論文 / 著書情報
Article / Book Information

題目(和文)	
Title(English)	Incorporating Multi-granularity Linguistic Units in Character-based Word Segmentation
著者(和文)	CHAY-INTR Thodsaporn
Author(English)	Thodsaporn Chay-intr
出典(和文)	学位:博士(工学), 学位授与機関:東京工業大学, 報告番号:甲第12542号, 授与年月日:2023年9月22日, 学位の種別:課程博士, 審査員:奥村 学,熊澤 逸夫,中山 実,篠崎 隆宏,船越 孝太郎
Citation(English)	Degree:Doctor (Engineering), Conferring organization: Tokyo Institute of Technology, Report number:甲第12542号, Conferred date:2023/9/22, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Tokyo Institute of Technology
Department of Information and Communications Engineering

Academic Year 2023

Incorporating Multi-granularity Linguistic Units in Character-based Word Segmentation

Supervisor OKUMURA Manabu, Professor

September 2023

A thesis submitted for the degree of
Doctor of Engineering

Department of Information and Communications Engineering
School of Engineering

CHAY-INTR Thodsaporn

Abstract

A character sequence tends to comprise segmentation alternatives, leading to segmentation ambiguity. Properly handling this ambiguity using multi-granularity linguistic units, such as character clusters, subwords, and words, can improve word segmentation performance and lessen ambiguous boundary decisions. We conduct a study to investigate the potential of using various linguistic units and leveraging segmentation alternatives for character-based word segmentation. Our experimental results demonstrated improvements in segmentation performance, outperforming previous work on the BCCWJ, CTB6, and BEST2010 datasets in Japanese, Chinese, and Thai, respectively.

Key Words: *Word segmentation, representation learning, linguistic units*

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Goals

Word segmentation is a fundamental task in understanding natural languages, especially for most Asian languages such as Japanese, Chinese, and Thai. The task is to determine word boundaries from a running text; in other words, it segments a character sequence into a sequence of word units. Incorrect segmentation can lead to error propagation in subsequent tasks, such as Named Entity Recognition (NER), part-of-speech (POS) tagging, and parsing [Qian and Liu, 2012, Zhang and Yang, 2018], emphasizing the importance of accurate word information.

However, while it may seem logical to use a word unit as a fundamental component in word segmentation models, this word-based approach has some substantial drawbacks. Specifically, these word-based models often struggle with issues such as ambiguity, data sparsity, and the presence of out-of-vocabulary (OOV) words [Li et al., 2019]. In contrast to the word-based models, character-based models utilize a character unit as a foundational feature, potentially alleviating the challenges. These models emphasize word-internal structures, providing a stronger word-induction ability, especially for the induction of new words [Sun, 2010]. Given its effectiveness, the character-based model serves as an effective approach for word segmentation, treating it as a sequence-labelling task. This method assigns word-boundary labels to characters in a sequence using the fine-grained BMES tagging scheme (beginning, middle, end, singleton), as demonstrated in Figure 1.1,[1] while also considering adjacent labels within the sequence. The success of this approach has been evident in recent studies for Asian languages, including Japanese, Chinese, and Thai [Higashiyama et al., 2019, Ke et al., 2021, Seeha et al., 2020].

Characters serve as fundamental units that, in various combinations, can form new words with different roles, meanings, or grammatical properties. Consequently, a character sequence inherently contains segmentation alternatives [Dyer et al., 2008], which gives rise to segmentation ambiguity. This ambiguity comes from the fact that a character sequence can be segmented into words in multiple valid ways, depending on context or intended meaning. This type of ambiguity poses a significant challenge in word segmentation, as it can lead to incorrect segmentation

[1] Here, 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.

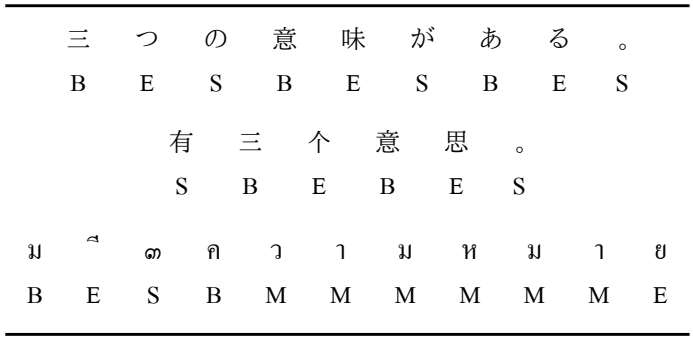


Figure 1.1: Examples of the sequence-labeling task for word segmentation using the BMES tagging scheme in Japanese (top), Chinese (middle), and Thai (bottom) languages.

results if not handled properly, thereby reducing the performance of the model. In particular, character-based models attempt to resolve this ambiguity implicitly through segmentation alternatives by learning underlying patterns and relationships between character units. Although these models could lessen ambiguity issues compared to word-based models, they rely solely on character units, which may lack the inherent meaning often found in larger units such as words. This limitation may lead character-based models to produce sub-optimal segmentation results, restricting the potential for performance improvement in the task (see Appendix A). Given this consideration, the inclusion of additional linguistic units, such as word units, alongside character units, may enhance the effectiveness of character-based models in handling segmentation alternatives, ultimately contributing to improved segmentation performance.

Previous studies have successfully utilized linguistic units either subwords [Sennrich et al., 2016] or words, in addition to character units, to alleviate the ambiguity problem in character-based word segmentation [Higashiyama et al., 2019, Yang et al., 2019, Chay-intr et al., 2021]. Notably, Yang et al. [2019], Higashiyama et al. [2019] focus on constructing a set of either potential subwords or words from a character sequence, with the aim of implicitly deriving multiple different segmentation alternatives. Subsequently, they leverage these different segmentation alternatives by using methods, such as the attention mechanism [Bahdanau et al., 2015] or long short-term memory (LSTM) [Hochreiter and Schmidhuber, 1997], to extract context features from the character sequence with its corresponding subwords or words. Finally, the context features are used to complement character representations through operations, such as concatenation or averaging.

Despite the progress made by these studies, there are still limitations to be addressed, particularly in fully utilizing various linguistic units, to enhance word segmentation performance. First, their approaches explore only one fundamental unit in addition to a character unit. Second, they

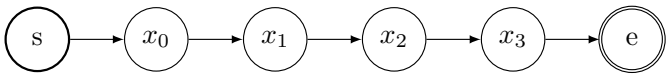
do not jointly utilize multi-granularity linguistic units such as subwords and words together. Thus, further handling segmentation alternatives using a broader range of multi-granularity structures jointly may not be fully exploited and becomes a research aspect.

A set of possible segmentation alternatives can be represented in a graph structure, specifically a multi-path lattice as shown in Figure 1.2. This allows for the explicit capture of dependencies between different linguistic units in segmentation alternatives based on multi-granularity linguistic units. Previously, Huang et al. [2021] attempts to capture these alternatives by constructing a lattice based on character and word units, along with word-boundary nodes to extract boundary information. They initialize the representation of these nodes by pre-trained models (PTMs), such as bidirectional encoder representations from transformers (BERT) [Devlin et al., 2019]. Subsequently, they employ graph neural networks (GNNs) [Kipf and Welling, 2016] to encode the lattice, thus preserving structural information and capturing dependencies between linguistic units [Yao et al., 2018]. This allows them to extract context features of these nodes in the lattice and integrate them into character representations through a concatenation operation. Despite its potential, this approach's segmentation performance is mostly on par with methods using multi-criteria (MC) segmentation across multiple datasets [Huang et al., 2020b, Ke et al., 2021], which are based on PTMs, specifically BERT.

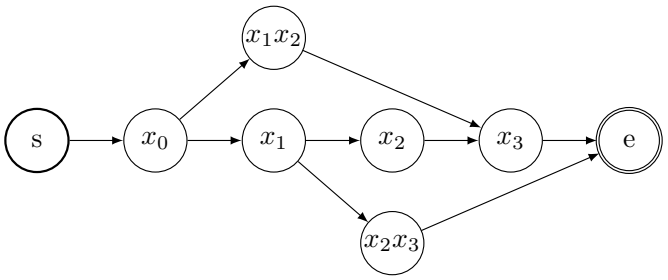
This may be due to two factors. First the method mainly focuses on constructing lattices to represent potential segmentation results. However, it only uses word boundary nodes to enhance character representations by a concatenation operation, rather than attentively using nodes from character and word units. Second, the method does not utilize multiple datasets as done by Huang et al. [2020b], Ke et al. [2021]. Instead, it relies on a single dataset for training, which might limit its effectiveness. Thus, to further improve the segmentation performance, it remains a challenge to effectively leverage segmentation alternatives based multi-granularity linguistic units through the use of lattices for complementing character representations. Addressing this challenge could enhance character representations and potentially yield better results.

In this thesis, we explore two main aspects that serve as our goals, which are derived from a thorough review of the literature. The first aspect aims to jointly utilize a broader range of multi-granularity linguistic units together in a character sequence using multiple attentions. This strategy is inspired by previous work that successfully employed multiple attentions in multi-task scenarios to estimate relationships between multiple types of knowledge [Zhang et al., 2018, Tian et al., 2020a]. To the best of our knowledge, such a strategy has not been exploited in word segmentation. Thus, we introduce multiple attentions to word segmentation, which jointly consider representations at different granularity levels, thereby enabling more effective handling of possible segmentation alternatives and improving segmentation performance.

Moreover, most studies on Asian Languages, such as Japanese and Chinese languages, rely only on subwords or words. In contrast, the Thai language offers a unique opportunity for a broader exploration due to the presence of character clusters (CCs) [Theeramunkong et al., 2000], which are indivisible units derived from predefined rules in the Thai writing system and have proven effective in Thai word segmentation [Lapjaturapit et al., 2018, Nararatwong et al., 2018]. Even though some languages, such as Japanese, comprise groups of characters



(a) Illustration of a single-path lattice



(b) Illustration of a multi-path lattice

Figure 1.2: Illustration of lattice structures: a single-path lattice and a multi-path lattice. In these diagrams, x represents a character, while **s** and **e** denote the initial and ending states, respectively. The single-path lattice depicts a character sequence $x_{0:3}$. In contrast, the multi-path lattice not only represents the character sequence but also incorporates a set of potential character sequences (words), specifically x_1x_2 and x_2x_3.

that form indivisible units similar to Thai CCs, to the best of our knowledge, these are not explicitly defined in previous studies. Instead, they typically rely on subword units, derived through statistical methods such as Byte Pair Encoding (BPE) [Sennrich et al., 2016], to extract frequently occurring consecutive characters in a corpus. Given this distinct characteristic, we introduce a method, particularly for the Thai language, that leverages the joint use of CCs, subwords, and words with multiple attentions.

The method iteratively employs an attention mechanism at each granularity linguistic unit, with character representations to gradually estimate relationships between characters. This process results in enriched features that complement the character representations. Specifically, we first extract a set of possible words from a character sequence. Then, we apply an attention mechanism to estimate a context feature that considers both the representation of each character and the representations of its possible words. Following this, we extract a set of potential CCs or subwords from the same character sequence. We then apply the attention mechanism again to estimate a more detailed context feature that considers the representation of each contextualized character feature and its potential CCs. Finally, we use these enhanced context features to complement the character representations through a concatenation operation. Experimental results regarding this method demonstrates that applying word attention followed by smaller units, either CC or subword attention, effectively improves segmentation performance on BEST2010, TNHC, and VISTEC datasets, outperforming previous Thai word segmentation methods.

The second aspect aims to incorporate multi-granularity linguistic units through the use of lattices in character-based word segmentation. We propose a method, called Lattice ATTentive Encoding (LATTE), that effectively leverages possible segmentation alternatives based on multi-granularity linguistic units, including character and word units, using a lattice structure. This

method constructs a lattice to represent possible segmentation alternatives, derived from a character sequence, using multi-granularity linguistic units, including character and word units. Subsequently, the representations of these units in the lattice are initialized with the PTM BERT and encoded using GNNs. The method then employs an attention mechanism to attentively estimate a context feature between the representation of each character with corresponding character-node and word-node features in the lattice. Finally, we integrate the context features into the representation of characters through a concatenation operation. Our experimental results regarding this method show improved segmentation performance on the BCCWJ, CTB6, and BEST2010 datasets for Japanese, Chinese, and Thai languages.

1.2 Contributions of the Thesis

Following the exploration of incorporating multi-granularity linguistic units into character-based word segmentation, as described earlier, our contributions are divided into two parts: (1) incorporating multi-granularity linguistic units using multiple attentions in character-based word segmentation, particularly for the Thai language,[2] and (2) incorporating multi-granularity linguistic units through the use of lattices in character-based word segmentation (LATTE).[3]

Regarding incorporating multi-granularity linguistic units with multiple attentions, we state our contributions as follows:

- We introduce a method for character-based Thai word segmentation that utilizes multi-granularity linguistic units with multiple attention mechanisms. This approach allows for a deeper understanding of the relationships between characters in a character sequence, generating enhanced features to complement character representations.
- Our model achieves improvements over baseline models and outperforms previous work in Thai word segmentation on three well-known benchmark datasets: BEST2010, TNHC, and VISTEC.
- Our analysis provides insightful analysis on the effectiveness of applying an attention mechanism with CCs over subword units, along with a word attention, in our model. This highlights the significance of using CCs, which adhere to the rules of the Thai writing system, for improved segmentation performance.

In reference to incorporating multi-granularity linguistic units through the use of lattices, our contributions are as follows:

- We propose a method, namely LATTE, that effectively leverages possible segmentation alternatives based on multi-granularity linguistic units through the use of lattices with

[2] https://github.com/tchayintr/thwcc-attn
[3] https://github.com/tchayintr/latte-ws

multi-criteria PTM, GNNs, and an attention mechanism. This approach generates enhanced context features from the lattice for complimenting character representations in character-based word segmentation.

- Our model demonstrates improvements over baseline models and outperforms previous work in word segmentation across three languages on well-known benchmark datasets: BCCWJ for Japanese, CTB6 for Chinese, and BEST2010 for Thai.
- Our analysis provides a detailed comparison between our model and baseline models in various aspects, including segmentation performance with additional datasets, an ablation study, and a comparison of real segmentation results. This analysis emphasizes the superior performance of our approach.

1.3 Outline of the Thesis

In the following chapters, we will first present related work in **Chapter 2**. These include a revisit of Thai word segmentation and relevant studies, particularly for Thai word segmentation, for the first aspect. For the second aspect, we introduce relevant work, including character-based and word-based approaches; segmentation alternatives; multi-granularity linguistic units; lattices; attention mechanism; GNNs; and MC word segmentation.

Chapter 3 delves into the first aspect, detailing our methodology for incorporating multi-granularity linguistic units with multiple attentions, including model architecture, experimental settings, results, and conclusion. The second aspect, which focuses on incorporating multi-granularity linguistic units through the use of lattices (i.e., LATTE), is discussed in **Chapter 4** along with experimental settings, results, and corresponding conclusion. Finally, **Chapter 5** provides the overall conclusion of the study and outlines future work.

Chapter 2

Related Work

In this chapter, we review prior studies, focusing particularly on word segmentation. We structure this chapter into two primary sections. The first section revisits studies related to our first aspect: incorporating multi-granularity linguistic units with multiple attentions, particularly for Thai character-based word segmentation. Here, we first provide a historical overview of Thai word segmentation from its early beginnings to recent developments. Subsequently, we introduce multi-granularity linguistic units, explore the application of the attention mechanism, and discuss the use of pre-trained models, all within the scope of Thai word segmentation.

The second part summarizes relevant work aligned with our second aspect: incorporating multi-granularity linguistic units through the use of lattices. We describe studies on word segmentation that are strongly related to this aspect, covering a variety of methods and approaches. These include a discussion on character-based and word-based approaches, the use of multi-granularity linguistic units, and the examination of segmentation alternatives. We also focus on studies that utilize lattices in sequence labelling, the application of the attention mechanism, and the employment of graph neural networks (GNNs). Finally, we introduce the definition of multi-criteria (MC) word segmentation and explore its potential.

2.1 Incorporating Multi-granularity Linguistic Units with Multiple Attentions

2.1.1 Historical Background of Thai Word Segmentation

Thai running text has unique characteristics as it lacks clear word boundaries and sentence periods. Spaces are not consistently used to separate words, phrases, and sentences. These characteristics make word segmentation in Thai potentially more difficult than in other languages such as Japanese and Chinese, which have delimiters to identify word and sentence boundaries.

During the early stages of Thai word segmentation, word-based methods were developed. These methods relied on pre-defined word units from dictionaries and were integrated in conjunction with machine-learning techniques, such as Markov models [Kawtrakul and Thumkanon, 1997], decision trees [Sornlertlamvanich et al., 2000, Theeramunkong and Usanavasin, 2001], and conditional random fields (CRFs) [Haruechaiyasak et al., 2008]. CRFs, in particular, have

shown effectiveness in Thai word segmentation [Kruengkrai et al., 2006, Haruechaiyasak and Kongyoung, 2009, Kruengkrai et al., 2009, Nararatwong et al., 2018].

Following this initial period, various neural network models, such as LSTM, Bidirectional LSTM (BiLSTM) [Hochreiter and Schmidhuber, 1997, Gers et al., 2000], and convolutional neural networks (CNNs) [Lecun et al., 1998], started playing a crucial role in advancing Thai word segmentation. These models were particularly employed to develop methods that fundamentally rely on character units, introducing a shift from traditional word-based approaches [Treeratpituk, 2017, Jousimo et al., 2017, Kittinaradorn et al., 2019, Chormai et al., 2019]. Remarkably, these character-based methods not only introduced a new paradigm in Thai word segmentation but also exhibited promising performance when compared to the prior word-based methods, marking a significant advancement in Thai word segmentation.

Subsequent studies have shown that the effectiveness of these neural network models can be further enhanced through the incorporation of additional knowledge such as character clusters (CCs) [Lapjaturapit et al., 2018, Nararatwong et al., 2018] and character types [Kittinaradorn et al., 2019]. Moreover, techniques such as transfer learning [Seeha et al., 2020] and stacking ensemble strategies [Limkonchotiwat et al., 2020, 2021] have also been employed to improve the model performance.

2.1.2 Multi-granularity Linguistic Units in Thai Word Segmentation

Thai comprises a variety of characters, including consonants, vowels, tones, and special characters. Words in Thai are formed by combining these characters in different ways. These linguistic units, including characters and words, have been widely used in machine learning and neural network models for Thai word segmentation. In addition to fundamental these units, Thai also presents unique linguistic phenomena where certain sequences of characters form the smallest indivisible units. These sequences adhere to specific rules of the Thai writing system; for example, a tone cannot be separated from a consonant. To capture these phenomena, Theeramunkong et al. [2000] introduced the concept of character clusters (CCs), which are defined as indivisible units conforming to these Thai writing system rules.

A CC can be described as a linguistic unit that is larger than a single character but typically smaller than a word. This concept is analogous to a subword [Sennrich et al., 2016], which also resides between a character and a word in terms of its length. Previous studies have significantly improved segmentation performance by integrating CCs into Thai word segmentation models [Theeramunkong and Tanhermhong, 2004, Sutantayawalee et al., 2014, Lapjaturapit et al., 2018, Nararatwong et al., 2018]. However, to the best of our knowledge, subwords have not yet been exploited in the context of Thai word segmentation. In particular, while CCs have been successfully employed for Thai word segmentation, subwords have demonstrated their effectiveness in Chinese word segmentation [Yang et al., 2019, Li et al., 2019].

CCs help prevent segmentation that could violate the Thai writing system [Limcharoen et al., 2009]. In contrast, subword units, lacking specificity to Thai language rules, might not fully leverage language morphology [Provilkov et al., 2020], potentially introducing noise and

reducing segmentation performance. This can be attributed to the fact that CCs are guided by Thai language principles, limiting their length to comply with specific language rules, thus generally making them smaller than subwords. This discrepancy is demonstrated in Figure 2.1, which provides a sample comparison of segmentation results from coarse to fine (top-down) across various units of linguistic granularity. Moreover, while decomposing subword units from words often requires specific settings, such as BPE [Sennrich et al., 2016], CCs do not require any settings, offering a simpler and more straightforward implementation.

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 2.1: Segmentation results comparing different levels of granularity in linguistic units. The levels are marked as S, W, Sub, CC, and C, indicating segmentation levels of sentence, word, subword, character cluster, and character, respectively. The figure illustrates the contrast in segmentation results when applying these different linguistic units.

Given these characteristics and advantages of CCs and subwords, our study aims to explore their potential for enhancing the segmentation performance of the Thai word segmentation by incorporating these linguistic units alongside character and word units.

2.1.3 Attention Mechanism in Thai Word Segmentation

The attention mechanism [Bahdanau et al., 2015, Luong et al., 2015] was initially proposed for neural machine translation, focusing on proper parts of sentences. It is fundamentally a method for estimating dependencies between source and target information. Recent studies have widely applied this method to various downstream tasks in NLP, such as word segmentation [Higashiyama et al., 2019, Tian et al., 2020a], machine translation [Luong et al., 2015, Vaswani et al., 2017], and constituency parsing [Kitaev and Klein, 2018]. Specifically, in the context of employing additional information for character-based word segmentation, the source information can refer to character units, and the target information encompasses additional linguistic units such as words related to these units [Higashiyama et al., 2019].

However, to the best of our knowledge, only our preliminary work [Chay-intr et al., 2021] has

introduced a method for applying the attention mechanism specifically to Thai character-based word segmentation. This method particularly estimates the relationships between character units and their corresponding linguistic units, including CCs, subwords, and words.

2.1.4 Pre-trained Models in Thai Word Segmentation

Recent studies have demonstrated the utility of pre-trained models (PTMs) across a variety of downstream tasks, particularly in Chinese Word Segmentation (CWS) [Yang, 2019, Qiu et al., 2020, Ke et al., 2020, Huang et al., 2020b, Ke et al., 2021]. For example, Yang [2019] incorporated a BERT to CWS and achieved superior segmentation performance compared to numerous neural models, such as CNNs and BiLSTM-based models.

However, to the best of our knowledge, only Seeha et al. [2020] have applied a transfer-learning approach for Thai character-based word segmentation using PTMs. Specifically, they pre-trained a character-based language model with BiLSTM architecture and then transferred its parameters for fine-tuning the character-based word segmentation task. This approach achieves state-of-the-art performance on the BEST2010 dataset,[1] which is the most well-known Thai dataset for evaluating a model for Thai word segmentation. Despite their model exhibiting state-of-the-art performance, it merely uses characters and neglects other linguistic units such as subwords and words [Sennrich et al., 2016, Kudo, 2018].

2.2 Incorporating Multi-granularity Linguistic Units through the Use of Lattices

2.2.1 Character-based and Word-based Approaches

Word segmentation can be categorized into two major approaches: character-based and word-based [Nakagawa, 2004, Sun, 2010]. The key distinction between these two approaches lies in the primary linguistic unit that they utilize to process a character sequence. Word-based approaches generally employ a predefined dictionary or vocabulary, leveraging these known word units to train word segmentation models for mapping a character sequence into a word sequence Zhang and Clark [2007], Cai and Zhao [2016]. These approaches have the advantage of incorporating context and semantic information at the word level, which can aid in the understanding of complex phrases and idiomatic expressions

On the other hand, character-based approaches focus on individual characters for building segmentation models. These models learn the representation of characters and assign a label for each character in a character sequence by using the BMES tagging scheme [Xue, 2003, Zheng et al., 2013, Chen et al., 2015]. Character-based approaches are particularly effective at handling out-of-vocabulary (OOV) words. They can recognize complex character patterns and the internal structures of words, thereby effectively inducing new words without dependence

[1] `https://thailang.nectec.or.th`

on predefined dictionaries [Sun, 2010]. Moreover, character-based approaches are generally simpler and more computationally efficient than word-based approaches. They require less feature engineering and can be trained on smaller datasets to achieve comparable performance.

Despite their merits, both approaches have their challenges. Word-based approaches primarily struggle with handling out-of-vocabulary (OOV) words, limiting their performance when encountering words not present in the training data or the predefined dictionary. In contrast, character-based approaches encounter the challenge of character ambiguity, where a single character, in various combinations, can form different words, each carrying a distinct role, meaning, or grammatical property. This can lead to multiple plausible segmentations of the same character sequence [Dyer et al., 2008]. Nevertheless, due to their effectiveness in handling OOV words and their overall simplicity and computational efficiency, character-based approaches have been gaining increasing attention over word-based approaches in word segmentation.

2.2.2 Segmentation Alternatives in Word Segmentation

A character sequence inherently consists of segmentation alternatives, which represent a set of potential segmentation results [Dyer et al., 2008]. These alternatives give rise to segmentation ambiguity, as a single character sequence can be interpreted in multiple ways. The lack of proper handling of these alternatives could negatively affect segmentation performance.

In prior studies, linguistic units such as subwords or words have been utilized, either implicitly or explicitly, to represent possible segmentation alternatives within a character sequence. For instance, Higashiyama et al. [2019] implicitly represented possible segmentation alternatives by utilizing a set of possible word units that correspond to a character sequence. They leveraged these segmentation alternatives to enrich character representation through the use of an attention mechanism, significantly improving segmentation performance.

Conversely, Yang et al. [2019], Huang et al. [2021] explicitly represented segmentation alternatives using a lattice-based graph structure, which is based on subword or word units. They employed neural networks models such as LSTM to capture contextual features from these segmentation alternatives for enhancing character representations. In particular, explicitly leveraging segmentation alternatives demonstrated significant improvement in segmentation performance over the implicit approach. Given these insights, our study focuses on explicitly leveraging segmentation alternatives based on multi-granularity linguistic units for this aspect, with the goal of improving segmentation performance.

2.2.3 Multi-granularity Linguistic Units in Word Segmentation

Character and word units have traditionally served as essential linguistic units for word segmentation. These units have been exploited alongside empirical methods to enhance segmentation performance [Nakagawa, 2004, Higashiyama et al., 2019, Huang et al., 2021]. Within the scope of character-based approaches, the integration of character and word units has proven successful in enriching character representations, thereby significantly improving segmentation

performance [Higashiyama et al., 2019, Tian et al., 2020c, Huang et al., 2021]. Likewise, the incorporation of subwords into character-based word segmentation has also been adopted to capture more fine-grained information from a character sequence [Yang et al., 2019].

In addition, Asian languages, such as Japanese, comprise groups of characters that form indivisible units, bearing similarity to Thai CCs. However, these units are not explicitly defined or subjected to specific rules in previous studies. Instead, such studies have relied more on subword units, extracted through statistical methods such as BPE, to identify and capture the most frequent consecutive characters in a corpus.

Despite the merits of incorporating subwords, existing studies have demonstrated that integrating word units into character-based approaches yields superior segmentation performance [Yang et al., 2019, Higashiyama et al., 2019]. Considering these findings, word units have emerged as the preferred additional linguistic unit in recent character-based word segmentation studies [Huang et al., 2021, Tang et al., 2022]. Accordingly, our study, particularly for this aspect, aims to incorporate multi-granularity linguistic units, including character and words into character-based word segmentation.

2.2.4 Attention Mechanism in Word Segmentation

As highlighted in Section 2.1.3, the attention mechanism proves particularly useful in incorporating additional information into character-based word segmentation. Specifically, it enables attentive production of context features between a character and its corresponding information.

In the domain of word segmentation of Asian languages, other than Thai, Higashiyama et al. [2019] proposed two attention-based composition functions: weighted average (WAVG) and weighted concatenation (WCON). These allow a model to effectively focus the relationships between a character and its candidate words. Both functions summarize the relationship between a character with its candidate words into a summary vector. Their work achieved state-of-the-art performance in BCCWJ, a well-known Japanese dataset, using the WCON function, despite it requiring more computational resources than WAVG. Similarly, Tian et al. [2020c] introduced a framework that incorporates wordhood information built from n-grams using memory networks. This framework also utilizes the attention mechanism through several popular encoder-decoder combinations. Alternatively, Tian et al. [2020a] utilized the attention mechanism to incorporate multiple types of linguistic knowledge by introducing a two-way attention mechanism for joint CWS and POS tagging. This method separately integrates two different types of linguistic information, including context features and linguistic knowledge, to enrich character representations. For the same task, Tian et al. [2020b] presented multi-channel attentions using various lengths of n-grams.

These approaches, i.e., Higashiyama et al. [2019], Tian et al. [2020a,b,c], that utilize the attention mechanism with additional linguistic information, have demonstrated promising performance, not only on the BCCWJ, but also across numerous Chinese datasets.

2.2.5 Lattice in Word Segmentation

A lattice, due to its ability to capture a graph or a set of possible paths, have been successfully incorporated in sequence labeling tasks such as word segmentation and NER. Various linguistic units, including characters, subwords, and words, have been employed as features in lattices, both individually and in combination to represent possible segmentation alternatives. Zhang and Yang [2018] proposed the Lattice LSTM for Chinese NER. This model outperformed common character-based and word-based models by incorporating lattices to represent possible segmentation paths, which included both characters and words, and control the information flow from the start of the sentence to the end. Li et al. [2020] introduced the flat-lattice transformer (FLAT), a transformer-based model for Chinese NER that uses a word-character lattice and its position information in a flat structure. This model demonstrated strong performance and efficiency. Furthermore, a lattice has also been deployed alongside coarse-granularity knowledge for model pre-training [Lai et al., 2021], yielding promising results on vital downstream tasks such as text classification, machine reading comprehension, and sequence labeling.

In the context of word segmentation, previous studies have typically employed lattices to represent segmentation alternatives [Nakagawa, 2004, Nakagawa and Uchimoto, 2007, Yang et al., 2019, Huang et al., 2021], leading to reliable improvements in segmentation performance and the capability to disambiguate character ambiguity [Yang et al., 2019]. Considering this advantage, we thus utilize lattices to effectively leverage segmentation alternatives based on multi-granularity linguistic units for this aspect.

2.2.6 Graph Neural Networks in Word Segmentation

Recently, there has been growing interest in applying graph neural networks (GNNs) to downstream tasks in NLP, leveraging graph structures such as lattices. A variety of GNN architectures, such as graph convolutional networks (GCNs) [Kipf and Welling, 2016], graph attention networks (GATs) [Veličković et al., 2017], and heterogeneous graph attention network (HAN or HGNN) [Wang et al., 2019], have been proposed to address specific challenges in NLP tasks.

While several studies have notably applied GNNs in the context of NER [Cetoli et al., 2017, Gui et al., 2019], their direct application to word segmentation remains less explored. To the best of our knowledge, only a few studies, including Huang et al. [2021] and Tang et al. [2022] have utilized GNNs in the context of character-based word segmentation to produce node features to complement character presentations. In their work, Huang et al. [2021] employed GCNs to aggregate word-character nodes along with its additional nodes, specifically boundary-label nodes. This approach performed well and can alleviate the problem of insufficient training from the small-scale annotated corpus. Tang et al. [2022] achieved promising results by proposing HGNSeg, a word segmentation framework that employs multi-level features including character, word, n-grams, and syntax, using a HGNN.

However, these approaches do not fully exploit the potential of GNNs, as they simply concatenate node features and character representations without explicitly considering their relation-

ships. Recognizing this, our study aims to further explore the potential of GNNs in improving character-based word segmentation by explicitly and attentively leveraging the relationships between node features and character representations.

2.2.7 Multi-Criteria Word Segmentation

A PTM is generally built on large-scale corpora, such as the SIGHAN2005[2], which includes multiple corpora to acquire prior knowledge. However, a significant challenge arises as these individual corpora are annotated according to different segmentation criteria, resulting in multi-criteria (MC) segmentations. He et al. [2017] proposed a simple yet effective model for pre-training MC word segmentation models, benefiting from multi-criteria segmentation across multiple corpora. Their approach involves adding an artificial token, referred to as a corpus-name token, at the start and end of a sentence, which serves to indicate the target corpus for the pre-training process. Despite its simplicity, this method has shown impressive segmentation performance in word segmentation. Due to its efficacy, this mechanism has been adapted by recent state-of-the-art studies in word segmentation, often in combination with empirical methods to further enhance segmentation performance [Huang et al., 2020a,b, Ke et al., 2021].

Nevertheless, it has not been extensively explored in the context of character-based word segmentation. Given its effectiveness, we thus integrate this approach, by adapting the strategy from He et al. [2017] into our study, with the goal of improving segmentation performance.

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

Chapter 3

Incorporating Multi-granularity Linguistic Units with Multiple Attentions

3.1 Methodology

Incorporating a set of possible words into the character-based BiLSTM-CRF architecture, as shown in Figure 3.1, using an attention mechanism has the potential to improve segmentation performance [Higashiyama et al., 2019]. Notably, the attention mechanism demonstrates its flexibility by allowing the integration of additional linguistic information into a character unit. Moreover, jointly employing multiple types of linguistic knowledge with multiple attentions has been proven to be effective in multi-task scenarios [Zhang et al., 2018, Tian et al., 2020a].

Building upon these insights, we explore the potential to incorporate a broader range multi-granularity linguistic units for this aspect. We utilize CCs with an attention mechanism in character-based word segmentation, as illustrated in Figure 3.2, by extending the BiLSTM-CRF architecture with word attention as proposed by Higashiyama et al. [2019]. In other words, we use characters, CCs, and words with multiple attentions for character-based word segmentation.

In our methodology, an attention mechanism is iteratively applied at each granularity level of linguistic units in conjunction with character representations. This process is designed to gradually estimate the relationships between characters, resulting in enriched features that complement the character representations. Specifically, *CC-integrated character vectors* ($\mathbf{z}$) are estimated and incorporated atop the *word-integrated character vectors* ($\mathbf{g}$), both of which have nearly identical architectures. In the following sections, we discuss the major components of our model. These include the character-embedding layer, word- and CC-embedding layers, BiLSTM layers for character representation, attention integrations for integrated representations, CRF layer, and optional BERT layers.

3.1.1 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 using a lookup table operation [Bengio et al., 2003, Collobert et al., 2011]. The lookup table is

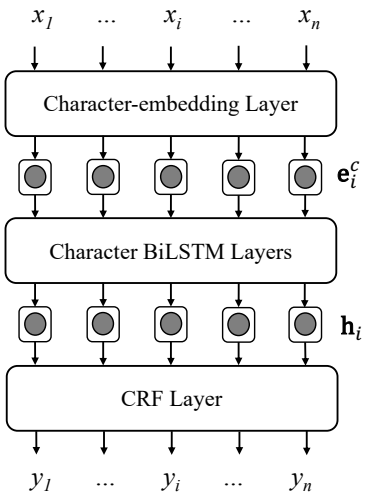


Figure 3.1: Character-based BiLSTM-CRF architecture used as a baseline, where x represents individual characters from an input sequence and y represents the predicted label for each character; $\mathbf{e}^c$ and $\mathbf{h}$ indicate an embedding representation and BiLSTM-encoded representation for each character, respectively.

defined as $\mathrm{E}^c \in \mathbb{R}^{d_c \times |V_c|}$, where d_c denotes the dimensions of the embeddings and V_c denotes a vocabulary of characters.

3.1.2 Word- and CC-Embedding Layers

Using the word-embedding layer as an example, let V_w be a word vocabulary. Given the character sequence $x_{1:n}$, V_w is searched for words within a maximum word length K, corresponding to that of the character subsequence. A candidate word list $\mathcal{W}_x \equiv (w_1, \ldots, w_m)$ (each of size within K) with m candidate words is then obtained, as shown in Figure 3.2. Each word $w_j \in \mathcal{W}_x \subseteq V_w$ was transformed into a word embedding $\mathbf{e}^w$ of a d_w-dimensional vector. The word-embedding matrix is defined as $\mathrm{E}^w \in \mathbb{R}^{d_w \times |V_w|}$, where d_w denotes the dimensions of the embeddings. This procedure is also applied to obtain a candidate CC list 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 $\mathrm{E}^{cc} \in \mathbb{R}^{d_{cc} \times |V_{cc}|}$, where d_{cc} denotes the dimensions of the embeddings, and V_{cc} denotes a CC vocabulary.

3.1.3 BiLSTM Layers for Character Representation

The character embedding sequence $\mathbf{e}^c_{1:n}$ is provided to the BiLSTM layers to acquire the *character context vectors* $\mathbf{h}_{1:n}$. The current character context vector $\mathbf{h}^l_i \in \mathbf{h}^l_{1:n}$ of the l-th layer BiLSTM

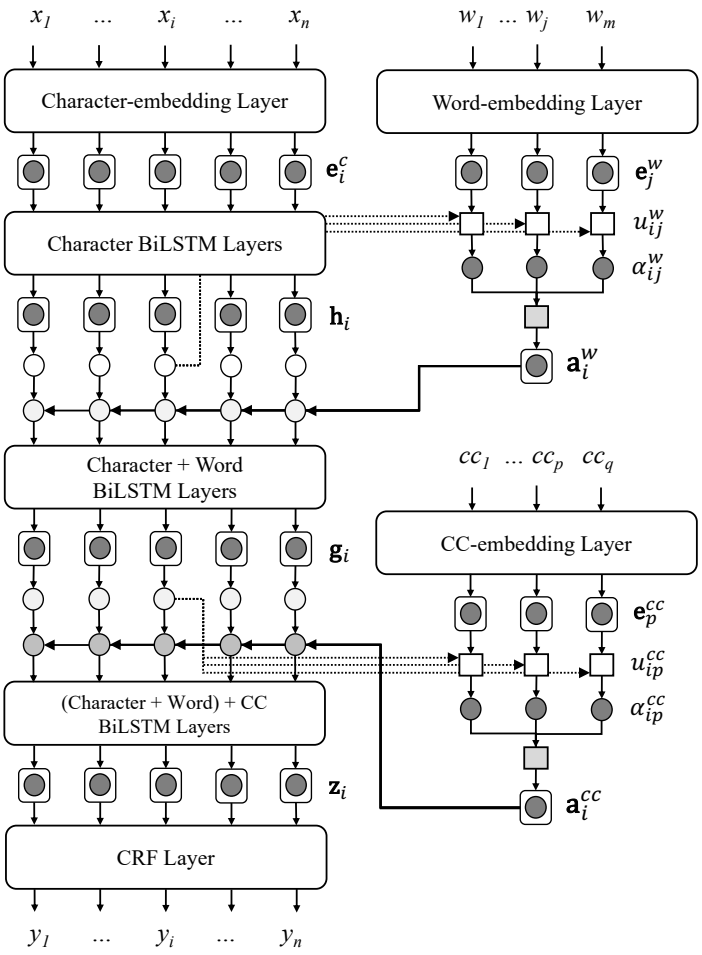


Figure 3.2: Our proposed model that integrates word and CC attentions into a character-based BiLSTM-CRF architecture.

can be computed bidirectionally, as follows:

$$\begin{aligned}\mathbf{h}_i^l &= \text{BiLSTM}(\mathbf{h}_{1:n}^{l-1}, i) \\ &\equiv \text{LSTM}_f(\mathbf{h}_{1:n}^{l-1}, i) \\ &\oplus \text{LSTM}_b(\mathbf{h}_{n:1}^{l-1}, n-i+1),\end{aligned} \tag{3.1.1}$$

where $\mathbf{h}_{1:n}^0 = \mathbf{e}_{1:n}^c$, 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.

3.1.4 Attention Integrations for Integrated Representations

We use two attention integrations, word attention and CC attention, to estimate a *word-integrated summary vector* $\mathbf{a}_i^w$ and *CC-integrated summary vector* $\mathbf{a}_i^{cc}$, respectively, for each character in the character sequence. These integrations, which are equal in architecture, summarize

the relationships among characters, words, and CCs. We apply the composition function *weight concatenation* (WCON) [Higashiyama et al., 2019, Higashiyama, 2022] to estimate both summary vectors. This function produces a word-integrated summary vector based on the relationship between a character and the corresponding candidate word. It can also be used implicitly to produce a CC-integrated summary vector based on the relationship between a character and the corresponding candidate 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} based on the character context vector $\mathbf{h}_i$ and the candidate word embedding $\mathbf{e}_j^{w}$, as follows:

$$u_{ij}^{w} = \mathbf{h}_i^{T} W_a^{w} \mathbf{e}_j^{w}, \tag{3.1.2}$$

$$\alpha_{ij}^{w} = \frac{\delta_{ij} \exp(u_{ij}^{w})}{\sum_{k=1}^{m} \delta_{ik} \exp(u_{ik}^{w})}, \tag{3.1.3}$$

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 the candidate word w_j. The word-integrated summary vector $\mathbf{a}_i^{w}$ for the character x_i can be calculated as

$$\mathbf{a}_i^{w} = \mathrm{WCON}^{w}(x_i, \{w_j\}_{j=1}^{m}) = \bigoplus_{l=1}^{L^w} \alpha_{i,i_l}^{w} \mathbf{e}_{i_l}^{w}, \tag{3.1.4}$$

where $\{w_j\} = \mathcal{W}_x$. If K^w is the maximum word length, then $L^w = \sum_{k=1}^{K^w} k$. The symbol $\bigoplus$ denotes concatenation and i_l is the corresponding index of the 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 3.1.4 when $w'_l \notin V_w$.

For example, to compose the list of words w'_4 containing a character x_4 for estimating the word-importance score $u_{4,j}^{w}$; word-attention weight $\alpha_{4,j}^{w}$; and summary vector $\mathbf{a}_4^{w}$, let the candidate word list $\mathcal{W}_x = \{w_1, \ldots, w_8\}$, maximum word length $K^w = 4$, and $L^w = \sum_{k=1}^{K^w} k = 10$. The procedure can be performed as follows:

$$\begin{aligned} w'_4 &= \bigcup_{k=1}^{4} \bigcup_{s=-k+1}^{0} \{x_{4+s:4+s+k-1}\} \\ &= \{x_{4:4}, x_{3:4}, x_{4:5}, x_{2:4}, x_{3:5}, x_{4:6}, x_{1:4}, x_{2:5}, x_{3:6}, x_{4:7}\}, \end{aligned} \tag{3.1.5}$$

where the words $x \in w'_4$ contained in the training vocabulary are used. We then use the BiLSTM layers to transform the word-integrated summary vectors $\mathbf{a}^{w}$ into word-integrated character vectors $\mathbf{g}$ using the corresponding character context vectors $\mathbf{h}$, as follows:

$$\mathbf{g}_i = \mathrm{BiLSTM}(\mathbf{h}_i \oplus \mathbf{a}_i^{w}). \tag{3.1.6}$$

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

$$u_{ip}^{cc} = \mathbf{g}_i^T W_a^{cc} \mathbf{e}_p^{cc}, \tag{3.1.7}$$

$$\alpha_{ip}^{cc} = \frac{\delta_{ip} \exp(u_{ip}^{cc})}{\sum_{k=1}^{q} \delta_{ik} \exp(u_{ik}^{cc})}, \tag{3.1.8}$$

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 candidate CC cc_p. The CC-integrated summary vector $\mathbf{a}_i^{cc}$ for the character x_i can be calculated as

$$\mathbf{a}_i^{cc} = \mathrm{WCON}^{cc}(x_i, \{cc_p\}_{p=1}^{q}) = \bigoplus_{l=1}^{L^{cc}} \alpha_{i,i_l}^{cc} \mathbf{e}_{i_l}^{cc}, \tag{3.1.9}$$

where $\{cc_p\} = CC_x$. If K^{cc} is the maximum CC length, then $L^{cc} = \sum_{k=1}^{K^{cc}} k$. Here, i_l is the corresponding index of the potential CC list CC_x for character x_i, which is represented by $\{cc'_1, \ldots, cc'_{L^{cc}}\} \equiv \bigcup_{k=1}^{K^{cc}} \bigcup_{s=-k+1}^{0} \{x_{i+s:i+s+k-1}\}$. As before, a zero vector is applied to Equation 3.1.9 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 $\mathbf{z}$ based on a CC-integrated summary vector $\mathbf{a}_i^{cc}$ and its corresponding word-integrated character vector $\mathbf{g}_i$, as

$$\mathbf{z}_i = \mathrm{BiLSTM}(\mathbf{g}_i \oplus \mathbf{a}_i^{cc}). \tag{3.1.10}$$

Finally, a CRF is used to estimate the probability of the optimal label sequence y.

3.1.5 CRF Layer

A CRF [Lafferty et al., 2001], combined with explicit consideration of the correlations between adjacent labels, has been successfully applied to sequence-labelling-related tasks [Collobert et al., 2011]. We developed a CRF layer as follows. 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 follows:

$$\mathbf{s}_i = W_s \mathbf{z}_i + \mathbf{b}_s, \tag{3.1.11}$$

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 from the transition matrix A and segmentation label scores $\mathbf{s}$, as follows:

$$\mathrm{score}(x, y) = \sum_{i=1}^{n} (A_{y_{i-1}, y_i} + \mathbf{s}_i[y_i]). \tag{3.1.12}$$

The probability of the label sequence can then be obtained as

$$P(y|x) = \frac{\text{score}(x, y)}{\sum_{y' \in T^n} \text{score}(x, y')}. \tag{3.1.13}$$

The optimal label sequence $y^\star$ can be obtained by maximising the sentence score using the Viterbi algorithm:

$$y^\star = \text{argmax}_{y \in T^n} \text{score}(x, y). \tag{3.1.14}$$

The loss function $\mathcal{L}$ is minimised by backpropagation during the training process:

$$\mathcal{L}(x, y) = -\log P(y|x). \tag{3.1.15}$$

3.1.6 BERT Layers

In this aspect of our study, BERT layers were used for comparison with BiLSTM-based models. A $\text{BERT}_{\text{base}}$ layer[1] was used to extract three types of representations: contextual-character-, word-, and CC-representation. Specifically, the BiLSTM layers for character representation in Section 3.1.3 were replaced with a BERT-encoder layer to compute the *character context vectors* $\mathbf{h}_{1:n}$, as shown in Figure 3.3. In this architecture, the current character context vector $\mathbf{h}_i$ is computed as follows:

$$\mathbf{h}_i^0 = \mathbf{e}^c{}_i + \mathbf{e}_i^p, \tag{3.1.16}$$

$$\mathbf{h}_i^l = \text{Transformer}(\mathbf{h}_i^{l-1}), \tag{3.1.17}$$

where $l = \{1, 2, \ldots, L\}$ is the number of transformer layers [Vaswani et al., 2017], $\mathbf{e}^c$ is the embedding obtained from the BERT embedding layer, and $\mathbf{e}^p$ is the positional embedding. Because we adopted $\text{BERT}_{\text{base}}$, $L = 12$ and $e^c \in \mathbb{R}^{d_c \times |V_{\text{BERT}}|}$, where $d_c = 768$ and V_{BERT} denotes the vocabulary in $\text{BERT}_{\text{base}}$. The word- and CC-representations (Section 3.1.2 are obtained simply from the BERT embedding layer. In other words, word- and CC-embedding layers (as well as subword-embedding layers) were replaced with a BERT-embedding layer. When using the BERT embedding layers, the word- and CC-embedding matrices are defined as $\text{E}^w \in \mathbb{R}^{d_w \times |V_{\text{BERT}}|}$ and $\text{E}^{cc} \in \mathbb{R}^{d_{cc} \times |V_{\text{BERT}}|}$, where $d_w = 768$ and $d_{cc} = 768$. Finally, the BERT-pooler layer and CRF layer conditionally project the BERT representations from the encoder layers into the optimal label sequences $y^\star$.

[1] https://huggingface.co/bert-base-multilingual-cased

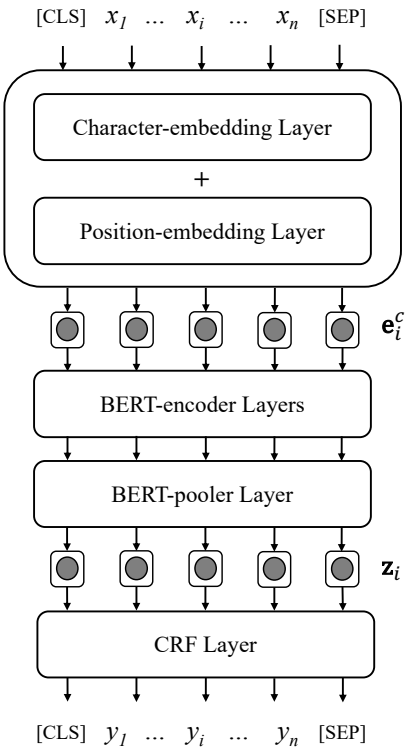


Figure 3.3: BERT-integrated character-based word segmentation model.

3.2 Experiments

3.2.1 Datasets

We trained and evaluated several versions of our model on three datasets: BEST2010, TNHC,[2], and VISTEC.[3] BEST2010 corpus consists of four different domains of textual documents—articles, encyclopaedia, news, and novels—and has frequently been used to evaluate Thai word segmentation models. TNHC and VISTEC are collections of Thai classical literature and social media datasets, respectively, for Thai text processing. The latter two datasets were recently used to evaluate domain-dependent word segmentation. For our purposes, we randomly split the BEST2010 corpus into three sets:[4] 80% for training, 10% for validation, and 10% for testing. For TNHC and VISTEC, we applied that data splits as in Limkonchotiwat et al. [2020, 2021]. The sizes of all the datasets are listed in Table 3.1.

Dataset	**Set**	**|S|**	**|W|**	**|V|**	**|Ch|**
BEST2010	Train	119K	4M	72.9K	16M
	Valid	14.9K	501.4K	23K	1.9M
	Test	14.9K	500.4K	23K	1.9M
TNHC	Train	12.7K	0.3M	16.8K	1.3M
	Valid	1.4K	47.2K	5.3K	0.1M
	Test	4.4K	125K	9.2K	0.4M
VISTEC	Train	36K	2.4M	98.5K	9.5M
	Valid	4K	270K	23.7K	1.1M
	Test	10K	677.4K	42.9K	2.6M

Table 3.1: Data sizes, including the number of sentences (S); words (W); characters (Ch); and the size of the vocabulary (V), for the BEST2010, TNHC, and VISTEC datasets.

3.2.2 Subword-Integration

In our exploration of a broader range of utilizing multi-granularity linguistic units for character-based word segmentation task, we also consider the integration of subword units. While subword units have been successfully applied to these tasks, they may generate noise that decreases segmentation performance when a word dictionary already exists. Therefore, using the different versions of our proposed model, we conducted a comparison between using subword units and CCs because of their similarity. To illustrate, let V_{sw} be a subword vocabulary decomposed from a dataset. We simply replaced the CC vocabulary V_{cc} with the decomposed subword vocabulary V_{sw}. Thus, a candidate list of subwords $\mathcal{SW}_x$ can be acquired and used for subword attention integration by applying Equations 3.1.7 and 3.1.9.

[2] https://attapol.github.io/tlc

[3] https://github.com/mrpeerat/OSKut/tree/main/VISTEC-TP-TH-2021

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

3.2.3 Attention Integration Order

Given that attention integrations in our model are applied sequentially, the order in which these integrations occur can potentially influence the segmentation performance, and it can be switched. For instance, in the base version, our model first performs CC attention integration to estimate the relationships between characters and CCs, which is then followed by word attention integration This might affect the segmentation performance because each integration provides different information. To explore this possibility, we implemented a swapped version of our model (Swap) in which the integration order was switched and compared its segmentation performance with that of the base model.

3.2.4 Pre-Trained Model Integration

Fine-tuning a PTM for word segmentation has proven to be effective in previous studies. Thus, as mentioned prior, we conducted character-based Thai word segmentation with a PTM (shown in Figure 3.3), by simply replacing the character BiLSTM layers in Figure 3.1 with BERT layers, as described in Section 3.1.6. We chose M-BERT[5] for our experiment because of its originality and accessibility. Furthermore, we applied this approach to the baseline model and our proposed model, where CCs, subwords, or words were used for model comparison. In our experiment, each character x_i was transformed into a BERT-token id. Thereafter, each token id was encoded with a BERT encoder to obtain a character BERT representation. Similarly, candidate word w_j and either candidate CC cc_p or candidate subword unit sw_p, obtained from Section 3.1.4, were also transformed into a BERT-token id. Unlike the character BERT-representation, such candidate units are directly encoded into CC, subword, and word BERT-representations from the BERT-embedding layer. The character BERT-representation, along with the corresponding word BERT-representation and either CC BERT-representation or subword BERT-representation, were used to estimate the importance scores u and the attention weights α (Section 3.1.4).

Note that the BERT tokenizer was not used to tokenise the sentence s into subword units. Hence, the candidate CC, subword, and word units, can potentially become unknown tokens (UNK). This circumstance arises when some tokens are not included in the BERT tokenizer and therefore cannot be converted into BERT-token ids. Therefore, the resulting tokenisation heavily depends on the specific PTM and the approach used to construct subword unit in this experiment. In addition, we included special tokens, including CLS and SEP tokens, in the training, validation, and test steps as the S (single-word) label of BMES tagging scheme. However, these special tokens were not counted during the testing step to avoid overestimation.

[5]https://huggingface.co/bert-base-multilingual-cased

3.2.5 Hyperparameters

We used CCs from publicly available libraries, including Phatthiyaphaibun et al. [2016] and TCCSEG[6], to build the CC vocabulary V_{cc}. To generate the subword vocabulary V_{sw} for each dataset, we decomposed raw sentences from the training sets into subword units of various sizes using BPE implemented in SentencePiece [Kudo and Richardson, 2018]. The generated subword vocabulary for each dataset was not shared with the other experimental datasets.

We used common hyperparameters for training the different versions of our proposed model (hereafter, referred to as "our models"), as shown in Table 3.2. Dropout [Srivastava et al., 2014] was applied to the BiLSTM layers to avoid overfitting and non-recurrent layers [Zaremba et al., 2015]. The model parameters were optimised using the Adam optimiser [Kingma and Ba, 2015]. We trained our models up to 20 epochs and chose the best one based on a validation process involving word-level evaluation.[7]

Parameter	**Value**
Character-embedding size	128
BiLSTM layers	2
BiLSTM hidden size	600
Mini-batch size	128
BiLSTM Initial learning rate	0.001
Learning rate decay	0.99
Recurrent layer dropout rate	0.4
Word-embedding size	300
Word-vector dropout rate	0.4
Maximum word (chunk) length	4
BERT hidden size	768
BERT initial learning rate	0.00002
Maximum sequence length	512
CC/subword-embedding size	300
CC/subword-vector dropout rate	0.4
Maximum CC/subword length	0.4

Table 3.2: Common hyperparameters for Baselines and our models (top/middle), with exclusive values to our models (bottom). Note: hyperparameters designated for CC integration can also be applied to subword integration.

3.2.6 Compared Models

The following models were evaluated:

[6] https://github.com/tchayintr/tccseg
[7] https://github.com/spyysalo/conlleval.py

- **Baseline**: A character-based BiLSTM-CRF architecture, as shown in Figure 3.1.
- **Baseline w/ Word**: An extension of the Baseline that integrates word attention (BiLSTM-CRF with word attention) [Higashiyama et al., 2019].
- **Baseline w/o BiLSTM w/ BERT**: A character-based BERT-CRF architecture that replaces BiLSTM layers for character representation with BERT layers, as shown in 3.3.
- **Baseline w/ BERT w/ Word**: The Baseline w/ Word that replaces BiLSTM layers for character representation with BERT (BERT-BiLSTM-CRF with word attention).
- **OURS**: Our proposed model that integrates word and CC attentions (BiLSTM-CRF with word and CC attentions), as shown in Figure 4.1.
- **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 w/o Word**: Our proposed model that removes word attention (BiLSTM-CRF with CC attention).
- **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.
- **OURS w/ BERT**: Our proposed model that integrates word and CC attentions, and replaces BiLSTM layers for character representation with BERT (BERT-BiLSTM-CRF with word and CC attentions).
- **OURS w/ BERT w/o CC w/ Sub**: Our proposed model that integrates word and subword unit attentions, and replaces BiLSTM layers for character representation with BERT (BERT-BiLSTM-CRF with word and subword attentions).
- **Others**: Reproduced Thai non-neural/neural word-segmentation models, including well-known models, the state-of-the-art Thai word-segmentation model [Seeha et al., 2020], and recent domain-adaptation models [Limkonchotiwat et al., 2020, 2021].

3.2.7 Evaluation Metrics

We evaluated our models on the test data using two evaluation metrics: character-level evaluation, word-level evaluation. We also used them to compare our model with recent Thai domain-adaptation word segmentation works [Limkonchotiwat et al., 2020, 2021].

Note that our F_1 scores were based on the micro-averaged F_1 scores for all evaluation matrices. We conducted statistical significance tests using paired bootstrap resampling [Koehn, 2004] on our results, particularly on OURS with state-of-the-art Thai word segmentation [Seeha et al., 2020], OURS with Baseline w/ Word [Higashinaka et al., 2021], and OURS w/o CC w/ Sub. We set the resampling size to 100,000 iterations and the sample size for each resampling to 10% of the test data.

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

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

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

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.

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

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

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

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.

3.3 Results and Analysis

3.3.1 Main Results

Table 3.3[8] illustrates the evaluation results of the compared models on BERT2010, while Table 3.4 shows the architectures of the compared models. According to these results, the best of our models was OURS, which performed better than all the other models. The statistical significance test further confirmed that OURS outperformed both the state-of-the-art and strong baseline models, including Baseline w/ Word and OURS w/o CC w/ Sub. Although using subword units showed the potential to slightly improve the performance compared with that when using CCs, OURS outperformed OURS w/o CC w/ Sub in multiple runs and performed significantly better at p-level < 0.05. This demonstrates the superior effectiveness of CCs over subword units for segmentation performance. Moreover, unlike constructing subwords, CCs do not require hyperparameter adjustments. OURS, which used word and CC units, outperformed Baseline w/ Word, indicating that CCs can be used to complement word units.

No non-neural models were able to outperform any neural model. However, TLTK and New Multi-Cut, which incorporate additional linguistic information, that is, syllables and CCs, demonstrated superior performance over LexTo and Multi-Cut. The results indicate that the inclusion of syllables can enhance the Word-F1 score, while integrating CCs can improve Char-F1. In addition, we implemented an additional model that replaces the BiLSTM layers with transformer encoder layers [Vaswani et al., 2017] in Baseline, using the hyperparameters for the transformer layers from Vaswani et al. [2017]. However, the results were noticeably poorer than those of other models.

3.3.2 Subword-Integration Performance

We implemented OURS w/o CC w/ Sub on various subword unit vocabulary sizes, as shown in Table 3.5. The results indicate that vocabulary size affects the performance of subword-integration. Specifically, providing a larger subword vocabulary to the model consistently improves its overall performance. It is suggested that OURS w/o CC w/ Sub may match the performance of the OURS when the subword vocabulary size is sufficiently large. However, it may be a challenge to determine the appropriate vocabulary size to 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 on the performance of those with 3,200 and 6,400 subword tokens. Thus, the size of the subword vocabulary is a crucial parameter that affects the performance of this model.

We chose the best subword-integration model based on validation performance to compare with OURS, as shown in Table 3.3. Both subword- and CC-integration tended to act as an additional filter layer for word-integrated character representations and improved the segmentation performance. However, OURS outperformed OURS w/o CC w/Sub in every evaluation. We

[8] LexTo (http://www.sansarn.com/lexto), TLTK (https://github.com/attapol/tltk), Multi-cut, and New Multi-cut (https://github.com/PyThaiNLP) were used to produce non-neural segmentation results.

believe that the main reason for this is that subword units contain noise, while CCs do not. For example, there may be a unit that is included in the subword vocabulary but does not exist in the Thai word vocabulary and violates the Thai writing system (for example, a combination of one Thai vowel followed by one Thai consonant), whereas CCs will not include this type of unit.

3.3.3 Order-of-Integration Performance

As mentioned prior, we also compared the performance of our model when the order of attention integration was swapped. Table 3.3 shows that the swapped models exhibited decreased segmentation performance compared with the original models, especially in the case of the swapped subword-integration model. We suggest that subword-integration adds initial noise to character representations, as explained above, making it difficult for the model to complement such representations in the word-attention integration.

OURS Swap slightly increased in performance compared with OURS w/o CC w/ Sub Swap because the CC vocabulary consists of smaller units that reflect the Thai writing system and

Model	**Method**	$\mathbf{F_{char}}$ (σ)	$\mathbf{F_{word}}$ (σ)
LexTo $^{\circ\odot}$	Dict w/ Longest-matching	84.38	67.29
TLTK $^{\circ\odot}$	Dict w/ Maximum-collocation w/ Syllable	86.00	74.49
Multi-Cut $^{\circ\odot}$	Dict w/ Maximum-matching	83.34	60.36
New Multi-Cut $^{\circ\odot}$	Dict w/ Maximum-matching w/ CC	86.39	68.63
[Treeratpituk, 2017]$^{\circ}$	LSTM	96.53	92.49
[Chormai et al., 2019]$^{\circ}$	CNN w/ Syllable	91.36	93.79
[Kittinaradorn et al., 2019]$^{\circ}$	CNN w/ Char-type	98.17	95.82
[Lapjaturapit et al., 2018]$^{\circ}$	BiLSTM w/ CC	98.43	96.22
[Seeha et al., 2020]$^{\circ\star}$	BiLSTM w/ Char-PTM	98.80	<u>97.20</u>
Baseline	BiLSTM-CRF	98.28 (0.117)	96.71 (0.120)
Baseline w/ Word	BiLSTM-CRF w/ Word	98.94 (0.008)	<u>97.57</u> (0.003)
Baseline w/o BiLSTM w/ BERT	BERT-CRF	98.77 (0.029)	97.18 (0.012)
Baseline w/ BERT w/ Word	BERT-BiLSTM-CRF w/ Word	98.65 (0.095)	96.79 (0.310)
OURS	BiLSTM-CRF w/ Word w/ CC	**98.99** (0.005)	**97.67** (0.020)
OURS w/o CC w/ Sub	BiLSTM-CRF w/ Word w/ Sub	98.96 (0.021)	<u>97.60</u> (0.035)
OURS w/o Word	BiLSTM-CRF w/ CC	98.91 (0.012)	97.42 (0.020)
OURS Swap	BiLSTM-CRF w/ Word w/ CC Swap	98.97 (0.012)	97.61 (0.010)
OURS w/o CC w/ Sub Swap	BiLSTM-CRF w/ Word w/ Sub Swap	98.87 (0.046)	97.45 (0.041)
OURS w/ BERT	BERT-BiLSTM-CRF w/ Word w/ CC	98.92 (0.020)	97.24 (0.030)
OURS w/ BERT w/o CC w/ Sub	BERT-BiLSTM-CRF w/ Word w/ Sub	98.76 (0.049)	97.11 (0.081)

Table 3.3: Comparison of segmentation performance among our models, baselines, and others on the BEST2010 dataset. Best score for each metric is indicated in **bold**. OURS was significantly better than the state-of-the-art Thai word segmentation model, Baseline w/ Word, and OURS w/o CC w/ Sub (<u>underlined</u> scores) at p-level < 0.05 in pairwise comparison. All models were evaluated based on the same dataset division. The scores were obtained from the mean of three runs. OURS w/o CC w/ Sub scores were reported from the best validation performance among various subword vocabulary sizes. The symbols $\circ$, $\odot$, and $\star$ indicate reproduced, non-neural, and the state-of-the-art Thai word segmentation models, respectively; σ represents population standard deviation.

Model	Method	Word	CC	Sub	LSTM	PTM
LexTo °⊙	Dict w/ Longest-matching	✓				
TLTK °⊙	Dict w/ Maximum-collocation w/ Syllable	✓				
Multi-Cut °⊙	Dict w/ Maximum-matching	✓				
New Multi-Cut °⊙	Dict w/ Maximum-matching w/ CC	✓	✓			
[Treeratpituk, 2017]°	LSTM				✓	
[Chormai et al., 2019]°	CNN w/ Syllable					
[Kittinaradorn et al., 2019]°	CNN w/ Char-type					
[Lapjaturapit et al., 2018]°	BiLSTM w/ CC		✓		✓	
[Seeha et al., 2020]°⋆	BiLSTM w/ Char-PTM				✓	✓
Baseline	BiLSTM-CRF				✓	
Baseline w/ Word	BiLSTM-CRF w/ Word	✓			✓	
Baseline w/o BiLSTM w/ BERT	BERT-CRF					✓
Baseline w/ BERT w/ Word	BERT-BiLSTM-CRF w/ Word	✓			✓	✓
OURS	BiLSTM-CRF w/ Word w/ CC	✓	✓		✓	
OURS w/o CC w/ Sub	BiLSTM-CRF w/ Word w/ Sub	✓		✓	✓	
OURS w/o Word	BiLSTM-CRF w/ CC		✓		✓	
OURS Swap	BiLSTM-CRF w/ Word w/ CC Swap	✓	✓		✓	
OURS w/o CC w/ Sub Swap	BiLSTM-CRF w/ Word w/ Sub Swap	✓		✓	✓	
OURS w/ BERT	BERT-BiLSTM-CRF w/ Word w/ CC	✓	✓		✓	✓
OURS w/ BERT w/o CC w/ Sub	BERT-BiLSTM-CRF w/ Word w/ Sub	✓		✓	✓	✓

Table 3.4: Comparison of architectures among our models, baselines, and others. A ✓indicates whether Words, CCs, Subwords, LSTM, and PTMs were used in the model.

Model	$\mathbf{F}_{\text{char}}$	$\mathbf{F}_{\text{word}}$
OURS w/o CC w/ Sub800	98.94	97.56
OURS w/o CC w/ Sub1600	98.96	97.59
OURS w/o CC w/ Sub3200	98.95	97.65
OURS w/o CC w/ Sub6400	98.98	97.65
OURS w/o CC w/ Sub12800	98.95	97.65

Table 3.5: Results of segmentation performance for our subword-integration model (OURS w/o CC w/ Sub) with various subword vocabulary sizes from 800 to 12,800 tokens. The best result of three runs is reported. An underline indicates the model that obtained the best score in the validation process.

includes no noise information. Results showed that OURS outperformed both swapped models and that word information is preferential to complement a character representation, whereas fine-grained information, that is, CCs and subword units, is more suitable for use after word information as an additional filter layer.

3.3.4 Pre-Trained Model Performance

In this experiment, we replaced BiLSTM layers with BERT layers to extract *character context vectors* while simply extracting word and CC representations from the BERT-embedding layer. From the results shown in Table 3.3, applying the pre-trained BERT model to the baseline improved its segmentation performance on BEST2010. However, the BERT-integrated models

that incorporated additional linguistic units, that is, Baseline w/ BERT w/ Word, OURS w/ BERT, and OURS w/ BERT w/o CC w/ Sub, exhibited clearly reduced segmentation performance compared with the original models. We believe the main reason is that the pre-trained BERT model can produce unknown tokens (UNK) to represent unknown characters or candidate words, CCs, and subword units, which are not included in the BERT vocabulary, instead of using their own representation. This can be considered a limitation of applying the existing pre-trained BERT model for straightforward fine-tuning without expanding the BERT vocabulary to cover all possible candidate words, CCs, and subword units. We expect that the segmentation performance could be improved if the pre-trained BERT model were pre-trained with the training data, and the BERT vocabulary is appended with candidate CCs, subwords, and words.

Although the segmentation performance in our model, that is, OURS w/ BERT and OURS w/ BERT w/o CC w/ Sub, decreased when it was combined with the pre-trained BERT model, their results still outperformed Baseline w/ BERT w/ Word. This indicates that incorporating a pre-trained BERT model with multi-granularity linguistic units, including words, and either CCs or subwords, is useful. By comparing the BiLSTM-based models with the BERT-integrated models, most of the BiLSTM-based models outperformed their own BERT-integrated versions; only Baseline did not outperform Baseline w/o BiLSTM w/ BERT.

3.3.5 Comparison with Thai Domain-Adaption Models

To evaluate our models on additional datasets, we compared them with recent studies that focused on domain-adaptation scenarios using specific datasets: TNHC and VISTEC. The baseline and

Model	**Method**	**TNHC**	
		$\mathbf{F}_{\text{char}}$ (σ)	$\mathbf{F}_{\text{word}}$ (σ)
LexTo $^{\circ\odot}$	Dict w/ Longest-matching	88.10	69.92
TLTK $^{\circ\odot}$	Dict w/ Maximum-collocation w/ Syllable	86.95	71.81
Multi-Cut $^{\circ\odot}$	Dict w/ Maximum-matching	83.96	67.62
New Multi-Cut $^{\circ\odot}$	Dict w/ Maximum-matching w/ CC	88.56	70.32
[Limkonchotiwat et al., 2020]$^{\bullet}$	SE-DC (Stack Ensemble)	95.2	84.1
[Limkonchotiwat et al., 2021]$^{\circ}$	DSE-DC (Deep Stack Ensemble)	95.71	85.74
Baseline	BiLSTM-CRF	98.17 (0.052)	95.91 (0.028)
Baseline w/ Word	BiLSTM-CRF w/ Word	98.39 (0.051)	96.40 (0.163)
Baseline w/o BiLSTM w/ BERT	BERT-CRF	98.06 (0.041)	95.55 (0.037)
Baseline w/ BERT w/ Word	BERT-BiLSTM-CRF w/ Word	98.31 (0.035)	95.46 (0.040)
OURS	BiLSTM-CRF w/ Word w/ CC	98.54 (0.016)	**96.65** (0.028)
OURS w/o CC w/ Sub	BiLSTM-CRF w/ Word w/ Sub	98.41 (0.021)	96.39 (0.032)
OURS w/ BERT	BERT-BiLSTM-CRF w/ Word w/ CC	**98.63** (0.016)	96.36 (0.023)
OURS w/ BERT w/o CC w/ Sub	BERT-BiLSTM-CRF w/ Word w/ Sub	98.03 (0.042)	94.71 (0.030)

Table 3.6: Comparison of segmentation performance between our models, domain-adaptation models, baselines, and others on the TNHC dataset. Best score for each metric is indicated in **bold**. Scores were obtained from the mean of three runs. OURS performed significantly better than Baseline w/ Word and OURS w/o CC w/ Sub (underlined scores) at p-level < 0.05. A $\bullet$ indicates a reported score from literature; $^{\circ}$ and $\odot$ indicate scores reproduced from neural models and non-neural models, respectively. σ represents population standard deviation.

Model	Method	VISTEC	
		$\mathbf{F}_{\text{char}}$ (σ)	$\mathbf{F}_{\text{word}}$ (σ)
LexTo $^{\circ\odot}$	Dict w/ Longest-matching	87.54	70.43
TLTK $^{\circ\odot}$	Dict w/ Maximum-collocation w/ Syllable	88.66	76.73
Multi-Cut $^{\circ\odot}$	Dict w/ Maximum-matching	84.52	64.41
New Multi-Cut $^{\circ\odot}$	Dict w/ Maximum-matching w/ CC	88.75	73.37
[Limkonchotiwat et al., 2020]$^{\bullet}$	SE-DC (Stack Ensemble)	-	-
[Limkonchotiwat et al., 2021]$^{\circ}$	DSE-DC (Deep Stack Ensemble)	95.07	88.48
Baseline	BiLSTM-CRF	96.81 (0.016)	91.70 (0.041)
Baseline w/ Word	BiLSTM-CRF w/ Word	97.09 (0.029)	92.42 (0.067)
Baseline w/o BiLSTM w/ BERT	BERT-CRF	97.10 (0.020)	92.23 (0.020)
Baseline w/ BERT w/ Word	BERT-BiLSTM-CRF w/ Word	97.04 (0.016)	92.19 (0.008)
OURS	BiLSTM-CRF w/ Word w/ CC	**97.18** (0.012)	**92.65** (0.016)
OURS w/o CC w/ Sub	BiLSTM-CRF w/ Word w/ Sub	97.04 (0.016)	92.27 (0.008)
OURS w/ BERT	BERT-BiLSTM-CRF w/ Word w/ CC	97.10 (0.012)	92.40 (0.008)
OURS w/ BERT w/o CC w/ Sub	BERT-BiLSTM-CRF w/ Word w/ Sub	96.92 (0.074)	92.09 (0.084)

Table 3.7: Comparison of segmentation performance between our models, domain-adaptation models, baselines, and others on the VISTEC dataset. Best score for each metric is indicated in **bold**. Scores were obtained from the mean of three runs. OURS performed significantly better than Baseline w/ Word and OURS w/o CC w/ Sub (underlined scores) at p-level < 0.05. A • indicates a reported score from literature; $^{\circ}$ and $\odot$ indicate scores reproduced from neural models and non-neural models, respectively. σ represents population standard deviation.

our models were evaluated in an in-domain scenario, that is, the models were trained on the targeted domain training data and evaluated on the target domain test data. Specifically, we performed these model based on training, validation, and test splits for TNHC and VISTEC. For reference, the results of previous models [Limkonchotiwat et al., 2020, 2021] trained in a domain adaptation scenario (using both source and target domain training data) were reported.

Tables 3.6 and 3.7 shows the results of our models and other domain-adaptation models in character-level and word-level evaluations, as in the work of [Limkonchotiwat et al., 2020, 2021].[9] OURS and OURS w/ BERT surpassed all domain-adaptation models and outperformed every reproduced model. Specifically, mirroring the tendency observed in Section 3.3.1, OURS was statistically significantly better than OURS w/o CC w/ Sub at p-level < 0.05. Moreover, OURS outperformed OURS without CC w/Sub in multiple runs. This emphasises the benefits of using CCs over subword units. Although we reproduced Limkonchotiwat et al. [2021] from their original repository, the reproduced results were lower than their reported scores: by 3.4 for the Word-F1 scores on TNHC and by 2.29–4.43 for the Char/Word-F1 scores on VISTEC.

The results of the non-neural models showed the same tendency as in Section 3.3.1, that is, no non-neural model was able to outperform any neural models. In addition, the results of the non-neural models also supported the hypothesis that the use of additional linguistic information, that is, syllables and CCs, could improve segmentation performance, as in TLTK and New Multi-Cut. Specifically, by incorporating syllables, TLTK achieved outstanding Word-F1 scores, while New Multi-Cut, which incorporates CCs, performed well with regards to Char-F1.

[9]Results for the DSE-DC method from Limkonchotiwat et al. [2021] were obtained using the same dataset division as their official model from https://github.com/mrpeerat/OSKut.

3.3.6 Case Study: Segmentation Results

Figure 3.4 presents examples of segmentation results among four models: Baseline, Baseline/w Word, OURS, and OURS w/o CC w/ Sub, utilizing the BEST2010 dataset. OURS perfectly segmented the example sentence; however, other models yielded incorrect results. Specifically, underlined word violates the Thai writing system by combining the two consonants in the word. We believe that CC-integration successfully filters out this type of violation from the word-integrated character representations, enabling OURS to outperform the other models.

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
OURS	ป๊อ อุ้ยอ้าย ยก ลัง เข้า ไป เก็บ ไว้ ใน ตู้ เสื้อผ้า
OURS w/o CC w/ Sub	ป๊อ อุ้ยอ้ายย กลัง เข้า ไป เก็บ ไว้ ใน ตู้ เสื้อผ้า

ป๊ออุ้ยอ้ายยกลังเข้าไปเก็บไว้ในตู้เสื้อผ้า

"Por clumsily put a crate into a closet."

Reference	ปิยนุช เห็น ท่า ว่า เรื่อง ชัก จะ วก มา หา ตัว เธอ มาก ขึ้น ทุกที จึง รีบ เปลี่ยน เรื่อง พูด
Baseline	ปิยนุช เห็น ท่า ว่า เรื่อง ชัก จะวก มา หา ตัว เธอ มาก ขึ้น ทุก ที จึง รีบ เปลี่ยน เรื่อง พูด
Baseline w/ Word	ปิยนุช เห็น ท่า ว่า เรื่อง ชักจะวก มา หา ตัว เธอ มาก ขึ้น ทุก ที จึง รีบ เปลี่ยน เรื่อง พูด
OURS	ปิยนุช เห็น ท่า ว่า เรื่อง ชัก จะ วก มา หา ตัว เธอ มาก ขึ้น ทุกที จึง รีบ เปลี่ยน เรื่อง พูด
OURS w/o CC w/ Sub	ปิยนุช เห็น ท่า ว่า เรื่อง ชัก จะ วก มา หา ตัว เธอ มาก ขึ้น ทุก ที จึง รีบ เปลี่ยน เรื่อง พูด

ปิยนุชเห็นท่าว่าเรื่องชักจะวกมาหาตัวเธอมากขึ้นทุกทีจึงรีบเปลี่ยนเรื่องพูด

"Piyanuch realized that discussing the story could cause her problems,

so she quickly changed the topic of conversation."

Figure 3.4: Examples of segmentation results comparing baseline models with our models, using the BEST2010 dataset. The ground-truth segmentation result is indicated as "Reference" and incorrect segmentation results are in red. Underlines indicates segmentation results that violate the Thai writing system.

3.4 Conclusion for this Chapter

In this chapter, we presented a character-based Thai word segmentation model that explores a broader range of multi-granularity linguistic units, including CCs, subwords, and words, with multiple attentions. Our model outperformed the state-of-the-art Thai word segmentation model by using word attention along with CC attention in the BiLSTM-CRF architecture.

Comparisons between BiLSTM-based models and BERT-based models indicated that BiLSTM-based models surpassed BERT-based models, particularly when applying our method. However, we observed improvements over the baseline model when integrating a pre-trained BERT model into our proposed approach. Further analysis also suggests that the incorporation of CCs could lead to better performance than using subword units in character-based Thai word segmentation.

Chapter 4

Incorporating Multi-granularity Linguistic Units through the Use of Lattices

4.1 Methodology

In this section, we provide an overview of our LATTE methodology and then discuss it in detail. Given a sentence s with n characters, that is, a character sequence $x_{1:n} = (x_1, x_2, \ldots, x_n)$, the

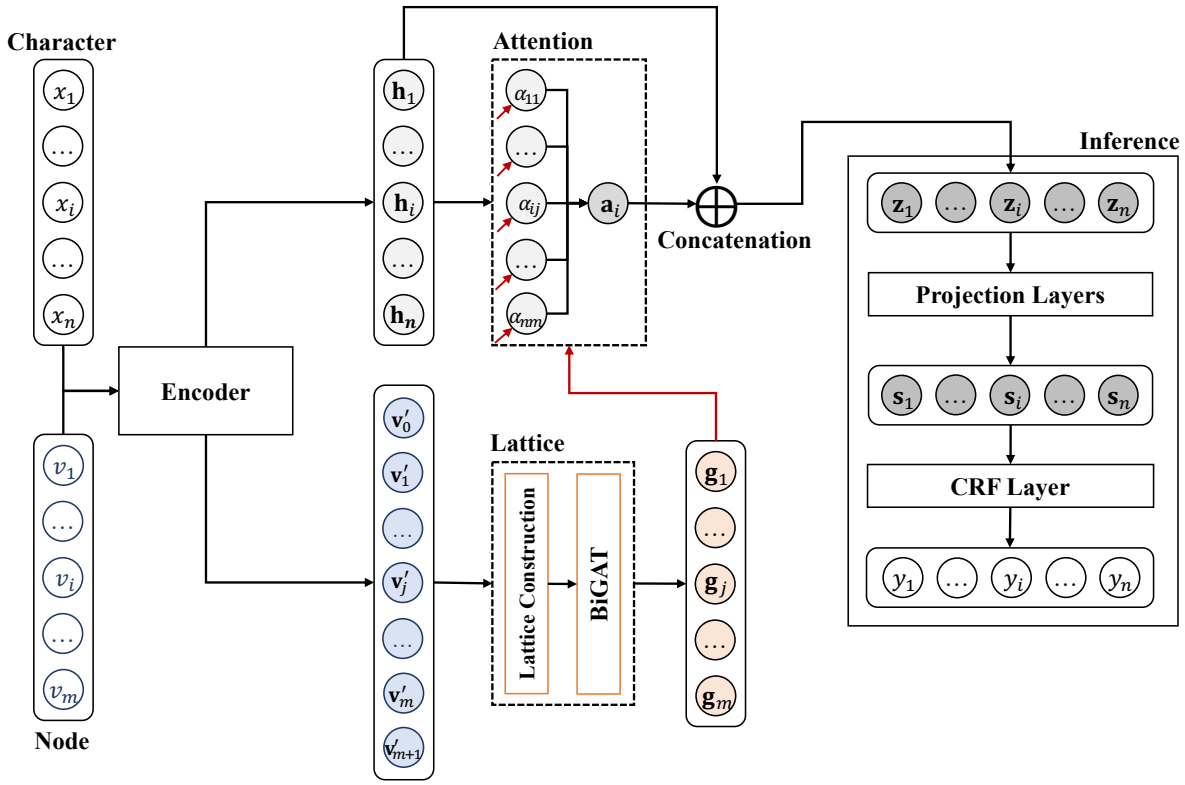


Figure 4.1: Our proposed model that integrates a lattice structure and GNNs into a character-based word segmentation model.

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 4.1, 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 characters and words based on the character sequence along with their edges. Subsequently, we encode lattice G to obtain *a multi-granularity 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.

4.1.1 Character Encoding

We employ either BiLSTM or BERT as an encoder, to transform a character sequence $x_{1:n}$ 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. Note that 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:

$$\begin{aligned} \mathbf{h}^l_i &= \text{BiLSTM}(\mathbf{h}^{l-1}_{1:n}, i) \\ &\equiv \text{LSTM}_f(\mathbf{h}^{l-1}_{1:n}, i) \\ &\oplus \text{LSTM}_b(\mathbf{h}^{l-1}_{n:1}, n-i+1), \end{aligned} \tag{4.1.1}$$

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:[1] 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 $\mathbf{u}_{0:n+1}$. Note that a character x_i can become [UNK] token assuming that the character does not exist in BERT vocabulary. The one-hot representation of characters $\mathbf{u}_{0:n+1}$ are then transformed into contextualized character vectors $\mathbf{h}_{0:n+1}$ of d_{BERT}-dimensional feature vector.

[1] Here, we used Multi-criteria BERT (MC-BERT) PTM as described in Section 4.1.4.

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

$$\mathbf{h}_i^0 = W_e \mathbf{u}_i + W_p[i], \tag{4.1.2}$$

$$\mathbf{h}_i^l = \text{Transformer}(\mathbf{h}_i^{l-1}), \tag{4.1.3}$$

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 weight of the positional encoding at the i^{th} index, and 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. In addition, the summation of the last four layers is used to obtain the contextualized character vector $\mathbf{h}_i$ as in Yang [2019].

4.1.2 Lattice Attentive Encoding

We propose the lattice attentive encoding method to attentively extract the representation of multi-granularity linguistic units from possible segmentation alternatives. These segmentation alternatives are represented by lattices based on multi-granularity units.[2]

Let $G = (V, E)$ where G is a directed acyclic graph (DAG) for a character sequence $x_{1:n}$. Here, $V = \{v_1, \ldots, v_{|V|}\}$ is the set of vertices or nodes, and $E \subseteq \{V \times V\}$ is the set of edges. To construct the nodes $v_{1:m} = V$ of the lattice G, we search for possible characters and words based on the character sequence $x_{1:n}$, where m is the number of nodes (i.e., the found characters and words). Each node preserves a character sequence of length up to k, where each sequence represents either a character or a word. Specifically, the lattice G comprises nodes that correspond to a character or word w of length $1 < |w| \leq k$ and edges that connect nodes of adjacent characters or words in the character sequence (the sentence).

We employ either BiLSTM-encoder or BERT-encoder to initialize feature vectors for the nodes in the lattice G, which include the 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 (word) 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 Section 4.1.1 (Equation (4.1.1)). 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

[2]Please refer to Section 4.1.4 for implementation details concerning lattice construction.

$E_w \in \mathbb{R}^{|\mathcal{VOC}_w| \times d_w}$ where $\mathcal{VOC}_w$ denotes the word vocabulary. Special nodes V^s, that is the [BOS] and [EOS] nodes, are used to specify the beginning ($\mathbf{v}_0$) and end ($\mathbf{v}_{m+1}$) of the lattice G, 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 using the contextualized character vectors $\mathbf{h}_{1:n}$ from the BERT encoder, as demonstrated in Section 4.1.1. To initialize word-feature vectors $\mathbf{v}^w$ of the word-node V^w, each word node is augmented with the special tokens, [CLS] and [SEP] tokens. These special tokens are placed at the front and the back of each word node in sequence, respectively. Each word node, now augmented with the [CLS] and [SEP] tokens, is then individually encoded using the BERT-encoder (Equations (4.1.2) and (4.1.3)), yielding the word-feature vectors $\mathbf{v}^w$. The [CLS] and [SEP] nodes are used to specify the start (v_0) and end (v_{m+1}) of the the lattice G, respectively. Note that these special nodes are initialized from $\mathbf{h}_0$ and $\mathbf{h}_{n+1}$ that encode by the BERT-encoder in Section 4.1.1).

In summary, we acquire the contextualized node representations $\mathbf{v}' = \mathbf{v}'_{0:m+1}$, with $\{\mathbf{v}^c, \mathbf{v}^w, \mathbf{v}^s\} \in \mathbf{v}'$. This effectively transforms the original lattice G into a new lattice G', with $G' = (\mathbf{v}', E)$ Note that a single-character word is treated similarly to a character in the construction of a lattice and in deriving its representation from either the BiLSTM or BERT.

We then employ GAT to encode the lattice G', acquiring multi-granularity 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-granularity contextualized node representation $\mathbf{g}_{1:m}$ can be obtained from GAT[3] 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,[4] respectively. Finally, the multi-head attention is employed to compute the multi-granularity contextualized node representations $\mathbf{g}_j$.

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

where $\mathbf{g}_j \in \mathbb{R}^{d_g}$, d_g is a hyperparameter, Q indicates the number of attention-head, and σ

[3]We used a bidirectional variant of GAT (BiGAT) as described in Section 4.1.4 (Equation (4.1.5)) to obtain the multi-granularity contextualized node representations $\mathbf{g}$.

[4]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., $v_i \notin O_j$)

represents a nonlinear transformation, i.e., LeakyReLU.

To attentively project a multi-granularity representation out of multi-granularity 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 is involved with a set of nodes that includes the character x_i in the lattice G'. First, based on the contextualized character vector $\mathbf{h}_i$ and its corresponding multi-granularity contextualized node representations $\mathbf{g}_j$ 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:

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

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

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

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

4.1.3 Inference Layer

Projection layers are used to transform the multi-granularity 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, $\mathcal{T} = \{\text{B}, \text{M}, \text{E}, \text{S}\}$. The i^{th} multi-granularity 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 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 $\mathbf{s}$ 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:

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

4.1.4 Implementation Details

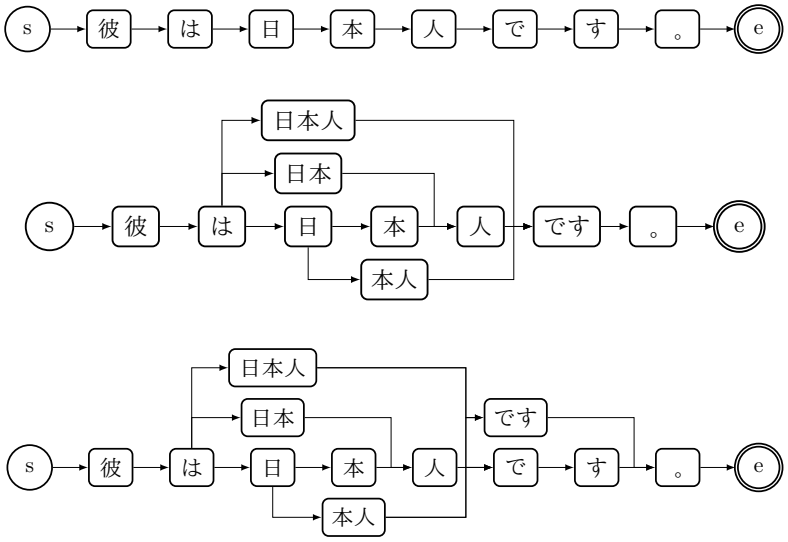
Lattice Construction

A 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 4.2. In this study, we opted for the construction of a word-character-lattice to handle segmentation alternatives comprehensively, leveraging the multi-granularity linguistic units for character-based word segmentation. We built a lattice using all possible combinations according to a character sequence from training vocabulary, which includes the training set and external dictionaries. The lattice also includes special nodes: the start node (s), ending node (e), and a dataset node[5] (criterion token). For the BERT encoder, [CLS] and [SEP] tokens are used as the start and end nodes, respectively. Conversely, for the BiLSTM encoder, [BOS] and [EOS] tokens serve as the start and end nodes. To obtain the substrings of the character sequence while reducing time complexity in lattice construction, we apply the Aho–Corasick algorithm [Aho and Corasick, 1975], which enables linear time complexity.

Additionally, we introduce *dynamic-lattice construction* (DyL). This concept adapts from Bagging (i.e., bootstrap aggregating) [Breiman, 1994] and Dropout [Srivastava et al., 2014] methods, aiming to minimize generalization error and overfitting. During the training process, edges in the lattice are randomly deleted. Consequently, the lattice corresponding to the same

[5]The dataset node represents a feature for the multi-criteria pre-training method, as described in the Multi-criteria Pre-training Method of Section 4.1.4.

character sequence can vary for each epoch.



彼は日本人です。
"He is Japanese."

Figure 4.2: Examples of lattice formation: character-lattice (top), word-lattice (middle), and word-character-lattice (bottom), that can be built for our model.

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 4.3. Parameters in GNN layers such as direction-dependent and trainable parameters are separately exploited according to the direction.

$$\text{BiGNN} = \text{GNN}_f(G_f, \theta_f) \oplus \text{GNN}_b(G_b, \theta_b), \tag{4.1.4}$$

where G denotes lattice, $\theta_\cdot$ denotes the parameters for GNN layers, and $\oplus$ denotes concatenation. f and b represent forward- and backward-direction, respectively. We can also apply variants of GNNs, such as GAT, to Equation (4.1.4).

$$\text{BiGAT} = \text{GAT}_f(G_f, \theta_f) \oplus \text{GAT}_b(G_b, \theta_b). \tag{4.1.5}$$

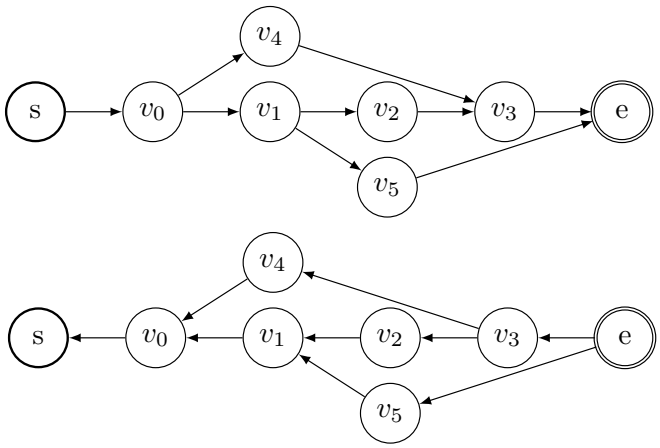


Figure 4.3: Examples of direction-aware lattice: forward-lattice (top) and backward-lattice (bottom).

Multi-criteria Pre-training Method

In this study, we present our multi-criteria pre-training method that is implemented on the BERT architecture, hereby referred to as MC-BERT. This method is an extension of the approach proposed by Ke et al. [2021], incorporating our LATTE during the pre-training of MC-BERT.

To begin with, each input sequence is augmented with the special tokens [CLS] and [SEP] at the start and end, respectively. Following the approach of Ke et al. [2021], we introduce criterion tokens after the [CLS] token to allows the model to learn both criterion-dependent and criterion-independent segmentation knowledge from multiple datasets. The criterion tokens used in this study include criterion-dependent tokens, for example, [CTB6] and [BCCWJ]; and an undefined-criterion token, [UNC] token. Notably, the [UNC] token was used similarly as in Ke et al. [2021]. Each input sequence is typically augmented with a criterion-dependent token, though it is randomly replaced with the undefined-criterion token [UNC] at a rate determined by a hyperparameter, which is set to 10% in this study.

Considering the requirement of multiple corpora for multi-criteria learning, we incorporate additional accessible corpora as specified in Section 4.2.3. Furthermore, we modify the conventional MC-BERT proposed by Ke et al. [2021] by integrating LATTE as a key component in the pre-training stage.[6] This includes the construction of a lattice from training data and the application of BiGAT for pre-training representations in MC-BERT. Once pre-trained, MC-BERT is used to initialize the node representations as detailed in Sections 4.1.1 and 4.1.2, thereby serving as the encoder depicted in Figure 4.1.

[6]The implementation details can be accessed at `https://github.com/tchayintr/latte-ptm-ws`

4.2 Experiments

4.2.1 Datasets

Three datasets in three languages, i.e., Japanese, Chinese, and Thai, were used to evaluate our model. Table 4.1 shows the statistics of the datasets, including sentences, words, vocabulary, and characters. (1) **BCCWJ**:[7] A Japanese word-segmented corpus that is primarily used in word segmentation experiments. (2) **CTB6**:[8] A Chinese Treebank corpus that is one of the most popular benchmark datasets for Chinese word segmentation. (3) **BEST2010**:[9] 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].

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 4.1: Data sizes, in terms of the number of sentences (S); words (W); vocabulary (V); and characters (Ch), for the BCCWJ, CTB6, and BEST2010 datasets.

4.2.2 External Dictionary and Pre-trained Word Vectors

In building lattices, which are based on vocabulary (characters and words), we supplemented our datasets with an external dictionary for each language. This ensures a comprehensive vocabulary coverage. Note that we used only the training data from our datasets. **Japanese**: UniDic[10] and IPADic[11] for MeCab. **Chinese**: BLCU balanced corpus[12], Train data from SIGHAN2005, and

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

Jieba[13]. **Thai**: HSE Thai Corpus[14] and LEXiTRON[15]. Additionally, to initialize robust word embeddings for the BiLSTM-encoder, we used fastText to generate features for character and word nodes, and kept these embeddings frozen during the training step.

4.2.3 Pre-training Models

Given the variety of existing Pre-Trained Models (PTMs), we selected PTMs based on their originality and accessibility. (1) **Japanese BERT:**[16] We selected the character-level Japanese BERT for the Japanese dataset due to its accessibility and alignment with the character-based approach. (2) **Chinese BERT**:[17] This PTM has proven effective in various neural network models on Chinese datasets. Therefore, we included it in our experiment on the Chinese dataset. Thus, we selected it for use in our experiment on the Chinese dataset. (3) **Multilingual BERT**:[18] Owing to the lack of a Thai pre-trained model similar to the Japanese and Chinese models in terms of originality and accessibility, we opted for this PTM for the Thai dataset. All models are $\text{BERT}_{\text{base}}$ models.

To construct MC-BERT models for Japanese, Chinese, and Thai, we supplemented our main datasets, i.e., BCCWJ, CTB6, and BEST2010, with two, six, and four additional accessible datasets, respectively. Specifically, we used the UD Japanese treebank,[19] and Kyoto University Text Corpus[20] to build the Japanese MC-BERT. For the Chinese MC-BERT, we added six Chinese datasets, four from SIGHAN2005[2] (AS, CITYU, MSRA, and PKU), SXU from SIGHAN2008, and CNC, all obtained from the public repository.[21] All traditional Chinese corpora, such as AS and CITYU, were converted into simplified Chinese. For the Thai MC-BERT, we utilized four Thai datasets, LST20,[22] TNHC,[23] VISTEC, [24] and WS160. [25]

In total, we used three datasets for Japanese, seven for Chinese, and five for Thai in the construction of the respective MC-BERT models. For accessibility reasons, we performed the pre-training methods for Chinese MC-BERT on seven datasets instead of nine as in the previous work [Ke et al., 2021].

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

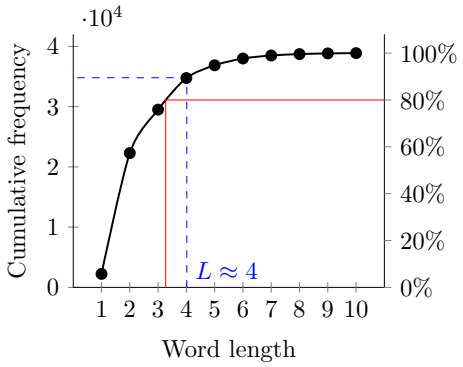
Table 4.2: Common hyperparameters and BERT hyperparameters for reproduced models and our proposed model (top and middle); and essential hyperparameters for our proposed model (bottom).

4.2.4 Hyperparameters

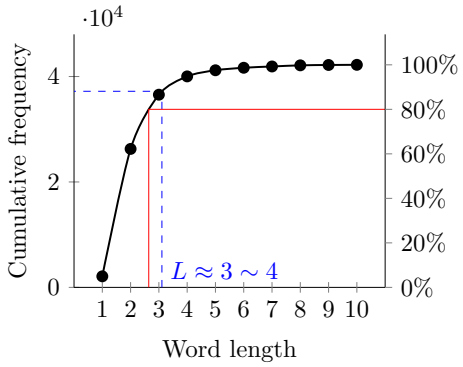
We used the essential hyperparameters for models as shown in Table 4.2. The AdamW optimizer [Loshchilov and Hutter, 2017] was used to optimize the model parameters. Every model was trained for 20 epochs. We selected the best model to perform on the test set based on the validation process by word-level 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.[26] We used

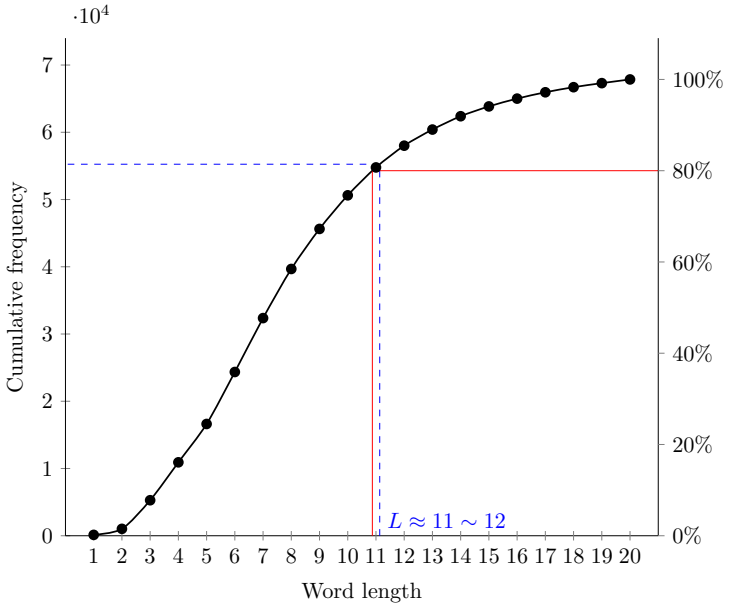
[13]https://github.com/fxsjy/jieba
[14]http://web-corpora.net/ThaiCorpus
[15]https://lexitron.nectec.or.th
[16]https://huggingface.co/cl-tohoku/bert-base-japanese-char-v2
[17]https://huggingface.co/bert-base-chinese
[18]https://huggingface.co/bert-base-multilingual-cased
[19]https://github.com/UniversalDependencies/UD_Japanese-GSD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



(a) Word length and cumulative frequency in BCCWJ



(b) Word length and cumulative frequency in CTB6



(c) Word length and cumulative frequency in BEST2010

Figure 4.4: Word length and cumulative frequency in BCCWJ, CTB6, and BEST2010. Red line indicates the cumulative frequency at 80%. Blue dashed line denotes selected *maximum word length* hyperparameter.

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 4.4(a), 4.4(b), and 4.4(c), 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.

4.2.5 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 4.1.
- **Others**: Popular Well-known word-segmentation models [Neubig et al., 2011, Kitagawa and Komachi, 2018, Qiu et al., 2020, Tian et al., 2020c, Huang et al., 2020b, 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].

4.2.6 Evaluation Metrics

Previous works commonly evaluate models using different evaluation metrics depending on the language. 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 [Ke et al., 2021]. Previous works on Thai word-segmentation interchangeably evaluated on the character-level-F_1 and word-level-F_1 [Limkonchotiwat et al., 2021, Chay-intr et al., 2021].

We chose to evaluate all models for our main results by using these three evaluation metrics: character-level-F_1 and word-level-F_1, and OOV-recall scores. We adopted word-level-F_1 and OOV-recall score as in Qiu et al. [2020], and we followed character-level-F_1 evaluation method from Limkonchotiwat et al. [2021], Chay-intr et al. [2021].

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

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

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

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.

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

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

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

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.

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

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

where pred, 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.

4.3 Results and Analysis

4.3.1 Main Results

Tables 4.3, 4.4, and 4.5 show comparisons of previous methods, baselines, and our proposed model, LATTE, on the three datasets: BCCWJ, CTB6, and BEST2010, respectively. The results indicate that LATTE outperformed both the baseline models and the previous methods across all selected evaluation metrics.

Model	$\mathbf{F}_{\text{char}}$ (σ)	$\mathbf{F}_{\text{word}}$ (σ)	$\mathbf{R}_{\text{oov}}$ (σ)
[Neubig et al., 2011]	-	98.2	-
[Kitagawa and Komachi, 2018]	-	98.4	-
[Higashiyama et al., 2019]	-	98.9	-
BiLSTM-CRF	99.2 (0.005)	98.2 (0.0200)	81.5 (0.060)
BiLSTM-WAVG-CRF	99.3 (0.025)	98.2 (0.005)	70.3 (0.690)
BERT-CRF	99.7 (0.005)	99.3 (0.030)	92.0 (0.010)
BERT-MC-CRF	99.7 (0.005)	99.3 (0.015)	91.6 (0.060)
LATTE w/ BiLSTM	99.5 (0.005)	99.0 (0.005)	83.6 (0.040)
LATTE	**99.8 (0.005)**	**99.4 (0.005)**	**92.1 (0.005)**

Table 4.3: Comparison of segmentation performance among Others; Baselines; and our proposed model, on BCCWJ dataset. The best score for each metric is indicated in **bold**. $\mathbf{F}_{\text{char}}$, $\mathbf{F}_{\text{word}}$, and $\mathbf{R}_{\text{oov}}$ denote the Char-level-F_1, 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	$\mathbf{F}_{\text{char}}$ (σ)	$\mathbf{F}_{\text{word}}$ (σ)	$\mathbf{R}_{\text{oov}}$ (σ)
[Qiu et al., 2020]	-	97.0	87.0
[Tian et al., 2020a]	-	97.4	88.5
[Tian et al., 2020c]	-	97.2	88.0
[Huang et al., 2020b]	-	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	97.7 (0.005)	94.4 (0.010)	75.5 (0.005)
BiLSTM-WAVG-CRF	98.1 (0.005)	95.1 (0.005)	63.3 (0.005)
BERT-CRF	99.2 (0.005)	97.8 (0.035)	89.2 (0.505)
BERT-MC-CRF	99.2 (0.035)	97.9 (0.035)	90.5 (0.120)
LATTE w/BiLSTM	98.4 (0.010)	95.8 (0.000)	78.5 (0.050)
LATTE	**99.3 (0.005)**	**98.1 (0.020)**	**90.6 (0.135)**

Table 4.4: Comparison of segmentation performance among Others; Baselines; and our proposed model, on the CTB6 dataset. The best score for each metric is indicated in **bold**. $\mathbf{F}_{\text{char}}$, $\mathbf{F}_{\text{word}}$, and $\mathbf{R}_{\text{oov}}$ denote the Char-level-F_1, 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	$\mathbf{F}_{\text{char}}$ (σ)	$\mathbf{F}_{\text{word}}$ (σ)	$\mathbf{R}_{\text{oov}}$ (σ)
[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)	57.0 (0.020)
BiLSTM-WAVG-CRF	98.9 (0.005)	97.2 (0.005)	57.3 (0.003)
BERT-CRF	99.0 (0.005)	97.3 (0.045)	62.7 (0.045)
BERT-MC-CRF	99.0 (0.010)	97.6 (0.005)	66.1 (0.050)
LATTE w/BiLSTM	99.0 (0.005)	97.3 (0.015)	62.9 (0.025)
LATTE	**99.1 (0.005)**	**97.7 (0.015)**	**67.9 (0.035)**

Table 4.5: Comparison of segmentation performance among Others; Baselines; and our proposed model, on the BEST2010 dataset. The best score for each metric is indicated in **bold**.$\mathbf{F}_{\text{char}}$, $\mathbf{F}_{\text{word}}$, and $\mathbf{R}_{\text{oov}}$ denote the Char-level-F_1, 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.

LATTE outperforms 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. [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 the 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, Chay-intr et al. [2021] achieved promising segmentation performance by incorporating multiple attentions from word and character-cluster information, which is exclusive knowledge for Thai writing system, 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.

4.3.2 Segmentation Performance with Additional Datasets

We tested our model when some unseen 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}[27] (Chinese). Note that we pulled UD_{JA} from Japanese MC-BERT pre-training to conduct this test.

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

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

Table 4.6 displays the segmentation performance results for the two additional datasets. As the results indicate, our proposed model outperformed the BiLSTM-CRF, BERT-CRF, and BERT-MC-CRF baselines on both datasets. The use of MC-BERT contributed to the improvement in segmentation performance on UDJA and UDZH. Moreover, our proposed model was able to further enhance segmentation performance when we used LATTE as a component in conjunction with MC-BERT.

4.3.3 Ablation Study

LATTE achieved superior segmentation performance beyond previous works by integrating three major components: BiGAT, MC-BERT, and DyL. To analyze the effect of these components on our proposed model, we conducted an ablation study. This study was based on LATTE integrated with the BERT-encoder on three datasets: BCCWJ, CTB6, and BEST2010.

Dataset	**BiGAT**	**MC-BERT**	**DyL**	$\mathbf{F}_{\text{word}}$
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 4.7: 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**.

The results from this study, as displayed in Table 4.7, indicate that the integration of BiGAT, MC-BERT, and DyL enhances segmentation performance. However, the application of MC-BERT to the BCCWJ corpus did not significantly improve segmentation performance as observed in the case of CTB6 and BEST2010. Interestingly, while the incorporation of BiGAT consistently improved performance across all datasets, the inclusion of MC-BERT did not sig-

nificantly improve performance on the BCCWJ corpus to the same extent as on the CTB6 and BEST2010 corpora. Similarly, while the use of DyL generally enhanced segmentation performance, particularly on the BCCWJ and CTB6 datasets, it slightly lessened the performance on the BEST2010 dataset. This BEST2010 dataset includes longer character sequences than the other two, which might explain this difference.

4.3.4 Case Study: Segmentation Results

To show whether specific cases from segmentation results[28] 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 4.5 and 4.6 show segmentation results between BERT-MC-CRF and the proposed LATTE, respectively. Figure 4.5 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 4.6 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. This suggests that, while LATTE excels in certain areas, there may be room for improvement in its handling of idiomatic language.

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

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 4.5: Examples of segmentation results between BERT-MC-CRF and LATTE on the CTB6 dataset. 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 4.6: Examples of segmentation results between BERT-MC-CRF and LATTE on the CTB6 dataset. 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.

Additionally, we selected segmentation results from BCCWJ and BEST2010 to illustrate the cases where LATTE outperformed BERT-MC-CRF as shown in Figures 4.7 and 4.8.

In terms of meaning, the segmentation results from BCCWJ are not significantly different. The major difference between the performance of BERT-MC-CRF and LATTE lies in how they handle 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 “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. Although both BERT-MC-CRF and LATTE segmented the honorific prefix “お” separately from “古 (old)”, this did not change the overall meaning of the phrase. 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. LATTE was able to accurately segment the sentence, preserving its original meaning, while the segmentation result from BERT-MC-CRF yielded a sentence with a completely different meaning Ultimately, LATTE significantly outperformed BERT-MC-CRF based on this sample.

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 4.7: Examples of segmentation results between BERT-MC-CRF and LATTE on the BCCWJ dataset. 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 4.8: Examples of segmentation results between BERT-MC-CRF and LATTE on the BEST2010 dataset. 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.

In summary, the comparative analysis of segmentation results between BERT-MC-CRF and LATTE demonstrates that the incorporation of LATTE leads to improvements in the segmentation process, especially in terms of preserving word categories and handling character connections. However, challenges remain when dealing with idiomatic language, indicating potential areas for future model enhancements. Despite these challenges, the examples clearly demonstrate the significant performance improvements achieved by LATTE over BERT-MC-CRF across different language datasets.

4.4 Conclusion for this Chapter

In this chapter, we proposed Lattice ATTentive Encoding (LATTE), a method that uses lattices to leverage potential segmentation alternatives based on multi-granularity linguistic units, including character and word units for character-based word segmentation. LATTE build a lattice based on character and word units. The representations of these units were initialized and encoded with PTM BERT and GNNs, respectively. Thereafter, we incorporated an attention mechanism to attentively extract multi-granularity representations from the lattice to complement the character representations.

According to our experimental results, it showed that LATTE improved segmentation performance on three well-known datasets, including BCCWJ, CTB6, and BEST2010. We conducted various analyses, such as examining segmentation performance with additional datasets, conducting an ablation study, and inferring segmentation results, to validate the effectiveness of our model. Our analysis consistently affirmed the superiority of LATTE over previous works.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

Character-based word segmentation has been successfully applied to Asian languages, including Japanese, Chinese, and Thai, by using a character unit as the fundamental information source. However, the character unit may lack inherent meaning compared to larger linguistic units such as CC, subword, and word units, leading to segmentation ambiguity in a character sequence. Although either subword or word unit has been incorporated into character-based word segmentation through various methods, the full potential of jointly using multi-granularity linguistic units to handle possible segmentation alternatives remains a challenge.

We conducted a study exploring utilizing a broader range of multi-granularity linguistic units and properly leveraging a set of potential segmentation alternatives based on these units to improve character-based word segmentation. In the first aspect, we presented a method for jointly utilizing multi-granularity linguistic units, including CCs, subwords, and words, in addition to a character sequence, particularly for the Thai language. Our approach employs attention mechanisms at each granularity level to establish relationships between character representations and multi-granularity units. This information is then utilized to compute context features to enrich the character representations.

Our experiments demonstrated that by utilizing multi-granularity units with multiple attention mechanisms, our method outperforms previous work in Thai word segmentation on three benchmark datasets: BEST2010, TNHC, and VISTEC. Furthermore, our case study highlights the advantages of using CCs with an attention mechanism in our model over subwords, evidenced improved segmentation performance and better adherence to the Thai writing system rules. These improvements were observed when comparing our results to the actual segmentation outputs from previous work.

Regarding the second aspect, we proposed Lattice ATTentive Encoding (LATTE), a method to leverage possible segmentation alternatives based on multi-granularity linguistic units through the lattice for generating context features to complement the representation of characters. LATTE uses a multi-path lattice to handle possible segmentation alternatives based on character and word units. The representations of these units are then initialized and encoded with the PTM BERT and GNNs, respectively. Subsequently, we employed an attention mechanism to

attentively extract multi-granularity representations from the lattice to estimate a context feature between the representation of each character and its corresponding character- and word-node features. The context features are then used to complement character representations through a concatenation operation.

Our method demonstrated an improvement in segmentation performance by outperforming previous work on the BCCWJ, CTB6, and BEST2010 datasets in Japanese, Chinese, and Thai, respectively. In addition, we conducted various analyses, including segmentation performance with additional datasets, ablation study, and inference of segmentation results, to verify the effectiveness of our model. Our analysis consistently demonstrated the superiority of LATTE over previous work.

5.2 Future Work

Regarding the first aspect, while our study advances segmentation performance, using multi-granularity linguistic units with attention mechanisms and concatenation operations increases computational demands and time complexity. Combining these multiple attentions into a single attention mechanism could alleviate this limitation. Moreover, we have explored the potential of using these linguistic units specifically for the Thai language, due to its existing broader range of linguistic units, including CCs, subwords, and words. In cases where other linguistic units can be extracted from Japanese and Chinese languages, the utilization of these units with our approach may lead to an improvement in segmentation performance. Additionally, we used multilingual BERT for its accessibility and Thai language effectiveness. However, creating new pre-trained models (PTMs) for fine-tuning our method is a potential area for future work, considering time complexity implications.

Regarding Lattice ATTentive Encoding (LATTE), our current model uses only characters and words. The integration of such linguistic units in future research could potentially lead to further improvements in segmentation performance. In addition, because we evaluated our LATTE on several languages, applying it to other languages could further emphasize our contributions, especially if it successfully outperforms other methods in these languages.

Acknowledgement

I would like to express my heartfelt gratitude to several individuals who have significantly contributed to my doctoral journey. Without their influence and faith in my abilities, this thesis would not have been possible. Their impact extends beyond the confines of this academic work, influencing my growth and approach towards future challenges.

Firstly, I am deeply thankful to Okumura sensei, Takamura sensei, and Funakoshi sensei. Your academic support, invaluable guidance, and patience have been instrumental in shaping my research work. You have fostered an environment of rigorous intellectual curiosity that has helped me learn and grow beyond my expectations. Despite my shortcomings, you never ceased to believe in my potential, and for that, I am forever grateful. Your teachings will continue to inspire and guide me in my future endeavors, be it in academia or industry. I am also profoundly indebted to Thanaruk sensei, whose encouragement played a pivotal role in my decision to pursue a doctoral degree. Your faith in my capabilities has been a source of motivation, and your insights have enriched my academic perspective.

Secondly, I would like to extend my sincere thanks to all my lab members in the Okumura-Takamura Laboratory, now known as the Okumura-Funakoshi Laboratory. Your camaraderie, collaboration, and shared insights have been an integral part of this journey. Thirdly, I must express my deep gratitude to the NSK Scholarship Foundation. The trust and financial support I received from the foundation was crucial in enabling me to complete this study

Lastly, my profound gratitude goes to my family. Your unwavering faith in me, constant encouragement, and emotional support have been my source of strength throughout this journey. You have always been there for me, even when times were tough, and I could not have accomplished this without you. Once again, I am immensely grateful to all of you. Your support and belief in me will always hold a special place in my heart.

References

Alfred V. Aho and Margaret J. Corasick. Efficient string matching: An aid to bibliographic search. *Communication of the ACM*, 18(6):333–340, jun 1975. ISSN 0001-0782. doi: 10.1145/360825.360855. URL https://doi.org/10.1145/360825.360855.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1409.0473.

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, 2003. URL https://dl.acm.org/doi/10.5555/944919.944966.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017. doi: 10.1162/tacl_a_00051. URL https://aclanthology.org/Q17-1010.

Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):114–133, 1994. doi: 10.48550/ARXIV.1712.02856.

Deng Cai and Hai Zhao. Neural word segmentation learning for Chinese. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 409–420, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1039. URL https://aclanthology.org/P16-1039.

Alberto Cetoli, Stefano Bragaglia, Andrew O'Harney, and Marc Sloan. Graph convolutional networks for named entity recognition. In *Proceedings of the 16th International Workshop on Treebanks and Linguistic Theories*, pages 37–45, Prague, Czech Republic, 2017. URL https://aclanthology.org/W17-7607.

Thodsaporn Chay-intr, Hidetaka Kamigaito, and Manabu Okumura. Character-based Thai word segmentation with multiple attentions. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 264–273, Held Online, September 2021. INCOMA Ltd. URL https://aclanthology.org/2021.ranlp-1.31.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, and Xuanjing Huang. Gated recursive neural network for Chinese word segmentation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1744–1753, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1168. URL https://aclanthology.org/P15-1168.

Pattarawat Chormai, Ponrawee Prasertsom, and Attapol T. Rutherford. Attacut : A fast and accurate neural thai word segmenter, 2019. URL https://arxiv.org/abs/1911.07056.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011. URL https://arxiv.org/abs/1103.0398.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

Christopher Dyer, Smaranda Muresan, and Philip Resnik. Generalizing word lattice translation. In *Proceedings of ACL-08: HLT*, pages 1012–1020, Columbus, Ohio, June 2008. Association for Computational Linguistics. URL https://aclanthology.org/P08-1115.

Felix Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. *Neural Computation*, 12(10):2451–2471, 2000. URL https://doi.org/10.1162/089976600300015015.

Tao Gui, Yicheng Zou, Qi Zhang, Minlong Peng, Jinlan Fu, Zhongyu Wei, and Xuanjing Huang. 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)*, pages 1040–1050, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1096. URL https://aclanthology.org/D19-1096.

Choochart Haruechaiyasak and Sarawoot Kongyoung. 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, 2009.

Choochart Haruechaiyasak, Sarawoot Kongyoung, and Matthew Dailey. 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, 2008. URL https://ieeexplore.ieee.org/document/4600388.

Han He, Lei Wu, Hua Yan, Zhimin Gao, Yi Feng, and George Townsend. Effective neural solution for multi-criteria word segmentation, 2017. URL https://arxiv.org/abs/1712.02856.

Ryuichiro Higashinaka, Masahiro Araki, Hiroshi Tsukahara, and Masahiro Mizukami. Integrated taxonomy of errors in chat-oriented dialogue systems. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 89–98, Singapore and Online, July 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.sigdial-1.10.

Shohei Higashiyama. Word segmentation and lexical normalization for unsegmented languages. 2022. URL https://library.naist.jp/opac/en/book/102590.

Shohei Higashiyama, Masao Utiyama, Eiichiro Sumita, Masao Ideuchi, Yoshiaki Oida, Yohei Sakamoto, and Isaac Okada. 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, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1276. URL https://aclanthology.org/N19-1276.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. URL https://dl.acm.org/doi/10.1162/neco.1997.9.8.1735.

Kaiyu Huang, Degen Huang, Zhuang Liu, and Fengran Mo. 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)*, pages 3873–3882, Online, November 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.318. URL https://aclanthology.org/2020.emnlp-main.318.

Kaiyu Huang, Hao Yu, Junpeng Liu, Wei Liu, Jingxiang Cao, and Degen Huang. Lexicon-based graph convolutional network for Chinese word segmentation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2908–2917, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.248. URL https://aclanthology.org/2021.findings-emnlp.248.

Weipeng Huang, Xingyi Cheng, Kunlong Chen, Taifeng Wang, and Wei Chu. Towards fast and accurate neural Chinese word segmentation with multi-criteria learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2062–2072, Barcelona, Spain (Online), December 2020b. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.186. URL https://aclanthology.org/2020.coling-main.186.

Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging, 2015. URL https://arxiv.org/abs/1508.01991.

Jussi Jousimo, Natsuda Laokulrat, Ben Carr, Ekkalak Thongthanomkul, and Vee Satayamas. Thai word segmentation with bi-directional rnn, 2017. URL https://github.com/sertiscorp/thai-word-segmentation.

Asanee Kawtrakul and Chalathip Thumkanon. A statistical approach to Thai morphological analyzer. In *Fifth Workshop on Very Large Corpora*, 1997. URL https://aclanthology.org/W97-0126.

Zhen Ke, Liang Shi, Erli Meng, Bin Wang, Xipeng Qiu, and Xuanjing Huang. Unified multi-criteria chinese word segmentation with bert. arXiv:2004.05808, 2020. doi: 10.48550/ARXIV.2004.05808. URL https://arxiv.org/abs/2004.05808.

Zhen Ke, Liang Shi, Songtao Sun, Erli Meng, Bin Wang, and Xipeng Qiu. 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*, pages 5514–5523, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.436. URL https://aclanthology.org/2021.naacl-main.436.

Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In *In Proceedings of the 3rd International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6980.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2016. URL https://arxiv.org/abs/1609.02907.

Nikita Kitaev and Dan Klein. 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, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1249. URL https://aclanthology.org/P18-1249.

Yoshiaki Kitagawa and Mamoru Komachi. Long short-term memory for Japanese word segmentation. In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation*, Hong Kong, 1–3 December 2018. Association for Computational Linguistics. URL https://aclanthology.org/Y18-1033.

Rakpong Kittinaradorn, Titipat Achakulvisut, Korakot Chaovavanich, Kittinan Srithaworn, Pattarawat Chormai, Chanwit Kaewkasi, Tulakan Ruangrong, and Krichkorn Oparad. Deepcut: A thai word tokenization library using deep neural network, 2019. URL http://doi.org/10.5281/zenodo.3457707.

Philipp Koehn. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-3250.

Canasai Kruengkrai, Virach Sornlertlamvanich, and Hitoshi Isahara. A conditional random field framework for Thai morphological analysis. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy, May 2006. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2006/pdf/137_pdf.pdf.

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

Taku Kudo. 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, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1007. URL https://aclanthology.org/P18-1007.

Taku Kudo and John Richardson. 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, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-2012. URL https://aclanthology.org/D18-2012.

John Lafferty, Andrew Mccallum, and Fernando Pereira. Conditional random fields : Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, 2001. URL https://dl.acm.org/doi/10.5555/645530.655813.

Yuxuan Lai, Yijia Liu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 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*, pages 1716–1731, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.137. URL https://aclanthology.org/2021.naacl-main.137.

Theerapat Lapjaturapit, Kobkrit Viriyayudhakom, and Thanaruk Theeramunkong. 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, 2018. URL https://ieeexplore.ieee.org/document/8442053.

Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Xiaonan Li, Hang Yan, Xipeng Qiu, and Xuanjing Huang. FLAT: Chinese NER using flat-lattice transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6836–6842, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.611. URL https://aclanthology.org/2020.acl-main.611.

Xiaoya Li, Yuxian Meng, Xiaofei Sun, Qinghong Han, Arianna Yuan, and Jiwei Li. 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, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1314. URL https://aclanthology.org/P19-1314.

Piya Limcharoen, Cholwich Nattee, and Thanaruk Theeramunkong. 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*, 2009.

Peerat Limkonchotiwat, Wannaphong Phatthiyaphaibun, Raheem Sarwar, Ekapol Chuangsuwanich, and Sarana Nutanong. 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, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.315. URL https://aclanthology.org/2020.emnlp-main.315.

Peerat Limkonchotiwat, Wannaphong Phatthiyaphaibun, Raheem Sarwar, Ekapol Chuangsuwanich, and Sarana Nutanong. Handling cross- and out-of-domain samples in Thai word segmentation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1003–1016, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.86. URL https://aclanthology.org/2021.findings-acl.86.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2017. URL https://arxiv.org/abs/1711.05101.

Thang Luong, Hieu Pham, and Christopher D. Manning. 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, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1166. URL https://aclanthology.org/D15-1166.

Mieradilijiang Maimaiti, Yang Liu, Yuanhang Zheng, Gang Chen, Kaiyu Huang, Ji Zhang, Huanbo Luan, and Maosong Sun. Segment, mask, and predict: Augmenting Chinese

word segmentation with self-supervision. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2068–2077, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.158. URL https://aclanthology.org/2021.emnlp-main.158.

Tetsuji Nakagawa. Chinese and Japanese word segmentation using word-level and character-level information. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 466–472, Geneva, Switzerland, aug 23–aug 27 2004. COLING. URL https://aclanthology.org/C04-1067.

Tetsuji Nakagawa and Kiyotaka Uchimoto. A hybrid approach to word segmentation and POS tagging. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 217–220, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://aclanthology.org/P07-2055.

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

Graham Neubig, Yosuke Nakata, and Shinsuke Mori. Pointwise prediction for robust, adaptable Japanese morphological analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 529–533, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-2093.

Wannaphong Phatthiyaphaibun, Korakot Chaovavanich, Charin Polpanumas, Arthit Suriyawongkul, Lalita Lowphansirikul, and Pattarawat Chormai. Pythainlp: Thai natural language processing in python, 2016. URL http://doi.org/10.5281/zenodo.3519354.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. BPE-dropout: Simple and effective subword regularization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1882–1892, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.170. URL https://aclanthology.org/2020.acl-main.170.

Xian Qian and Yang Liu. 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*, pages 501–511, Jeju Island, Korea, July 2012. Association for Computational Linguistics. URL https://aclanthology.org/D12-1046.

Xipeng Qiu, Hengzhi Pei, Hang Yan, and Xuanjing Huang. A concise model for multi-criteria Chinese word segmentation with transformer encoder. In *Findings of the Association for*

Computational Linguistics: EMNLP 2020, pages 2887–2897, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.260. URL https://aclanthology.org/2020.findings-emnlp.260.

Suteera Seeha, Ivan Bilan, Liliana Mamani Sanchez, Johannes Huber, Michael Matuschek, and Hinrich Schütze. ThaiLMCut: Unsupervised pretraining for Thai word segmentation. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 6947–6957, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.858.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 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, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://aclanthology.org/P16-1162.

Virach Sornlertlamvanich, Tanapong Potipiti, and Thatsanee Charoenporn. Automatic corpus-based Thai word extraction with the C4.5 learning algorithm. In *COLING 2000 Volume 2: The 18th International Conference on Computational Linguistics*, 2000. URL https://aclanthology.org/C00-2116.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Salakhutdinov Ruslan. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014. URL http://jmlr.org/papers/v15/srivastava14a.html.

Weiwei Sun. Word-based and character-based word segmentation models: Comparison and combination. In *Coling 2010: Posters*, pages 1211–1219, Beijing, China, August 2010. Coling 2010 Organizing Committee. URL https://aclanthology.org/C10-2139.

Vipas Sutantayawalee, Peerachet Porkeaw, Thepchai Supnithi, Prachya Boonkwan, and Sitthaa Phaholphinyo. Character-cluster-based segmentation using monolingual and bilingual information for statistical machine translation. In *Proceedings of the Fifth Workshop on South and Southeast Asian Natural Language Processing*, pages 94–101, Dublin, Ireland, August 2014. Association for Computational Linguistics and Dublin City University. doi: 10.3115/v1/W14-5513. URL https://aclanthology.org/W14-5513.

Xuemei Tang, Jun Wang, and Qi Su. Chinese word segmentation with heterogeneous graph neural network, 2022. URL https://arxiv.org/abs/2201.08975.

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

Thanaruk Theeramunkong and Sasiporn Usanavasin. Non-dictionary-based Thai word segmentation using decision trees. In *Proceedings of the First International Conference on*

Human Language Technology Research, 2001. URL https://aclanthology.org/H01-1057.

Thanaruk Theeramunkong, Virach Sornlertlamvanich, Thanasan Tanhermhong, and Wirat Chinnan. Character cluster based thai information retrieval. In *Proceedings of the Fifth International Workshop on Information Retrieval with Asian Languages*, pages 75–80, 2000. URL https://dl.acm.org/doi/10.1145/355214.355225.

Yuanhe Tian, Yan Song, Xiang Ao, Fei Xia, Xiaojun Quan, Tong Zhang, and Yonggang Wang. 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*, pages 8286–8296, Online, July 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.735. URL https://aclanthology.org/2020.acl-main.735.

Yuanhe Tian, Yan Song, and Fei Xia. 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), December 2020b. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.187. URL https://aclanthology.org/2020.coling-main.187.

Yuanhe Tian, Yan Song, Fei Xia, Tong Zhang, and Yonggang Wang. Improving Chinese word segmentation with wordhood memory networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8274–8285, Online, July 2020c. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.734. URL https://aclanthology.org/2020.acl-main.734.

Pucktada Treeratpituk. Thai word-segmentation with lstm in tensorflow, 2017. URL https://github.com/pucktada/cutkum.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. URL https://arxiv.org/abs/1706.03762.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2017. URL https://arxiv.org/abs/1710.10903.

Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Peng Cui, P. Yu, and Yanfang Ye. Heterogeneous graph attention network. 2019. doi: 10.48550/ARXIV.1903.07293. URL https://arxiv.org/abs/1903.07293.

Nianwen Xue. Chinese word segmentation as character tagging. In *International Journal of Computational Linguistics & Chinese Language Processing, Volume 8, Number 1, February*

2003: Special Issue on Word Formation and Chinese Language Processing, pages 29–48, February 2003. URL https://aclanthology.org/O03-4002.

Haiqin Yang. Bert meets chinese word segmentation, 2019. URL https://arxiv.org/abs/1909.09292.

Jie Yang, Yue Zhang, and Shuailong Liang. 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, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1278. URL https://aclanthology.org/N19-1278.

Liang Yao, Chengsheng Mao, and Yuan Luo. Graph convolutional networks for text classification, 2018. URL https://arxiv.org/abs/1809.05679.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. In *In Proceedings of the 3rd International Conference on Learning Representations*, 2015. URL http://arxiv.org/abs/1409.2329.

Meishan Zhang, Nan Yu, and Guohong Fu. A simple and effective neural model for joint word segmentation and pos tagging. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26:1528–1538, 2018.

Yue Zhang and Stephen Clark. Chinese segmentation with a word-based perceptron algorithm. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 840–847, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://aclanthology.org/P07-1106.

Yue Zhang and Jie Yang. Chinese NER using lattice LSTM. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1554–1564, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1144. URL https://aclanthology.org/P18-1144.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. Deep learning for Chinese word segmentation and POS tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 647–657, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://aclanthology.org/D13-1061.

Appendix A

Upper-bound Score Test

We made a hypothesis that segmentation results produced from a model may not be the best results, although, 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 $r = \{1, 2, 4, 8, 16\}$. CRF layer is used along with 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, 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.

Dataset	r	$\mathbf{F}_{\text{char}}$
BCCWJ	1	99.2
	2	99.5 (+0.3)
	4	99.7 (+0.5)
	8	99.8 (+0.6)
	16	**99.8 (+0.6)**
CTB6	1	97.8
	2	98.3 (+0.5)
	4	98.7 (+0.9)
	8	98.9 (+1.1)
	16	**99.1 (+1.3)**
BEST2010	1	98.9
	2	99.2 (+0.3)
	4	99.4 (+0.5)
	8	99.5 (+0.6)
	16	**99.6 (+0.7)**

Table A.1: 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 round brackets represent the different from the model where r=1.

Table A.1 shows a comparison of top-r segmentation performance. The results show the same tendency on three datasets, i.e., segmentation performance of the model depends on the increase of r. By comparing with the segmentation performance from top-r segmentation results, where r >1, it simply demonstrates that the best results (r=1) is 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 value to produce more segmentation results from the model to be used for the evaluation, it could obtain superior segmentation performance. These accordingly indicates that in case of top-r information is handled explicitly and properly, it could lead to the improvement of segmentation performance.

Publication

Journal

- Thodsaporn Chay-intr, Kamigaito Hidetaka, and Manabu Okumura. Character-Based Thai Word Segmentation with Multiple Attentions. Journal of Natural Language Processing, 30(2):372-400, 2023.
- Thodsaporn Chay-intr, Kamigaito Hidetaka, Kotaro Funakoshi, and Manabu Okumura. LATTE: Lattice ATTentive Encoding for Character-based Word Segmentation. Journal of Natural Language Processing, 30(2):456-488, 2023.

Conference

- Thodsaporn Chay-intr, Hidetaka Kamigaito, and Manabu Okumura. Character-based Thai word segmentation with multiple attentions. In Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021), pages 264–273, Held Online, September 2021. INCOMA Ltd.