Character-based Thai Word Segmentation with Multiple Attentions

Thodsaporn Chay-intr, Hidetaka Kamigaito, Manabu Okumura
Institute of Innovative Research, Tokyo Institute of Technology
{chayintr@lr., kamigaito@lr., oku@}pi.titech.ac.jp

Abstract

Character-based word-segmentation models have been extensively applied to agglutinative languages, including Thai, due to their high performance. These models estimate word boundaries from a character sequence. However, a character unit in sequences has no essential meaning, compared with word, subword, and character cluster units. We propose a Thai word-segmentation model that uses various types of information, including words, subwords, and character clusters, from a character sequence. Our model applies multiple attentions to refine segmentation inferences by estimating the significant relationships among characters and various unit types. The experimental results indicate that our model can outperform other state-of-the-art Thai word-segmentation models.

1 Introduction

Thai running text has neither essential word delimiters nor sentence periods. However, spaces are arbitrarily allowed to separate words, phrases, clauses, and sentences. These characteristics make word segmentation in Thai more difficult than in other languages, such as English, German, and Finnish, which have spaces and periods to identify word and sentence boundaries. Thai word segmentation can be categorized as a sequential-labeling task that assigns a word-boundary label to each character in a fine-grained tagging scheme such as BIES (beginning, inside, end, and singleton) (Xue, 2003), as shown in Figure 1.

ม	ี	๓	ค	ว	า	ม	ห	ม	า	ย
B	E	S	B	I	I	I	I	I	I	E

มี ๓ ความหมาย

"There are three meanings."

Figure 1: Thai word segmentation as sequence-labeling task on BIES (beginning, inside, end, and singleton) tagging scheme

Neural-network models have been applied and perform well on character-based Thai word segmentation. Jousimo et al. (2017) applied bidirectional recurrent neural networks with gated recurrent units, while Chormai et al. (2019) proposed an AttaCut, a convolutional-neural-network (CNN)-based model that mainly provides faster and more accurate word inferring motivated by DeepCut (Kittinaradorn et al., 2019).

Despite the fact that neural-network models with linguistic knowledge can perform almost perfectly, adapting models with pre-trained neural networks, such as word vectors and language models, is still useful (Shao et al., 2018). Seeha et al. (2020) proposed a transfer-learning approach for Thai word segmentation by using a pre-trained character language model for character-based word segmentation. Although it exhibited state-of-the-art performance regarding Thai, it merely uses characters and does not use other information such as word and subword (Sennrich et al., 2016; Kudo, 2018). Additional linguistic knowledge, such as Thai character clusters (CCs) (Theeramunkong et al., 2000) and subword units, has also been successfully used for word segmentation and related tasks (Sutantayawalee et al., 2014; Lapjaturapit et al., 2018; Nararatwong et al., 2018; Yang et al., 2019; Li et al., 2019). An attention mechanism (Bahdanau et al., 2015) has been successfully applied to various downstream tasks, particularly a sequence-labeling task (Higashiyama et al., 2019; Tian et al., 2020).

We propose a character-based Thai word-segmentation model with multiple attentions that jointly uses corresponding words and CCs. Our model is based on the bidirectional long short-term

Proceedings of Recent Advances in Natural Language Processing, pages 264–273
Sep 1–3, 2021.
https://doi.org/10.26615/978-954-452-072-4_031

memory with conditional random field (BiLSTM-CRF) architecture, which is the baseline model for this study, because it has been successfully applied in sequence-labeling tasks for Thai and other languages (Jousimo et al., 2017; Nararatwong et al., 2018; Higashiyama et al., 2019; Seeha et al., 2020; Tian et al., 2020). Our contributions are as follows:

- We use word, subword units, and CCs with multiple attentions to estimate the relationships of characters in character-based Thai word segmentation.
- Our model outperforms the state-of-the-art models in Thai word segmentation, showing the validity of using CCs over subword units.
- Our code will be made publicly available.[1]

2 Background and Related Work

2.1 Thai Word Segmentation Revisited

In the early stage of Thai word segmentation, dictionary-based learning techniques had been used along with machine-learning techniques, for instance, Markov models (Kawtrakul and Thumkanon, 1997), decision trees (Sornlertlamvanich et al., 2000; Theeramunkong and Usanavasin, 2001), and CRFs (Haruechaiyasak et al., 2008). CRFs have been shown to be particularly suitable for Thai sequence-labeling tasks (Kruengkrai et al., 2006; Haruechaiyasak and Kongyoung, 2009; Kruengkrai et al., 2009; Nararatwong et al., 2018).

In parallel with CRFs, neural-network models, e.g., CNNs (Kittinaradorn et al., 2019; Chormai et al., 2019), LSTM (Treeratpituk, 2017), and BiLSTM (Jousimo et al., 2017), have been applied and performed excellently for character-based Thai word segmentation. Using additional knowledge, such as CC (Lapjaturapit et al., 2018; Nararatwong et al., 2018), transfer learning (Seeha et al., 2020), and stacking ensemble (Limkonchotiwat et al., 2020), along with neural-network models could improve performance.

2.2 Character Clusters in Thai Word Segmentation

Compared with English, the Thai language has various types of characters, i.e., consonants, vowels, tones, and special characters. A word can be formed from a combination of these characters.

[1] https://github.com/tchayintr/thwcc-attn

S	มี ๓ ความหมาย									
W	มี	๓	ความหมาย							
Sub	มี	๓	ความ				หมาย			
CC	มี	๓	ค	วา		ม	ห	มา		ย
C	ม	ี	๓	ค	ว	า	ม	ห	ม	า	ย

มี ๓ ความหมาย

"There are three meanings."

Figure 2: Comparison of zero-shot segmentation results. S, W, Sub, CC, and C indicate segmentation levels of sentence, word, subword, character cluster, and character, respectively.

Thai also has unique linguistic phenomena, for example, some sequential characters tend to be indivisible units. Thus, Theeramunkong et al. (2000) introduced the concept of a CC, which is a set of predefined rules for an indivisible unit on the basis of the Thai writing system.

A CC is smaller than a word but larger than a character. This concept is roughly comparable to a subword unit that is also in the middle of a character and word in terms of length. Using subword units, as well as CCs (Theeramunkong and Tanhermhong, 2004; Sutantayawalee et al., 2014; Lapjaturapit et al., 2018; Nararatwong et al., 2018), in word-segmentation tasks could yield good segmentation performance (Yang et al., 2019; Li et al., 2019). However, decomposing subword units from words requires appropriate parameters and training data, while CCs provide a greater advantage by not requiring any additional parameters. A CC helps avoid segmenting that might violate a writing system (Limcharoen et al., 2009), while subword units will likely not thoroughly exploit morphology (Provilkov et al., 2020), which could generate noise and weaken segmentation performance. This generally makes CCs smaller than subword units, as shown in Figure 2, which enables a comparison of zero-shot segmentation from coarse to fine (top-down) information.

2.3 Attention Mechanism

An attention mechanism was initially proposed by Bahdanau et al. (2015) for neural machine translation focusing on proper parts in sentences, particularly long sentences. It has been successfully applied to downstream tasks, including machine translation (Luong et al., 2015; Vaswani et al., 2017), constituency parsing (Kitaev and Klein, 2018), and sequence labeling (Higashiyama et al., 2019).

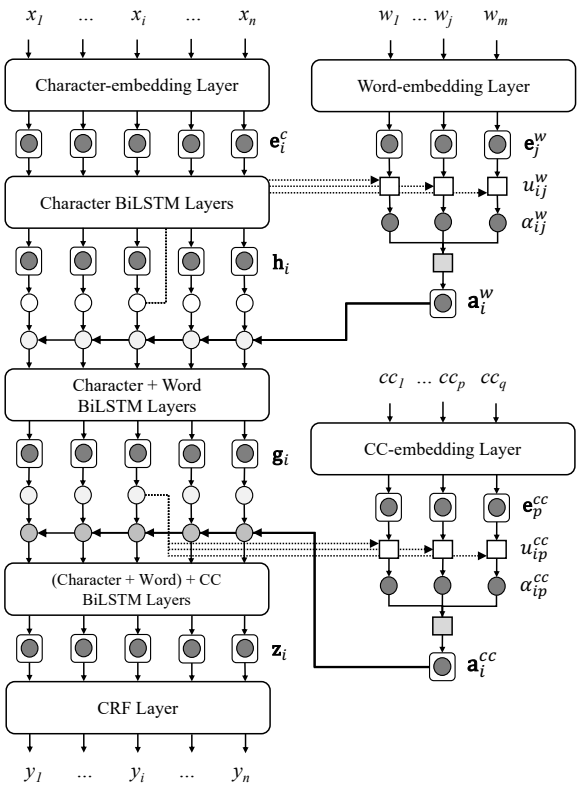

Figure 3: Our model integrating word and CC attentions into character-based BiLSTM-CRF architecture.

3 Proposed Model

Incorporating candidate words with an attention mechanism into the character-based BiLSTM-CRF architecture could yield superb segmentation performance (Higashiyama et al., 2019). An attention mechanism has the advantage of being flexible for use with additional linguistic knowledge such as CCs and subword units. Thus, we use the concept of CC with an attention mechanism in character-based word segmentation, as shown in Figure 3, by extending the BiLSTM-CRF architecture with word attention from Higashiyama et al. (2019).

Our model estimates *CC-integrated character vectors* $\mathbf{z}$ are incorporated on top of *word-integrated character vectors* $\mathbf{g}$, which are almost identical in architecture. We discuss the major components of our model, i.e., the character-embedding layer, word- and CC-embedding layers, BiLSTM layers for character representation, attention integrations with the BiLSTM layers for integrated representations, and CRF layer.

3.1 Character-embedding Layer

Given a sentence s with n characters that can be represented as $x_{1:n} \equiv (x_1, x_2, \ldots, x_n)$, each character $x_i \in x_{1:n}$ is transformed into a character embedding $\mathbf{e}^c_i$ of a d_c-dimensional vector (Bengio et al., 2003; Collobert et al., 2011) using lookup-table operation. The lookup table is defined as $\mathrm{E}^c \in \mathbb{R}^{d_c \times |\mathcal{V}_c|}$, where d_c denotes the dimension of embeddings and $\mathcal{V}_c$ denotes a character vocabulary.

3.2 Word- and CC-embedding Layers

Using the word embedding layer as an example, let V_w be a word vocabulary.

Given the character sequence $x_{1:n}$, words are searched on the basis of V_w within a maximum word length K of the character subsequence. A candidate word list $\mathcal{W}_x \equiv (w_1, \ldots, w_m)$ of size K with m candidate words is then obtained, as shown in Figure 3. Each word $w_j \in \mathcal{W}_x \subseteq V_w$ is transformed into a word embedding $\mathbf{e}^w$ of a d_w-dimensional vector. The word-embedding matrix is defined as $\mathrm{E}^w \in \mathbb{R}^{d_w \times |\mathcal{V}_w|}$, where d_w denotes the dimension of embeddings. This procedure is also applied to obtain a candidate CC list $\mathcal{CC}_x$, which is transformed into a CC-embedding layer $\mathbf{e}^{cc}$ of a d_{cc}-dimensional vector. The CC-embedding matrix is defined as $\mathrm{E}^{cc} \in \mathbb{R}^{d_{cc} \times |\mathcal{V}_{cc}|}$, where d_{cc} denotes the dimension of embeddings and V_{cc} denotes a CC vocabulary.

3.3 BiLSTM Layers for Character Representation

The character embedding sequence $\mathbf{e}^c_{1:n}$ is provided to the BiLSTM (Hochreiter and Schmidhuber, 1997; Gers et al., 2000) layers to contextually acquire *character context vectors* $\mathbf{h}_{1:n}$.

A current character context vector $\mathbf{h}^l_i \in \mathbf{h}^l_{1:n}$ of the l-th layer BiLSTM can be computed bidirectionally:

$$\begin{aligned}\mathbf{h}^l_i &= \mathrm{BiLSTM}(\mathbf{h}^{l-1}_{1:n}, i) \\ &\equiv \mathrm{LSTM}_f(\mathbf{h}^{l-1}_{1:n}, i) \\ &\oplus \mathrm{LSTM}_b(\mathbf{h}^{l-1}_{n:1}, n-i-1),\end{aligned} \tag{1}$$

where $\mathbf{h}^0_{1:n} = \mathbf{e}^c_{1:n}$, LSTM_f denotes forward LSTM, LSTM_b denotes backward LSTM, $\oplus$ denotes concatenation, and $\mathbf{h} \in \mathbb{R}^{2d_r}$ and d_r are hyperparameters.

3.4 Attention Integrations with BiLSTM Layers for Integrated Representations

We use two attention integrations, including word attention and CC attention, to respectively estimate a *word-integrated summary vector* $\mathbf{a}^w_i$ and

CC-integrated summary vector $\mathbf{a}_i^{cc}$ for each character in the character sequence. These integrations, which are equal in architecture, accordingly summarize the relationship among characters, words, and CCs.

We apply the composition function *weight concatenation* (WCON) (Higashiyama et al., 2019) to estimate both summary vectors. This function produces a word-integrated summary vector on the basis of the relationship between a character and their corresponding candidate words. It also can be used to implicitly produce a CC-integrated summary vector on the basis of the relationship between the character with its corresponding candidates words and candidate CCs.

Starting with word-attention integration, we estimate the word-importance score u_{ij}^w and word-attention weight α_{ij}^w on the basis of the character context vector $\mathbf{h}_i$ and candidate word embedding $\mathbf{e}_j^w$ as

$$u_{ij}^w = \mathbf{h}_i^T W_a^w \mathbf{e}_j^w, \tag{2}$$

$$\alpha_{ij}^w = \frac{\delta_{ij} \exp(u_{ij}^w)}{\sum_{k=1}^m \delta_{ik} \exp(u_{ik}^w)}, \tag{3}$$

where $W_a^w \in \mathbb{R}^{2d_r \times d_w}$ denotes a trainable weight matrix and $\delta_{ij} \in \{0, 1\}$ indicates whether character x_i is included in candidate word w_j. The word-integrated summary vector $\mathbf{a}_i^w$ for character x_i can be calculated as

$$\mathbf{a}_i^w = \text{WCON}^w(x_i, \{w_j\}_{j=1}^m) = \bigoplus_{l=1}^{L^w} \alpha_{i,i_l}^w \mathbf{e}_{i_l}^w, \tag{4}$$

where $\{w_j\} = \mathcal{W}_x$. Let K^w be the maximum word length, $L^w = \sum_{k=1}^{K^w} k$, $\bigoplus$ denote concatenation, and i_l is the corresponding index of candidate word list $\mathcal{W}_x$ for character x_i that is $\{w'_1, \ldots, w'_{L^w}\} \equiv \bigcup_{k=1}^{K^w} \bigcup_{s=-k+1}^{0} \{x_{i+s:i+s+k-1}\}$. A zero vector is applied to Equation 4 when $w'_l \notin \mathcal{V}_w$.

We then use the BiLSTM layers for transforming the word-integrated summary vectors $\mathbf{a}^w$ into word-integrated character vectors $\mathbf{g}$. The operation of word-integrated character vector $\mathbf{g}_i$ is computed using the BiLSTM layers on the basis of word-integrated summary vector $\mathbf{a}_i^w$ with its corresponding character context vector $\mathbf{h}_i$ as

$$\mathbf{g}_i = \text{BiLSTM}(\mathbf{h}_i \oplus \mathbf{a}_i^w). \tag{5}$$

However, the candidate CCs that correspond to the character are used on top of $\mathbf{g}_i$ as

$$u_{ip}^{cc} = \mathbf{g}_i^T W_a^{cc} \mathbf{e}_p^{cc}, \tag{6}$$

$$\alpha_{ip}^{cc} = \frac{\delta_{ip} \exp(u_{ip}^{cc})}{\sum_{k=1}^q \delta_{ik} \exp(u_{ik}^{cc})}, \tag{7}$$

where $W_a^{cc} \in \mathbb{R}^{4d_r \times d_{cc}}$ denotes a trainable weight matrix and $\delta_{ip} \in \{0, 1\}$ indicates whether character x_i is included in the candidate CC cc_p. The CC-integrated summary vector $\mathbf{a}_i^{cc}$ for character x_i can be calculated as

$$\mathbf{a}_i^{cc} = \text{WCON}^{cc}(x_i, \{cc_p\}_{p=1}^q) = \bigoplus_{l=1}^{L^{cc}} \alpha_{i,i_l}^{cc} \mathbf{e}_{i_l}^{cc}, \tag{8}$$

where $\{cc_p\} = \mathcal{CC}_x$. Let K^{cc} is the maximum CC length, $L^{cc} = \sum_{k=1}^{K^{cc}} k$, and i_l is the corresponding index of potential CC list $\mathcal{CC}_x$ for character x_i, i.e., $\{cc'_1, \ldots, cc'_{L^{cc}}\} \equiv \bigcup_{k=1}^{K^{cc}} \bigcup_{s=-k+1}^{0} \{x_{i+s:i+s+k-1}\}$. A zero vector is applied to Equation 8 when $cc'_l \notin \mathcal{V}_{cc}$.

Next, we use additional BiLSTM layers to transform the CC-integrated summary vectors $\mathbf{a}^{cc}$ into CC-integrated character vectors $\mathbf{z}$ on the basis of a cluster-integrated summary vector $\mathbf{a}_i^{cc}$ and its corresponding word-integrated character vector $\mathbf{g}_i$ as

$$\mathbf{z}_i = \text{BiLSTM}(\mathbf{g}_i \oplus \mathbf{a}_i^{cc}). \tag{9}$$

A CRF is finally used to estimate the probability of the optimal label sequences y.

3.5 CRF Layer

A CRF (Lafferty et al., 2001) along with explicitly considering the correlations between adjacent labels has been successfully applied for sequence-labeling-related tasks (Collobert et al., 2011). Let $A \in \mathbb{R}^{|T| \times |T|}$ be a transition matrix for correlations between adjacent labels, where T denotes a set of all possible label sequences, for instance, $T = \{B, I, E, S\}$. The CC-integrated character vector $\mathbf{z}_i$ is transformed into an un-normalized label score $\mathbf{s}_i$ of the $|T|$-dimensional vector for character x_i as

$$\mathbf{s}_i = W_s \mathbf{z}_i + \mathbf{b}_s, \tag{10}$$

where $W_s \in \mathbb{R}^{|T| \times 4d_r}$ denotes a trainable weight matrix, and $\mathbf{b}_s \in \mathbb{R}^{|T|}$ denotes a trainable bias. Given the input sequence $x_{1:n}$, the corresponding scores for the label sequence $y_{1:n}$ are computed on the basis of transition matrix A and the segmentation label scores $\mathbf{s}$ as follows:

$$\text{score}(x, y) = \sum_{i=1}^{n} (A_{y_{i-1}, y_i} + \mathbf{s}_i[y_i]) \tag{11}$$

Domain	|S|	|W|	|V|	|Ch|
Article	16732	1018907	25658	4178690
Ency	50629	1041067	26466	4231447
News	31225	1448604	37076	6174642
Novel	50134	1522258	22062	5428205
Total	148720	5030836	81688	20012984

Table 1: Data lengths of BEST2010 corpus, including length of sentences (S), words (W), vocabulary (V), and characters (Ch). "Ency" denotes encyclopedia.

The probability of the label sequence can then be obtained as

$$P(y|x) = \frac{\text{score}(x, y)}{\sum_{y' \in T^n} \text{score}(x, y')}, \quad (12)$$

We can obtain the optimal label sequence $y^\star$ by maximizing the sentence score with the Viterbi algorithm:

$$y^\star = \text{argmax}_{y \in T^n} \text{score}(x, y) \quad (13)$$

The loss function $\mathcal{L}$ is minimized by back propagation during the training process:

$$\mathcal{L}(x, y) = -\log P(y|x) \quad (14)$$

4 Experiments

4.1 Dataset

We trained and evaluated several versions of our model on the BEST2010 corpus[2], which is the most well-known Thai word-segmented corpus. It contains 5 million words with 20 million characters. The categories and their data lengths are listed in Table 1. We randomly split this corpus into three sets[3]: 80% for a training, 10% for a validation, and 10% for a test.

4.2 Subword-unit Integration

Although subword units have been successfully applied to word-segmentation tasks, they might generate noise that decreases segmentation performance when the word dictionary already exists. Therefore, we conducted a comparison of using either subword units or CCs due to their similarity on the different versions of our proposed model.

[2] https://thailang.nectec.or.th

[3] https://resources.aiat.or.th/thwcc-attn/datasets

Let V_{sw} be a subword vocabulary decomposed from the dataset. We simply replace the CC vocabulary V_{cc} with the decomposed subword vocabulary V_{sw}. Thus, a candidate subwords list can be acquired and used for the subword-attention integration by applying Equations 6 and 8.

4.3 Integration Order

Considering the flexibility of attention integration, the integration order in our model can be switched. For instance, our model executes CC-attention integration to estimate the relationship between characters and CCs before word-attention integration. This might affect segmentation performance because each integration provides different knowledge. Thus, we implemented a swapped version of our model (Swap) that switches the integration order for comparing the segmentation performance.

4.4 Compared Models

We evaluated the following models:

- **Baseline** A character-based BiLSTM-CRF architecture.
- **Baseline w/ Word** An extension of Baseline that integrates word attention (BiLSTM-CRF with word attention) (Higashiyama et al., 2019).
- **OURS** Our proposed model that integrates word and CC attentions (BiLSTM-CRF with word and CC attentions), as shown in Figure 3.
- **OURS w/o Word** Our proposed model that removes word attention (BiLSTM-CRF with CC attention).
- **OURS w/o CC w/ Sub** Our proposed model that replaces the CC with various sizes of subword units (800-12,800) (BiLSTM-CRF with word and subword attentions).
- **OURS Swap** Our proposed model that swaps the order of word and CC attentions.
- **OURS w/o CC w/ Sub Swap** OURS w/o CC w/ Sub model that swaps the order of word and subword unit attentions.
- **Others** Reproduced Thai word-segmentation models, including well-known models and the state-of-the-art Thai word-segmentation model (Seeha et al., 2020).

We used CCs with publicly available libraries, including Phatthiyaphaibun et al. (2016) and TCC-SEG[4], to build the CC vocabulary V_{cc}. To generate the subword vocabulary V_{sw}, we decomposed raw sentences from the dataset into various sizes of subword units using byte-pair encoding (Sennrich et al., 2016) implemented by SentencePiece (Kudo and Richardson, 2018).

We used the common hyperparameters for training the different versions of our proposed model (hereafter, our models), as shown in Table 2 Dropout (Srivastava et al., 2014) was applied to the BiLSTM layers to avoid overfitting as well as non-recurrent layers (Zaremba et al., 2015). We also optimized the model parameters using the Adam optimizer (Kingma and Ba, 2015). We trained our models up to 20 epochs and chose the best one on the basis of the validation process involving the CoNLL[5] evaluation.

We evaluated our models on the test data by using three evaluation metrics, i.e., CoNLL (word-level evaluation), BIES tagging scheme (character-level evaluation), and Bound (boundary-level evaluation) (Seeha et al., 2020). Although the CoNLL and BIES tagging schemes are often used for evaluating the performance of sequence-labeling tasks, the boundary-level evaluation has been used in Thai word-segmentation evaluations. Thus, we also used the boundary-level evaluation in our experiments. Note that our F_1 scores are based on the micro-averaged F_1 score for all evaluation matrices. We conducted a statistical significance test using paired bootstrap resampling (Koehn, 2004) on our results. We set the resampling size to 100,000 iterations and sample size for each resampling to 10% of the test data.

4.5 Main Results[6]

Table 3 illustrates the evaluation results among the compared models.

The best of our models, i.e., OURS, achieved the state-of-the-art performance compared with all the other models. From the statistical significance test results, we concluded that OURS surpasses the state-of-the-art model. However, using subword units could slightly improve its performance on average compared with using CCs. This indicates that CCs are more beneficial than subword units, which might generate further noise. OURS, which uses word and CC information, outperformed Baseline w/ Word, which indicates that CCs can be used to complement linguistic knowledge, particularly word information.

Parameter	Value
Character-embedding size	128
BiLSTM layers	2
BiLSTM hidden size	600
Mini-batch size	128
Initial learning rate	0.001
Recurrent layer dropout rate	0.4
Word-embedding size	300
Word-vector dropout rate	0.4
Maximum word (chunk) length	4
CC/subword-embedding size	300
CC/subword-vector dropout rate	0.4
Maximum CC/subword length	4

Table 2: Common hyperparameters for Baselines and our models (top/middle) with exclusive values for our models (bottom). CC hyperparameters can be applied to subword integration.

4.6 Analysis

Subword-integration Performance: We implemented OURS w/o CC w/ Sub on various vocabulary sizes of subword units, as shown in Table 4. The results indicate that the vocabulary size clearly affects subword-integration performance improvement. Specifically, by providing more subword vocabulary to the model, we could consistently increase the overall performance. It may reach the performance of OURS when the subword vocabulary size is enormous. However, it might be difficult to appropriately determine the vocabulary size that will benefit the model. For instance, OURS w/o CC w/Sub with 12,800 subword tokens (OURS w/o CC w/ Sub12800) failed to improve in performance compared with those with 3,200 and 6,400 subword tokens. Thus, the size of subword vocabulary is a crucial parameter that affects the performance for this model.

We chose the best subword-integration model on the basis of validation performance to compare it with OURS, as shown in Table 3. Both subword and CC integrations tended to act as an additional filter layer for word-integrated character

[4] https://github.com/tchayintr/tccseg

[5] https://github.com/spyysalo/conlleval.py

[6] We implemented an additional model that replaces the BiLSTM layers with Transformer layers (Vaswani et al., 2017) in Baseline. However, the results were noticeably lower than all other models. Note that we used the hyperparameters for the Transformer layers by referring to Vaswani et al. (2017).

Model	CoNLL	Bound	BIES
(Treeratpituk, 2017)°	92.49	97.94	96.53
(Chormai et al., 2019)°	93.79	98.36	91.36
(Kittinaradorn et al., 2019)°	95.82	98.87	98.17
(Lapjaturapit et al., 2018)°	96.22	99.03	98.43
(Seeha et al., 2020)°	97.20	99.27	98.80
Baseline	96.78	99.16	98.29
Baseline w/ Word	97.57	99.35	98.94
OURS	**97.67**	**99.38**	**98.99**
OURS w/o Word	97.41	99.32	98.91
OURS w/o CC w/ Sub	97.65	99.37	98.98
OURS Swap	97.60	99.36	98.98
OURS w/o CC w/ Sub Swap	97.47	99.33	98.89

Table 3: Comparison among our models, baselines, and Others. Best score for each metric is indicated in **bold**. Our models were significantly better than state-of-the-art model Thai word segmentation model (underline scores) at p-level < 0.01 in pairwise comparison. All models were evaluated on basis of same dataset division. Scores were obtained from mean of two runs. OURS w/o CC w/ Sub scores were reported on the basis of best validation performance among various subword vocabulary sizes. ○ indicates reproduced Thai word-segmentation models

Model	CoNLL	Bound	BIES
OURS w/o CC w/ Sub800	97.56	99.35	98.94
OURS w/o CC w/ Sub1600	97.59	99.36	98.96
OURS w/o CC w/ Sub3200	97.65	99.37	98.95
OURS w/o CC w/ Sub6400	97.65	99.37	98.98
OURS w/o CC w/ Sub12800	97.65	99.37	98.95

Table 4: Results of our subword-integration model (OURS w/o CC w/ Sub) with various subword-vocabulary sizes from 800 to 12,800 tokens. underline indicates model that obtained best score in validation process.

representations and improve segmentation performance. However, OURS outperformed OURS w/o CC w/ Sub for every evaluation matrix. We think the main reason is that subword units contain noise while CCs do not. For example, the unit "าว", which is included in the subword vocabulary, does not exist in Thai word vocabulary and violates the Thai writing system, whereas CCs will not include this type of unit.

Order-of-integration Performance: We compared the performance of our model when the order of attention integrations are swapped.

Table 3 shows that the swapped models decrease segmentation performance compared with their original models, especially the swapped subword-integration model. We argue that subword integration initially adds noise to the character representations, e.g., a subword unit that does not exist

Reference	ป๊อ อุ้ยอ้าย ยก ลัง เข้า ไป เก็บ ไว้ ใน ตู้ เสื้อผ้า
Baseline	ป๊อ อุ้ย อ้ายย กลัง เข้า ไป เก็บ ไว้ ใน ตู้ เสื้อผ้า
Baseline w/ Word	ป๊ออุ้ยอ้ายย กลัง เข้า ไป เก็บ ไว้ ใน ตู้ เสื้อผ้า
Our	ป๊อ อุ้ยอ้าย ยก ลัง เข้า ไป เก็บ ไว้ ใน ตู้ เสื้อผ้า
Our w/o CC w/ Sub	ป๊อ อุ้ยอ้ายย กลัง เข้า ไป เก็บ ไว้ ใน ตู้ เสื้อผ้า

ป๊ออุ้ยอ้ายยกลังเข้าไปเก็บไว้ในตู้เสื้อผ้า

"Por clumsily put a crate into a closet."

Figure 4: Examples of segmentation results among baseline models and our models. Ground-truth segmentation result is indicated as "Reference" and incorrect segmentation results are in red.

in Thai word vocabulary. Therefore, it is difficult for the model to complement such representations in the word-attention integration afterwards. The swapped CC-integration model slightly decreased in performance compared with subword integration because CC vocabulary consists of smaller units that reflect the Thai writing system and includes no noise information. This indicates that OURS outperformed both swapped models and the word information is the priority knowledge to complement a character representation, whereas fine-grained information, i.e., CCs and subword units, are suitable for use after word information as an additional filter layer.

Case Study: Figure 4 shows examples of segmentation results among four models, i.e., Baseline, Baseline /w Word, OURS, and OURS w/o CC w/ Sub. OURS could perfectly segment the example sentence; however, the other models yielded incorrect results. Specifically, the word "อุ้ยอ้ายย" violates the Thai writing system by combining the two consonants "ยย" in the word. We think that CC integration filters this type of violation out of the word-integrated character representations, enabling OURS to outperform the other models.

5 Conclusion

We proposed a character-based Thai word-segmentation model that uses multiple attentions on various types of linguistic knowledge, i.e., words, subwords, and CCs. The best version of our model achieved Thai state-of-the-art performance by using the word attention along with CC attention in the BiLSTM-CRF architecture. Further analysis also indicates that using CC can be more beneficial than using subword units in word-segmentation tasks.

References

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *In Proceedings of the 3rd International Conference on Learning Representations*.

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

Pattarawat Chormai, Ponrawee Prasertsom, and Attapol T. Rutherford. 2019. Attacut : A fast and accurate neural thai word segmenter.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Felix Gers, Jürgen Schmidhuber, and Fred Cummins. 2000. Learning to forget: Continual prediction with lstm. *Neural Computation*, 12(10):2451–2471.

Choochart Haruechaiyasak and Sarawoot Kongyoung. 2009. Tlex: Thai lexeme analyser based on the conditional random fields. In *Proceedings of International Joint Symposium on Artificial Intelligence and Natural Language Processing 2009*, pages 13–17.

Choochart Haruechaiyasak, Sarawoot Kongyoung, and Matthew Dailey. 2008. A comparative study on thai word segmentation approaches. In *Proceedings of the 5th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, pages 125–128.

Shohei Higashiyama, Masao Utiyama, Eiichiro Sumita, Masao Ideuchi, Yoshiaki Oida, Yohei Sakamoto, and Isaac Okada. 2019. Incorporating word attention into character-based word segmentation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2699–2709, Minneapolis, Minnesota. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Jussi Jousimo, Natsuda Laokulrat, Ben Carr, Ekkalak Thongthanomkul, and Vee Satayamas. 2017. Thai word segmentation with bi-directional rnn.

Asanee Kawtrakul and Chalatip Thumkanon. 1997. A statistical approach to thai morphological analyzer. In *Proceedings of of the 5th Workshop on Very Large Corpora*, pages 289–296.

Diederik P. Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic optimization. In *In Proceedings of the 3rd International Conference on Learning Representations*.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia. Association for Computational Linguistics.

Rakpong Kittinaradorn, Titipat Achakulvisut, Korakot Chaovavanich, Kittinan Srithaworn, Pattarawat Chormai, Chanwit Kaewkasi, Tulakan Ruangrong, and Krichkorn Oparad. 2019. Deepcut: A thai word tokenization library using deep neural network.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, volume 4, pages 388–395.

Canasai Kruengkrai, Virach Sornlertlamvanich, and Hitoshi Isahara. 2006. A conditional random field framework for thai morphological analysis. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, pages 2419–2424.

Canasai Kruengkrai, Kiyotaka Uchimoto, Kazama Jun'ichi, Kentaro Torisawa, Hitoshi Isahara, and Chuleerat Jaruskulchai. 2009. A word and character-cluster hybrid model for thai word segmentation. In *In Proceedings of InterBEST 2009 Thai Word Segmentation*.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

John Lafferty, Andrew Mccallum, and Fernando Pereira. 2001. Conditional random fields : Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289.

Theerapat Lapjaturapit, Kobkrit Viriyayudhakom, and Thanaruk Theeramunkong. 2018. Multi-candidate word segmentation using bi-directional lstm neural networks. In *Proceedings of 2018 International Conference on Embedded Systems and Intelligent*

Technology and International Conference on Information and Communication Technology for Embedded Systems, pages 30–35.

Xiaoya Li, Yuxian Meng, Xiaofei Sun, Qinghong Han, Arianna Yuan, and Jiwei Li. 2019. Is word segmentation necessary for deep learning of Chinese representations? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3242–3252, Florence, Italy. Association for Computational Linguistics.

Piya Limcharoen, Cholwich Nattee, and Thanaruk Theeramunkong. 2009. Thai word segmentation based-on glr parsing technique and word n-gram model. In *In Proceedings of the 8th International Symposium on Natural Language Processing*.

Peerat Limkonchotiwat, Wannaphong Phatthiyaphaibun, Raheem Sarwar, Ekapol Chuangsuwanich, and Sarana Nutanong. 2020. Domain adaptation of Thai word segmentation models using stacked ensemble. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3841–3847, Online. Association for Computational Linguistics.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Rungsiman Nararatwong, Natthawut Kertkeidkachorn, Nagul Cooharojananone, and Hitoshi Okada. 2018. Improving thai word and sentence segmentation using linguistic knowledge. *IEICE Transactions on Information and Systems*, E101D(12):3218–3225.

Wannaphong Phatthiyaphaibun, Korakot Chaovavanich, Charin Polpanumas, Arthit Suriyawongkul, Lalita Lowphansirikul, and Pattarawat Chormai. 2016. Pythainlp: Thai natural language processing in python.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2020. BPE-dropout: Simple and effective subword regularization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1882–1892, Online. Association for Computational Linguistics.

Suteera Seeha, Ivan Bilan, Liliana Mamani Sanchez, Johannes Huber, Michael Matuschek, and Hinrich Schütze. 2020. Thailmcut: Unsupervised pretraining for thai word segmentation. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 11–16.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Yan Shao, Christian Hardmeier, and Joakim Nivre. 2018. Universal word segmentation: Implementation and interpretation. *Transactions of the Association for Computational Linguistics*, 6(2002):421–435.

Virach Sornlertlamvanich, Tanapong Potipiti, and Thatsanee Charoenporn. 2000. Automatic corpus-based thai word extraction with the c4.5 learning algorithm. In *Proceedings of the 18th Conference on Computational Linguistics*, pages 802–807.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Salakhutdinov Ruslan. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Vipas Sutantayawalee, Peerachet Porkeaw, Thepchai Supnithi, Prachya Boonkwan, and Sitthaa Phaholphinyo. 2014. Character-cluster-based segmentation using monolingual and bilingual information for statistical machine translation. In *Proceedings of the 5th Workshop on South and Southeast Asian Natural Language Processing*, pages 94–101.

Thanaruk Theeramunkong, Virach Sornlertlamvanich, Thanasan Tanhermhong, and Wirat Chinnan. 2000. Character cluster based thai information retrieval. In *Proceedings of the Fifth International Workshop on Information Retrieval with Asian Languages*, pages 75–80.

Thanaruk Theeramunkong and Thanasan Tanhermhong. 2004. Pattern-based features vs. statistical-based features in decision trees for word segmentation. *IEICE Transactions on Information and Systems*, E87-D(5):1254–1260.

Thanaruk Theeramunkong and Sasipron Usanavasin. 2001. Non-dictionary-based thai word segmentation using decision trees. In *Proceedings of the 1st International Conference on Human Language Technology Research*, pages 251–256.

Yuanhe Tian, Yan Song, and Fei Xia. 2020. Joint Chinese word segmentation and part-of-speech tagging via multi-channel attention of character n-grams. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2073–2084, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Pucktada Treeratpituk. 2017. Thai word-segmentation with lstm in tensorflow.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *In Proceedings of the 31st Conference on Neural Information Processing Systems*.

Nianwen Xue. 2003. Chinese word segmentation as character tagging. *Computational Linguistics and Chinese Language Processing*, 8(1):29–48.

Jie Yang, Yue Zhang, and Shuailong Liang. 2019. Subword encoding in lattice LSTM for Chinese word segmentation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2720–2725, Minneapolis, Minnesota. Association for Computational Linguistics.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2015. Recurrent neural network regularization. In *In Proceedings of the 3rd International Conference on Learning Representations*.