General Paper Peer-Reviewed

LATTE: Lattice ATTentive Encoding for Character-based Word Segmentation

Thodsaporn Chay-intr[†], Hidetaka Kamigaito[††], Kotaro Funakoshi[†††] and Manabu Okumura[†††]

A character sequence comprises at least one or more segmentation alternatives. This can be considered segmentation ambiguity and may weaken segmentation performance in word segmentation. Proper handling of such ambiguity lessens ambiguous decisions on word boundaries. Previous works have achieved remarkable segmentation performance and alleviated the ambiguity problem by incorporating the lattice, owing to its ability to capture segmentation alternatives, along with graph-based and pre-trained models. However, multiple granularity information, including character and word, in a lattice that encodes with such models may not be attentively exploited. To strengthen multi-granularity representations in a lattice, we propose the **L**attice **ATT**entive **E**ncoding (**LATTE**) method for character-based word segmentation. Our model employs the lattice structure to handle segmentation alternatives and utilizes graph neural networks along with an attention mechanism to attentively extract multi-granularity representation from the lattice for complementing character representations. Our experimental results demonstrated improvements in segmentation performance on the BCCWJ, CTB6, and BEST2010 datasets in three languages, particularly Japanese, Chinese, and Thai.

Key Words: *Word Segmentation, Representation Learning*

1 Introduction

Word segmentation is a fundamental task for an understanding of natural language. The task is to determine word boundaries from a running text; in other words, it segments a character sequence into word units. Inadequate segmentation causes error propagation in consequential tasks, such as Named Entity Recognition (NER), part-of-speech (POS) tagging, and parsing (Qian and Liu 2012; Zhang and Yang 2018). This indicates the priority of word information.

Although a word unit is the most natural concept for word segmentation, word-based models suffer from significant issues such as ambiguity, data sparsity, and out-of-vocabulary (OOV) (Li

[†] School of Engineering, Tokyo Institute of Technology
[††] Division of Information Science, Nara Institute of Science and Technology
[†††] Institute of Innovative Research, Tokyo Institute of Technology

et al. 2019). Unlike word-based models, character-based models, which use a character unit as an essential feature, may mitigate these obstacles because word-internal structures are more focused, having a stronger word-induction ability, particularly for the induction of new words (Sun 2010). Character-based word segmentation can be categorized as a sequential labelling task. It aims to assign a word-boundary label to each character from a character sequence into a fine-grained tagging scheme, such as beginning, middle, end, and singleton (BMES), as shown in Figure 1.[1] Recent studies on Asian languages, including Japanese, Chinese, and Thai, have strongly relied on this character-based model (Higashiyama et al. 2019; Ke et al. 2021; Seeha et al. 2020).

Naturally, a character sequence consists of at least one segmentation alternative (Dyer et al. 2008). This ambiguity is considered a problem in word segmentation. Previous studies have successfully incorporated various types of linguistic units such as words and subwords (Sennrich et al. 2016), on top of the character sequence to alleviate the ambiguity problem (Higashiyama et al. 2019; Yang et al. 2019; Tian et al. 2020a; Chay-intr et al. 2021). In particular, Higashiyama et al. (2019), Tian et al. (2020a) applied the attention mechanism (Bahdanau et al. 2015) to extract context features associated with a character and its corresponding knowledge, including words and syntactic structures. These approaches are primarily based on a recurrent neural

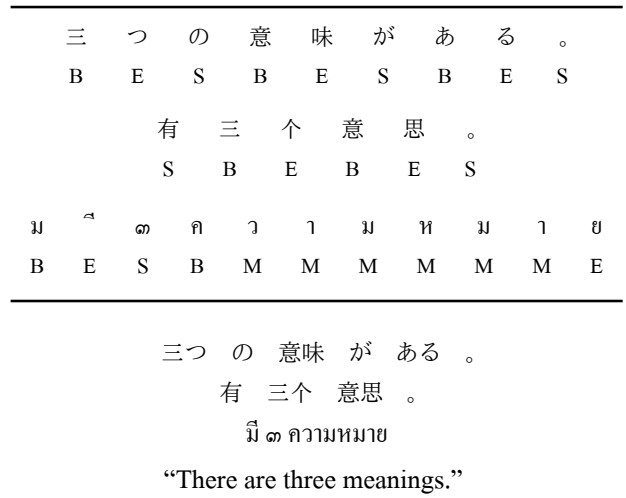


Figure 1 Word segmentation as a sequence-labeling task for Japanese (top), Chinese (middle), and Thai (bottom), respectively, on BMES tagging scheme.

[1] We used MeCab (`https://taku910.github.io/mecab`), Jieaba (`https://github.com/fxsjy/jieba`), and Deepcut (`https://github.com/rkcosmos/deepcut`) to produce segmentation results for Japanese, Chinese, and Thai, respectively.

network (RNN) architecture that handles character sequences consecutively; however, linguistics is not rigidly sequential (Shen et al. 2018). Consequently, segmentation alternatives that can be produced from character sequences based on such linguistic units are not implicitly exploited.

Lattice, a graph-based representation, is employed to capture an arbitrary number of segmentation alternatives[2] as shown in Figure 2; in particular, they represent ambiguous decisions on word boundaries in a multi-path lattice (Dyer et al. 2008).

Various types of neural network models, such as RNN-, Transformer-, and graph-models, have been successfully applied along with lattice structure in downstream tasks besides word segmentation, including NER, Machine Reading Comprehension (MRC), and text classification (Zhang and Yang 2018; Gui et al. 2019; Li et al. 2020; Lai et al. 2021).

However, using pre-trained models (PTMs) with diverse methods to fine-tune word segmentation may significantly advance segmentation performance owing to their ability to provide prior knowledge (He et al. 2017; Seeha et al. 2020; Ke et al. 2021; Maimaiti et al. 2021), particularly Ke et al. (2021) that introduces multi-criteria pre-training with meta-learning (Chen et al. 2017; Finn et al. 2017) to minimize discrepancies in pre-training tasks. However, the performance of PTMs in fine-tuning word segmentation is limited by the scale and quality of the annotated corpus (Huang et al. 2021). Moreover, multi-grained linguistic units, such as unseen character combinations and ambiguous contextual information from segmentation alternatives, may not be

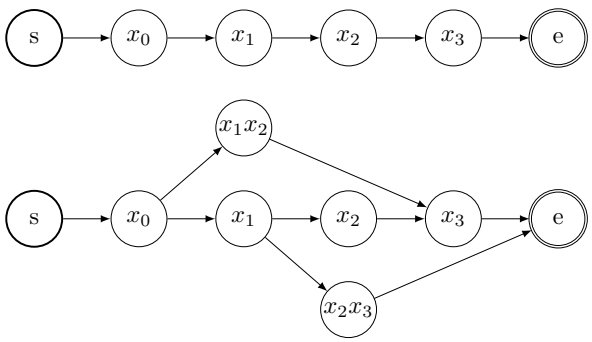


Figure 2 Examples of lattice structures: single-path lattice (top) and multi-path lattice (bottom), where x represents a character. **s** and **e** indicate initial and ending states, respectively. While the single-path lattice merely represents a character sequence $x_{0:3}$, the multi-path lattice additionally includes a group of possible character sequences (words), that is, x_1x_2 and x_2x_3 on top of the character sequence.

[2] In sequential labelling tasks, for example, word segmentation and NER, a node is conventionally used to represent either character or word information, and an edge is used to represent a transition between nodes.

effectively exploited in fine-tuning methods.

To handle such information from segmentation alternatives, graph neural networks (GNNs) are exploited for their ability to preserve relational and global structure information in a graph (Yao et al. 2018). They have been utilized to encode fine-grained lattice with additional multi-level linguistic features, including word-boundary nodes, n-grams, and dependency syntactic trees (Huang et al. 2021; Tang et al. 2022). Although their works performed well in segmentation, they exhibited similar results to Huang, Cheng et al. (2020), Ke et al. (2021). These empirical approaches rely on pre-training methods and almost a dozen datasets, likely owing to discrepancies between lattice and dataset segmentation alternatives. It indicates that an exponential number of segmentation alternatives in lattice encoded with GNN must be handled attentively.

In this study, we propose **L**attice **ATT**entive **E**ncoding (**LATTE**), a method to strengthen multi-granularity knowledge in the lattice that captures an exponential number of segmentation alternatives[3] for character-based word segmentation. In practice, we utilize lattice to capture segmentation alternatives from a character sequence and apply GNN simultaneously with the PTM BERT (Devlin et al. 2019) and an attention mechanism to enable neural models to attentively extract contextual features from lattice for complementing character representations.

After empirically conducting experiments towards extracting essential representations from the use of lattice, we can state our contributions as follows.

- We proposed a method to strengthen multi-granularity representations, including character and word, in lattice that captures an exponential number of segmentation alternatives for complementing character representations with an attention mechanism.
- Incorporating the bidirectional graph attention network (BiGAT) and attention mechanism into lattice, discrepancy in an arbitrary of segmentation alternatives captured in the lattice tends to decrease, leading to an improvement in the segmentation performance.
- Experimental results demonstrate that our model outperforms previous models on three datasets in three languages and yields superior performance compared with baseline methods in our analysis.
- Our code is publicly available.[4]

[3] We have conducted a preliminary experiment in Appendix A to verify the benefits of using segmentation alternatives. Our experiment indicates that applying several segmentation alternatives in word segmentation is useful.

[4] `https://github.com/tchayintr/latte-ws`

2 Related Work

In this section, we present an overview of prior studies analyzing sequence labeling tasks, with a special focus on word segmentation.

2.1 Lattice in Sequence Labelling Tasks

Lattice has been successfully incorporated in sequence labelling tasks such as word segmentation and NER owing to its ability to capture sequential paths. Different types of linguistic knowledge such as character, word, and subword, have been employed as a feature unit in lattice. Zhang and Yang (2018) proposed the Lattice LSTM for Chinese NER and obtained superior performance over character-based and word-based models by incorporating lattices to control information flow, including characters and words, from the beginning of the sentence to the end. Yang et al. (2019) successfully incorporated a subword unit into lattice for Chinese NER atop a character representation, using a model similar to that of Zhang and Yang (2018).

Li et al. (2020) introduced FLAT, a transformer-based model for Chinese NER that applies word-character lattice and its position information as a flat structure. Their model exhibited good performance and efficiency. As with various abilities of PTMs, lattice has also been employed with coarse-granularity knowledge for pre-training a model (Lai et al. 2021). Their approach yielded satisfying performance on essential downstream tasks such as text classification, MRC, and sequence labeling.

2.2 Graph Neural Networks in Word Segmentation

Recently, GNNs have been explored and applied along with graph structures for tasks downstream of natural language processing (NLP). Various GNN architectures have been proposed to address particular problems, such as graph convolutional network (GCN) (Kipf and Welling 2016), graph attention network (GAT) (Veličković et al. 2017), and heterogeneous graph attention network (HAN) (Wang et al. 2019).

To the best of our knowledge, only Huang et al. (2021), Tang et al. (2022) utilize GNN in word segmentation; however, most of the related studies focused on sequential labeling, particularly NER. Huang et al. (2021) used GCN to aggregate word-character node along with its additional nodes, i.e., boundary-label node. Their work achieved high segmentation performance and alleviated the problem of insufficient training by the small-scale annotated corpus. Tang et al. (2022) achieved excellent results by proposing HGNSeg, a framework that employs multi-level features including character, word, n-grams, and dependency syntactic tree, using a heterogeneous graph neural network (HGNN).

2.3 Multi-Criteria Word Segmentation

A PTM is generally built on large-scale corpora, for example, SIGHAN2005[5] that include several corpora, to acquire prior knowledge. However, these corpora are individually annotated in different segmentation criteria, that is, they exhibit multi-criteria segmentation. He et al. (2017) proposed a simple model that benefits from multi-criteria segmentation across multiple corpora by adding an artificial token at the start and end of a sentence to specify the target corpus – a corpus-name token. Although their method is remarkably simple, it yielded good segmentation performance. Therefore, recent state-of-the-art word segmentation research adapted this mechanism with empirical methods to boost the segmentation performance (Huang, Cheng et al. 2020; Huang, Huang et al. 2020; Ke et al. 2021).

2.4 Attention Mechanism in Word Segmentation

An attention mechanism (Bahdanau et al. 2015) has been demonstrated to be effective for incorporating knowledge in word segmentation (Tian et al. 2020a). It computes an importance score between source and target information, particularly in character-based word segmentation, which produces context features correlated with a character and its corresponding knowledge.

Higashiyama et al. (2019) proposed two attention-based composition functions: weighted average (WAVG) and weighted concatenation (WCON). These allow a model to alternate focus between a character and its candidate words. Both functions summarize a relationship between a character with its candidate words as a summary vector. Their work achieved state-of-the-art performance in BCCWJ, a well-known Japanese dataset, using the WCON function, which requires more computational resources than WAVG. Tian et al. (2020a) proposed two-way attentions for joint Chinese word segmentation (CWS) and POS tagging that integrates two different linguistic information, including context features and linguistic knowledge, separately, for complementing a character representation. Tian et al. (2020b) introduced a framework to properly incorporate wordhood information built from n-grams using memory networks; it also exploits the attention mechanism using several popular encoder-decoder combinations. These approaches, i.e., Tian et al. (2020a) and Tian et al. (2020b) performed well for numerous Chinese datasets.

3 Approach

In this section, we provide an overview of our LATTE framework and then discuss it in detail.

[5] http://sighan.cs.uchicago.edu/bakeoff2005

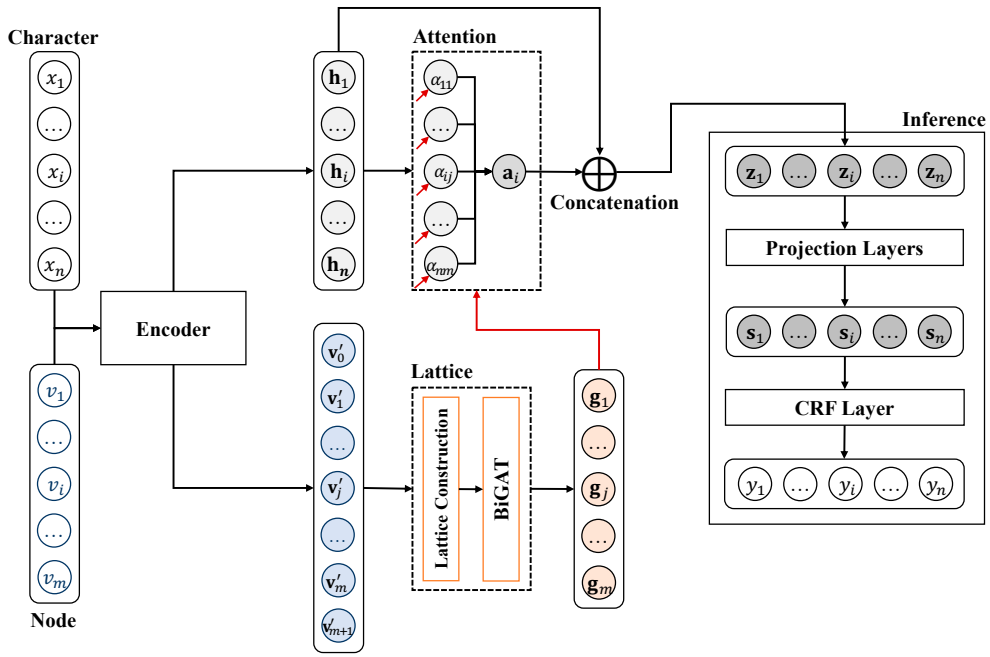


Figure 3 Proposed model integrating lattice and GNN into character-based word segmentation model.

Given a sentence s with n characters, that is, a character sequence $x_{1:n} = (x_1, x_2, \ldots, x_n)$, the task is to assign a segmentation label y_i based on the BMES tagging scheme $\mathcal{T} = \{\text{B, M, E, S}\}$ (beginning, middle, end, and singleton), which is a word-boundary label, to a character x_i. Our approach, as illustrated in Figure 3, utilizes either BiLSTM- or BERT-encoder to obtain *a contextualized character representation* $\mathbf{h}_i$ for each character in a character sequence. Consequently, we construct lattice G that includes nodes v built from possible words based on the character sequence along with their edges. Subsequently, we encode lattice G to obtain *a multi-grained contextualized node representation* $\mathbf{g}_j$ and a *lattice-attention summary vector* $\mathbf{a}_i$ using either BiLSTM- or BERT-encoder, GNN, and attention mechanism, sequentially. The contextualized character representation is subsequently concatenated with the lattice-attention summary vector. Finally, a conditional random field (CRF) layer is used to conditionally estimate the label sequence score for the character sequence.

We describe three major components, which perform the above operations in detail, including character encoding, lattice attentive encoding, and inference layer.

3.1 Character Encoding

We employ either BiLSTM (Hochreiter and Schmidhuber 1997; Gers et al. 2000) or BERT (Devlin et al. 2019), to transform a character sequence into contextualized character vectors.

BiLSTM: The character sequence $x_{1:n}$ is transformed into character embeddings $\mathbf{e}^c_{1:n}$ of d_c-dimensional feature vector. The character embedding matrix is defined as $E_c \in \mathbb{R}^{|\mathcal{VOC}_c| \times d_c}$, where $\mathcal{VOC}_c$ denotes the character vocabulary. Pre-trained word vectors such as fastText (Bojanowski et al. 2017) can be used to initialize character embeddings $\mathbf{e}^c_{1:n}$. BiLSTM layers are used to obtain contextualized character vectors $\mathbf{h}_{1:n}$ by subsequently encoding the character embeddings $\mathbf{e}^c_{1:n}$.

A current contextualized character vector $\mathbf{h}^l_i$ of the l^{th} BiLSTM layer can be computed bidirectionally as follows:

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

where $\mathbf{h}^0_{1:n} = \mathbf{e}^c_{1:n}$, f denotes the forward direction, b denotes the backward direction, $\oplus$ denotes concatenation, and $\mathbf{h}_i \in \mathbb{R}^{d_r}$ and d_r are hyperparameters.

BERT: Apart from the character sequence $x_{1:n}$, special tokens, namely the [CLS] and [SEP] tokens, are augmented at the beginning x_0 and end x_{n+1} of the character sequence, respectively. The character sequence that includes the special tokens $x_{0:n+1}$ is converted into a one-hot representation of characters $\mathbf{u}_{0:n+1}$. Subsequently, the one-hot representation of characters $\mathbf{u}_{0:n+1}$ are transformed into contextualized character vectors $\mathbf{h}_{0:n+1}$ of d_{BERT}-dimensional feature vector.

A current contextualized character vector $\mathbf{h}_i$ can be computed as follows:

$$\mathbf{h}^0_i = W_e \mathbf{u}_i + W_p[i], \tag{2}$$

$$\mathbf{h}^l_i = \text{Transformer}(\mathbf{h}^{l-1}_i), \tag{3}$$

where $l = \{1, 2, \ldots, L\}$, which is the number of transformer layers (Vaswani et al. 2017). W_e is the weight of the BERT-embedding layer while $W_p[i]$ is the positional encoding at the i^{th} index. u_i is one-hot representation of the i^{th} character. The BERT-embedding matrix is determined as $\mathbb{R}^{|\mathcal{VOC}_{\text{BERT}}| \times d_{\text{BERT}}}$, where $\mathcal{VOC}_{\text{BERT}}$ denotes the vocabulary in BERT. Notably, a character x can become [UNK] token assuming that the character does not exist in BERT vocabulary because the one-hot representation u_i of the i^{th} character is obtained from the BERT-embedding layer.

In addition, the summation of the last four layers is used to acquire the contextualized character vector $\mathbf{h}_i$ as in Yang (2019).

3.2 Lattice Attentive Encoding

We introduce the lattice attentive encoding method to attentively acquire multi-grained knowledge representations from an arbitrary number of segmentation alternatives, that is, lattice.[6]

Let $G = (V, E)$ where G is a directed acyclic graph (DAG) for a character sequence $x_{1:n}$, $V = \{v_1, \ldots, v_{|V|}\}$ is the set of vertices or nodes, and $E \subseteq \{V \times V\}$ is the set of edges. Possible words and characters are searched based on the character sequence $x_{1:n}$ to build nodes $v_{1:m} = V$, where m is the number of nodes (characters and words found) for lattice G. Each node preserves a character sequence within length k. Lattice G consists of nodes corresponding to a character or word w of length $1 < |w| \leq k$ and edges between nodes of adjacent characters/words in the sentence (the character sequence). We introduce approaches based on either BiLSTM-encoder or BERT-encoder to encode lattice G by initializing feature vectors for lattice nodes, that is, character-node V^c, word-node V^w, and special-node V^s.

$$V^c = \{v_j \in V | |v_j| = 1 \wedge v_j \notin S\},$$
$$V^w = \{v_j \in V | |v_j| > 1 \wedge v_j \notin S\},$$
$$V^s = \{v_j \in V | v_j \in S\},$$

where $|v_j|$ represents the length of characters in the j^{th} node. $\mathcal{S}$ denotes the set of special tokens, that is, [CLS], [SEP], [BOS], [EOS], and [UNK].

BiLSTM: Character-feature vectors $\mathbf{v}^c$ of the character-node V^c are initialized by the contextualized character vectors $\mathbf{h}_{1:n}$ from the BiLSTM encoder in Equation (1). Word-feature vectors $\mathbf{v}^w$ of the word-node V^w are initially generated from word embeddings $\mathbf{e}^w_{1:m}$ of the d_w-dimensional feature vector. The word embedding matrix is defined as $E_w \in \mathbb{R}^{|\mathcal{VOC}_w| \times d_w}$ where $\mathcal{VOC}_w$ denotes the word vocabulary. In addition, special nodes V^s, that is the [BOS] and [EOS] nodes, are used to specify the beginning and end of the sentence, respectively. These special-node features are obtained from the word embeddings matrix E_w.

BERT: Character-feature vectors $\mathbf{v}^c$ of the character-node V^c are initialized by the contextualized character vectors $\mathbf{h}_{1:n}$ from the BERT encoder in Equations (2) and (3). Word-feature vectors $\mathbf{v}^w$ of the word-node V^w are individually augmented with two special tokens, the [CLS] and [SEP] tokens, that are placed in the front and the back of each word-node, respectively. Consequently, the augmented word nodes are individually encoded using the BERT-encoder, i.e., Equations (2) and (3), to obtain contextualized node representations $\mathbf{v}' = \mathbf{v}'_{0:m+1}$, where $\{\mathbf{v}^c$,

[6] Please refer to Section 3.4.1 for implementation details concerning lattice construction.

$\mathbf{v}^w\} \in \mathbf{v}'$. As for special nodes V^s in lattice G, [CLS] and [SEP], are used to initialize the start and end nodes for BERT, respectively. These approaches eventually transform lattice G into lattice G', where $G' = (\mathbf{v}', E)$. Notably, a single-character word is treated similarly to a character in building a lattice and obtaining its representation from either BiLSTM or BERT.

Lattice G' is encoded by GAT to acquire multi-grained contextualized node representations $\mathbf{g} = \mathbf{g}_{1:m}$.

$$\mathbf{g} = \text{GAT}\left(G', \theta_G\right),$$

where θ_G denotes parameters of GAT such as the number of GAT layers and the number of attention heads, among others. The multi-grained contextualized node representation $\mathbf{g}_{1:m}$ can be obtained from GAT[7] by estimating the importance score u^g_{jk} of node k to node j and their attention weight α^g_{jk} as follows:

$$u^g_{jk} = \text{FFNN}(W_g\mathbf{v}'_j \oplus W_g\mathbf{v}'_k),$$
$$\alpha^g_{jk} = \frac{\exp(\text{LeakyReLU}(u^g_{jk}))}{\sum_{l\in O_j}\exp(\text{LeakyReLU}(u^g_{jl}))},$$

where j and k are neighbouring nodes, and FFNN is a single-layer feed-forward neural network. W_g and O_j denote a shared weight matrix and the set of neighbourhoods of node j,[8] respectively. Finally, the multi-head attention is employed to compute the multi-grained contextualized node representations $\mathbf{g}_j$.

$$\mathbf{g}_j = \sigma(\frac{1}{Q}\sum_{q=1}^{Q}\sum_{k\in O_j}\alpha^{g,q}_{jk}W^q_g\mathbf{v}'_k),$$

where $\mathbf{g}_j \in \mathbb{R}^{d_g}$, d_g is a hyperparameter, Q indicates the number of attention-head, and σ represents a nonlinear transformation, i.e., LeakyReLU.

To attentively project a proper representation out of multi-grained contextualized node representation $\mathbf{g}$, we employed a WAVG from Higashiyama et al. (2019), which is an attention-based composition function. This function summarizes the relationship for each character representation and its corresponding nodes by estimating a *lattice-attention summary vector* $\mathbf{a}_i$. Specifically, a contextualized character representation $\mathbf{h}_i$ originated from a character x_i corresponding to either the hidden vector of the final BiLSTM or BERT layer, are involved with a set of nodes that

[7] We used a bidirectional variant of GAT (BiGAT) as described in Section 3.4.2 (Equation (5)) to obtain the multi-grained contextualized node representations $\mathbf{g}$.

[8] When BiGAT is used if $(v_i, v_j) \in E$ and $(v_j, v_i) \notin E$, then v_j is the neighbourhood of v_i (i.e., $v_j \in O_i$) but v_i is not in the neighbourhood of v_j (i.e., $v_i \notin O_j$).

includes the character x_i in lattice G'. First, based on the contextualized character vector $\mathbf{h}_i$ and its corresponding multi-grained contextualized node representations $\mathbf{g}_{1:m}$ in lattice G', the node-importance score u^a_{ij} and lattice-attention weight α^a_{ij} are estimated accordingly.

$$u^a_{ij} = \mathbf{h}_i^T W_a \mathbf{g}_j,$$
$$\alpha^a_{ij} = \frac{\delta_{ij} \exp(u_{ij})}{\sum_{k=1}^{m} \delta_{ik} \exp(u_{ik})},$$

where $W_a \in \mathbb{R}^{d_c \times d_g}$ denotes a trainable weight matrix and $\delta_{ij} \in \{0, 1\}$ indicates whether character x_i is included in node v_j. The lattice-attention summary vector $\mathbf{a}_i$ for character x_i can be calculated as follows:

$$\mathbf{a}_i = \mathrm{WAVG}(x_i, \{v_j\}_{j=1}^{m}) = \sum_{j=1}^{m} \alpha^a_{ij} \mathbf{g}_j,$$

where $\{v_j\}$ is a node in lattice G' and $\mathbf{a}_i \in \mathbb{R}^{d_g}$. Finally, a multi-grained contextualized character vector $\mathbf{z}_i$ is produced by concatenating a contextualized character vector $\mathbf{h}$ with the lattice-attention summary vector $\mathbf{a}_i$ as,

$$\mathbf{z}_i = \mathbf{h}_i \oplus \mathbf{a}_i,$$

where $\mathbf{z}_i \in \mathbb{R}^{d_c + d_g}$.

3.3 Inference Layer

Projection layers are used to transform the multi-grained contextualized character vectors into a vector $\mathbf{s}_i \in \mathbb{R}^{|\mathcal{T}|}$. Considering CRF (Lafferty et al. 2001) has been successfully applied for sequence-labeling related tasks (Collobert et al. 2011), we adopted it to estimate the probability of the optimal label sequence $y = y_{1:n}$ for the character sequence $x = x_{1:n}$ by measuring the correlations between adjacent labels as in previous studies (Higashiyama et al. 2019; Chay-intr et al. 2021). Let $A \in \mathbb{R}^{|\mathcal{T}| \times |\mathcal{T}|}$ be a transition matrix for correlations between adjacent labels, where $\mathcal{T}$ denotes a set of all possible label sequences, for instance, $\mathcal{T} = \{\mathrm{B}, \mathrm{M}, \mathrm{E}, \mathrm{S}\}$. The i^{th} multi-grained contextualized character vector $\mathbf{z}_i$ can be transformed into an un-normalized label score $\mathbf{s}_i \in \mathbb{R}^{|\mathcal{T}|}$ as follows:

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

where $W_s \in \mathbb{R}^{|\mathcal{T}| \times (d_c + d_g)}$ is a trainable parameter matrix, and $\mathbf{b}_s \in \mathbb{R}^{|\mathcal{T}|}$ denotes a trainable bias vector.

Given the input sequence $x_{1:n}$, the corresponding scores for the label sequence $y_{1:n}$ can be obtained as follows:

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

where $s[y]$ represents the dimension of a vector s according to a label y. The probability of the label sequence can be obtained afterwards as follows:

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

To obtain the optimal label sequence $y^\star$, we adopt the Viterbi algorithm to maximize the sentence score:

$$y^\star = \arg\max_{y \in T^n} \text{score}(x, y).$$

Finally, we adopt the negative log-likelihood as our loss function and minimize it by backpropagation during the training process:

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

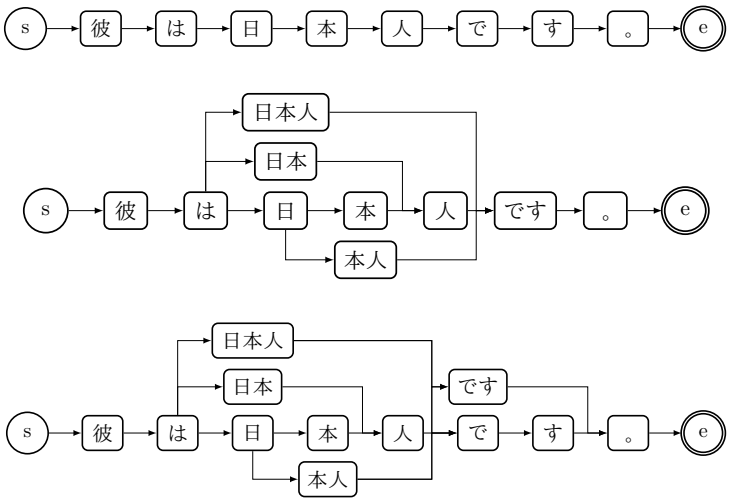
3.4 Implementation Details

3.4.1 Lattice Construction

Lattice can be built on the basis of three formations: character-lattice (ChL), word-lattice (WL), and word-character-lattice (WChL), as shown in Figure 4. We constructed a word-character-lattice to comprehensively handle segmentation alternatives, leveraging character knowledge as a foundation for character-based word segmentation. We built a lattice using all possible combinations according to a character sequence from training vocabulary, including the training set and external dictionary. Special nodes, including start node (s), ending node (e), and dataset node[9] (criterion token), are also included in the lattice. [CLS] and [SEP] are used as the start and end nodes, respectively, for the BERT encoder while [BOS] and [EOS] are used as the start and end nodes, respectively, for the BiLSTM encoder. The Aho–Corasick algorithm (Aho and Corasick 1975) is applied to obtain the substrings of the character sequence while reducing the time complexity in lattice construction to linear time complexity.

Furthermore, we introduce *dynamic-lattice construction* (DyL) by adapting concepts from Bagging, i.e., bootstrap aggregating (Breiman 1994) and Dropout (Srivastava et al. 2014) to

[9] Dataset node represents a feature for multi-criteria pre-training method as described in Section 3.4.3.



彼は日本人です。
“He is Japanese.”

Figure 4 Examples of lattice formation: character-lattice (top), word-lattice (middle), and word-character-lattice (bottom), that can be built for our model.

minimize generalization error and overfitting. Edges in the lattice are randomly deleted during the training step. This means the lattice for each epoch in relation to the same character sequence can be different.

3.4.2 Bidirectional Graph Neural Networks

As for a concept of BiLSTM architecture in sequence labelling that considers both forward and backward information (Huang et al. 2015), it can also be applied to a graph structure and GNN architecture. Gui et al. (2019) additionally built a transpose-graph from a directed graph where the graph comprises the same set of nodes but all edges in the graph are reversed. They concatenated the forward- and backward-state as the final result for node classification. In this study, we build direction-aware GNN layers based on the direction information, i.e., forward-GNN and backward-GNN layers, as shown in Figure 5. Parameters in GNN layers such as direction-dependent and trainable parameters are separately exploited according to the direction.

$$\mathrm{BiGNN} = \mathrm{GNN}_f(G_f, \theta_f) \oplus \mathrm{GNN}_b(G_b, \theta_b), \tag{4}$$

where G denotes lattice, $\theta.$ denotes the parameters for GNN layers, and $\oplus$ denotes concatenation. f and b represent forward- and backward-direction, respectively. A variant of GNNs such as GAT

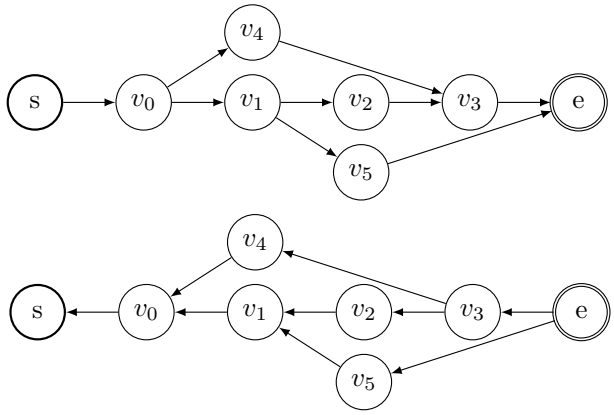


Figure 5 Examples of direction-aware lattice, including forward-lattice (top) and backward-lattice (bottom).

can also be applied to Equation (4).

$$\text{BiGAT} = \text{GAT}_f(G_f, \theta_f) \oplus \text{GAT}_b(G_b, \theta_b). \tag{5}$$

3.4.3 Multi-criteria Pre-training Method

The special tokens [CLS] and [SEP] are augmented at the beginning and end of each input sequence, respectively. We adopted a multi-criteria pre-training method from Ke et al. (2021) by adding criterion tokens after the [CLS] token to allow a model in learning criterion-dependent and criterion-independent segmentation knowledge among corpora. The criterion tokens, including criterion-dependent tokens such as [CTB6] and [BCCWJ], and an undefined-criterion token, that is, [UNC] token, were used in this study. Notably, the [UNC] token was used similarly as in Ke et al. (2021). Each input sequence is augmented with the criterion-dependent token; however, it is randomly replaced with the undefined-criterion token based on a hyperparameter.

We performed the multi-criteria pre-training method on BERT architecture, namely MC-BERT. Because the multi-criteria learning method requires more than one corpus, we additionally included accessible corpora to produce MC-BERT. Moreover, we also applied our proposed method, that is, LATTE, as a component.[10]

[10] We modified the MC-BERT pre-training method from Ke et al. (2021) by including LATTE (building lattice from training data and applying BiGAT to pre-train representations in MC-BERT) as a component. We released our implementation at `https://github.com/tchayintr/latte-ptm-ws` in pre-training MC-BERT rather than pre-training conventional MC-BERT proposed by Ke et al. (2021) to pre-train MC-BERT.

4 Experiment

4.1 Experimental Settings

4.1.1 Dataset

Three datasets in three languages, i.e., Japanese, Chinese, and Thai, were used to evaluate our model. Table 1 shows the statistics of the datasets, including sentences, words, vocabulary, and characters. (1) **BCCWJ**:[11] A Japanese word-segmented corpus that is primarily used in word segmentation experiments. (2) **CTB6**:[12] A Chinese Treebank corpus that is one of the most popular benchmark datasets for Chinese word segmentation. (3) **BEST2010**:[13] A large-scale Thai word-segmented corpus in four domains, which include article, encyclopaedia, news, and novel. While we followed the official data splits for both BCCWJ and CTB6 as in the previous works (Higashiyama et al. 2019; Huang et al. 2021), we used the same data splits for BEST2010 as Chay-intr et al. (2021).

4.1.2 External Dictionary and Pre-trained Word Vectors

Because lattice is built on the basis of vocabulary (characters and words), to enable the coverage of vocabulary in building lattice, we also included an external dictionary on top of the datasets, where only the training data was used, for each language individually. **Japanese**: UniDic[14] and IPADic[15] for MeCab. **Chinese**: BLCU balanced corpus,[16] Train data from

Dataset	Set	|S|	|W|	|V|	|Ch|
BCCWJ	Train	51.4K	1.2M	39.3K	1.7M
	Valid	5.7K	130.6K	13.2K	189.1K
	Test	3.0K	74.0K	7.2K	105.8K
CTB6	Train	24.4K	678.8K	43.9K	1.1M
	Valid	1.9K	51.2K	8.8K	83.3K
	Test	1.9K	52.9K	8.9K	86.8K
BEST2010	Train	119K	4.0M	72.9K	16.0M
	Valid	14.9K	501.4K	23.0K	1.9M
	Test	14.9K	500.4K	23.0K	1.9M

Table 1 Data sizes, in terms of the number of sentences (S), words (W), vocabulary (V), and characters (Ch).

[11] https://clrd.ninjal.ac.jp/bccwj/en
[12] https://catalog.ldc.upenn.edu/LDC2007T36
[13] https://thailang.nectec.or.th
[14] https://clrd.ninjal.ac.jp/unidic
[15] https://taku910.github.io/mecab
[16] http://bcc.blcu.edu.cn

SIGHAN2005, and Jieba.[17] **Thai**: HSE Thai Corpus[18] and LEXiTRON.[19] Additionally, to initialize robust word embeddings for BiLSTM-encoder, we employed fastText to produce a feature for characters and nodes, as well as freezing it during the training step.

4.1.3 Pre-training Models

Owing to various existing PTMs, we selected PTMs for use on the basis of their originality and accessibility. (1) **Japanese BERT**:[20] We chose character-level Japanese BERT to perform on the Japanese dataset owing to its accessibility and consistency with the character-based approach. (2) **Chinese BERT**:[21] This pre-trained model has been effectively used in neural models on Chinese datasets. Thus, we selected it for use in our experiment on the Chinese dataset. (3) **Multilingual BERT**:[22] We selected this pre-trained model to use on the Thai dataset because we could not find any Thai pre-trained model that is similar to Japanese and Chinese pre-trained models in terms of originality and accessibility. All models are $\text{BERT}_{\text{base}}$ models.

To build MC-BERT for Japanese, Chinese, and Thai, we additionally collected two, six, and four accessible datasets, respectively, and appended these additional datasets on top of the main datasets, i.e., BCCWJ, CTB6, and BEST2010. Two Japanese datasets, UD Japanese treebank,[23] and Kyoto University Text Corpus[24] were added to produce Japanese MC-BERT. Six Chinese datasets, four corpora from SIGHAN2005[5] (AS, CITYU, MSRA, and PKU), SXU from SIGHAN2008, and CNC, were appended to build Chinese MC-BERT. Both SXU and CNC corpora are obtained from public repositories.[25] All traditional Chinese corpora, such as AS and CITYU, are converted into simplified Chinese. Four Thai datasets, LST20,[26] TNHC,[27] VISTEC, [28] and WS160,[29] were used to construct the Thai MC-BERT. The number of datasets used to construct MC-BERT for each language was three for Japanese, seven for Chinese, and five for Thai. For accessibility, we perform the pre-training methods for Chinese MC-BERT on seven datasets rather than nine compared with the previous work (Ke et al. 2021).

[17] https://github.com/fxsjy/jieba
[18] http://web-corpora.net/ThaiCorpus
[19] https://lexitron.nectec.or.th
[20] https://huggingface.co/cl-tohoku/bert-base-japanese-char-v2
[21] https://huggingface.co/bert-base-chinese
[22] https://huggingface.co/bert-base-multilingual-cased
[23] https://github.com/UniversalDependencies/UD_Japanese-GSD
[24] https://github.com/ku-nlp/KyotoCorpus
[25] https://github.com/hankcs/multi-criteria-cws
[26] https://aiat.or.th/lst20-corpus/
[27] https://attapol.github.io/tlc
[28] https://github.com/mrpeerat/OSKut/tree/main/VISTEC-TP-TH-2021
[29] https://github.com/PyThaiNLP/wisesight-sentiment/tree/master/word-tokenization

4.1.4 Hyperparameters

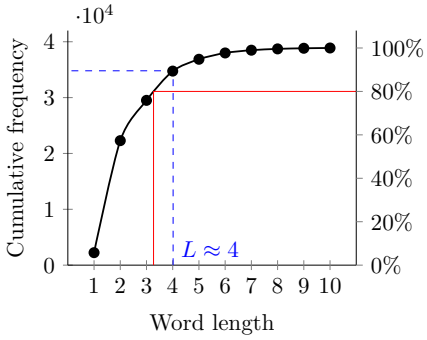
We used the essential hyperparameters for models as shown in Table 2. The AdamW optimizer (Loshchilov and Hutter 2017) was used to optimize the model parameters. Every model was trained for 20 epochs. We selected the best model to perform on the test set based on the validation process by word-level F_1 evaluation. Because several types of neural networks were utilized in the proposed method, the initial learning rate was set separately by the neural network type, that is, 2e-5 for BERT, 1e-3 for GNN, and 1e-3 for others. Learning rate decay is also applied and was set to 0.9.

To select an optimal maximum word length for building nodes in lattice among the datasets equitably, we reversely adapted the 80/20 rule also known as the Pareto principle.[30] We used only words within the optimal maximum word length in relation to the cumulative frequency of word length at 80% to build nodes. As shown in Figures 6a, 6b, and 6c, we selected the maximum word length of four, four, and twelve for the Japanese, Chinese, and Thai datasets, respectively. We set the [UNC] token rate as 0.1; practically, 10% of sentences among the corpora are augmented with the undefined-criterion token rather than the criterion-dependent tokens.

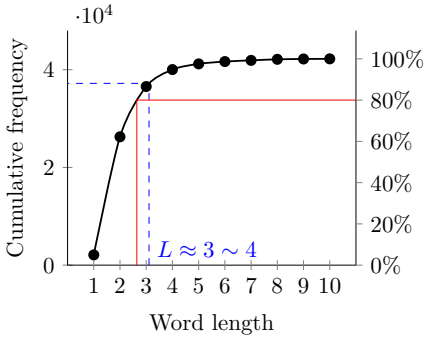
Parameter	Value
Character-embedding size	128
BiLSTM layers	2
BiLSTM hidden size	300
Initial learning rate	1e-3
Dropout rate	0.2
BERT-embedding size	768
BERT learning rate	2e-5
Max sequence length	512
Node-embedding size	300
GAT layers	2
GAT hidden size	300
GAT heads	2
GAT dropout rate	0.2
GAT learning rate	1e-3
Lattice dropout rate (DyL)	0.2

Table 2 Common hyperparameters and BERT hyperparameters for reproduced models and our proposed model (top and middle), and essential hyperparameters for our proposed model (bottom).

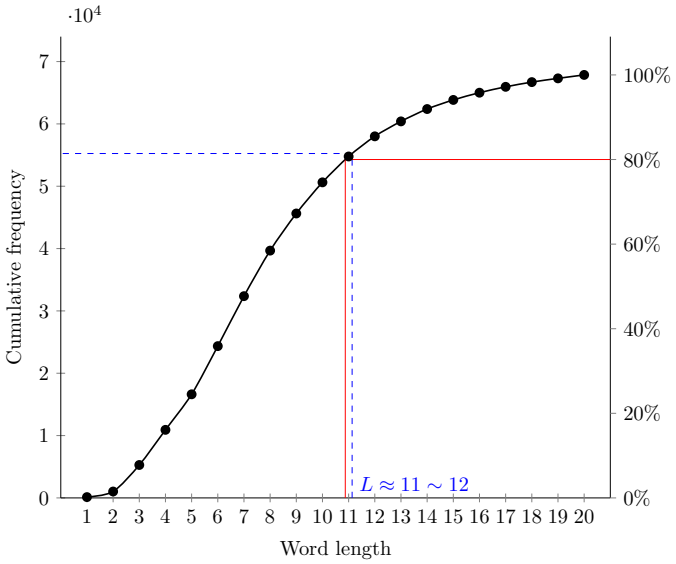
[30] https://en.wikipedia.org/wiki/Pareto_principle



(a) Word length and its cumulative frequency on BCCWJ



(b) Word length and its cumulative frequency on CTB6



(c) Word length and its cumulative frequency on BEST2010

Figure 6 Word length and its cumulative frequency on three datasets. Red line indicates the cumulative frequency at 80%. Blue dashed line denotes selected *maximum node length* hyperparameter.

4.1.5 Compared Models

We evaluated the following models:

- **Baselines**: Character-based models with different architectures, including BiLSTM-CRF, BiLSTM-WAVG-CRF (Higashiyama et al. 2019), BERT-CRF, and BERT-MC-CRF (Multi-criteria BERT).
- **LATTE w/ BiLSTM** (BiLSTM-BiGAT-CRF): Our proposed model integrating a lattice

attentive encoder with BiLSTM-encoder to generate features.

- **LATTE** (BERT-MC-BiGAT-CRF): Our proposed model integrating a lattice attentive encoder and using BERT-encoder, which is fine-tuned by multi-criteria BERT, to extract features, as shown in Figure 3.
- **Others**: Popular Well-known word-segmentation models (Neubig et al. 2011; Kitagawa and Komachi 2018; Qiu et al. 2020; Tian et al. 2020b; Huang, Cheng et al. 2020; Maimaiti et al. 2021; Huang et al. 2021; Tang et al. 2022; Treeratpituk 2017; Lapjaturapit et al. 2018; Chormai et al. 2019; Kittinaradorn et al. 2019; Seeha et al. 2020) and state-of-the-art word-segmentation models (Higashiyama et al. 2019; Ke et al. 2021; Chay-intr et al. 2021).

4.1.6 Evaluation Metrics

We selected evaluation metrics to evaluate models per language based on previous research. Word-level-F_1 score has been used to evaluate recent Japanese word-segmentation models (Higashiyama et al. 2019). However, two types of evaluation metrics, word-level-F_1 and OOV-recall score, have been used to evaluate models on Chinese word-segmentation task (Ke et al. 2021). Owing to the similarity of vocabulary length in both languages as shown in Figures 6a and 6b, we used word-level-F_1 and OOV-recall score to evaluate models on Japanese and Chinese datasets. Previous works on Thai word-segmentation task interchangeably evaluated on the character-level-F_1 and word-level-F_1 (Limkonchotiwat et al. 2021; Chay-intr et al. 2021). Thus, we chose character-level-F_1 and word-level-F_1 to evaluate models on Thai dataset. We evaluated word-level-F_1 and OOV-recall score as in Qiu et al. (2020), and we followed character-level-F_1 evaluation from Limkonchotiwat et al. (2021), Chay-intr et al. (2021).

$$\mathbf{Precision}_{\text{char}} = \frac{\#\text{char}_{\text{gold(B)}\cap\text{pred(B)}}}{\#\text{char}_{\text{pred(B)}}},$$

$$\mathbf{Recall}_{\text{char}} = \frac{\#\text{char}_{\text{gold(B)}\cap\text{pred(B)}}}{\#\text{char}_{\text{gold(B)}}},$$

$$\mathbf{F1}_{\text{char}} = 2 \times \frac{\mathbf{Precision}_{\text{char}} \times \mathbf{Recall}_{\text{char}}}{\mathbf{Precision}_{\text{char}} + \mathbf{Recall}_{\text{char}}},$$

where #char represents the number of characters in a sequence. gold(B) and pred(B) denote gold boundary characters from a dataset and predicted boundary characters from a model, respectively.

$$\mathbf{Precision}_{\text{word}} = \frac{\#\text{word}_{\text{gold}\cap\text{pred}}}{\#\text{word}_{\text{pred}}},$$

$$\mathbf{Recall}_{\text{word}} = \frac{\#\text{word}_{\text{gold}\cap\text{pred}}}{\#\text{word}_{\text{gold}}},$$

$$\mathbf{F1}_{\text{word}} = 2 \times \frac{\mathbf{Precision}_{\text{word}} \times \mathbf{Recall}_{\text{word}}}{\mathbf{Precision}_{\text{word}} + \mathbf{Recall}_{\text{word}}},$$

where #word represents the number of words found in a sequence. gold and pred denote a set of gold words from a dataset and a set of predicted words from a model, respectively.

$\mathbf{Recall}_{\text{oov}}$ represents the recalls for OOV words that exist in inference phrase while not existing in the training phase.It can be computed as follows:

$$\mathbf{Recall}_{\text{oov}} = \frac{\#\text{word}_{\text{infer}\cap(\text{gold}\setminus\text{train})}}{\#\text{word}_{\text{gold}\setminus\text{train}}},$$

where infer, train, and gold denote a set of words produced in the inference phase, a set of words from the Train set, and a set of gold words from a dataset, respectively.

4.2 Results and Analysis

4.2.1 Main Results

Tables 3, 4, and 5 show comparisons of previous works, baselines, and our proposed model on the three datasets, that is, BCCWJ, CTB6, and BEST2010, respectively. The results indicate that LATTE outperformed previous works and Baselines in every selected evaluation metric.

LATTE could outperform Higashiyama et al. (2019) which integrates either the WAVG function or WCON function to estimate the relationships between a character and its candidate words. While incorporating the WCON function with the word-segmentation model in Higashiyama et al.

Model	F_{word} (σ)	R_{oov} (σ)
(Neubig et al. 2011)	98.2	—
(Kitagawa and Komachi 2018)	98.4	—
(Higashiyama et al. 2019)	98.9	—
BiLSTM-CRF	98.2 (0.0200)	81.5 (0.060)
BiLSTM-WAVG-CRF	98.2 (0.005)	70.3 (0.690)
BERT-CRF	99.3 (0.030)	92.0 (0.010)
BERT-MC-CRF	99.3 (0.015)	91.6 (0.060)
LATTE w/ BiLSTM	99.0 (0.005)	83.6 (0.040)
LATTE	**99.4 (0.005)**	**92.1 (0.005)**

Table 3 Comparison among Others, Baselines, and our proposed model, on BCCWJ dataset. The best score for each metric is indicated in **bold**. F_{word} and R_{OOV} denote the word-level-F_1 and OOV-recall scores, respectively. σ represents the population standard deviation. The scores were obtained from the mean of two runs.

Model	F_{word} (σ)	R_{oov} (σ)
(Qiu et al. 2020)	97.0	87.0
(Tian et al. 2020a)	97.4	88.5
(Tian et al. 2020b)	97.2	88.0
(Huang, Cheng et al. 2020)	97.8	89.4
(Maimaiti et al. 2021)	97.7	—
(Huang et al. 2021)	97.8	90.2
(Ke et al. 2021)	97.9	89.2
(Tang et al. 2022)	97.8	89.7
BiLSTM-CRF	94.4 (0.010)	75.5 (0.005)
BiLSTM-WAVG-CRF	95.1 (0.005)	63.3 (0.005)
BERT-CRF	97.8 (0.035)	89.2 (0.505)
BERT-MC-CRF	97.9 (0.035)	90.5 (0.120)
LATTE w/BiLSTM	95.8 (0.000)	78.45 (0.050)
LATTE	**98.1 (0.020)**	**90.6 (0.135)**

Table 4 Comparison among Others, Baselines, and our proposed model, on the CTB6 dataset. The best score for each metric is indicated in **bold**. F_{word} and R_{OOV} denote the Word-level-F_1 and OOV-recall scores, respectively. σ represents the population standard deviation. The scores were obtained from the mean of two runs.

Model	F_{char}	F_{word}
(Treeratpituk 2017)	97.1	92.5
(Lapjaturapit et al. 2018)	98.4	96.2
(Chormai et al. 2019)	98.4	96.2
(Kittinaradorn et al. 2019)	97.1	93.8
(Seeha et al. 2020)	98.8	97.2
(Chay-intr et al. 2021)	99.0	**97.7**
BiLSTM-CRF	98.9 (0.005)	97.1 (0.020)
BiLSTM-WAVG-CRF	98.9 (0.005)	97.2 (0.005)
BERT-CRF	99.0 (0.005)	97.3 (0.045)
BERT-MC-CRF	99.0 (0.010)	97.6 (0.005)
LATTE w/BiLSTM	99.0 (0.005)	97.3 (0.015)
LATTE	**99.1 (0.005)**	**97.7 (0.015)**

Table 5 Comparison among Others, Baselines, and our proposed model, on the BEST2010 dataset. The best score for each metric is indicated in **bold**. F_{char} and F_{word} denote the character-level-F_1 and word-level-F_1 scores, respectively. σ represents the population standard deviation. The scores were obtained from the mean of two runs.

(2019) achieves superior segmentation performance to the WAVG function, which is an average-based function, it is computationally intensive owing to its concatenation mechanism that logically consumes more memory and computational time. Although LATTE is incorporated with

the WAVG function only, it outperformed their model that integrates the WCON function. Comparing our model to a similar lattice-based work (Huang et al. 2021), our method surpasses it through various approaches to handle, encode, and interact between a character sequence and a lattice. For BEST2010, superb segmentation performance has been obtained on Chay-intr et al. (2021) by incorporating multiple attentions from word and character-cluster information, which is an exclusive Thai writing system knowledge, with a WCON function. Although LATTE obtained comparable results on word-level-F_1 with Chay-intr et al. (2021), our model outperformed it on character-level-F_1 using WAVG function which requires fewer computational resources. In addition, while LATTE w/BiLSTM could not surpass BERT-based models, it outperformed previous works and baselines, particularly the BiLSTM-based model, on three datasets in each evaluation metric.

4.2.2 Segmentation Performance with Additional Datasets

We tested our model when some unknown datasets were not included in pre-training MC-BERT. The task is to additionally fine-tune MC-BERT with an additional dataset based on its training set along with a validation set and evaluate the fine-tuned model on the testing set. We augmented the undefined-criterion token [UNC] after the [CLS] token to each sentence, where the representation of the [UNC] token was transferred from MC-BERT. The criterion-dependent tokens, such as [CTB6], were not used in this test. We conducted this experiment on two datasets: UD_{JA} (Japanese) and UD_{ZH}[31] (Chinese). Note that we pulled UD_{JA} out from Japanese MC-BERT pre-training to conduct this test.

Table 6 shows the results of segmentation performance on two additional datasets. The results showed that our proposed model outperformed the baselines on both datasets. MC-BERT could help in improving segmentation performance on UD_{JA} and UD_{ZH}. Moreover, our proposed

Model	UD_{JA}		UD_{ZH}	
	F_{word}	R_{oov}	F_{word}	R_{oov}
BiLSTM-CRF	96.1	82.1	90.7	75.1
BERT-CRF	98.9	93.3	98.2	93.4
BERT-MC-CRF	99.2	94.3	98.4	93.4
LATTE	**99.3**	**95.1**	**98.5**	**93.5**

Table 6 Results of segmentation performance on additional datasets, including UD_{JA} (Japanese) and UD_{ZH} (Chinese).

[31] https://github.com/UniversalDependencies/UD_Chinese-GSDSimp

model could further enhance segmentation performance when MC-BERT was employed while using LATTE as a component.

4.2.3 Ablation Study

LATTE achieved superior segmentation performance beyond previous works by integrating three major components: BiGNN, MC-BERT, and DyL. To analyze the effect of these components on our proposed model, we conducted an ablation study based on LATTE incorporated with BERT-encoder on the three datasets.

Table 7 shows the results of segmentation performance on the ablation study. The results demonstrated that combining BiGNN, MC-BERT, and DyL advanced segmentation performance. However, employing MC-BERT on the BCCWJ corpus did not improve segmentation performance as CTB6 and BEST2010. Incorporating BiGNN into our model consistently improved segmentation performance on all dataset. However, LATTE that incorporated BiGAT, MC-BERT, and DyL did not enhance segmentation performance on BCCWJ compared with LATTE that applied BiGAT and DyL; however, it could perform well on CTB6 and BEST2010. The DyL helped boost segmentation performance on BCCWJ and CTB6; however, it lessened seg-

Dataset	BiGAT	MC-BERT	DyL	F_{word}
BCCWJ				99.29
	✓			99.32
	✓	✓		99.32
	✓		✓	**99.36**
	✓	✓	✓	99.35
CTB6				97.80
	✓			97.83
	✓	✓		98.00
	✓		✓	97.92
	✓	✓	✓	**98.07**
BEST2010				97.45
	✓			97.49
	✓	✓		97.61
	✓		✓	97.46
	✓	✓	✓	**97.69**

Table 7 Results of Ablation Study on BCCWJ, CTB6, and BEST2010. MC-BERT denotes BERT with multi-criteria learning and DyL represents dynamic lattice construction. All models are the proposed model incorporated with BERT-encoder. ✓ indicates whether the feature is incorporated into the word-segmentation model, and the best scores are indicated in **bold**.

mentation performance on BEST2010, which includes longer character sequences than BCCWJ and CTB6, and led to lower performance.

4.2.4 Case Study: Segmentation Results

To show whether specific cases from segmentation results[32] were improved or worsened by incorporating LATTE, we conducted a comparison of segmentation results between BERT-MC-CRF and LATTE. We selected two Chinese test samples from the CTB6 dataset to be our case study because Chinese word segmentation is the most competitive among the three languages, i.e., Chinese, Japanese, and Thai.

Figures 7 and 8 show segmentation results between BERT-MC-CRF and the proposed

Reference	为 此 , 省 政府 将 “龙开河 治理 开发 工程 ” 纳入 了 省 重点 防洪 工程 。
BERT-MC-CRF	为 此 , 省政府 将 “龙开河 治理 开发 工程 ” 纳入 了 省 重点 防洪 工程 。
LATTE	为 此 , 省 政府 将 “龙开河 治理 开发 工程 ” 纳入 了 省 重点 防洪 工程 。

为此，省政府将“龙开河治理开发工程”纳入了省重点防洪工程。

“For this reason, the provincial government incorporated
the “Longkai River Treatment and Development Project”
into the provincial key flood control project.”

Figure 7 Examples of segmentation results between BERT-MC-CRF and LATTE. Ground-truth segmentation result is indicated as “Reference” and incorrect segmentation results are in red. While LATTE completely segments the correct results, BERT-MC-CRF produces incorrect results.

Reference	从 那时 起 , 欧洲 政局 无 一 日 安宁 , 危机重重 。
BERT-MC-CRF	从 那时 起 , 欧洲 政局 无 一 日 安宁 , 危机重重 。
LATTE	从 那时 起 , 欧洲 政局 无 一 日 安宁 , 危机 重重 。

从那时起，欧洲政局无一日安宁，危机重重。

“Since then, the political situation in Europe has never been peaceful,
and there are crisis-ridden”

Figure 8 Examples of segmentation results between BERT-MC-CRF and LATTE. Ground-truth segmentation result is indicated as “Reference” and incorrect segmentation results are in red. While LATTE produces incorrect results, BERT-MC-CRF completely segments the correct results.

[32] A collection of segmentation results are publicly available at `https://github.com/tchayintr/latte-ws`

LATTE, respectively. Figure 7 illustrates a case where LATTE outperforms BERT-MC-CRF by segmenting “省政府 (Provincial Government)” as “省 (Provincial) and 政府 (Government)” while BERT-MC-CRF preserves the “省政府” as it is. By considering a word category (part-of-speech) of “省 (Provincial)” and “政府 (Government)”, that is, adjective and noun, respectively, it is the smallest piece of words that can be divided, where the word category is still preserved. This indicates a tendency towards less ambiguity in terms of word units by segmenting them correctly into small units. In case both words are combined into a noun phrase that produces from BERT-MC-CRF, it gathers more complex structures.

However, Figure 8 shows other results where LATTE could not outperform BERT-MC-CRF by segmenting “危机重重 (crisis-ridden)” as “危机 (crisis)” and “重重 (ridden)” while BERT-MC-CRF preserves “危机重重 (crisis-ridden)” as it is. “危机重重 (crisis-ridden)” is a Chinese idiom, where its character sequence is fixed to present certain meanings with more complex structures. Regardless of the idiom structures, it is also legitimate to segment “危机重重 (crisis-ridden)” into “危机 (crisis)” and “重重 (ridden)” because both words contain meanings by themselves. Therefore, although LATTE could not recognize the idiom, it could produce results according to the meanings.

Additionally, we selected segmentation results from BCCWJ and BEST2010 to illustrate the cases where LATTE outperformed BERT-MC-CRF as shown in Figures 9 and 10.

In terms of meaning, the segmentation results from BCCWJ are not significantly different. The major difference between BERT-MC-CRF and LATTE lies in the connection between the character “着” and “せ”. LATTE produced the segmentation result where “着” and “せ” are combined as “着せ (dress up)”, which forms the verb “着せる (to dress)” by considering the word category. BERT-MC-CRF segmented the sentence differently, by combining “着” with

Reference	えぇ〜い 、親 なら 子供 に お古 着せ て 節約 する な〜! !
BERT-MC-CRF	えぇ〜い 、親 なら 子供 に お 古着 せ て 節約 する な〜! !
LATTE	えぇ〜い 、親 なら 子供 に お 古 着せ て 節約 する な〜! !

えぇ〜い、親なら子供にお古着せて節約するな〜!!

“Come on, if you’re a parent, don’t dress your kids
in hand-me-downs to save money!!”

Figure 9 Examples of segmentation results between BERT-MC-CRF and LATTE. Ground-truth segmentation result is indicated as “Reference” and incorrect segmentation results are in red. LATTE segments better results than BERT-MC-CRF.

Reference	ฝีมือ ประณีต กว่า ที่ อื่น ๆ ใน ภาค เดียว กัน
BERT-MC-CRF	ฝีมือ ประณีตก ว่า ที่ อื่น ๆ ใน ภาค เดียว กัน
LATTE	ฝีมือ ประณีต กว่า ที่ อื่น ๆ ใน ภาค เดียว กัน

ฝีมือประณีตกว่าที่อื่นๆในภาคเดียวกัน

"The level of craftsmanship is more refined than
that found in other areas within the same region."

Figure 10 Examples of segmentation results between BERT-MC-CRF and LATTE. Ground-truth segmentation result is indicated as "Reference" and incorrect segmentation results are in red. While LATTE completely segments the correct results, BERT-MC-CRF produces incorrect results.

"古" into "古着" rather than forming the verb "着せる (to dress)". This leaves "せ" by itself, which is grammatically incorrect and does not convey the intended meaning of the verb "着せる (to dress)", resulting in a grammatically incorrect sentence. Although both BERT-MC-CRF and LATTE separately segmented "お", which is an honorific prefix, from "古 (old)", the overall meaning is not changed. However, this results in a less natural expression, as the honorific "お" and the character "古" are not commonly separated when referring to second-hand clothes in Japanese. However, the segmentation results from BEST2010 could represent two different meanings. While LATTE could accurately segment the sentence, thus producing the correct meaning, the segmentation result from BERT-MC-CRF represents the sentence with a completely different meaning. Ultimately, LATTE significantly outperformed BERT-MC-CRF based on this sample.

5 Conclusion

In this study, we proposed a lattice attentive encoding method for character-based word segmentation that uses lattice to handle segmentation alternatives along with GNNs and attention mechanisms. Our proposed model attentively extracts multi-granularity representations from such a lattice for complementing character representations. Experimental results showed that our method could improve segmentation performance on three popular datasets: BCCWJ, CTB6, and BEST2010. Moreover, pre-training a model with LATTE as a component enhanced segmentation performance as well.

References

Aho, A. V. and Corasick, M. J. (1975). "Efficient String Matching: An Aid to Bibliographic Search." *Communication of the ACM*, **18** (6), pp. 333–340.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). "Neural Machine Translation by Jointly Learning to Align and Translate." In Bengio, Y. and LeCun, Y. (Eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings*.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). "Enriching Word Vectors with Subword Information." *Transactions of the Association for Computational Linguistics*, **5**, pp. 135–146.

Breiman, L. (1994). "Bagging Predictors." *Machine Learning*, **24** (2), pp. 114–133.

Chay-intr, T., Kamigaito, H., and Okumura, M. (2021). "Character-based Thai Word Segmentation with Multiple Attentions." In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pp. 264–273, Held Online. INCOMA Ltd.

Chen, X., Shi, Z., Qiu, X., and Huang, X. (2017). "Adversarial Multi-Criteria Learning for Chinese Word Segmentation." In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1193–1203, Vancouver, Canada. Association for Computational Linguistics.

Chormai, P., Prasertsom, P., and T. Rutherford, A. (2019). "AttaCut: A Fast and Accurate Neural Thai Word Segmenter." `https://arxiv.org/abs/1911.07056`.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). "Natural Language Processing (almost) from Scratch." *Journal of Machine Learning Research*, **12**, pp. 2493–2537.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." 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)*, pp. 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dyer, C., Muresan, S., and Resnik, P. (2008). "Generalizing Word Lattice Translation." In *Proceedings of ACL-08: HLT*, pp. 1012–1020, Columbus, Ohio. Association for Computational Linguistics.

Finn, C., Abbeel, P., and Levine, S. (2017). "Model-Agnostic Meta-Learning for Fast Adaptation

of Deep Networks." `https://arxiv.org/abs/1703.03400`.

Gers, F., Schmidhuber, J., and Cummins, F. (2000). "Learning to Forget: Continual Prediction with LSTM." *Neural Computation*, **12** (10), pp. 2451–2471.

Gui, T., Zou, Y., Zhang, Q., Peng, M., Fu, J., Wei, Z., and Huang, X. (2019). "A Lexicon-Based Graph Neural Network for Chinese NER." In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 1040–1050, Hong Kong, China. Association for Computational Linguistics.

He, H., Wu, L., Yan, H., Gao, Z., Feng, Y., and Townsend, G. (2017). "Effective Neural Solution for Multi-Criteria Word Segmentation." `https://arxiv.org/abs/1712.02856`.

Higashiyama, S., Utiyama, M., Sumita, E., Ideuchi, M., Oida, Y., Sakamoto, Y., and Okada, I. (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)*, pp. 2699–2709, Minneapolis, Minnesota. Association for Computational Linguistics.

Hochreiter, S. and Schmidhuber, J. (1997). "Long Short-Term Memory." *Neural Computation*, **9** (8), pp. 1735–1780.

Huang, W., Cheng, X., Chen, K., Wang, T., and Chu, W. (2020). "Towards Fast and Accurate Neural Chinese Word Segmentation with Multi-Criteria Learning." In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 2062–2072, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Huang, K., Huang, D., Liu, Z., and Mo, F. (2020). "A Joint Multiple Criteria Model in Transfer Learning for Cross-domain Chinese Word Segmentation." In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 3873–3882, Online. Association for Computational Linguistics.

Huang, K., Yu, H., Liu, J., Liu, W., Cao, J., and Huang, D. (2021). "Lexicon-Based Graph Convolutional Network for Chinese Word Segmentation." In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 2908–2917, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Huang, Z., Xu, W., and Yu, K. (2015). "Bidirectional LSTM-CRF Models for Sequence Tagging.".

Ke, Z., Shi, L., Sun, S., Meng, E., Wang, B., and Qiu, X. (2021). "Pre-training with Meta Learning for Chinese Word Segmentation." In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5514–5523, Online. Association for Computational Linguistics.

Kipf, T. N. and Welling, M. (2016). "Semi-Supervised Classification with Graph Convolutional Networks." https://arxiv.org/abs/1609.02907.

Kitagawa, Y. and Komachi, M. (2018). "Long Short-Term Memory for Japanese Word Segmentation." In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation*, pp. 279–288, Hong Kong. Association for Computational Linguistics.

Kittinaradorn, R., Achakulvisut, T., Chaovavanich, K., Srithaworn, K., Chormai, P., Kaewkasi, C., Ruangrong, T., and Oparad, K. (2019). "DeepCut: A Thai word tokenization library using Deep Neural Network." https://github.com/rkcosmos/deepcut.

Lafferty, J., Mccallum, A., and Pereira, F. (2001). "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data." In *Proceedings of the 18th International Conference on Machine Learning*, pp. 282–289.

Lai, Y., Liu, Y., Feng, Y., Huang, S., and Zhao, D. (2021). "Lattice-BERT: Leveraging Multi-Granularity Representations in Chinese Pre-trained Language Models." In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1716–1731, Online. Association for Computational Linguistics.

Lapjaturapit, T., Viriyayudhakom, K., and Theeramunkong, T. (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*, pp. 30–35.

Li, X., Yan, H., Qiu, X., and Huang, X. (2020). "FLAT: Chinese NER Using Flat-Lattice Transformer." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 6836–6842, Online. Association for Computational Linguistics.

Li, X., Meng, Y., Sun, X., Han, Q., Yuan, A., and Li, J. (2019). "Is Word Segmentation Necessary for Deep Learning of Chinese Representations?" In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3242–3252, Florence, Italy. Association for Computational Linguistics.

Limkonchotiwat, P., Phatthiyaphaibun, W., Sarwar, R., Chuangsuwanich, E., and Nutanong, S. (2021). "Handling Cross- and Out-of-Domain Samples in Thai Word Segmentation." In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 1003–1016, Online. Association for Computational Linguistics.

Loshchilov, I. and Hutter, F. (2017). "Decoupled Weight Decay Regularization." https://arxiv.org/abs/1711.05101.

Maimaiti, M., Liu, Y., Zheng, Y., Chen, G., Huang, K., Zhang, J., Luan, H., and Sun, M.

(2021). "Segment, Mask, and Predict: Augmenting Chinese Word Segmentation with Self-Supervision." In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 2068–2077, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Neubig, G., Nakata, Y., and Mori, S. (2011). "Pointwise Prediction for Robust, Adaptable Japanese Morphological Analysis." In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 529–533, Portland, Oregon, USA. Association for Computational Linguistics.

Qian, X. and Liu, Y. (2012). "Joint Chinese Word Segmentation, POS Tagging and Parsing." In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 501–511, Jeju Island, Korea. Association for Computational Linguistics.

Qiu, X., Pei, H., Yan, H., and Huang, X. (2020). "A Concise Model for Multi-Criteria Chinese Word Segmentation with Transformer Encoder." In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 2887–2897, Online. Association for Computational Linguistics.

Seeha, S., Bilan, I., Mamani Sanchez, L., Huber, J., Matuschek, M., and Schütze, H. (2020). "ThaiLMCut: Unsupervised Pretraining for Thai Word Segmentation." In *Proceedings of the 12th Language Resources and Evaluation Conference*, pp. 6947–6957, Marseille, France. European Language Resources Association.

Sennrich, R., Haddow, B., and Birch, A. (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)*, pp. 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Shen, Y., Tan, S., Sordoni, A., and Courville, A. (2018). "Ordered Neurons: Integrating Tree Structures into Recurrent Neural Networks." `https://arxiv.org/abs/1810.09536`.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Ruslan, S. (2014). "Dropout: A Simple Way to Prevent Neural Networks from Overfitting." *Journal of Machine Learning Research*, **15**, pp. 1929–1958.

Sun, W. (2010). "Word-based and Character-based Word Segmentation Models: Comparison and Combination." In *Coling 2010: Posters*, pp. 1211–1219, Beijing, China. Coling 2010 Organizing Committee.

Tang, X., Wang, J., and Su, Q. (2022). "Chinese Word Segmentation with Heterogeneous Graph Neural Network." `https://arxiv.org/abs/2201.08975`.

Tian, Y., Song, Y., Ao, X., Xia, F., Quan, X., Zhang, T., and Wang, Y. (2020a). "Joint Chinese Word Segmentation and Part-of-speech Tagging via Two-way Attentions of Auto-analyzed Knowledge." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8286–8296, Online. Association for Computational Linguistics.

Tian, Y., Song, Y., Xia, F., Zhang, T., and Wang, Y. (2020b). "Improving Chinese Word Segmentation with Wordhood Memory Networks." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8274–8285, Online. Association for Computational Linguistics.

Treeratpituk, P. (2017). "Thai Word-Segmentation with LSTM in Tensorflow." `https://github.com/pucktada/cutkum`.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). "Attention Is All You Need." `https://arxiv.org/abs/1706.03762`.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2017). "Graph Attention Networks." `https://arxiv.org/abs/1710.10903`.

Wang, X., Ji, H., Shi, C., Wang, B., Cui, P., Yu, P., and Ye, Y. (2019). "Heterogeneous Graph Attention Network.". `https://arxiv.org/abs/1903.07293`.

Yang, H. (2019). "BERT Meets Chinese Word Segmentation." `https://arxiv.org/abs/1909.09292`.

Yang, J., Zhang, Y., and Liang, S. (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)*, pp. 2720–2725, Minneapolis, Minnesota. Association for Computational Linguistics.

Yao, L., Mao, C., and Luo, Y. (2018). "Graph Convolutional Networks for Text Classification." `https://arxiv.org/abs/1809.05679`.

Zhang, Y. and Yang, J. (2018). "Chinese NER Using Lattice LSTM." In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1554–1564, Melbourne, Australia. Association for Computational Linguistics.

Appendix

A. Upper-bound Score Test

We hypothesised that segmentation results produced from a model may not be the best results; however, it generalizes the model to minimize the segmentation errors. The test is performed

by training a character-based BiLSTM-CRF model to predict a label sequence $\hat{y}$ for each input sequence up to r segmentation results, i.e. $\hat{\mathbf{Y}} = \{\hat{y}^1, \hat{y}^2, \ldots, \hat{y}^r\}$, where $\hat{\mathbf{Y}}$ denotes a set of label sequences $\hat{y}$, and $r = \{1, 2, 4, 8, 16\}$. The CRF layer is used along with the Viterbi algorithm to produce top-r segmentation results. To evaluate segmentation performance in this test, we aggregated scores, including character-level-F_1 score and OOV-recall score, according to the best segmentation result that yields the highest score among the top r results. For example, in case of $r = 8$, the top 8 possible segmented sentences $\hat{\mathbf{Y}} = \{\hat{y}^1, \hat{y}^2, \ldots, \hat{y}^r\}$ for a sentence will be produced from the model. Subsequently, each segmentation result $\hat{y}^i \in \{\hat{y}^1, \hat{y}^2, \ldots, \hat{y}^8\}$ will be evaluated with the reference sentence y; the highest scores from among the top-eight sentences will be used to aggregate the scores.

Table 8 shows a comparison of top-r segmentation performance. The results show the same tendency on three datasets; therefore, segmentation performance of the model depends on the increase of r. Comparison with the segmentation performance from top-r segmentation results where $r > 1$ demonstrates that the best results ($r = 1$) are not the truly best segmentation results. In addition, however, the model where $r = 1$ implicitly and generally considers such top-r information in principle to produce the results. On the other hand, by increasing the r

Dataset	r	F_{char}	R_{oov}
BCCWJ	1	99.2	88.8
	2	99.5 (+0.3)	92.4 (+3.6)
	4	99.7 (+0.5)	94.8 (+6.0)
	8	99.8 (+0.6)	95.9 (+7.1)
	16	**99.8 (+0.6)**	**96.8 (+8.0)**
CTB6	1	97.8	79.6
	2	98.3 (+0.5)	82.7 (+3.1)
	4	98.7 (+0.9)	85.7 (+6.1)
	8	98.9 (+1.1)	88.2 (+8.6)
	16	**99.1 (+1.3)**	**90.4 (+10.8)**
BEST2010	1	98.9	75.6
	2	99.2 (+0.3)	77.8 (+2.2)
	4	99.4 (+0.5)	80.8 (+5.2)
	8	99.5 (+0.6)	83.2 (+7.6)
	16	**99.6 (+0.7)**	**85.6 (+10.0)**

Table 8 Comparison of segmentation performance in upper-bound score test on the basis of character-based BiLSTM architecture (Baseline). The scores are aggregated from the best results in top-r segmentation results. The numbers in parentheses represent the differences from the model where $r = 1$.

value to produce more segmentation results from the model to be used for the evaluation, it could obtain superior segmentation performance. Accordingly, this indicates that if top-r information is handled explicitly and properly, it could lead to the improvement of segmentation performance as in LATTE method.

Thodsaporn Chay-intr: T. Chay-intr received his M. Eng. from Sirindhorn Institute of Technology, Thailand, and is currently a doctoral student at Tokyo Institute of Technology, Japan. His research interests include natural language processing, particularly text classification; computer linguistics; and text mining.

Hidetaka Kamigaito: H. Kamigaito received his Dr. Eng. from Tokyo Institute of Technology, Japan, and is currently an associate professor at Nara Institute of Science and Technology, Japan. His research interests include natural language processing with a specific focus on document-level text processing and knowledge graph completion.

Kotaro Funakoshi: K. Funakoshi received his Dr. Eng. from Tokyo Institute of Technology, Japan, and is currently an associate professor at Institute of Innovative Research, Tokyo Institute of Technology. Formerly, he was with Honda Research Institute Japan Co., Ltd. from 2006 and served a program-specific associate professor of the Cooperative Intelligence Joint Chair at Kyoto University from 2017. His research interests include natural language processing, particularly computational linguistics; multi-modal dialogue systems; and human-machine interaction.

Manabu Okumura: M. Okumura received his Dr. Eng. from Tokyo Institute of Technology, Japan, and is currently a professor at Institute of Innovative Research, Tokyo Institute of Technology. His research interests include natural language processing, particularly text summarising; computer-assisted language learning; sentiment analysis; and text mining.

(Received September 1, 2022)
(Revised January 5, 2023)
(Accepted February 13, 2023)