ct, Cs, CtCs, CsCt, CcGt, CgCc, CgCcCt |
| Category      | Single [8] | Tt |
|               | Combine-1 | Tt, TtTb, TsTt, TtTb, TsTt |
|               | Combine-2 | Tt, TtTb, TsTt, TtTb, TsTt |
|               |         | TtTbTt, TtTbTt, TtTbTt |

by Haruechaiyasak and Kongyoung [8] (Table 1). They found that using character and character-type feature sets combined yielded a more accurate result (an F-score of 0.94 as opposed to 0.92 for character-only and 0.63 for character-type-only feature sets).

Haruechaiyasak and Kongyoung’s settings regard each character and its character type individually. We consider up to two characters (including their types) before and after the character of interest. Suppose $C_t$ is the character of interest and $T_t$ is its character-type: $C_{t-l}$ and $T_{t-l}$ would be the previous ones, and $C_{t+l}$ and $T_{t+l}$ would be the ones that follow. We introduce three feature-set templates: Single, which comprises only the character of interest and its character-type, Combine-1, which adds combinations of up to one character before and after, and Combine-2, which adds up to two characters. Looking further than two adjacent characters causes an inordinate leap in demand for computational resources; thus, is not cost-effective. Table 2 describes our feature-set templates.

For example, the word ฮานี has six characters: ฮ, อ, น, ี, ว. Suppose the forth character is the character of interest, the set of characters should be: $C_{t-2} = ฮ, C_{t-1} = อ, C_{t} = น, C_{t+1} = ี, C_{t+2} = ว$. In the combined sequence, we would have, for example, $C_{t-2}C_{t-1}C_{t} = ฮอนำ$. The characters’ category should be: $T_{t-2} = ฮ, T_{t-1} = อ, T_{t} = น, T_{t+1} = ี, T_{t+2} = ว$, $T_{t+3} = ว$.

We expected to boost the accuracy of the CRF model via the introduction of the feature-set templates. In fact, a similar idea was mentioned in Susatpanit et al.’s work [20]. Zhao et al. [21] also defined feature-set templates, partly from the unigrams and bigrams of the characters, and observed an improvement in their Chinese word-segmentation model.

4.2 Dictionary-Based Word Merging

When the BEST2009 corpus was released, NECTEC published detailed instructions of how it was constructed [9]. The document explained relevant linguistic definitions and segmentation rules, laying the foundation upon which we built our study. As explained in the linguistic fundamentals section, we hypothesized that compound words, conjunctions, and interrogative words with semantically comparable roots should not be chunked; otherwise, we would risk altering their message or context. The rules are different in the BEST2009 corpus; in which these words are all chunked. That being said, we are not repudiating the original labeling. Instead, we prefer the idea that different chunking methods serve different purposes. In an attempt to reverse the chunking, we could relabel the corpus and preserve the compound words, but that would not be cost-effective. Instead, since compound words are considered to be a single word in terms of their dictionary definition, they ought to be defined in a dictionary. Therefore, if any sequential combination of multiple segmented words appeared in a dictionary, we merged it.

For a set of all vocabularies $V$ in a dictionary, suppose a sentence $S$ contains the words $w_1 \ldots w_n$. For each $w_j$ where $1 \leq j \leq n(S)$, the merging algorithm first attempts to find the longest recognizable sequential combinations $w_j \ldots w_{j+k-1}$, then the shorter ones until $w_jw_{j+k}$. The longest sequence possible is the length of the longest word ($l$) in the dictionary. This is the case when the longest word appears in the sentence and its characters are all separated, which is highly unlikely. If a combination of $m = w_j \ldots w_{j+k}$ where $1 \leq k \leq l-1$ is a word, $w_j$ would be replaced by $m$ and $w_{j+k}, \ldots, w_{j+k}$ would be removed from the sentence. Figure 1 shows our word-merging algorithm.

Our aim is for the algorithm to be able to preserve compound words efficiently without compromising on the accuracy achieved by the CRF model. The algorithm will also offer the simplicity of not having to change the design of the CRF model to perform the task.

4.3 POS-Based Word Splitting

Nothing marks the end of a sentence in Thai. A space may be the most consistent as a sentence delimiter, but it is used for other purposes as well, such as separating numbers from characters. Moreover, Zhou et al. [18] found no spaces at the end of about 23% of the sentences in TalAPi’s news corpus (5,489 articles; over three million words) [22]. For developing a reliable sentence segmentation method, Zhou et al. turned to CRF. The task is to determine whether a word, with a space included, is the last in a sentence. Since this is a word-level operation, they have the advantage of utilizing POS as a feature set. However, following their method, we noted words lacking a POS tag. The dearth of POS tags...
could be the result of inaccurate word segmentation, or the inability of the POS tagger to cope with obscure words. If vocabulary size is the problem, then compound words may be the cause. First, as they are highly diverse, one corpus, even a large one, may omit a considerable number of them. Second, if compound words with roots resembling their meaning were chunked and their roots were assigned a POS tag in a corpus, it may be better to divide them whenever possible to increase POS tags.

We defined our POS-based word-splitting algorithm (Fig. 2) as follows: If a word does not have a POS tag but all its roots can be tagged, split it. For a word \( w \) consisting of characters \( c_1 \ldots c_n \), where \( n \) is the length of the word, we split the characters into \( p \) sequential groups, where \( 2 \leq p \leq n - 1 \). Each group, \( g \), becomes a candidate root. At every iteration, we generate all possible splitting outcomes; for example, at \( p = 2 \), \( w = c_1c_2c_3c_4 \), we would have \( g_{2,1} = c_1, c_2c_3c_4; g_{2,2} = c_1c_2, c_3c_4; g_{2,3} = c_1c_2c_3, c_4 \). For every splitting outcome, if all the roots can be tagged, the outcome would be considered a candidate. From the pool of candidates, we then choose the one with the most roots to be the outcome; that is, the split words.

More POS tags mean more information is available for the sentence segmentation model to make predictions; ergo, more accurate chunking. We hypothesized that an increased number of POS tags would make sentence segmentation more accurate.

5. Experiments and Results

5.1 CRF-Based Word Segmentation

Our methods modify the result of the CRF model. Thus, it was imperative that we first found the CRF model’s setting that yielded the highest accuracy. While we did not alter Haruechaiyasak and Kongyoung’s character-type feature set, we needed to choose from our three feature templates, Single, Combine-1, and Combine-2. We expected Combine-2 to outperform the other two, since this template takes the greatest length of the neighbors into consideration.

We trained and tested the CRF model using BEST2009. The corpus contains over five million words across eight genres. Eighty percent of the corpus is for training; the rest is for testing. We incorporated the Python package sklearn-crfsuite, which implements Lafferty et al.’s work [23]. The training implemented L-BFGS algorithm for gradient descent, with the coefficients of both L1 and L2 regularizations set to 0.1. The optimization algorithm was iterated at the maximum of 100 iterations. We configured the CRFSuite to generate state features that associated all of the possible combinations between attributes and labels, enabling the CRF model to learn the condition in which an item was not predicted by its reference label.

The model was trained on Amazon EC2 r3.4xlarge instance (16 vCPU on Intel Xeon E5-2670 v2 2.5 GHz, 122 GiB memory, and 320 GB SSD storage). The testing does not require a substantial amount of memory; thus, a computer with 1.7 GHz Intel Core i7 processor, 8 GB 1600 MHz DDR3 memory, and 256 GB SSD storage suffices; the same goes for every experiment hereafter.

In Table 3, with the Combine-2 feature template, we were able to increase the accuracy up by six percent. This

| Feature Template | Precision | Recall | F1 |
|------------------|-----------|--------|----|
| Single [8]       | 93        | 93     | 93 |
| Combine-1        | 96        | 96     | 96 |
| Combine-2        | 99        | 99     | 99 |

†Available at https://pypi.python.org/pypi/sklearn-crfsuite
††More information at https://sklearn-crfsuite.readthedocs.io
†††Available at https://aws.amazon.com/ec2

Fig. 1 Dictionary-based word merging algorithm

Fig. 2 POS-based word-splitting algorithm

Table 3 CRF-based word segmentation on BEST2009 corpus
increase corroborates Zhao et al.'s [21] finding that incorporating neighboring characters augments the accuracy of the CRF model. Once we had ascertained the preeminent feature template, if we were to have retained the original BEST2009 corpus’ rules without change, we would already have had a highly reliable CRF-based word segmentation model at hand; however, as we aimed to leave all the compound words intact, we continued.

### 5.2 Dictionary-Based Word Merging

When building a test corpus, we assigned one native Thai speaker to relabeled 509 paragraphs that were chosen randomly chosen from all the document categories in BEST2009, yielding a total of 25,603 words. Following the rules in the **linguistic fundamentals** section, compound words, conjunctions, and interrogative words with semantically comparable roots were all preserved; we changed nothing else.

The word-merging algorithm looks up its candidate words in three dictionaries – Wiktionary†, LibThai††, and LEXiTRON†††. In their work, Haruechaiyasak and Kongyoung proposed named-entity (NE) merging as a post-CRF process that helps enhance word segmentation. They extracted NEs from the BEST2009 corpus, and so did we. In addition, we merged names from LibThai and GeoNames†††† with our NE list.

Table 4 shows the evaluation results. We tested all the combinations of the six corpora and found a moderately positive correlation between the number of corpora and the accuracy (r = .48, n = 64, p < 0.01). The best combination for our test data was the one that incorporated all the corpora except GeoNames (F1 = .9835). The F1-score decreased slightly when all corpora were included (F1 = .9834). Therefore, it is best to include more corpora to increase the accuracy.

In the results, we found compound words that retained their meaning after the merging; for example, ‘ результат’ (result), ‘ассамбле’ (assemble), ‘задача’ (task) become ‘результат’. However, the algorithm sometimes mistakenly combined correctly chunked words. For example, ‘ чемпион’ (be number one) should not be chunked unless it is a part of ‘ чемпионство’. In this case, the correct segmentation would be ‘ чемпион’ (is), ‘ чемпионство’ (one), ‘ чемпионство’ (of). Such a case is not common, but doubtless contributed to the inaccuracy observed. Another unexpected case was word correction, as in the case of ‘и’ and ‘его’, which were segmented incorrectly; neither has a meaning individually, but become ‘его’ (cause) together.

### 5.3 POS-Based Word Splitting

We evaluated our word-splitting algorithm using the ORCHID corpus [24]. ORCHID contains the POS tags and sentence-level boundary annotations necessary for the evaluation. To train the sentence segmentation model, we employed sklearn-crfsuite with identical settings to those in the word segmentation experiment. Zhou et al.’s original CRF feature template [18] includes up to one neighboring word, word-type (English word, Thai word, punctuation, digits, and spaces), and POS. Our extended template covers two adjacent words. We used eighty percent of the corpus for training and the rest for testing.

As shown in Table 5, extending the feature set template boosted the F1-score by .15; thus, it became our choice for the evaluation of the word-splitting algorithm. We trained the sentence-segmentation model using the entire corpus, with word-level annotations and POS tags removed; we then chunked all the sentences and labeled each word as *E* or *I*, depending on its position in the sentence, and removed sentence-level annotations.

As expected, not all the segmented words could be tagged via POS; hence, the sentence-segmentation model decreased in accuracy. The splitting algorithm was able to recover the average F1-score by 3.58 percent in relation to the loss margin†††††, with 1.39 percent more POS be-

---

Table 4  Evaluation of word merging algorithm on relabeled corpus

| Template | Precision | Recall | F1  |
|----------|-----------|--------|-----|
| Single [8] | 88 | 85 | 86  |
| All NE | 88 | 82 | 83  |
| NE: BEST2009 | 92 | 86 | 88  |
| NE: LibThai | 94 | 91 | 92  |
| NE: GeoNames | 95 | 94 | 94  |
| All NE | 92 | 86 | 87  |

*All NE combines NEs from BEST2009, LibThai and GeoNames. All Dict. combines vocabularies from LEXiTRON, Wiktionary, and LibThai.*

Table 5  Accuracy measurement on original ORCHID corpus

| Template | Measure | Precision | Recall | F1  | #POS Tagged |
|----------|---------|-----------|--------|-----|-------------|
| Original [18] | E | .81 | .56 | .66  |
| I | .97 | .99 | .98  |
| Avg. | .96 | .96 | .96  |
| Extended | E | .86 | .77 | .81  |
| I | .98 | .99 | .99  |
| Avg. | .97 | .98 | .97  |

*E represents words at the end of a sentence; I represents the rest, including spaces between words.*

Table 6  Accuracy measurement on processed ORCHID corpus

| Template | Precision | Recall | F1  | POS Tagged |
|----------|-----------|--------|-----|-------------|
| WS + CRF | E | .80 | .73 | .76  |
| using ET | I | .98 | .99 | .99  |
| Avg. | .97 | .97 | .97  |
| WS + CRF | E | .79 | .74 | .77  |
| & Split using ET | I | .99 | .99 | .99  |
| Avg. | .97 | .98 | .97  |

*WS stands for word segmentation.*

---

†Available at https://www.wiktionary.org
††More information at https://linux.thai.net/projects/libthai
†††More information at https://www.nectec.or.th
††††More information at http://www.geonames.org
†††††(F1WS + CRF & Split − F1WS + CRF)/(F1CRF − F1WS + CRF) × 100
ing tagged. Overall, the word splitting algorithm improved CRF-based sentence segmentation.

6. Applications

6.1 Topic Extraction

The Latent Dirichlet Allocation (LDA) [25] and the Hierarchical Dirichlet Process (HDP) [26], [27] models are capable of discovering underlying topic structures in discrete data. Both generate small sets of words, each associated with a latent topic. From these words, humans interpret the topics, making it vital that the words’ meanings and contexts are easily comprehensible. Therefore, we hypothesized that by merging compound words, preserving their original meaning, would make the interpretation more accurate.

To investigate this, we collected 2,000 tweets, half of which were flood-related and the rest pertained to traffic. All the tweets in the flood corpus contained a hashtag #thaiflood. The traffic tweets were from two prominent Twitter accounts, @js100radio and @fm91trafficpro, both dedicated to providing traffic information. Each tweet had all the URLs, usernames, and hashtags removed before the words were chunked and merged. We then fed both corpora separately into LDA and HDP models of Python package gensim†, creating a total of four tests.

As shown in Table 7, in each of the four tests, we selected the 50 most relevant words from five topics, yielding 250 words in total. We found merged words, meaning that the algorithm affected topic extraction. The examples in Table 8 further explain the results.

If a topic extraction model regards the chunked words separately, it will omit the relation the words once had, potentially leading to poor interpretation. For example, ‘‘water’’ (water) as in ‘‘water level’’ may also be a part of ‘‘drinking water’’ in some other sentences. While ‘‘water level’’ conjures a sense of a situation report, ‘‘drinking water’’ may be in sanitation guidelines, which is a different topic. Isolating the words ‘‘water,’’ ‘‘level’’ and ‘‘drink’’ does not convey meaning to interpret. Thus, by applying our word-segmentation algorithm, compound words remained intact and their meaning was preserved, enhancing the interpretation of topic extraction.

6.2 Translation

We investigated whether splitting and merging words improved translation. Using the fact that the word-merging algorithm could merge incorrectly segmented words, we searched for words that were not found in any dictionary, attempted to split them to remove incomprehensible characters, and then merged them. For example, ‘‘น้ำ น้ำหน้า’’ are three wrongly separated tokens with no meaning. Splitting the sentence produces ‘‘น้ำ น้ำหน้า,’’ which consists of six tokens, two of which are meaningful (‘‘น้ำ’’ means in and ‘‘หน้า’’ means have). After merging, the sentence becomes ‘‘น้ำ น้ำหน้า,’’ which is sequentially translated as in (น้ำ), field (น้ำหน้า) has (น้ำ) and rice (ข้าว).

We collected 50 abstracts from the Journal of King Mongkut’s University of Technology North Bangkok††. All the abstracts had an English translation, which we used as a test data. We created two experimental conditions. In the first condition, we segmented words in the Thai abstracts. The chunked texts were then split and merged in the second condition.

We fed the processed Thai abstracts in both conditions into our pre-trained machine translation model and into Google Translate†††, creating four conditions in total. We trained the Thai-English Transformer model [28] using tensor2tensor†††† and 75,535 parallel sentences obtained from TED Talks†††††. The English translations were then compared to their human-translated references using three of ROUGE metrics [29].

As shown in Tables 9 and 10, Google Translate outperformed our pre-trained model by a wide margin. We speculate that this may have been the result of the small amount of training data. There was no significant difference regarding the accuracy between the two conditions when translated using the Transformer model. For Google Translate, on the other hand, a paired-samples t-test of the 50 abstracts’ ROUGE evaluations, as shown in Table 10, in-

| Table 7 | Word merging in topic extraction results |
|---------|----------------------------------------|
| Corpus  | Model  | Selected Most Relevant Words | Percentage of Merged Words in Relation to Selected Most Relevant Words |
| Flood   | LDA    | 250                           | 7.6%                                 |
| Flood   | HDP    | 250                           | 23.6%                                |
| Traffic | LDA    | 250                           | 10%                                  |
| Traffic | HDP    | 250                           | 16%                                  |

| Table 8 | Examples of merged words in topic extraction |
|---------|-----------------------------------------------|
| Corpus  | Chunked Compound Words | Merged Words |
| Flood   | น้ำ (level), น้ำ (water) | น้ำ (water) |
|         | น้ำหน้า (water level) | น้ำหน้า (water level) |
|         | ผู้ (person), ผู้เสีย (face disaster) | ผู้เสีย (victim) |
|         | น้ำใส (must), น้ำใส (fresh) | น้ำใส (fresh) |
| Traffic | รถ (car), รถติด (stuck) | รถติด (traffic jam) |

†Available at https://pypi.python.org/pypi/gensim

†††Available at http://ojs.kmutnb.ac.th/index.php/kjournal/
††††Available at https://translate.google.com
†††††Available at https://github.com/tensorflow/tensor2tensor
†††††Available at https://www.ted.com
dicated a significant improvement in translation. However, this improvement may be limited by factors beyond our control due to Google Translate’s ability to repair incorrectly separated words and merge compound words. That being said, the improvement observed tells us that our method can help to make word-level Thai-English translation more accurate. Note that character-level translation is beyond the scope of this study, since our focus is on word segmentation.

6.3 Summarization

A graph-based algorithm, TextRank [30], has long been well known in text summarization. The relationship between nodes determines a connection between two sentences in terms of similarity measured as a function of their content overlap. The overlap can be as simple as the number of common lexical tokens in the two sentences, which was the setting for our implementation, or it may involve more sophisticated syntactic features such as POS. The algorithm, operating at both the word level and sentence level, ranks important sentences based on the graph.

In Thai, the word-level operation is not a serious problem since, although not perfect, the CRF-based word chunking is reasonably reliable. Sentences, on the other hand, are a different conundrum to solve. If, for example, the important sentence is mistakenly segmented, parts of it may be omitted, while a segment of its less-important neighbors may be included. In either case, the error undermines what should have been, as decided by TextRank, the best summary of the document.

Our summarization experiment tested whether the improved sentence segmentation could enhance text summarization. We used the TextRank-based summarization function of gensim with a default summarization ratio (0.2). The test corpus, summarized manually by one Thai native speaker, contained 50 on-line articles from various publishers and across different topics. The first experimental condition involved word chunking; of an average of 1163.3 words per article, 85.31 percent could be tagged. In the second condition, we performed the POS-based word splitting on the segmented text from the first condition, thus increasing the POS tags by 1.19 percent.

As evidenced in the test results shown in Table 11, our POS-based word-splitting algorithm improved the summarization significantly. In conclusion, the splitting method increased the POS tags, leading to more accurate sentence segmentation and better summarization.

### Table 10 Paired samples test of F1-scores (Google Translate)

|              | WS    | WS + Split & Merge |
|--------------|-------|--------------------|
| Mean         | 0.61  | 0.63               |
| SD           | 0.07  | 0.06               |
| t            | 4.73  | 4.56               |
| df (2-tailed)| 49    | 49                 |
| Sig.         | <.001 | <.001              |

### Table 11 Paired samples test of F1-scores between the two conditions

|              | Split | POS |
|--------------|-------|-----|
| Mean         | 0.63  | 0.66 |
| SD           | 0.06  | 0.05 |
| t            | 4.56  | 4.73 |
| df (2-tailed)| 49    | 49   |
| Sig.         | <.001 | <.001|

* p ≤ 0.05, ** p ≤ 0.01; WS stands for word segmentation.

7. Conclusion

We introduced two post-processing methods for CRF-based word segmentation. The first method, a dictionary-based word-merging algorithm, addresses the compound-word ambiguity issue by implementing the notion that all kinds of compound words should stay intact to maintain their precise meaning and context. Our experiment has demonstrated that the algorithm can preserve the compound words, meaning that the entire corpus does not have to be recreated by the researchers for the task.

The second method, a POS-based word-splitting algorithm, targets the sentence-boundary ambiguity. The algorithm increases POS tags with segmented words, giving the CRF-based sentence-chunking model more information for the prediction of sentence boundaries. Our experiment verified that the method improves results.

The word-merging method benefits topic extraction by making human interpretation more comprehensive. In combination with dictionary-based word splitting, the algorithm enhances word-level Thai-English translations. Lastly, the POS-based word-splitting method improves both sentence segmentation and text summarization.

References

1. Y. Poowarawan, “Dictionary-based Thai Syllable Separation,” Proc. 9th Electronics Engineering con., pp. 409–418, 1986.
2. S. Rarunrom, “Dictionary-based Thai Word Separation,” Thesis, 1991.
3. A. Pornprasertkul, “Thai Syntactic Analysis,” Thesis, 1994.
4. A. Kawtrakul, S. Kuntanode, T. Jamjanya, and C. Jewriyavech, “A Lexibase Model for Writing Production Assistant System,” Proc. symp. on National Language Processing in Thailand, pp. 398–406, 1995.
5. S. Meknavin, P. Charoenpornsawat, and B. Kijsirikul, “Feature-based Thai Word Segmentation,” Proc. Natural Language Processing Pacific Rim symp., 1997.
6. T. Theeramunkong and S. Usanavasin, “Non-Dictionary-Based Thai Word Segmentation Using Decision Trees,” Proc. 1st intl. con. on Human Language Technology Research, pp. 1–5, 2001.
7. T. Theeramunkong, V. Sorntertlanvanich, T. Tanhemhong, and W. Chinnan, “Character-Cluster Based Thai Information Retrieval,” Proc. 5th intl. Workshop on Information Retrieval with Asian Languages, pp. 75–80, 2000.
8. C. Haruechaiyasak and S. Kongyoung, “Flex: Thai Lexeme Analyser Based on the Conditional Random Fields,” Proc. intl. symp. on Natural Language Processing, pp. 13–17, 2009.
9. Human Language Technology Laboratory, National Electronics and Computer Technology Center (NECTEC), “Benchmark for Enhancing the Standard of Thai Language Processing,” 2009.
10. C. Haruechaiyasak, C. Sangkettrakarn, P. Palingoon, S. Kongyoung, and C. Damrongrat, “A Collaborative Framework for
Rungsiman Nararatwong received the BSc in Computer Science from Chulalongkorn University, Thailand. He is now a graduate student at Sokendai (The Graduate University for Advanced Studies). His research interests include online marketing, media psychology, social media, and Thai language processing.

Nagul Cooharojananone is an associate professor at Chulalongkorn University, Thailand. He received his B.S. degree in 1996 from Mahidol University. He received his M.E. and Ph.D. degrees in 2000 and 2004 from the University of Tokyo. His research interests include image processing and multimedia database.

Hitoshi Okada is an Associate Professor at the National Institute of Informatics (NII), Japan. He got his Bachelor of Public Law (LLB) and Bachelor of Private Law (LLB) from the University of Tokyo, Japan. He completed his MA at the Osaka School of International Public Policy (OSIPP) at Osaka University, Japan. He got his Ph.D. in International Public Policy from Osaka University, Japan.

Collecting Thai Unknown Words from the Web,” Proc. COLING/ACL Main Conference Poster Sessions, pp.345–352, 2006.

K. Trakultaweekoon, W.N. Chai, and T. Supnithi, “An Expertise-based Vocabulary Suggestion System: A Case Study of LEXITRON,” National Electronics and Computer Technology Center, Thailand, 2009.

A. Kawtrakul and C. Thumkanon, “A Statistical Approach to Thai Morphological Analyzer,” The 5th Workshop on Very Large Corpora, pp.289–296, 1997.

T. Theeramunpong, V. Sornlertlamvanich, T. Tanhermhong, and W. Chimnan, “Character-Cluster Based Thai Information Retrieval,” Proc. 5th Intl. Workshop on Information Retrieval with Asian Languages, pp.75–80, 2000.

P. Limcharoen, C. Nattee, and T. Theeramunpong, “Thai Word Segmentation based-on GLR Parsing Technique and Word N-gram Model,” The 8th intl. symposium on Natural Language Processing, 2009.

C. Kruengkrai, K. Uchimoto, J. Kazama, K. Torisawa, H. Isahara, and C. Jaruskulchai, “A Word and Character-Cluster Hybrid Model for Thai Word Segmentation,” Proc. InterBEST 2009 Thai Word Segmentation Workshop, pp.24–29, 2009.

P. Bangcharoensap, P. Porkaew, and T. Supnithi, “A Statistical-Machine-Translation Approach to Word Boundary Identification: A Projective Analogy of Bilingual Translation,” National Science and Technology Development Agency, Thailand, 2009.

F. Peng, F. Feng, and A. McCallum, “Chinese segmentation and new word detection using conditional random fields,” Proc. 20th intl. conf. on Computational Linguistics, pp.562–568, 2004.

N. Zhou, A. AitT, N. Lertcheva, and W. Xuangcong, “A Word Labelling Approach to Thai Sentence Boundary Detection and POS Tagging,” Proc. 26th intl. conf. on Computational Linguistics: Technical Papers, pp.319–327, 2016.

A. McCallum and W. Li, “Early Results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-Enhanced Lexicons,” Proc. 7th conf. on Natural Language Learning, pp.188–191, 2003.

K. Suesatpanit, P. Punyabukkana, and A. Suchato, “Thai Word Segmentation using Character-level Information,” Proc. InterBEST 2009 Thai Word Segmentation Workshop, pp.18–23, 2009.

H. Zhao, C.N. Huang, and M. Li, “An Improved Chinese Word Segmentation System with Conditional Random Field,” Proc. 5th SIGIHAN Workshop on Chinese Language Processing, pp.162–165, 2006.

A. Aw, S.M. Aljunied, N. Lertcheva, and S. Kalunsima, “TaLAPi – A Thai Linguistically Annotated Corpus for Language Processing,” LREC, pp.125–132, 2014.

J. Lafferty, A. McCallum, and F. Pereira, “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data,” Proc. 18th intl. conf. on Machine Learning, pp.282–289, 2001.

V. Sornlertlamvanich, N. Takahashi, and H. Isahara, “Building a Thai part-of-speech tagged corpus (ORCHID),” J. the Acoustical Society of Japan, vol.20, no.3, pp.189–198, 1999.

D.M. Blei, A.Y. Ng, and M.I. Jordan, “Latent dirichlet allocation,” J. Machine Learning Research, 3, pp.993–1022, 2003.

Y.W. Teh, M.I. Jordan, M.J. Beal, and D.M. Blei, “Hierarchical Dirichlet Processes,” J. the American Statistical Association, vol.101, no.476, pp.1566–1581, 2006.

Y.W. Teh and M.I. Jordan, “Hierarchical Bayesian Nonparametric Models with Applications,” Bayesian Nonparametrics, Cambridge University Press, 2010.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” Advances in Neural Information Processing Systems, pp.5998–6008, 2017.

C.Y. Lin, “ROUGE: A Package for Automatic Evaluation of Summaries,” Text summarization branches out: Proc. ACL-04 workshop, pp.74–81, 2004.

R. Mihalcea and P. Tarau, “TextRank: Bringing Order into Texts,” Proc. on Empirical Methods in Natural Language Processing, pp.404–411, 2004.